RCA Engineer

Vol. 28 No. 6 Nov./Dec. 1983

Theme: Software

Cover illustration by Louise M. Carr

The allure of hardware . . .
but the need for software.

By incorporating our logo on a floppy disc, our cover illustration symbolizes the inextricably intertwined aspects of hardware and software in the RCA product lines. Software innovations amount to a growing portion of RCA's business, whether embedded in a product or system, or as a stand-alone marketable commodity, and whether stored on a mainframe or mini, or in a chip. One certainly cannot judge the contents of a floppy disc by its cover (nor for that matter, any computer program by the plethora of storage media used today). But between the covers of this timely issue, the first of two issues devoted to software, RCA engineers have written articles on artificial intelligence, communications satellite software, software checking programs, software management schemes, and several specific software applications.

—MRS

RCA Engineer

A technical journal published by
RCA Technical Excellence Center
13 Roszel Road
P.O. Box 432
Princeton, NJ 08540
TACNET: 226-3090 (609-734-3090)

• To disseminate to RCA engineers technical information of professional value • To publish in an appropriate manner important technical developments at RCA, and the role of the engineer • To serve as a medium of interchange of technical information between various groups at RCA • To create a community of engineering interest within the company by stressing the interrelated nature of all technical contributions • To help publicize engineering achievements in a manner that will promote the interests and reputation of RCA in the engineering field • To provide a convenient means by which the RCA engineer may review his professional work before associates and engineering management • To announce outstanding and unusual achievements of RCA engineers in a manner most likely to enhance their prestige and professional status.

A.T. Hospodor

Software engineering—Growing our new roots

Software engineering is the systematic design and development of software products, and the management of the software design, code, integration, test, validation, and verification process. Software engineering is a rigorously demanding and absolutely necessary way of life as we grow our new RCA roots for the complex products of the future.

In 1983, four types of RCA software products come to mind.

1. Software to control physical operations in the factory—*automation.*
2. Software to measure and report the results of operations—*management information systems.*
3. Software to control, monitor, and manage *communications and entertainment systems.*
4. Software *embedded in systems* delivered to our customers.

RCA acts as the customer for the first two types of software products. We design, build, specify, buy, integrate, use and support for our own consumption. The third type of software, for communications and entertainment systems, adds value to a service in which the software is not visible to the user, but is maintained, operated, and supported by RCA as part of its leased or owned plant. Software embedded in systems is totally invisible to the end user. He has only the dim perception of "a computer in there someplace" because he may be using a ubiquitous CRT terminal to enter questions and call for results. Most of our large-scale systems fall in this category.

The computers for which we engineer software range from simple microprocessor chips that might be used in portable test equipment or home entertainment units, to multitasked, multicomputer systems for the control of an air defense weapon system.

In support of this array of products are the tools—languages, editors, compilers, linking loaders, and software development facilities. Each has a special software engineering function, and each must fit exactly with our product development objectives.

"Software engineers" are not necessarily engineers. They come in both genders, and in all sizes, shapes, and backgrounds. Some have a genuine appreciation for the physical world, and others can be thought of as artists, unchained by the limited perceptions that physical models force.

As job complexity expands beyond what a small team can perform, the need for formal software engineering increases dramatically. In general, three good people can do any three-person software job—any way they wish. But in real-time applications, particularly those requiring groups larger than approximately seven people, rigorous application of software engineering discipline becomes mandatory.

The traditional roles of systems and hardware engineering, and quality assurance, are being altered radically as the software engineering content of new RCA systems increases. In many cases, the hardware value added by RCA will be small compared to the software value added. This means that our ability to satisfy our customer's needs will be determined almost entirely by our ability to do our software engineering job.

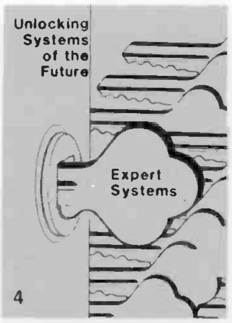
Andrew T. Hospodor

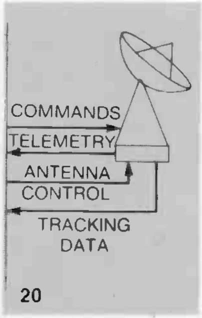
Andrew T. Hospodor
Vice-President and General Manager
RCA Automated Systems

RCA Engineer

Vol. 28|No. 6 Nov.|Dec. 1983

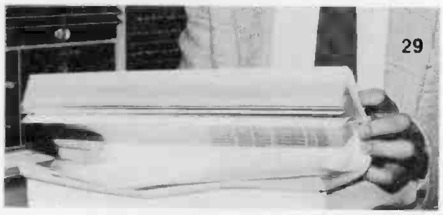
in this issue ...
software engineering





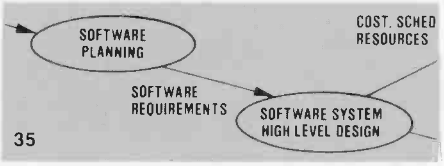
■ **Zapriala, et al.:** *"Artificial intelligence (AI) is a discipline that seeks computer methods to solve the class of problems generally thought to be solvable only by humans."*

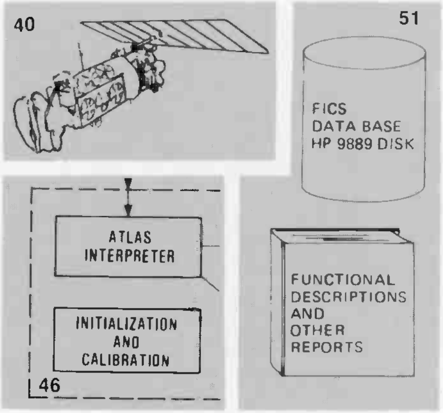
■ **Dusio/Murphy/Cashman:** *"Software has become an essential ingredient in the design, implementation, and operation of communications satellite systems."*



■ **Suhy:** *"Software tools can reduce cost and improve quality at every stage of the software development cycle."*

■ **Resnick:** *"To achieve success, the team must use a blend of development methodology, software tools, and progress-tracking techniques."*

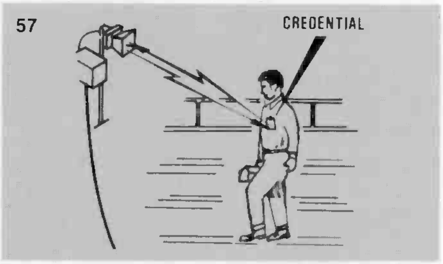



40
51
FICS DATA BASE HP 9889 DISK
ATLAS INTERPRETER
INITIALIZATION AND CALIBRATION
FUNCTIONAL DESCRIPTIONS AND OTHER REPORTS
46

■ **Goodwin:** *"Examples of the on-orbit experience show that hardware redundancy and software monitoring have allowed faults while extending the overall life of the satellites."*

■ **Prenoveau:** *"The augmentation design that was developed enables a method of implementing additional functions compatible with the existing RCA ATE or similar systems using ATLAS."*

■ **Felbinger/Leahy:** *"Because FICS can automatically detect, locate, and identify errors in the functional analysis data base, the resulting products are more accurate, consistent, and comprehensive . . . . "*

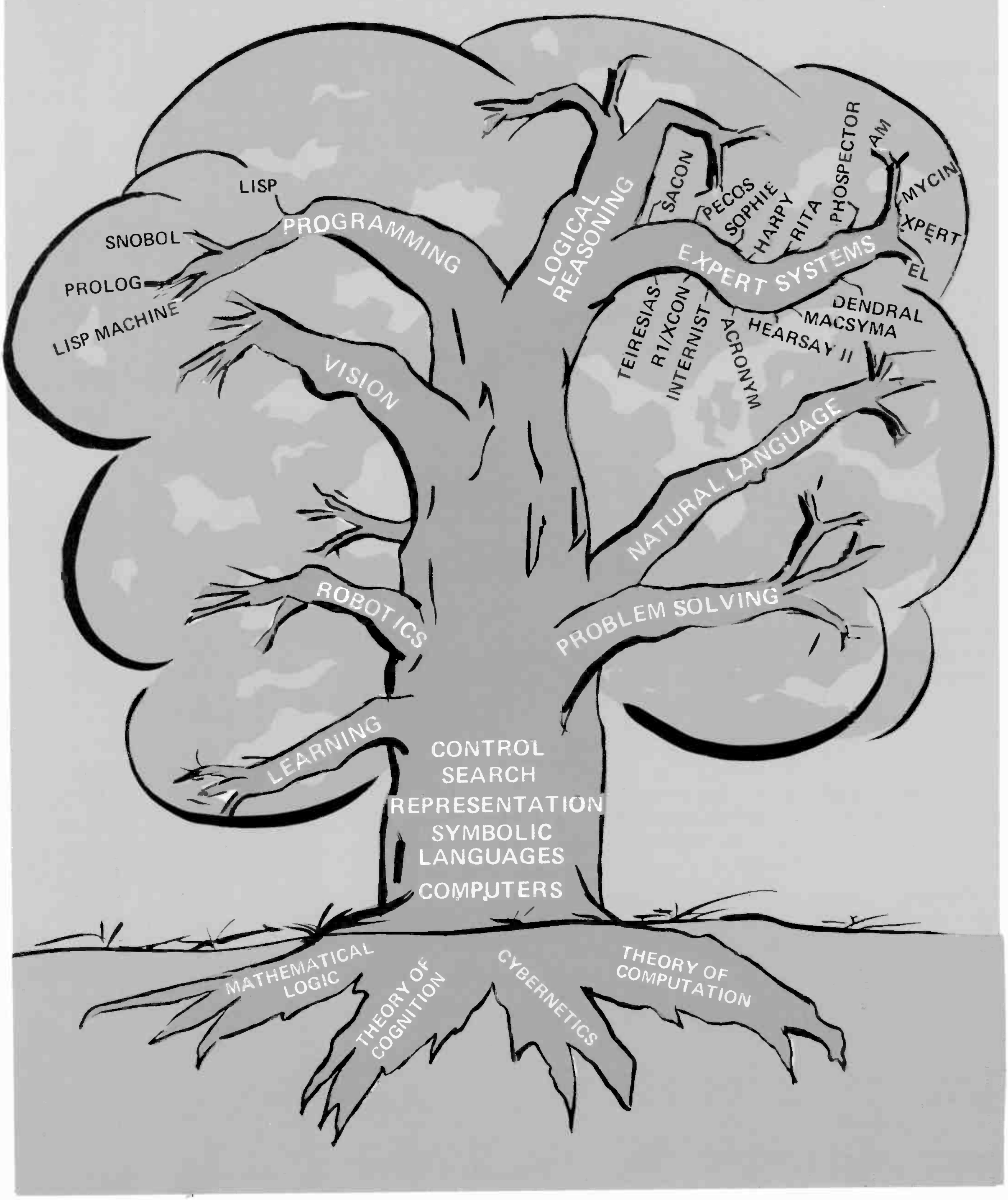


■ **McEachern, et al.:** *"In 1981, Signal Processing Engineering in Somerville, part of Government Communications Systems in Camden, demonstrated the feasibility of a personnel-tracking system to control access to various critical areas . . . . "*

■ **Kitchens:** *"The personal computer cuts the umbilical cord binding users to the mainframe and brings the processing power of the laboratory and the office right into the home."*

in future issues...
software,
manufacturing,
automating the engineer's workplace,
technical excellence

Growth of Artificial Intelligence



J.S. Zapriala | P.R. Spool | J.M. Whitehead
J.A. Gaev | P.J. Kleinosky | J.F. McAdams

Artificial intelligence research at ATL's Software Technology Laboratory

Researchers at RCA are building computerized expert systems by applying specialized software techniques to the analysis and recreation of human expertise.

Abstract: *An overview of the field of Artificial Intelligence is given along with an in-depth look at the subfield of expert systems. Design issues are discussed, and current as well as future research at the Advanced Technology Laboratories is presented.*

Final manuscript received August 15, 1983.
Reprint RE-28-6-1

Artificial Intelligence (AI) is a discipline that seeks computer methods to solve the class of problems generally thought to be solvable only by humans. These include problems that seem to require creativity or some intuitive reasoning, something computers don't seem to possess. AI also seeks to "give" machines some of the natural abilities humans take for granted. These abilities include comprehending spoken as well as written words and sentences, understanding what we see, physically moving in a coordinated way to accomplish tasks of our own choosing, learning from past experience, and reasoning by using abstract concepts.

Twenty-five years ago, when AI was in its infancy, early AI researchers thought that they would provide these "human" abilities to computers within a decade. They soon found that progress was much slower than originally expected. General solutions still elude researchers, but progress has been made in all of the above areas. Some of the results of this research are even beginning to find their way into commercially available products. For example:

- Texas Instruments' "Speak and Spell" electronic game uses a voice synthesis chip to talk.
- A computer program plans molecular genetics experiments.
- An automatic wheelchair and an automatic television set each respond to spoken commands.
- A system evaluates data obtained during oil-well drilling to determine the quality and quantity of the oil in the well.
- A factory visual inspection system recognizes defective parts and rejects them.
- A robot arm coordinated with a vision system reaches into a bin of jumbled parts and extracts a specified one.
- A system identifies organic chemical compounds from mass spectroscopy data with higher accuracy than any human can muster.

To accomplish these feats, AI researchers have had to develop techniques that allow computers to do things previously thought to be impossible. To do this, AI has drawn on knowledge from such fields as Psychology, Philosophy, Computer Science, Linguistics, Theoretical Mathematics, Mechanical Engineering, and Electrical Engineering. AI is very much an interdisciplinary endeavor (Fig. 1).

Expert systems

One of the most fruitful areas within AI is that of expert systems (Fig. 2). According to Edward Feigenbaum, one of the pioneers in expert systems,

> "An expert system is an intelligent computer program that uses knowledge and inference procedures to solve problems that are difficult enough to require significant human expertise for their solution. The knowledge necessary to perform at such a level, plus the inference procedures used, can be thought of as a model of the expertise of the best practitioners of the field."

An expert system consists of (Fig. 3):

- A knowledge base composed of facts as

well as rules of thumb used by human experts in solving problems in their specialty.

- An inference engine (control structure) for applying items from the knowledge base to solve the problem. This theorem prover makes hypotheses, seeks out evidence to support them, evaluates that evidence, chooses the best hypotheses, and keeps track of its line of reasoning.
- A working memory that can keep track of past history as well as the current status of the solution.
- A friendly man/machine interface to facilitate interaction with the system.

The knowledge in an expert system consists of both facts and heuristics. The facts constitute a body of information that is widely shared, publicly available, and generally accepted by experts in the field. The heuristics are mostly personal rules of thumb that experts have developed through years of experience in their particular domain. The performance level of an expert system is primarily a function of the size and quality of the knowledge base that it possesses ("In the Knowledge Lies the Power").

The rules of thumb used by human experts are a very important part of the system. They enable the inferencing portion of the system to rule out unreasonable avenues of possibilities while searching for a solution. It is this ability to trim the set of all possibilities down to some reasonable size that makes the problem solvable at all.

Consider, for example, the game of chess. To judge how wise each possible opening move is, a computer program could evaluate all the possible consequences of that move. That is, it would examine all possible moves its opponent might make, all possible moves it might make in return, and so on, back and forth, until it reaches end-game. The number of possibilities that must be considered using this method is on the order of 10^{120} (a prime example of what is generally termed a combinatorial explosion). Clearly, 10^{120} is an impossibly large set of possibilities to consider in its entirety. Even with a computer 100 times faster than the fastest one available today, it would still take over 10^{100} years to evaluate each move by this method.

A different method is required—an "intelligent" search strategy that eliminates most of these possibilities from consideration so that the problem is solvable in a reason-

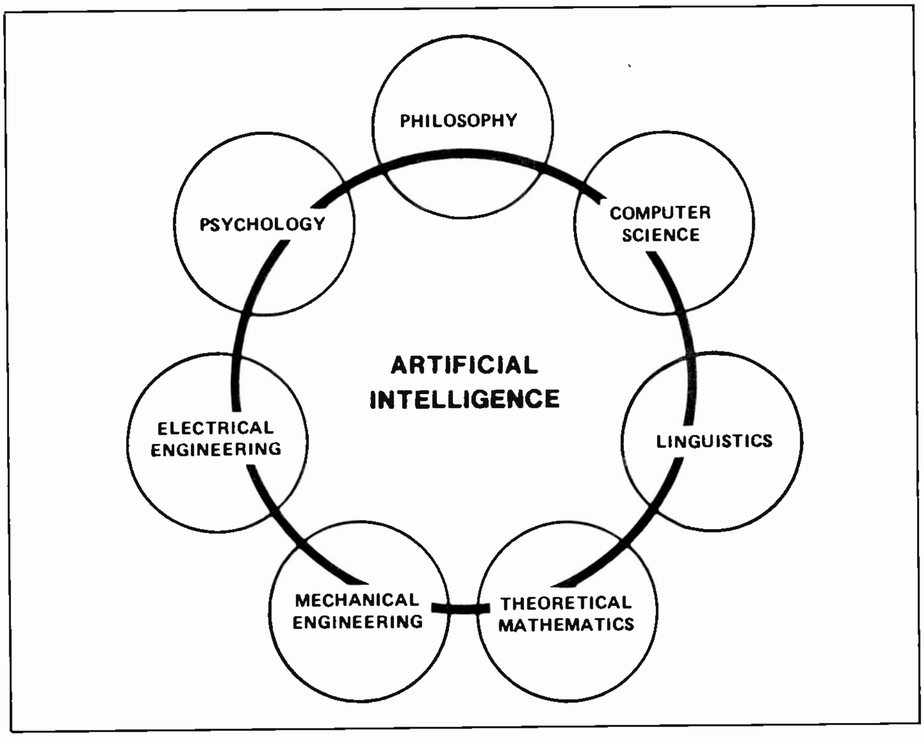


Fig. 1. *Artificial Intelligence is one of the few disciplines within the field of computer science which draws heavily on widely diverse fields such as psychology and mechanical engineering.*

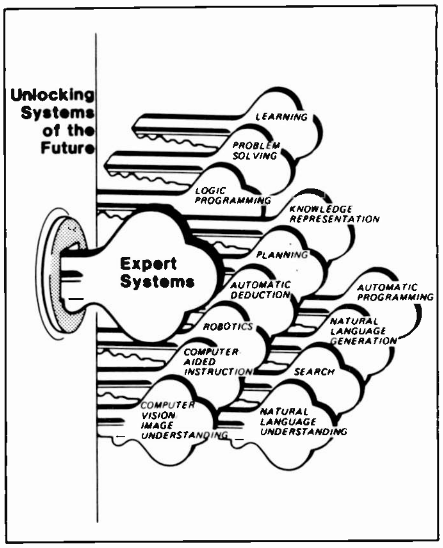

Unlocking Systems of the Future
Expert Systems
LEARNING
PROBLEM SOLVING
LOGIC PROGRAMMING
KNOWLEDGE REPRESENTATION
PLANNING
AUTOMATIC DEDUCTION
AUTOMATIC PROGRAMMING
ROBOTICS
NATURAL LANGUAGE GENERATION
COMPUTER AIDED INSTRUCTION
SEARCH
COMPUTER VISION IMAGE UNDERSTANDING
NATURAL LANGUAGE UNDERSTANDING

Fig. 2. *One of the most fruitful areas within Artificial Intelligence is that concerned with building expert systems.*

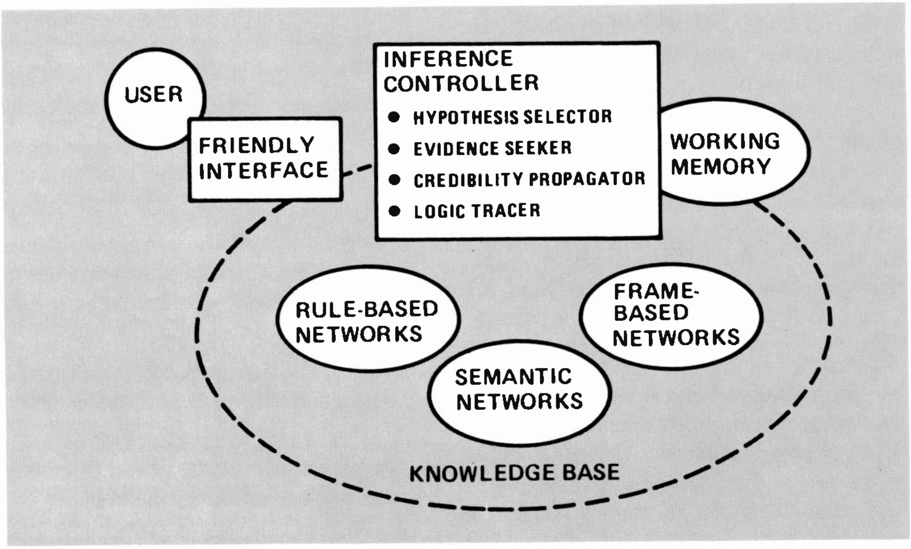


Fig. 3. *The various parts of an expert system, although often developed separately, mesh together to provide users with probable solutions to intricate and complex problems.*

able amount of time. Obviously, human chess players use rules of thumb to make the task of choosing the next move manageable, because they do not examine all possibilities. They concentrate on what they consider to be the best moves and use only a small amount of "look-ahead."

The construction of expert systems should be considered:

- When the use of conventional programming methods does not yield an adequate solution.
- When the problem is not manageable using conventional programming techniques.
- When the demand for expertise exceeds the supply.
- When the amount of data to be interpreted exceeds the human expert's capabilities.
- When human experts cannot produce solutions fast enough.
- When expertise must be made available where it has not been accessible before.

Research at the Advanced Technology Laboratories

The principal goal of our artificial intelligence effort is the demonstration of expert system feasibility. We hope that in doing so, we will be able to gain an intimate understanding of both the opportunities and limitations of knowledge-based systems.

Our approach to expert-system building begins with a careful examination of the problem area. From this examination, a suitable problem is chosen. The solution to the problem is achieved through the development of a series of models of the problem. Each successive model is more sophisticated than its predecessor; the final model represents the actual full-blown problem to be solved. This method of attack enables the expert system builder to progressively gain insight into the problem he is solving. Failure to do this has resulted in severe difficulties. For example, Digital Equipment Corporation had to abandon a one-year effort when they realized they were unable to completely specify what they were interested in achieving. By changing to a model-by-model approach, they were able to eventually reach their current level, where an expert system actually does configure the VAX computers that they produce (Fig. 4).

Our development as AI researchers is facilitated by the evolutionary approach that we have undertaken in that there are a number of areas in which we gain invaluable experience. These areas include scenario generation, the definition of the man/machine interface, the construction of suitable knowledge structures and inference routines, the appropriateness of tools, the all-important step known as knowledge acquisition, and the improvement of our models' explanation capabilities.

Knowledge acquisition

Obtaining and codifying the knowledge of experts is the most critical step in the building of any expert system. The importance of this facet of expert-system development cannot be overemphasized. Good "knowledge engineering" can make the difference between a system that works the second or third time as opposed to the fifth or sixth. This difference, when translated into time, may mean delays of months or even years.

A number of constraints must be dealt with when creating a knowledge-based system. They fall into two major categories. The first group deals with the experts themselves. Initially, they must be located and convinced to participate. It will be impossible to extract anything but the most basic rules from experts if they aren't convinced that what they are helping to create is both useful and technically feasible.

The second group of difficulties concerns the rules themselves. It may be difficult for the experts to introspect and realize that they go through various steps in solving the problem at hand. The expert is being asked to turn what is essentially a "reflex" into a set of If-Then rules. The rules must then be numerically quantified in terms of belief or confidence (which is not a common way of thinking for most experts).

Experience provides us with a number of techniques or "rules of thumb." Use of these techniques will increase the accuracy of the system, while significantly decreasing the time necessary for knowledge acquisition:

1. Begin by developing a primitive/precursor system. It is a useful vehicle for eliciting further information from the experts.
2. Do not misrepresent the capabilities of the system because it will alienate the experts.
3. Develop a script for use in questioning the expert. We have found that asking the same question in different ways may elicit responses that would indicate that completely different queries had been posed.
4. When, in the judgment of the knowledge engineer, a phase of development has been completed, turn the system over to any other interested experts. This may result in information that was missed during earlier questioning.
5. For best results, a team approach to knowledge engineering appears to be successful. An engineer familiar with the domain under consideration may recognize when crucial steps are being omitted and can also offer the experts some assistance in verbalizing their thought processes, while a software expert will ensure that the most correct and efficient software structures are used.

Knowledge engineering is one of the most time-consuming parts of building any expert system. For the best results, it should alter-

```
THE R1 SYSTEM (CONFIGURING VAX SYSTEMS):

    IF:  (1)  THE CURRENT CONTEXT IS ASSIGNING DEVICES TO
              UNIBUS MODULES, AND
         (2)  THERE IS AN UNASSIGNED DUAL-PORT DISK DRIVE, AND
         (3)  THE TYPE OF CONTROLLER IT REQUIRES IS KNOWN, AND
         (4)  THERE ARE TWO SUCH CONTROLLERS, NEITHER OF WHICH
              HAS ANY DEVICES ASSIGNED TO IT, AND
         (5)  THE NUMBER OF DEVICES THAT THESE CONTROLLERS CAN
              SUPPORT IS KNOWN

  THEN:  (1)  ASSIGN THE DISK DRIVE TO EACH OF THE CONTROLLERS, AND
         (2)  NOTE THAT THE TWO CONTROLLERS HAVE BEEN ASSOCIATED
              AND THAT EACH SUPPORTS ONE DEVICE
```

Fig. 4. *An example of a rule from an actual expert system, which underwent serious "growing pains" before achieving its current success.*

RCA PROGRAM	SPECIFIC RELEVANCE
AEGIS SHIPBOARD COMBAT SYSTEM	• Operations and Interface Analysis • System Integration
EQUATE AUTOMATED TEST EQUIPMENT	• Specialist Interview Technique • Diagnostic Process Control
TECHNICAL COMBAT ANALYSIS CENTER (CAC) FOR ARMY FIELD INTELLIGENCE	• Field Environment Data Base Management • Rules Development for Intelligence Fusion • Specialist Interview Techniques
INTEGRATED RADIO ROOM (IR^2) FOR TRIDENT SUBMARINES	• C^2 Human Factors • Flexible Interface Maintenance
CLASSIFIED PROGRAMS	• Rules for Intelligence Fusion • Distributed Multisource Data Base Management

Fig. 5. *Choosing an area for expert system development effort involves careful evaluation of programs which currently require a large amount of human expertise.*

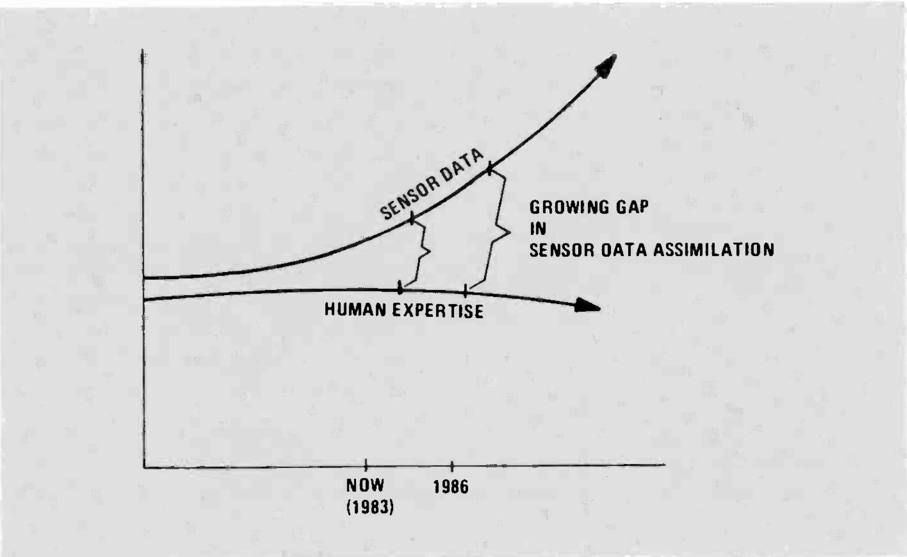


Fig. 6. *In fields such as Intelligence Fusion the amount of available information is greatly exceeding the supply of experienced personnel to analyze such information, making the application of expert systems not only valuable but necessary.*

nate with system testing so that test results can be presented to the experts in order to elicit further information. For a small, well-bounded system (250 rules, 10 possible conclusions), past experience indicates that, with an experienced knowledge engineer, it would take 8 to 12 months to develop and test the system with approximately 3 to 4 man months set aside for knowledge acquisition alone. If the knowledge engineer is inexperienced in building expert systems, another 2 to 3 man months may be necessary for the trial-and-error method of learning knowledge acquisition.

Explanation capabilities

One of the most interesting features of an expert system is its ability to explain its reasoning. When a hypothesis has been proposed to the users, they can ask the system to exhibit the chain of reasoning (as well as the pertinent rules that "fired") that the knowledge-based system used in determining both that hypothesis and the belief factor attributed to that hypothesis. Most, if not all, expert systems also possess a "why" feature. This feature comes into play when the system asks the users for some information that it does not have. Users can query the system (using "why") and obtain an explanation that details where the required information would be used and the reasoning that has preceded the query. Explanation is extremely important throughout the lifetime of a knowledge-based system. During the development phase, it is used to both verify and debug the system. We must be able to show that the reasoning employed by the expert system parallels the expert's exactly because that is why we're building the system in the first place. Later, when the system is actually being used, explanation once again comes into play. The users must have confidence in what the system "tells" them, and that is achieved by allowing them to "watch it think." Expert systems are generally used to raise the level of performance of a less experienced user to that approaching the expert's. Novices are not left out in the cold because the expert system can be used to tutor the novice by, once again, using its explanation capabilities.

Choosing an application

In choosing our initial application of knowledge-based technology, we examined many of RCA's major system development efforts and (Fig. 5) "played" them against the current limitations of the technology. At this time, the knowledge that is codified into the system's rule base can only come from one principal expert, unless the problem can be broken into disjoint subsets with one expert per subset. The problem domain must be well-bounded and relatively static in nature (not involving voluminous and/or rapidly changing data).

The problem domain that we opted for was Intelligence Fusion. The illustrative subproblem that we chose was Command-Post Identification. The application involves the interpretation of battlefield sensor data in order to identify enemy command posts with as much certainty as possible. Expert system technology lends itself to this problem because the tactical analyst currently responsible for this identification has to deal with incomplete and often inaccurate data. Many individuals foresee a growing gap in sensor data assimilation. The amount of data from sensors is growing exponentially due to improvements in sensor technology. This situation is coupled with a slow decline in human expertise, that is, the number of tactical experts available (Fig. 6).

Model-0

Our first "throwaway" model implemented an extremely simple problem with knowledge-based technology using a tool called Advice Language/X (AL/X). Since this was our first attempt at building expert systems, we made a number of assumptions that greatly simplified the problem we were attempting to solve. We devised an enemy hierarchy that was to be known beforehand along with a communications protocol between the nodes in the hierarchy. It was also assumed that any node was capable of only one "conversation" at any one time, with its superior, subordinate, or sibling (Fig. 7).

Advice Language/X

The (AL/X) is a 64-kb Pascal program used to facilitate the construction of single-user expert systems. It is especially useful for the development of diagnostic systems. Pieces of the expert's knowledge are represented in the form of "spaces." Spaces are analogous to nodes in a logical network. The interconnection of these spaces forms the knowledge base. The inference engine represents the thought processes of the expert as it traverses the knowledge base.

The main advantage of using AL/X is that it provides a ready-made explanation feature. This feature enables the user to ask the system why a question was asked. Most operators find that this makes the system easier to use than conventional problem-solving programs. Another aid provided by AL/X is a built-in method of handling uncertainty. If one is unsure of a piece of data, one can easily assign a degree of belief to it that reflects this uncertainty. These features combine to make the final system developed using AL/X very user-friendly.

Unfortunately, the same strict format required in order to exploit these features also restricts the system designer during development. A major disadvantage of AL/X is that it does not have variables. This complicates the interaction between the operator and the system. It also greatly lengthens development time. Network states cannot be saved or restored. This can be a major problem if a long consultation session is interrupted.

In summary, AL/X is a fairly good expert-system building tool. The features that it provides must be weighed against the developmental restrictions that must be tolerated in order to support these features. It has been our experience that when AL/X is used to construct expert systems that are not basically diagnostic in nature, its disadvantages greatly outweigh its advantages.

Model-1

We were interested in a somewhat more realistic system for our next model and were fortunate to have access to a number of ex-tactical analysts at our Automated Systems business unit in Burlington, Massachusetts. They were able to give us a number of realistic rules. The model was defined as follows:

Given

1. Some number of nodes and their locations,
2. A history of detected emissions (Fig. 8), as well as,
3. Several radio types and their characteristics.

- Determine the enemy battlefield configuration as accurately as possible;
- Establish confidence levels for results; and
- Be able to explain results.

Languages

PROLOG

Since we are also trying to evaluate advanced software as a parallel effort, we decided to implement this model in PROLOG (PROgramming in LOGic). PROLOG is a language whose overall design paradigm is that of being rule-oriented. It originated in France in 1973, and has only recently been used to any degree in the United States. In this, and

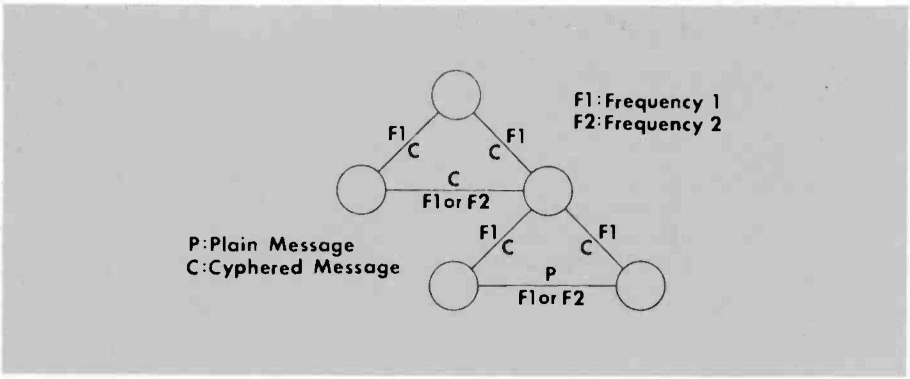


Fig. 7. *A simple prototype expert system analyzes enemy communications. Implementation – Advice Language/X (AL/X).*

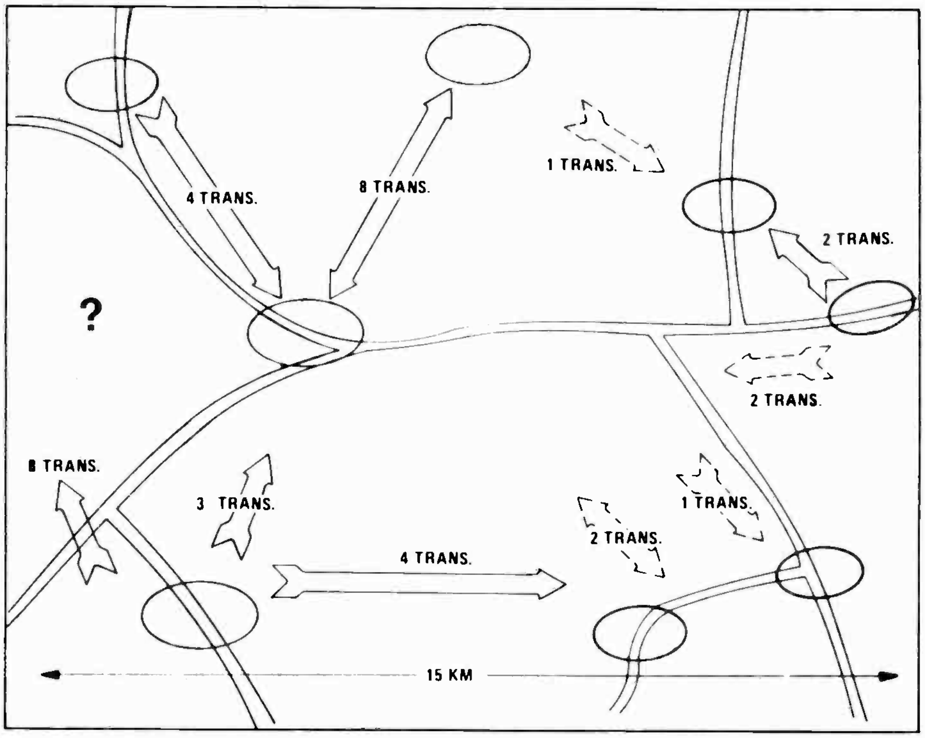


Fig. 8. *A slightly more complex expert system determines battlefield configuration using radio emissions. Implementation – PROLOG.*

indeed any, rule-oriented language, the behavior of the system is determined by sets of condition-action pairs, usually in the form of If-Then clauses (Fig. 9). A set of these pairs with a common function comprises a rule set. It may be convenient to visualize a rule set as a set of printed rules in a popular board game, such as Monopoly.

For example:

A. One rule set is designed to handle the game board, and the movement of pieces on it.

B. Another rule set can function as the Banker in the game, and ensure no cheating with monetary transactions.

```
(( CRITICAL-NODE *X )            : —
                ( PRIVILEGED *X )
                ( BETTER-DEFENDED *X )
                ( MESSAGES CYPHERED *X )
                ( OWNS SPECIAL-RADIOS *X ))
```

A critical node *X is one which is:

```
        Privileged
AND     Better defended
AND     Sends cyphered messages
AND     Owns special radios
```

To find a critical node Prolog attempts to satisfy each of these four subgoals in turn. If eventually this search through subgoals reaches facts which are defined to be true, then the top level goal succeeds, i.e., *X is a critical node.

```
(( PRIVILEGED *X )               : —
                ( SKIPPED-SUPERIOR *X ))
(( PRIVILEGED *X )
                ( SKIPPED-SUBORDINATE *X ))
```

A Privileged node *X is one which:

```
        Skipped a superior
OR      Skipped a subordinate
```

To find a privileged node Prolog attempts a depth first search on these subgoals until any one of them is satisfied.

```
(( SKIPPED-SUPERIOR *X )         : —
                ( CONVERSED *X *Z )
                ( SUPERIOR-OF *X *Y )
                ( SUPERIOR-OF *Y *Z ))
```

A node *X skipped a superior if:

```
        Node *X conversed with some node *Z
AND     there exists some node *Y such that
        *Y is a superior of *X
AND     *Z is a superior of *Y
```

Here, Prolog finds any *Z that conversed with *X. It then attempts to find a *Y for which the superior-of relationships hold. If such a *Y is found then (skipped-superior*X) succeeds which implies (privileged *X) succeeds. If no suitable *Y can be found Prolog backtracks to look for another *Z which conversed with *X and then begins another search for an appropriate *Y. If after looking at all *Z which conversed with *X no appropriate *Y could be found, Prolog would back up and attempt to satisfy (privileged *X) using the alternative (skipped subordinate *X) clause above.

```
(( SKIPPED-SUBORDINATE *X )  : —
                ( CONVERSED *X *Z )
                ( SUPERIOR-OF *Y *X )
                ( SUPERIOR-OF *Z *Y ))
```

A node *X skipped a subordinate if:

```
        Node *X conversed with some node *Z
AND     there exists some node *Y such that
        *X is a superior of *Y
AND     *Y is a superior of *Z
```

Here, as above, Prolog attempts to find a *Z that conversed with *X. It then attempts to find a *Y for which these superior-of relationships hold. Note here that variables with the same name in different clauses like *Y and *Z have no relationship to each other—all variables are local.

To make this a complete running program, we would need assertions to make up our data base such as

```
( conversed cluster1 cluster3 )
( conversed cluster3 cluster4 )
( conversed cluster 3 cluster 5 )

( subordinate-of cluster3 cluster2 )
( subordinate-of cluster1 cluster4 )
```

```
(( SUPERIOR-OF *X *Y )           : —
                ( SUBORDINATE-OF *Y *X ))
```

A node *Y is a superior of *X if the data base contains an assertion (fact) of the form (subordinate-of *y *X). This clause eliminates the necessity of including redundant information in the data base such as:

```
( superior-of cluster2 cluster3 )
( superior-of cluster4 cluster1 )
```

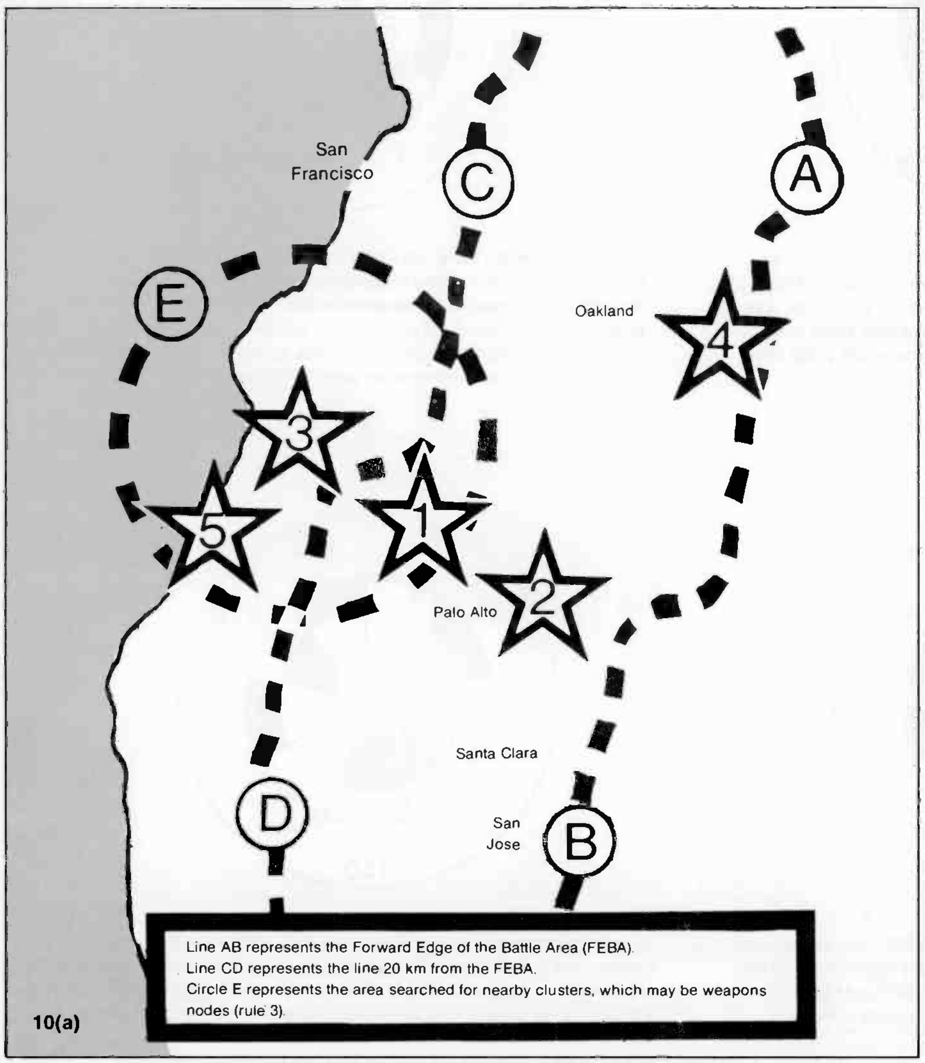
Fig. 9. *An example of the PROLOG language illustrates the use of If-Then rules to reach conclusions about enemy communications.*

C. One or more rule sets can act as opponents in the game.

Each rule set has a number of semi-independent rules to handle the game problems such as "landing in jail," buying and selling houses, and handling other incidental problems. As the user gets experience with each rule set, he can easily replace the old "jail rules" with new "jail rules" which, for instance, decide whether or not it is more advantageous to buy one's way out. Changing the jail rules without the need for modifying other rules makes for a very flexible and fluid system of rules. Rule-set players can even keep statistics as to the value of their rules, and perhaps even "swap" rules back and forth to improve their performance!

PROLOG has a built-in feature that performs backward-chaining from a high-level goal through one or more subgoals, until it reaches "facts," which are defined to be true in the database. PROLOG has no features to automatically trace the path it used to satisfy its goal; nor can it handle probabilistic rules. These were added on top of PROLOG. Fortunately, this was not extremely difficult. The most serious deficiency may be its unalterable search strategy. Successful AI efforts have found that careful control over the search was essential to achieving reasonable execution times.

Our completed system is capable of telling us (along with a degree of belief) which nodes are most likely to be the command post and, conversely, which of several node types (command post, support node, weapons node, and so on) a given node may be. New information concerning the battlefield situation can be added at any time, analyzed, and subsequently compared with past states using an implemented feature that we call the "snapshot" (Fig. 10a-d, pages 12-14).



Line AB represents the Forward Edge of the Battle Area (FEBA).
Line CD represents the line 20 km from the FEBA.
Circle E represents the area searched for nearby clusters, which may be weapons nodes (rule 3).

10(a)

Fig. 10. *Identification of battlefield command posts based on a given set of radio emission data includes explanations of reasoning leading to the conclusions.*

LISP

A successful working environment for Artificial Intelligence can derive great benefit from special-purpose hardware and software. We are purchasing two Xerox LISP (LIST Processing) Machines, special-purpose stand-alone workstations that possess a specially-designed architecture for executing LISP code.

LISP is a symbol-processing language that operates on lists of objects. Procedures in LISP are functions that return a value. LISP has a simple structure and syntax, but is nevertheless a very powerful and flexible programming language. Part of its power derives from the fact that it can treat any list as a series of instructions to be executed. This means that a LISP program can be used to create another program that can then be executed. Pieces of LISP code can even be passed as parameters to a LISP function. In addition, LISP is an extensible language. This means that new features can easily be added to the base-level LISP language. New applications and programming languages written in LISP inherit its power and flexibility. Such applications can often be written in a very short time. Since LISP is an interactive language, programs written in LISP are also inherently interactive.

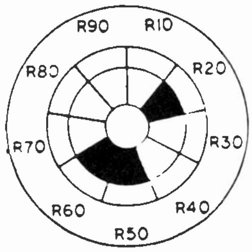
LISP was one of the first languages used to write Artificial Intelligence programs. Even today, most AI programs are still written in LISP. Many dialects have evolved over the past 25 years since LISP was first created by John McCarthy, including the following:

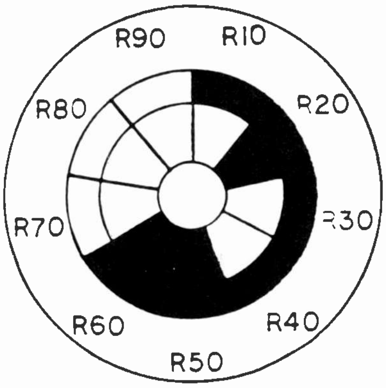
- MACLISP at MIT;
- INTERLISP at Stanford University and SRI International;
- INTERLISP-D at the Xerox Palo Alto Research Center (PARC);
- UCILISP at the University of California at Irvine;
- RUTLISP at Rutgers University; and
- FRANZ LISP at the University of California at Berkeley.

This lack of a standard LISP language has created a problem in transporting programs written in LISP between computers since many of these dialects are only available on one kind of machine. The Department of Defense is now sponsoring a project to produce a COMMON LISP with standardized specifications to try to eliminate the problem in the future. COMMON LISP will be much like MACLISP in appearance.

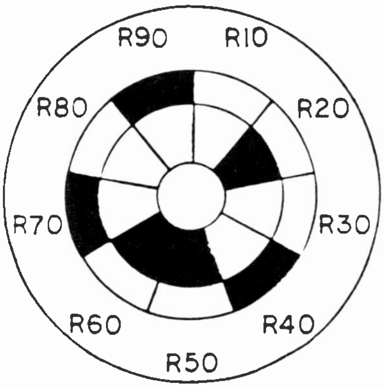
(Continued on page 15)

1) From the emission frequency and mod type we can determine that this emission came from an R20, R50, or R60 radio (see figure at right). We see below how well these possible radios compare with the radios the various node types are known to possess.

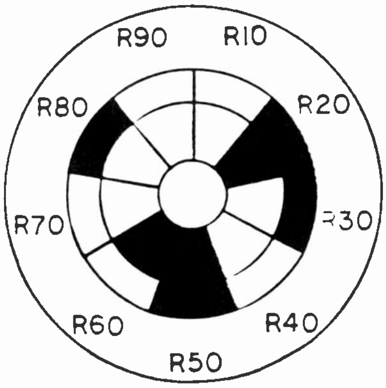




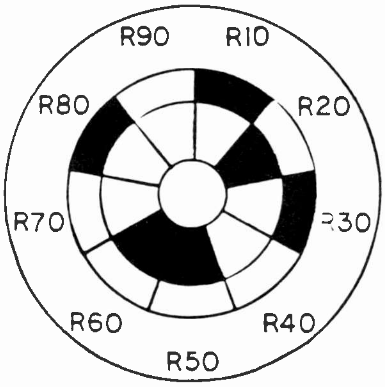
2) Command Posts have radio types: R10, R20, R30, R40, R50, R60 (Outer ring of diagram). Note the overlap of three radio types (R20, R50, R60) with the radios possibly responsible for the emission. This is strong evidence in favor of cluster 3 being the command post since command posts are known to possess all three of the radios which could have been responsible for the emission.



3) Weapons Nodes have radio types: R40, R70, R90. There is no overlap between the radios possibly responsible for the emission and the radios known to be possessed by weapons nodes. This is conclusive evidence against the hypothesis that cluster 3 is a weapons node—the set of radios which could have been responsible for the emission and the set of radios known to be possessed by weapons nodes are disjoint.

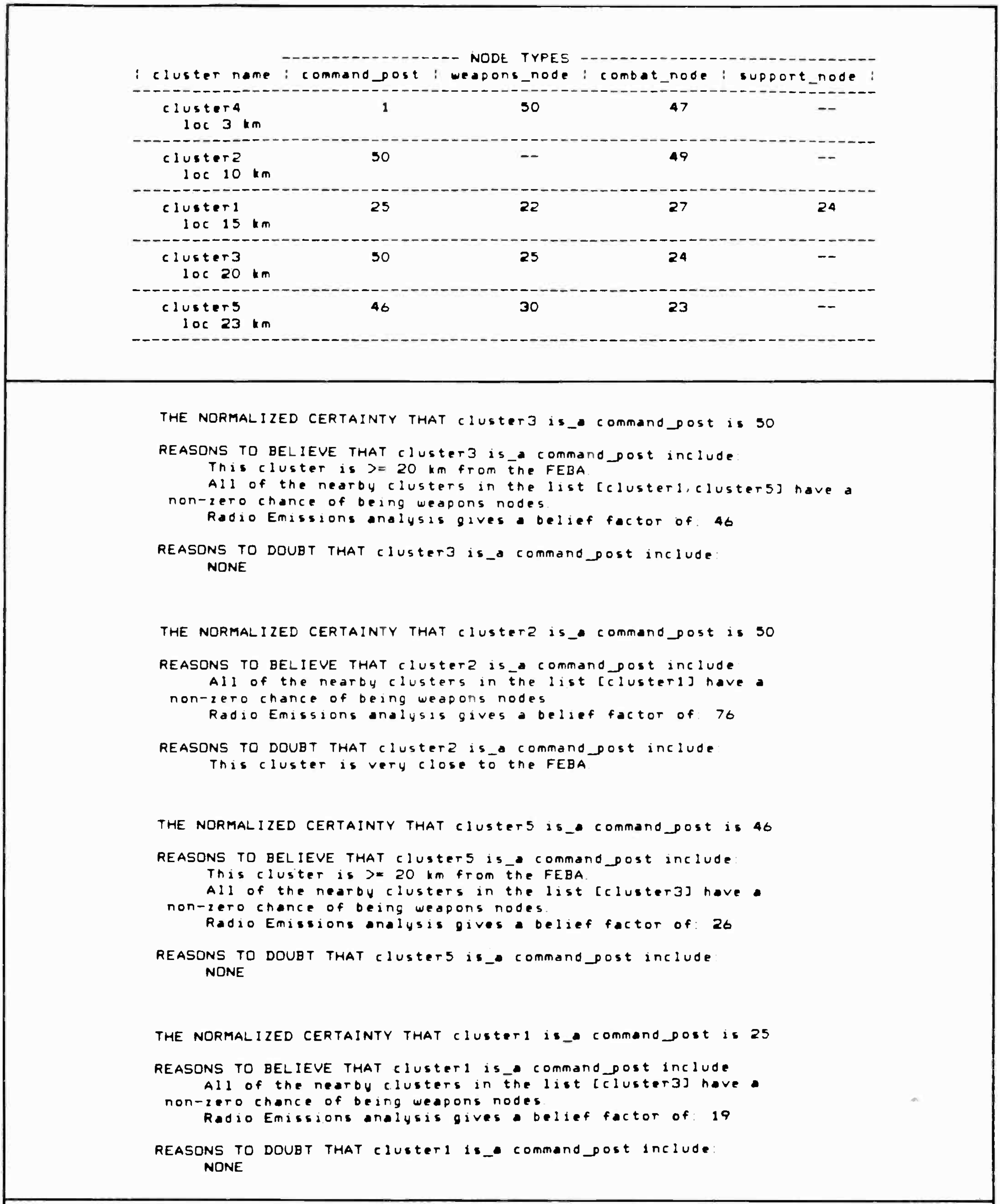


4) Combat Nodes have radio types: R20, R30, R60, R80. There is overlap of two radio types (R20, R50) with the radios possibly responsible for the emission in favor of cluster 3 being a combat node since combat nodes are known to possess two of the three radios which could have been responsible for the emission.



5) Support Nodes have radio types: R10, R30, R80. There is no overlap between the radios possibly responsible for the emission and the radios known to be possessed by support nodes. This is conclusive evidence against the hypothesis that cluster 3 is a support node—the set of radios which could have been responsible for the emission and the set of radios known to be possessed by support nodes are disjoint.

10(b)

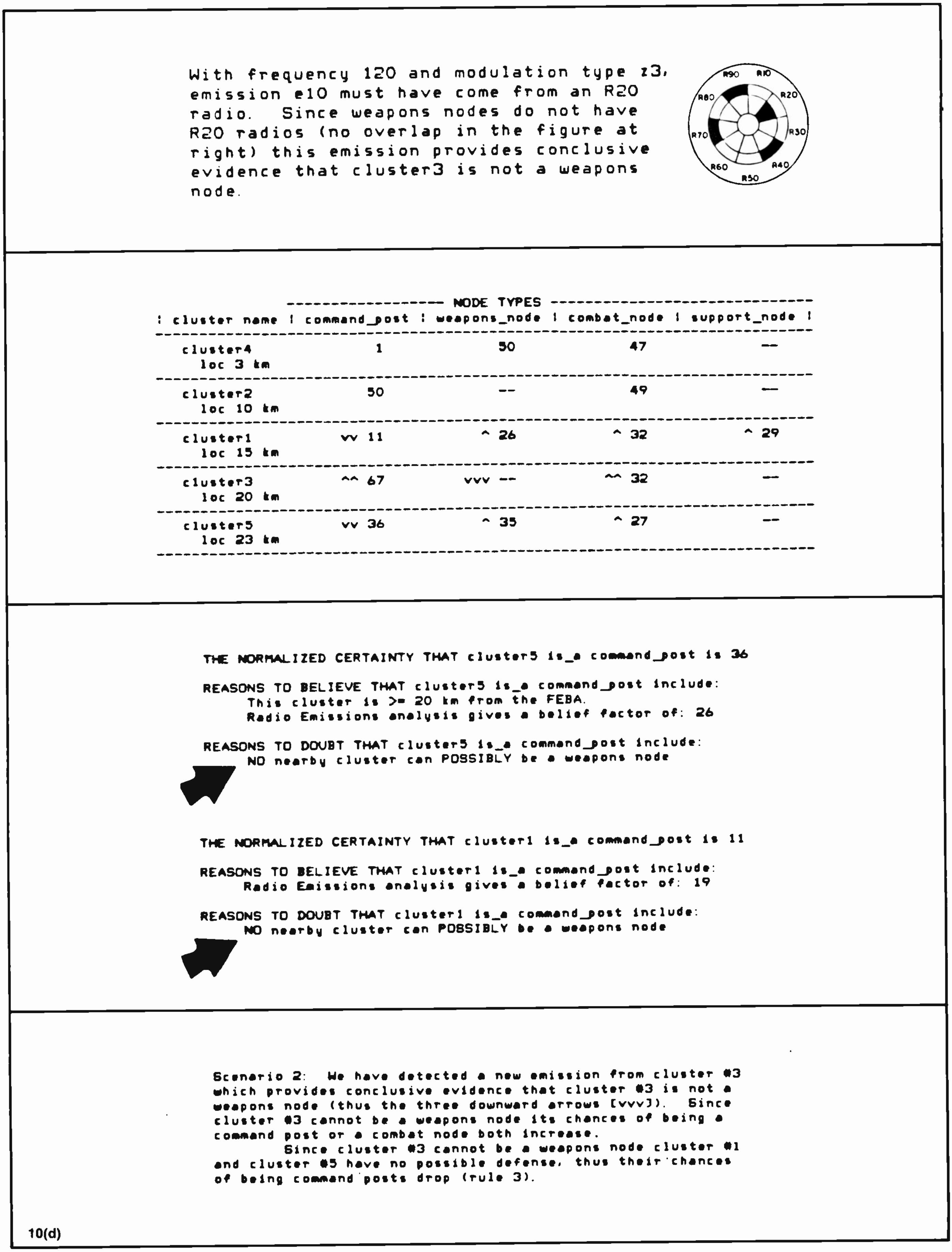


```
                  ------------------ NODE TYPES -----------------------------
 | cluster name | command_post | weapons_node | combat_node | support_node |
 --------------------------------------------------------------------------
    cluster4            1            50            47           --
      loc 3 km
 --------------------------------------------------------------------------
    cluster2           50            --            49           --
      loc 10 km
 --------------------------------------------------------------------------
    cluster1           25            22            27           24
      loc 15 km
 --------------------------------------------------------------------------
    cluster3           50            25            24           --
      loc 20 km
 --------------------------------------------------------------------------
    cluster5           46            30            23           --
      loc 23 km
 --------------------------------------------------------------------------
```

```
THE NORMALIZED CERTAINTY THAT cluster3 is_a command_post is 50

REASONS TO BELIEVE THAT cluster3 is_a command_post include:
     This cluster is >= 20 km from the FEBA.
     All of the nearby clusters in the list [cluster1,cluster5] have a
 non-zero chance of being weapons nodes.
     Radio Emissions analysis gives a belief factor of: 46

REASONS TO DOUBT THAT cluster3 is_a command_post include:
     NONE


THE NORMALIZED CERTAINTY THAT cluster2 is_a command_post is 50

REASONS TO BELIEVE THAT cluster2 is_a command_post include:
     All of the nearby clusters in the list [cluster1] have a
 non-zero chance of being weapons nodes.
     Radio Emissions analysis gives a belief factor of: 76

REASONS TO DOUBT THAT cluster2 is_a command_post include:
     This cluster is very close to the FEBA.


THE NORMALIZED CERTAINTY THAT cluster5 is_a command_post is 46

REASONS TO BELIEVE THAT cluster5 is_a command_post include:
     This cluster is >= 20 km from the FEBA.
     All of the nearby clusters in the list [cluster3] have a
 non-zero chance of being weapons nodes.
     Radio Emissions analysis gives a belief factor of: 26

REASONS TO DOUBT THAT cluster5 is_a command_post include:
     NONE


THE NORMALIZED CERTAINTY THAT cluster1 is_a command_post is 25

REASONS TO BELIEVE THAT cluster1 is_a command_post include:
     All of the nearby clusters in the list [cluster3] have a
 non-zero chance of being weapons nodes.
     Radio Emissions analysis gives a belief factor of: 19

REASONS TO DOUBT THAT cluster1 is_a command_post include:
     NONE
```

Scenario 1   Original data base is analyzed

10(c)



With frequency 120 and modulation type z3, emission e10 must have come from an R20 radio. Since weapons nodes do not have R20 radios (no overlap in the figure at right) this emission provides conclusive evidence that cluster3 is not a weapons node.

```
                  ------------------ NODE TYPES ------------------------------
! cluster name ! command_post ! weapons_node ! combat_node ! support_node !
---------------------------------------------------------------------------
   cluster4              1            50            47            --
     loc 3 km
---------------------------------------------------------------------------
   cluster2             50            --            49            --
     loc 10 km
---------------------------------------------------------------------------
   cluster1          vv 11          ^ 26          ^ 32          ^ 29
     loc 15 km
---------------------------------------------------------------------------
   cluster3          ^^ 67        vvv --         ^^ 32            --
     loc 20 km
---------------------------------------------------------------------------
   cluster5          vv 36          ^ 35          ^ 27            --
     loc 23 km
---------------------------------------------------------------------------
```

```
THE NORMALIZED CERTAINTY THAT cluster5 is_a command_post is 36

REASONS TO BELIEVE THAT cluster5 is_a command_post include:
     This cluster is >= 20 km from the FEBA.
     Radio Emissions analysis gives a belief factor of: 26

REASONS TO DOUBT THAT cluster5 is_a command_post include:
     NO nearby cluster can POSSIBLY be a weapons node


THE NORMALIZED CERTAINTY THAT cluster1 is_a command_post is 11

REASONS TO BELIEVE THAT cluster1 is_a command_post include:
     Radio Emissions analysis gives a belief factor of: 19

REASONS TO DOUBT THAT cluster1 is_a command_post include:
     NO nearby cluster can POSSIBLY be a weapons node
```

Scenario 2: We have detected a new emission from cluster #3 which provides conclusive evidence that cluster #3 is not a weapons node (thus the three downward arrows [vvv]). Since cluster #3 cannot be a weapons node its chances of being a command post or a combat node both increase.

Since cluster #3 cannot be a weapons node cluster #1 and cluster #5 have no possible defense, thus their chances of being command posts drop (rule 3).

10(d)

(Cont. from p. 11)

LISP machines

The Xerox LISP machines (Fig. 11) provide a personal INTERLISP-D environment that supports effective prototyping and the exploration of alternative designs. It is a unified environment for all programming phases. The workstations possess a high-resolution bit-map display. A complete set of raster-graphic functions to display text in multiple fonts, and manipulate raster images is provided as well as a "mouse" pointing device. The systems feature sophisticated display management with multiple overlapping windows (Fig. 12) and menu-driven selection. A structure-based editor that exploits the "form" of the object and reduces errors by emphasizing meaning is included. The environment provides a "programmer's assistant" that frees the programmer from mundane detail, as well as debugging and program analysis tools.

The standard software supplied with the Xerox LISP machines includes LOOPS, a revolutionary new expert system-building environment developed at Xerox PARC. LOOPS "sits atop" INTERLISP-D and executes exclusively on the Xerox LISP machines.

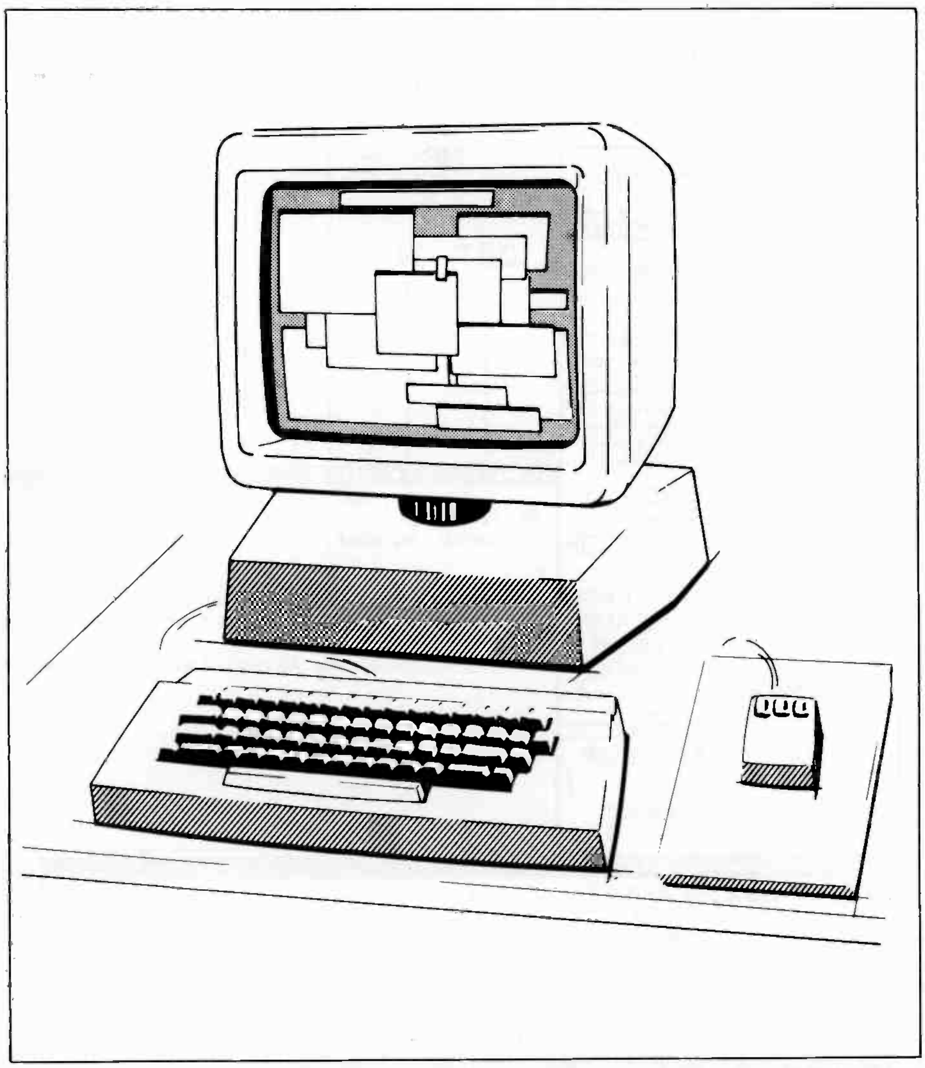


Fig. 11. *The development of expert systems (usually programmed in LISP) is aided by the introduction of processors designed specifically to run LISP. A drawing of the Xerox Dolphin is shown.*

LOOPS

The LOOPS developers identify the four programming paradigms useful for developing expert systems as: Procedural, Access, Rules, and Object-Oriented. Unlike other languages, LOOPS supports all four (Fig. 13). The paradigms can be mixed and matched to fit the given application. In the example given below (which expands on our earlier one), a Monopoly player may be a set of Rules. The Object-Oriented paradigm allows an object to inherit properties from other objects. The Access-Oriented approach is useful for monitoring independent processes. In our Monopoly example, it would be used to automatically display the current status of the game, the game board, and detect illegal accesses to the game's Bank.

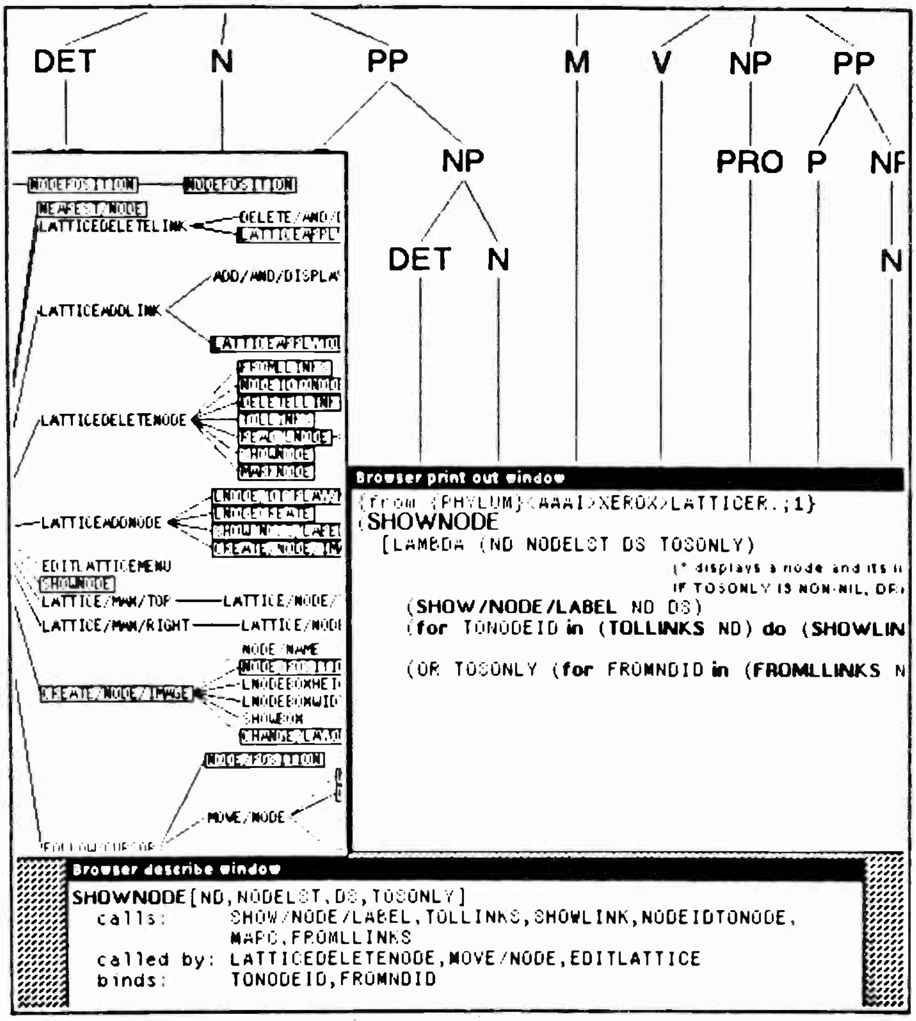
Historically, LOOPS is the latest of a long series of Xerox experiments in alternative representation structures. The main application has been the development of an expert "assistant" for Very Large Scale Integration (VLSI) integrated circuit design. These experiments helped identify a common "core" of paradigms with lasting value. Unlike the archtypical MYCIN system for medical diagnosis, or the PROSPECTOR system for mineral exploration, the application chosen is not a merely diagnostic task, but a theoretically and technically more ambitious design and planning task. Thus, Xerox's need for a better tool with which to construct expert systems may have prompted the design of a technically more sophisticated environment.

"Programs are easier to build in a language when there is an available paradigm that matches the structure of the program," claim the two major LOOPS creators, Dr. Daniel Bobrow and Dr. Mark Stefik of Xerox PARC. LOOPS offers a set of language features for each of four different programming paradigms. Existing expert system building tools were generally designed for, at most, only one or two of these paradigms. LOOPS creates an integrated hybrid of all four. Users may choose the style of programming that best suits their application, and use different paradigms for different parts of their application, as will be illustrated below with an example of a Monopoly game. Consider:

A. *Possible procedural-language approach:*

> Write procedures in a suitable language (FORTRAN, Ada, and so on). Store each square of the board in records and/or matrices (a tedious task). Store many other facts about the pieces in other records. Write Pascal or Ada code to move these pieces around. Embed within these sections of code many scattered "print" and "error" messages to keep track of the game. A more ambitious approach: Create a computer player, with a strategy, useful in solo mode for tuning its strategy.

The primary defect of these languages is that a large and difficult-to-maintain pro-



```
{from {PHYLUM}<AAAI>XEROX>LATTICER.;1}
(SHOWNODE
  [LAMBDA (ND NODELST DS TOSONLY)
                                    (* displays a node and its li
                                    IF TOSONLY IS NON-NIL, OR
    (SHOW/NODE/LABEL ND DS)
    (for TONODEID in (TOLLINKS ND) do (SHOWLIN

    (OR TOSONLY (for FROMNDID in (FROMLLINKS N
```

```
SHOWNODE[ND,NODELST,DS,TOSONLY]
  calls:      SHOW/NODE/LABEL,TOLLINKS,SHOWLINK,NODEIDTONODE,
              MAPC,FROMLLINKS
  called by:  LATTICEDELETENODE,MOVE/NODE,EDITLATTICE
  binds:      TONODEID,FROMNDID
```

Fig. 12. *The Xerox Dolphin uses multiple overlapping windows to provide the programmer with the ability to arrange his desired "desktop."*

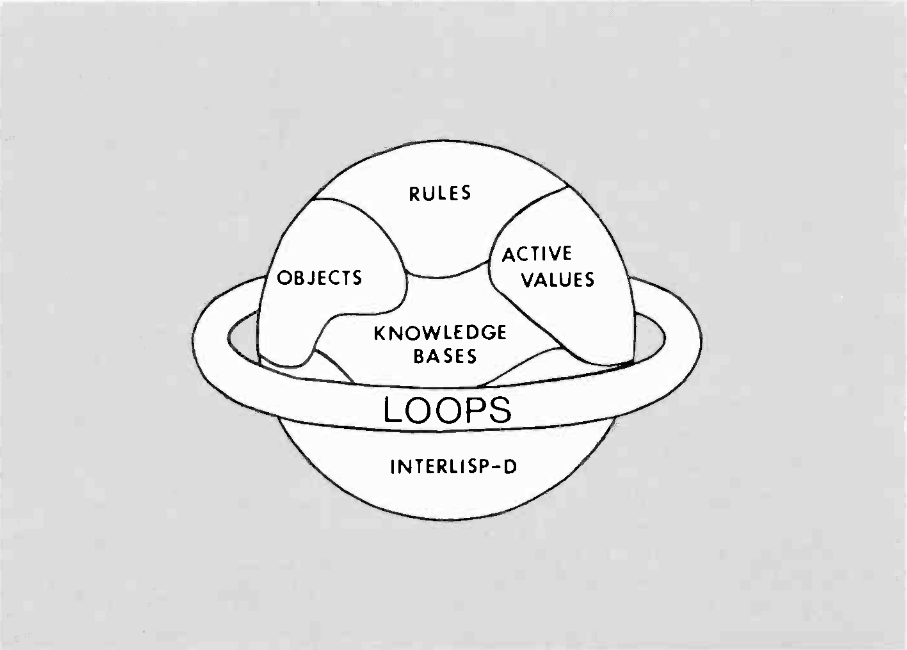


Fig. 13. *The Xerox LOOPS system is an expert-system building environment which supports and provides for the four major paradigms of expert-system design.*

gram would have to be written and changes would cause unexpected side-effects. Input/output commands are often scattered throughout the procedures where the various quantities are calculated, making changes difficult, if not impossible, to make.

B. *One possible LOOPS approach:*

1. Object-oriented paradigm: Create a rule set called the "Banker." Let this Banker, and all players (human or otherwise) communicate with it and each other using a common protocol. For example, a player could issue these instructions on his turn (assuming the following syntax:

<prompt *No.*>":"<*Subject* addressed> ":"<*Verb* command> <Opt. *Nouns*>"."):

1: Banker: *Mortgage* Park Place.
You roll a 7. You move to Boardwalk.
2: Banker: *Buy* Boardwalk.
3: END.

The game's Banker would then respond using this same protocol, along with a full printed explanation, if desired.

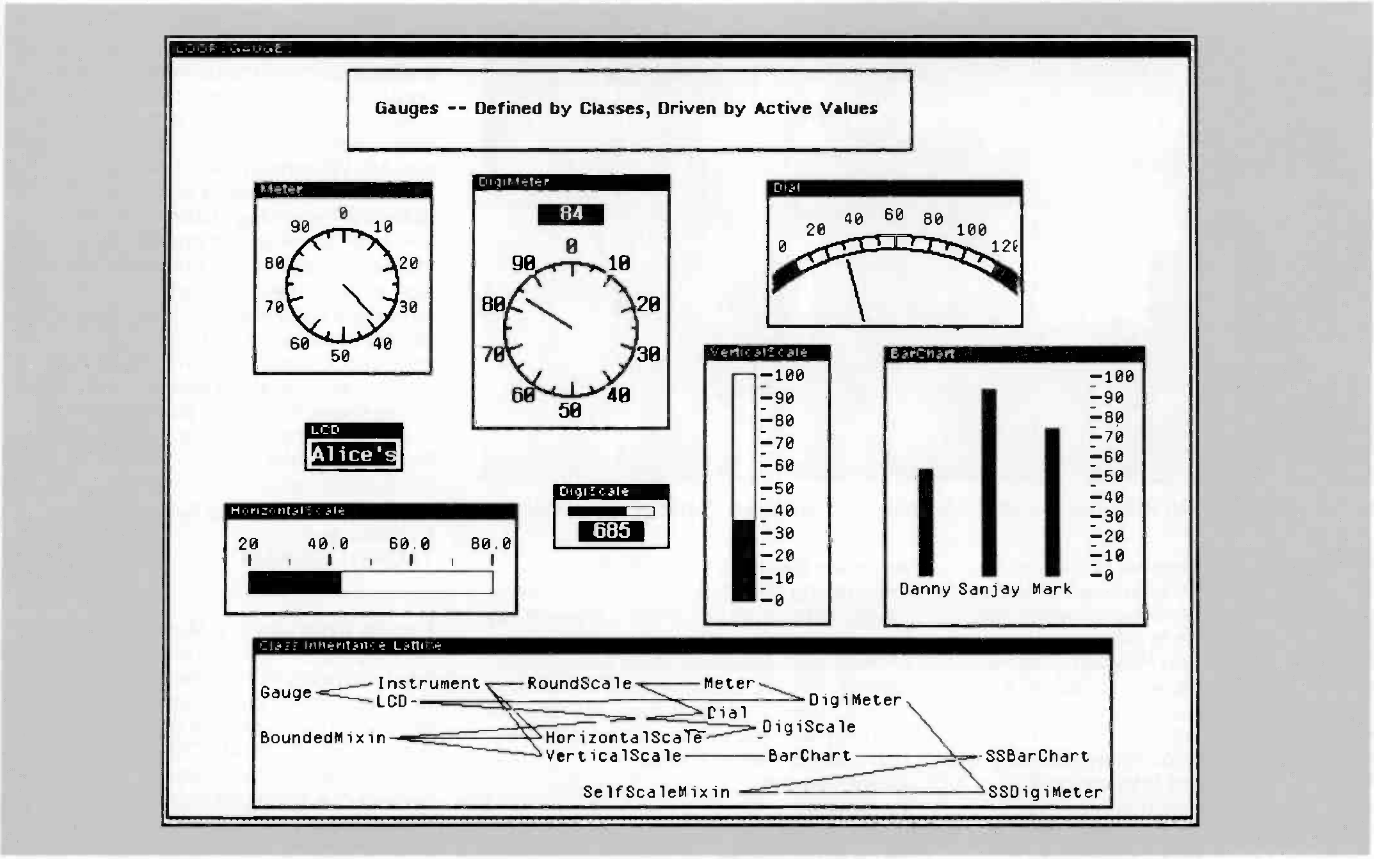
2. Rule-oriented paradigm: The computer players are defined by structures called Rule Sets. These succinct constructions are designed to act like any human player: tokens are moved on a board; properties are bought and sold; and so on. Each rule uses this form:

<pattern>. .<action>
(IF . . . THEN . . .)

These rules contain the "knowledge" of playing the game in separate small chunks, one or more for each situation (such as how to get out of the "Jail" square, or what passing "Go" will do). The built-in keyword-directed control structure chooses among the IF clauses in these rules, using information about the board, and thereby finds the most appropriate THEN action to perform on its turn.

3. The Procedure-oriented paradigm is satisfied by using INTERLISP-D. In the example above, calculation of various play odds are most efficiently done by using conventional procedural number-crunching.

4. The Access-oriented paradigm is most useful for the task of interfacing between very independent processes—defining a common protocol for these would not be very useful. Instead, a procedure call is triggered whenever a certain item of data is accessed.



Gauges -- Defined by Classes, Driven by Active Values

Fig. 14. *An example of the particularly effective man-machine interface which LOOPS employs.*

In the above example, the graphics display package (of various items in the game, such as the game board) is independent of the logic of play, yet it must be able to unobtrusively monitor the rest of the code. Automatic procedure calls to the graphics package also prevent scattering machine-specific input/output functions throughout all the other program code. Programs become more functionally decomposable, reducing complexity.

Unlike some other rule-oriented languages (PROLOG, and so on), LOOPS also contains functions for defining inheritance relationships. For example, suppose a rule set called "RandomPlayer" was defined with about 40 rules. Its only purpose is to make legal, but random moves. An improved player can be defined as a specialization, call it "CautiousPlayer," of only 15 rules unique to itself. CautiousPlayer performs elementary risk assessment, replacing all the "random" action rules of RandomPlayer. The rest of RandomPlayer's rules are implied from the inheritance link. This process may continue indefinitely: CautiousPlayer could, in turn, be further specialized into a rule set called "BeachBuyer" consisting of only one rule. This rule need only give the highest priority to the development of the "Boardwalk" and "Park Place" properties (this is a passable and common game strategy).

LOOPS' design incorporates and efficiently intermixes some of the best of several current programming paradigms as explained above. The resulting mix offers more capability than any other environment known. The built-in graphics are very attractive and seem unusually flexible. Gauge types (meters, dials, and so on) are easily defined and can be "tied" to values in the system via the "access-oriented" programming paradigms (Fig. 14).

Future directions

Future application areas for our Artificial Intelligence effort will include Automatic Test Equipment (ATE), Satellite Management as well as Command and Control. In ATE, our group has two points of focus. We are currently under contract to the Naval Air Engineering Center to study the application of Artificial Intelligence to Automatic Test Equipment. Work on this contract will include the development of criteria that will be used to evaluate expert systems as well as a survey of available expert-system-building tools. In addition, we will identify those areas of ATE in which AI will offer the greatest improvement over existing techniques. The completion of this contract in June 1984 will result in a definitive study of this merger, which both our customer and RCA will find useful.

Secondly, we are collaborating on an Independent Research and Development project, with RCA Automated Systems. It will include an analysis of existing ATE equipment, as well as an investigation into ways to improve future generations of ATE equipment using AI. This work has resulted in the development of a small-scale expert system, slated to be upgraded in the future, that has demonstrated the feasibility of expert systems in ATE. The objective of this work is to augment existing ATE software and thereby increase the amount of fault diagnosis that is possible.

In Satellite Management, we will be working with Astro-Electronics to develop an expert system to help diagnose and resolve problems with spacecraft once they have been launched. This project will include an Intelligent Man-Machine Interface that will use color graphics to exhibit satellite status and aid in rapid fault diagnosis and correction.

Authors (left to right) Kleinosky, Spool, McAdams, Whitehead, Gaev, and Zapriala.

John Zapriala is a Member of Engineering Staff in ATL's Software Technology Laboratory. He is presently developing an operational expert system to perform hostile-air-target identification for short-range air defense. Mr. Zapriala, who joined RCA in 1980 and had previously worked primarily in the areas of pattern recognition and image processing, is currently pursuing an MSE in Computer and Information Sciences at the University of Pennsylvania.
Contact him at:
Advanced Technology Laboratories
Camden, N.J.
TACNET: 222-6244

Peter Spool holds a Masters degree in Computer Science from Rutgers University, where he is currently pursuing a PhD in Artificial Intelligence. He is a regular consultant at ATL's Software Technology Laboratory in Camden, New Jersey. He has done research in machine learning, computer-assisted instruction, and software engineering.
Contact him at:
Advanced Technology Laboratories
Camden, N.J.
TACNET: 222-6218

Jonathan Gaev is a Member of Engineering Staff in ATL's Software Technology Laboratory in Camden, New Jersey. He is the principal investigator for a recently commissioned study of the applications of Artificial Intelligence to the field of Automated Test Equipment. Mr. Gaev is currently working on his doctorate at the University of Pennsylvania.
Contact him at:
Advanced Technology Laboratories
Camden, N.J.
TACNET: 222-6246

John Mc Adams is also a Member of Engineering Staff, in ATL's Software Technology Laboratory in Camden, New Jersey. He is presently constructing an expert system for Command Post Identification. Mr. Mc Adams holds a BA degree in Mathematics from the University of Pennsylvania where he is currently pursuing an MSE in Artificial Intelligence in the Computer and Information Science department.
Contact him at:
Advanced Technology Laboratories
Camden, N.J.
TACNET: 222-6246

Jim Whitehead holds a Masters degree in Computer Science (specializing in Artificial Intelligence) from the University of Wisconsin, where his research focused on the computer understanding of human metaphors. Following a period at Bell Laboratories, Mr. Whitehead joined RCA as a Member of Engineering Staff in ATL's Software Technology Laboratory. He is presently developing a prototype expert intelligence system—a computerized decision aid which will attempt to augment and reproduce the expertise of an experienced tactical intelligence analyst in performing intelligence fusion. Mr. Whitehead also designs and develops games for the interactive K-line VideoDisc player, currently under development at the David Sarnoff Research Center.
Contact him at:
Advanced Technology Laboratories
Camden, N.J.
TACNET: 222-6244

Pamela Kleinosky is a Member of Engineering Staff in ATL's Software Technology Laboratory in Camden, N.J. She joined RCA after developing several expert systems for on-line process diagnosis at the Westinghouse Electric Corporation. Her current responsibilities include the development of a prototype expert system which is designed to emulate the electronic troubleshooting expertise of RCA field engineers as well as research in the area of generic applications of Artificial Intelligence to Automatic Test Equipment.
Contact her at:
Advanced Technology Laboratories
Camden, N.J.
TACNET: 222-6246

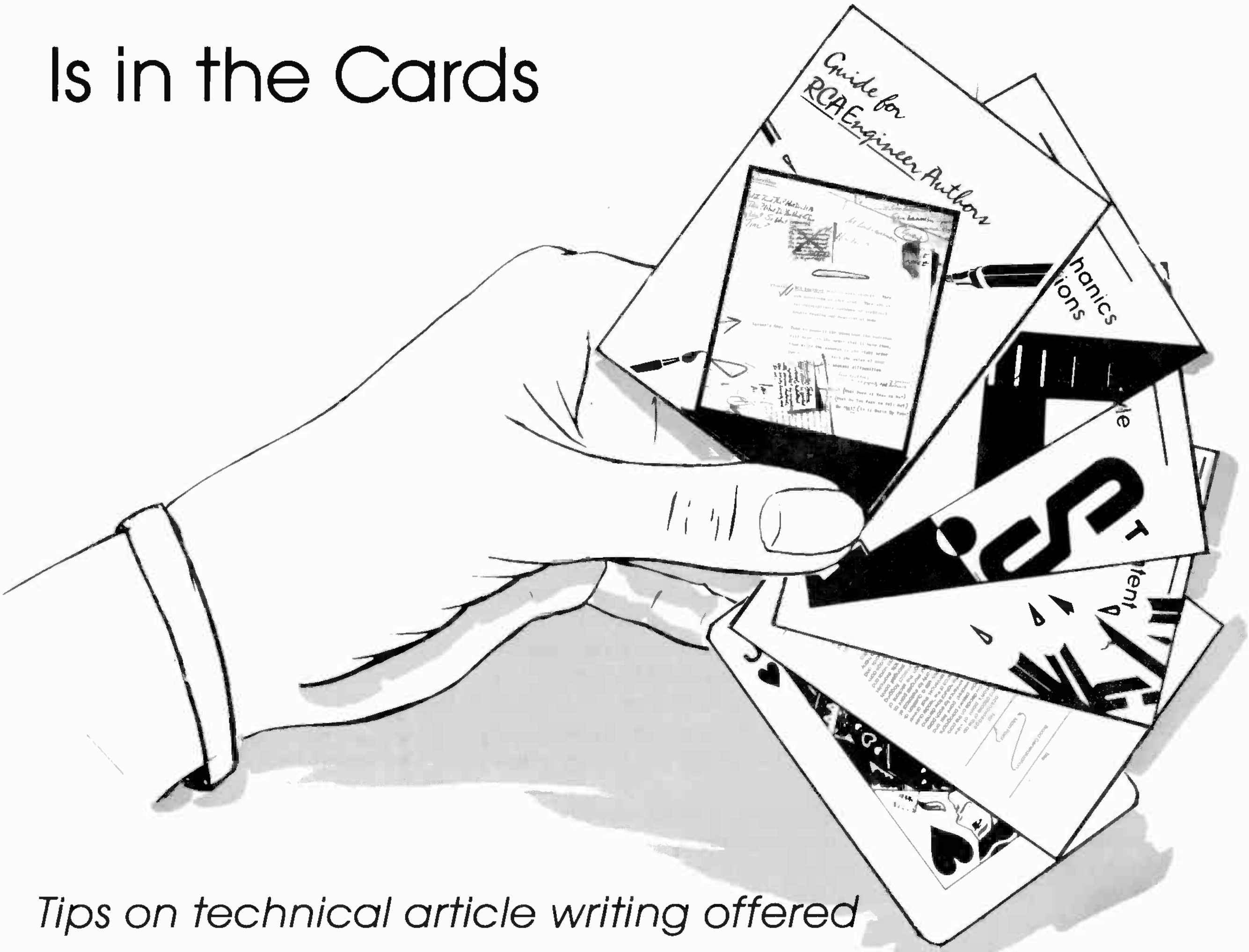
In the domain of Command and Control, we hope to be collaborating with Missile and Surface Radar on the construction of an expert system to be used for the Identification of Friend or Foe (IFF) in Tactical Army Air Defense. All of these efforts are sure to provide us with both interesting and challenging work and give us a chance to put our Xerox LISP Machines through their paces.

References

1. P.McCorduck, *Machines Who Think,* W.H. Freeman Co., San Francisco, Calif. (1979).
2. E. Feigenbaum and P. McCorduck, *The Fifth Generation: Artificial Intelligence and Japan's Computer Challenge to the World,* Addison Wesley Publishing Co., Reading, Mass. (1983).
3. A. Barr, *et al., The Handbook of Artificial Intelligence,* William Kaufmann, Los Altos, Calif. (1981).
4. F. Hayes-Roth, *et al., Building Expert Systems,* Addison Wesley Publishing Co., Reading, Mass. (1983).
5. H. Simon, *The Sciences of the Artificial,* MIT Press, Cambridge, Mass. (1981).
6. E.H. Shortliffe, *Computer-Based Medical Consultations: MYCIN,* American Elsevier Publishing Co., New York, N.Y. (1976).
7. A. Paterson, *AL/X User Manual,* Intelligent Terminal Ltd., Oxford (1981).
8. W.F. Clocksin and C.S. Mellish, *Programming in Prolog,* Springer-Verlag, New York, N.Y. (1981).
9. *INTERLISP-D Users Guide,* Xerox Electro-Optical Systems, Pasadena, Calif. (1982).
10. D.G. Bobrow and M. Stefik, *The LOOPS Manual,* Xerox Palo Alto Research Center, Palo Alto, Calif. (1983).

Your *RCA Engineer* Article . . .

Is in the Cards



Tips on technical article writing offered

If you are writing an article for the *RCA Engineer's* multidisciplinary engineering audience, the 48-page "Guide for *RCA Engineer* Authors" can show you practical ways to attract and maintain reader interest. The material specifically applies to RCA users. Experienced and novice authors can use many of the universally applicable principles, methods, and examples given here. Each chapter presents a series of Premises, Goals, and Methods that lead the author through the writing effort. Essays on side topics embellish the main text.

The first chapter, on "Article Mechanics & Specifications," defines the parts of a complete article package. The second chapter, on "Article Content," contains idea starters that will help you to gather the right information and artwork for a multidisciplinary audience, organize the material, establish and keep audience interest, and write for the reader. The third chapter, on "Writing Style," illustrates by example the five major ways to improve written expressions, by making them active, lean, clearly qualified, symmetric, and specific.

To order, send your request for a free copy of the "Guide for *RCA Engineer* Authors" to the RCA Technical Excellence Center, 13 Roszel Road, P.O. Box 432, Princeton, NJ 08540. Call TACNET: 226-3090.

E.W. Dusio | T.P. Murphy | W.F. Cashman

Communications satellite software tutorial

During the past decade, software has permeated every stage in the development and operation of communications satellites.

Abstract: *This tutorial on communications satellite software covers applications in analysis and design, manufacturing and testing of system components, and controlling and monitoring of system operation. To this end, the authors cover computer-aided engineering, system design, and launch and operation. Spacecraft control functions on board a satellite include on-orbit stationkeeping and attitude control. Ground software, supporting communications satellite operation, includes orbit determination and stationkeeping; and telemetry, processing, and control.*

This tutorial surveys the entire range of software applications for communications satellites. Software applications include analysis and design, manufacturing and testing of system components, and controlling and monitoring of system operation.

A communications satellite system consists of a satellite, a ground-based station to control its operation, and a network of earth stations providing transmission (uplink) and reception (downlink) facilities for communications traffic relayed through the satellite. The satellite itself has two major parts: a payload and a bus. The payload consists of the antennas, receivers, and transmitters that perform the communications signal transponding function; the bus provides the support (housekeeping) functions needed for the payload operation. The bus includes attitude control, thermal control, power management, and command and telemetry subsystems.

Before November 1982, all communications satellites were launched from expendable boosters of limited launch-weight capabilities. Figure 1 presents a typical launch sequence for an expendable booster. The Space Transportation System, now operational, provides much greater launch-weight capabilities and greatly relaxes the weight constraint.

Once in space, a communications satellite common carrier occupies an assigned orbital position, or slot, above the equator at a geostationary altitude. These slots are boxes with 0.1- to 0.2-degree sides at three- or four-degree spacings. Because the earth is not a perfect sphere, and because of solar/lunar gravity, a satellite in geostationary orbit wanders. The maneuvers that hold a satellite in its slot are commonly referred to as *stationkeeping* and are effected by firing on-board thrusters. When hydrazine (N_2H_4) is the propellant, as it is in many satellites, stationkeeping requires 20 to 30 pounds of propellant per year. Propellant use varies slightly with the position of the assigned slot relative to several "equilibrium positions" in the earth's gravity field.

The normal source of a satellite's electrical power is a solar cell array. A communications satellite, however, is eclipsed by the earth once per day around the time of the vernal and autumnal equinoxes. During these eclipse periods, which can be as long as 70 minutes, batteries power the satellite.

The attitude control system of a communications satellite has the task of providing a stable orientation with respect to the earth and sun. There are two basic

This paper was originally published in the April, 1983 issue of IEEE *Computer,* pages 21-34.

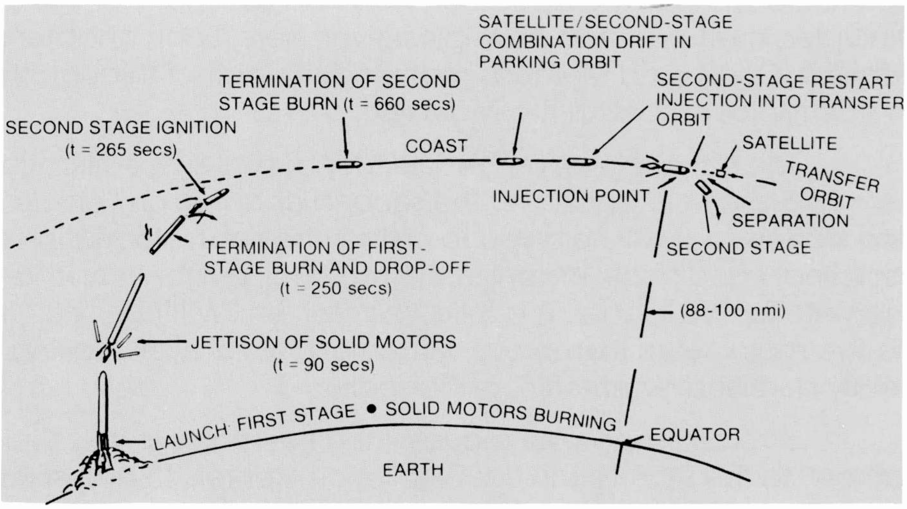

SATELLITE/SECOND-STAGE COMBINATION DRIFT IN PARKING ORBIT
TERMINATION OF SECOND STAGE BURN (t = 660 secs)
SECOND-STAGE RESTART INJECTION INTO TRANSFER ORBIT
SECOND STAGE IGNITION (t = 265 secs)
COAST
SATELLITE
TRANSFER ORBIT
INJECTION POINT
SEPARATION
SECOND STAGE
TERMINATION OF FIRST-STAGE BURN AND DROP-OFF (t = 250 secs)
(88-100 nmi)
JETTISON OF SOLID MOTORS (t = 90 secs)
LAUNCH
FIRST STAGE • SOLID MOTORS BURNING
EQUATOR
EARTH

Fig. 1. *Typical launch and injection sequence for a booster.*

Final manuscript received October 20, 1983.
Reprint RE-28-6-2

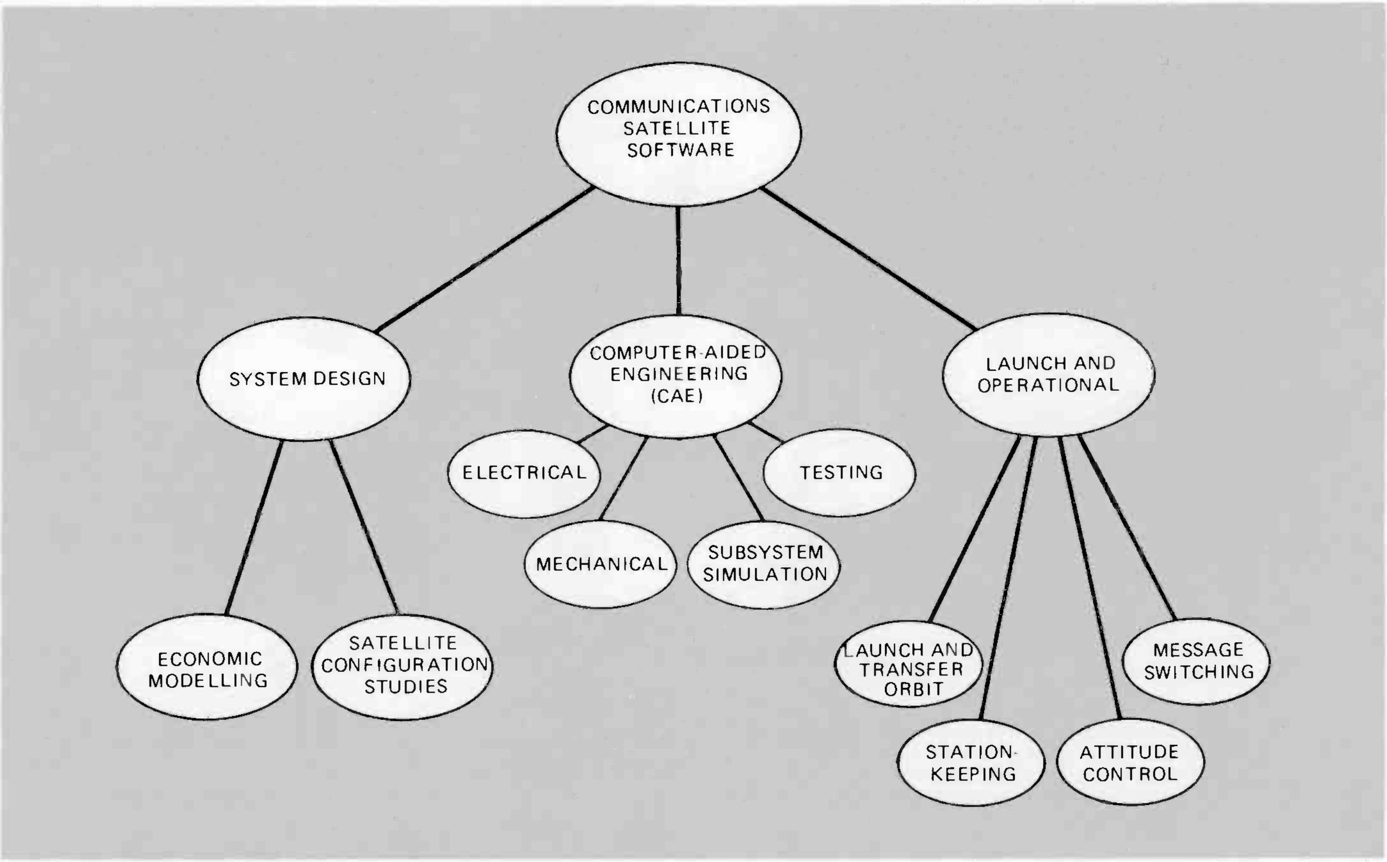


Fig. 2. *Time-phased representation of the software associated with a communications satellite life cycle.*

classes of communications satellites: spinners and three-axis stablized. A spinner uses the angular momentum of its spinning body to provide attitude stabilization in roll and yaw. The body of a three-axis stabilized satellite remains fixed relative to the earth's surface; an internal momentum wheel provides roll/yaw stabilization.

Software classification

The tree structure in Fig. 2 is a time-phased representation of the various classes of software associated with a communications satellite life cycle.

System design

System design software is of two types: economic modeling software and satellite configuration studies software. The application of economic modeling software to satellite systems is a normal, prudent activity for any business entering the field. Communications satellites are commercial ventures, driven by the forces of the marketplace. Economic modeling software helps assess the investment in a communications satellite system in relation to the carrier's corporate goals. Trade-offs involving geographic coverage, types of service provided, and tariff structure are part of each carrier's market strategy.

Satellite manufacturers perform satellite configuration studies as they seek to meet the performance requirements imposed by common carriers without exceeding size, weight, power, and thermal constraints. Software used to study satellite configurations extends the software nomographs and rules of thumb manufacturers have used for many years in designing satellites for the government.

Computer-aided engineering

Computer-aided engineering includes software used to design, produce, and test a satellite. CAE software for communications satellite design is usually general-purpose software developed by vendors for their regular markets or by spacecraft or subsystem manufacturers. Development of integrated circuits, printed circuit boards, and subsystems for communications satellites is similar to the development of these items for ground-based applications. Mechanical CAE software running on a large computer allows the rapid iteration of design and analysis required to minimize the weight of a satellite's structure while maximizing the likelihood of survival during the stresses of launch and the space environment. Subsystem simulation and design software addresses detailed subsystem characteristics and allows subsystem designers to evaluate the many possible trade-offs. In a power subsystem, for example, the designer has options in the following areas:

- Solar array
- Batteries
- Power supply electronics

The output of these trade-offs is a set of design specifications covering every component in the subsystem. Also, a rich set of software for attitude control subsystem design has been developed since the early 1960s.

BASIC and ATLAS are two languages commonly used to write subsystem test procedures. Most test activities develop libraries of subroutines or macros for frequently used test instrument setups to minimize the potential for procedure errors and reduce test procedure development costs. Computer graphics are especially helpful, because they reduce tremendous amounts of test data to easily understood

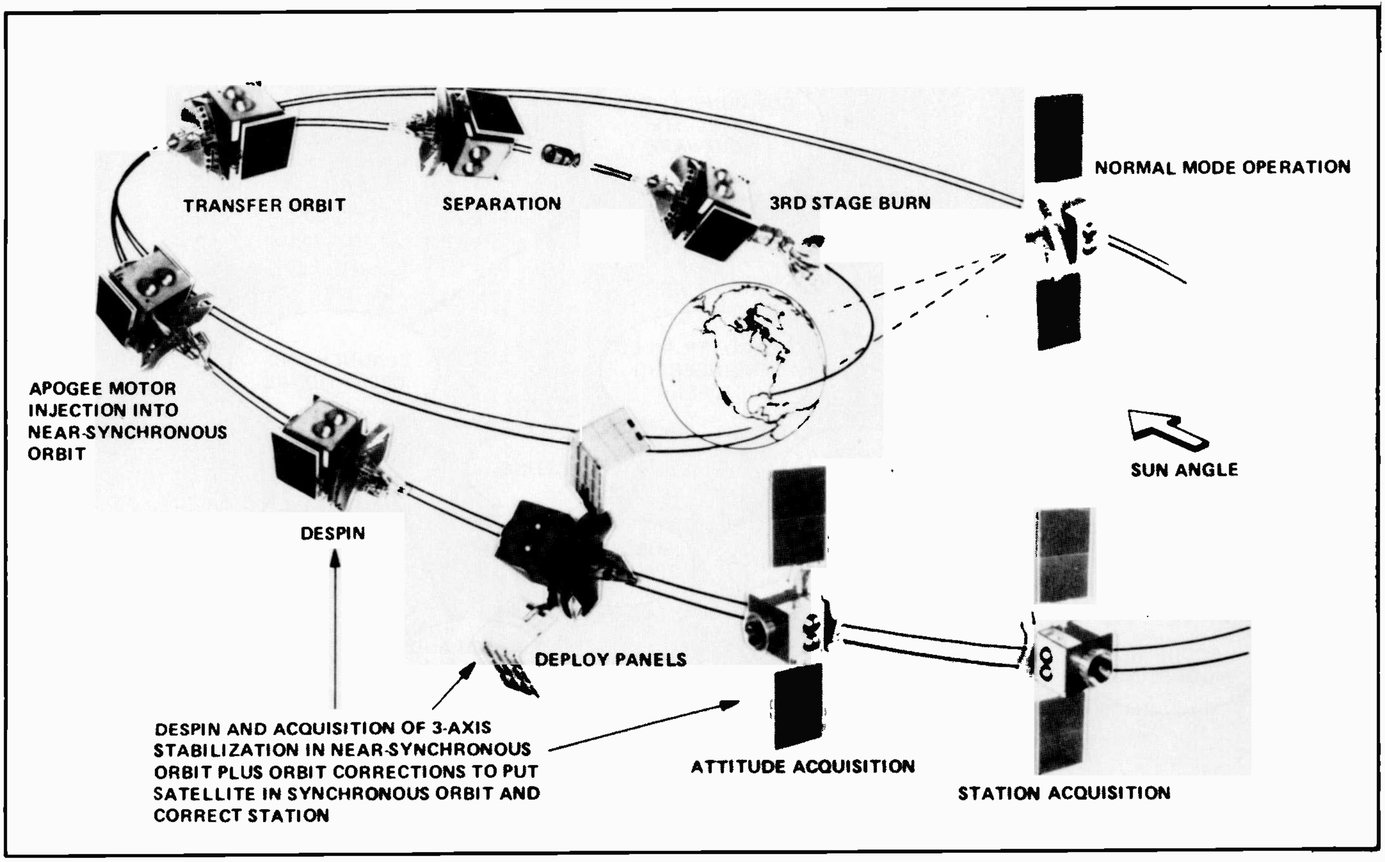

TRANSFER ORBIT
SEPARATION
3RD STAGE BURN
NORMAL MODE OPERATION
APOGEE MOTOR INJECTION INTO NEAR-SYNCHRONOUS ORBIT
SUN ANGLE
DESPIN
DEPLOY PANELS
DESPIN AND ACQUISITION OF 3-AXIS STABILIZATION IN NEAR-SYNCHRONOUS ORBIT PLUS ORBIT CORRECTIONS TO PUT SATELLITE IN SYNCHRONOUS ORBIT AND CORRECT STATION
ATTITUDE ACQUISITION
STATION ACQUISITION

Fig. 3. *Launch scenario for a three-axis stabilized satellite.*

plots of specified-versus-measured performance.

Launch and operation

The last major software class is concerned with launch and system operation, which comprises launch and transfer orbit, stationkeeping, attitude control, and message switching.

During powered flight, the launch vehicle computers match the vehicle's trajectory to a stored flight profile. The transfer-orbit portion of launch uses ground-based software, which computes the optimal firing sequence for the final motor burns. These take the satellite from the highly elliptic orbit provided by the launch vehicle to the desired circular geostationary orbit.

The transfer-orbit software determines the satellite orbital elements from tracking and ranging data and the satellite attitude from telemetered sun and earth sensor data. On the basis of this information, the transfer-orbit software computes the satellite attitude and time at which motor ignition will achieve geostationary orbit. Because a solid-propellant motor commonly moves the satellite from the transfer orbit into the desired geostationary orbit, motor orientation and ignition time are the only available parameters for controlling the change in the satellite velocity vector needed to bring the satellite close to its assigned slot.

Fine tuning of the satellite's position in geostationary orbit is accomplished by firing the liquid-propellant stationkeeping thrusters in accordance with the results of stationkeeping software runs. A poor injection into the final geostationary orbit reduces potential useful life, because it takes more stationkeeping propellant initially to reach the proper orbital slot.

At present, software or firmware is not used in on-board message switching in commercial communications satellites. However, use of on-board message-switching software has been proposed for several future applications.

Evolution of on-board computers and flight software

The spacecraft control functions on board a communications satellite support transfer/drift-orbit operations, on-orbit stationkeeping, and attitude control. Transfer/drift-orbit operations begin after satellite separation from the booster and end when the satellite is on station. During this phase, the crucial firing of the apogee kick motor, or AKM, occurs and circularizes the orbit.

Information telemetered during transfer orbit enables ground controllers to stabilize the spacecraft by spinning it to a desired angular velocity, determine its orientation by using on-board sensors, and reorient it by firing the thrusters.

After the AKM burn is complete, the spacecraft is reoriented, configured for on-station operations, and maneuvered into its designated orbital position. This sequence of events is depicted in Fig. 3 for a three-axis stabilized satellite.

A number of spacecraft housekeeping functions are candidates for on-board computer control. These include attitude control, thermal control, and power management. To date, interest has centered on attitude control, because this function shows the most opportunity for performance improvement.

The on-board attitude control function keeps the payload—the communications antennas—properly oriented. Attitude control processing translates earth sensor and gyro data into torques to be applied to the spacecraft body. These torques can be generated through earth-opposing magnetic dipoles, momentum wheels, or thrusters.

Figure 4 depicts a three-axis stabilized attitude control system. The horizon sen-

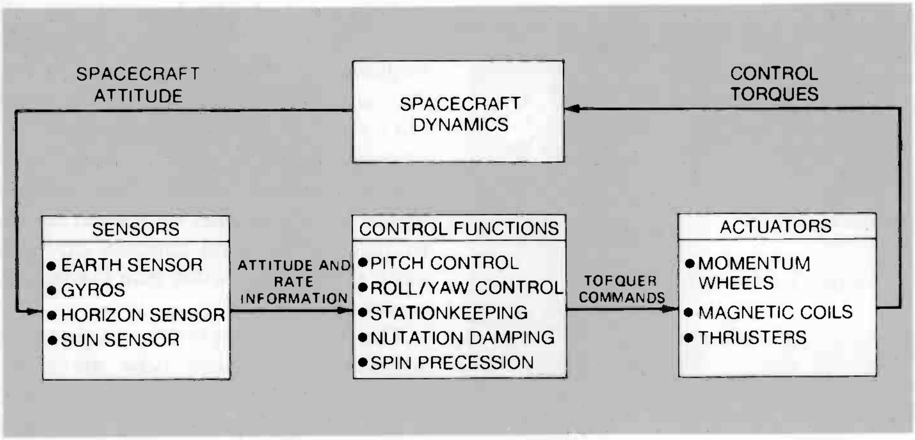


Fig. 4. *A three-axis stabilized attitude control system.*

sor, sun sensor, and spin precession control function support transfer-orbit operations. The sun sensor provides spin rate and yaw axis attitude data. Horizon sensors detect earth crossings while the spacecraft is spinning. Crossing times provide data for accurate spacecraft attitude determination.

Attitude control electronics

Spacecraft such as RCA Satcom I and II, launched in the mid-1970s, are of the three-axis stabilized type. These satellites handle attitude control and thruster firing functions through a hardware device known as the attitude control electronics, or ACE. Because of the fixed, limited logical capability of the ACE, stationkeeping required significant ground setup and monitoring; later designs reduced the level of these ground-control tasks.

Attitude logic processor

The next generation of Satcom satellites (Satcom III, IV, V, VI, and VII) consolidated some of the attitude control and thruster firing functions into a unit called the attitude logic processor. The ALP uses a CD1802 microprocessor with 4K bytes of ROM and 2K bytes of RAM. The ROM contains all thruster control logic for transfer-orbit and on-station mission operations; the RAM contains ground-commandable parameters for controlling these operations. The ROM software, more appropriately referred to as firmware, implements the following functions:

- power-on initialization;
- ALP execution control;
- spin precession logic;
- thruster control for east-west and north-south stationkeeping;
- momentum unloading logic;
- self-test and backup time-out functions; and
- telemetry of critical mission mode variables.

The ALP was flown on Satcom III and IV, launched in October 1981 and January 1982, respectively. It successfully performed all programmed functions. Onboard firmware-controlled functions have achieved better attitude control accuracy during stationkeeping thruster firing than could be achieved by ground control. Furthermore, the parts count is smaller in firmware-controlled systems, so they are lighter and more reliable.

Attitude processor electronics

The ALP's successor is the attitude processor electronics system, currently being built for future communications satellites. It retains all ALP functions and absorbs functions previously performed by other hardware units.

Transfer-orbit functions have been expanded to include attitude data collection and formatting. This increases the accuracy of data collected while the satellite performs a function that previously required ground hardware and software. Improved accuracy of attitude data allows more precise AKM pointing prior to firing and thus translates directly into fuel savings.

On-station attitude control functions have been expanded to include:

- pitch, roll, and yaw closed-loop control;
- nutation damping; and
- time-of-day-dependent pitch and roll offset pointing.

The time-of-day offsets compensate for antenna pattern distortions caused by conditions such as the temperature excesses of entering or exiting eclipse.

The firmware now requires 12K bytes of ROM; the microprocessor clock speed has been increased from 3.1 MHz to 3.6 MHz to provide the needed instruction execution rate. The firmware is written in the CD1802 assembler language.

A significant increase in the number and complexity of attitude control functions performed on board communications satellites has accompanied this giant step from hardwired logic control systems to microprocessor-based firmware control. Table I summarizes the software evolution.

Table I. *Evolution of satellite control functions.*

	RCA SATCOM I		***RCA SATCOM III***		***Present Generation***	
Control Function	*Hardware*	*Firmware*	*Hardware*	*Firmware*	*Hardware*	*Firmware*
Pitch Control	X		X			X
Roll/Yaw Control	X		X			X
Momentum Unloading				X		X
Stationkeeping	X		X (Backup)	X	X (Backup)	X
Nutation Damping	X		X			X
Spin Precession	X			X		X

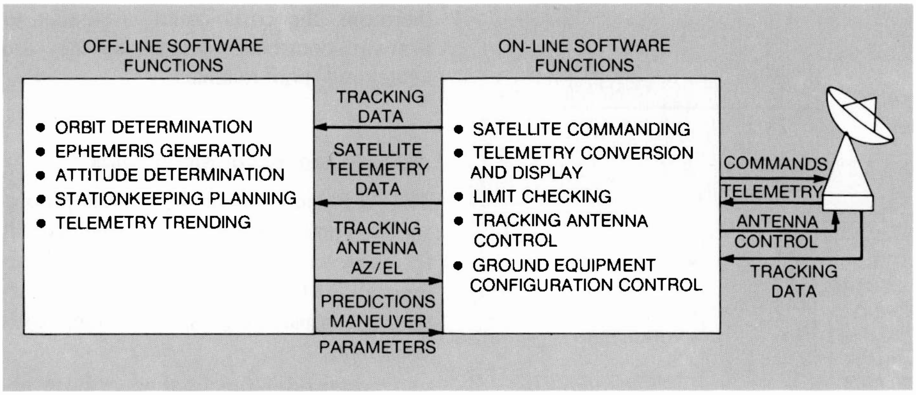


Fig. 5. *An overview of off-line and on-line functions.*

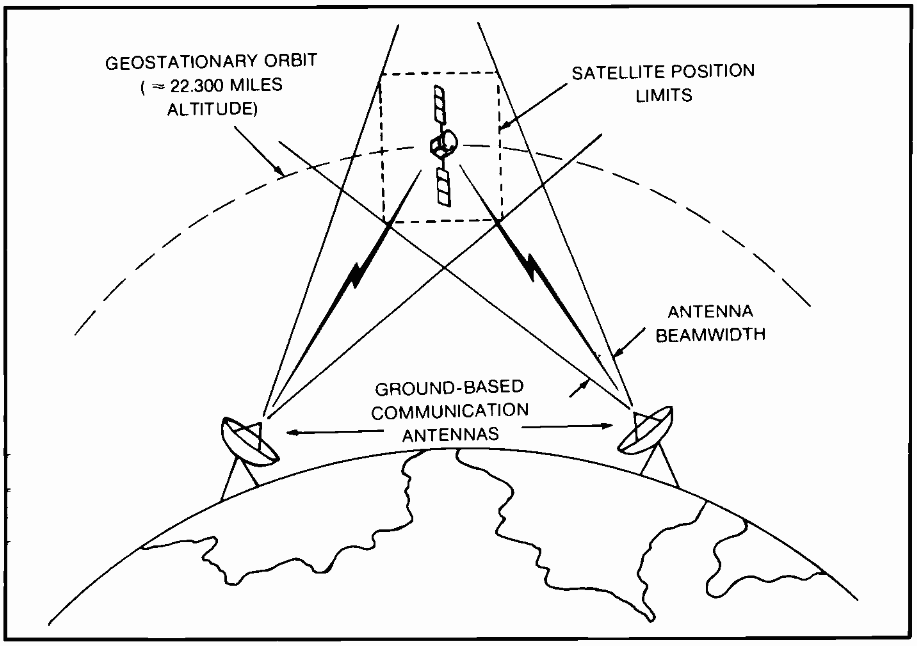

GEOSTATIONARY ORBIT (≈ 22.300 MILES ALTITUDE)
SATELLITE POSITION LIMITS
ANTENNA BEAMWIDTH
GROUND-BASED COMMUNICATION ANTENNAS

Fig. 6. *Satellite latitude and longitude must be tightly controlled to meet communications system requirements.*

Ground software supporting communications satellite operation

Although use of on-board processing to control satellite position and attitude is increasing, the bulk of required processing is still ground based.

Broadly speaking, ground processing falls into two categories. The first involves off-line computations associated with orbit determination and stationkeeping maneuver planning. The other contains on-line computations that command a satellite and monitor its status. Figure 5 is an overview of off-line and on-line functions.

Orbit determination and stationkeeping

The communications functions of geostationary communications satellites demand accurate control of orbit and attitude. Satellite attitude adustments are made to properly point on-board communications antennas.

Each communications satellite is assigned a longitude in the geostationary arc located some 35,900 km (22,300 miles) above the equator. Communications satellite common carriers exercise tight control over a satellite's orbit; the satellite's position must remain within a small tolerance band of the assigned longitude. The need for tight control arises because many ground-based receiving antennas are of the nontracking type. Thus, significant satellite motion would move the satellite from the center portion of the earth-station antenna beam and degrade signal reception. Figure 6 illustrates the need for close control of satellite position.

The size of the acceptable tolerance band depends on the communications frequency band.[1, 2] For the popular C-band systems (6-GHz uplink, 4-GHz downlink), the tolerance bands are typically ±0.1 degree in longitude and ±0.1 degree in latitude. For the increasingly used Ku-band systems (14-GHz uplink, 12-GHz downlink), the tolerance bands are ±0.05 degree in longitude and latitude. Ku-band systems have tighter bounds because, for a given diameter reflector, Ku-band antennas exhibit a much smaller beamwidth than C-band antennas.

The natural forces tending to perturb a geostationary satellite's orbit are small. Thus, once an acceptable orbit is achieved, it does not require correction for several weeks. In typical systems, orbit determination computations are performed once or twice per week; orbit corrections are implemented every two to six weeks, as required. Therefore, the processing to support orbit control can be performed effectively on a scheduled batch basis. Also, because the processing is low duty cycle and not time critical, many satellite operators use minicomputers with floating point hardware to perform it at minimum cost.

Orbit determination. To achieve the required control of a satellite's orbit, one must first be able to determine the current orbit and then plan any needed corrections. The orbit determination process consists of collecting a set of observations of the satellite's position over a measured time interval and then using orbit determination software to process these measurements and generate a best estimate of the satellite's orbit.

Another ingredient of the orbit determination process is a description of the set of natural forces acting on the satellite. Nominally, the satellite is in a circular orbit about the earth, with a period of about 1436 minutes. However, because the earth's equator is not perfectly circular (triaxiality), asymmetries are created in the gravity field, causing a satellite to drift from an initial longitude. Also, the gravitational attraction of the sun and moon inclines the satellite's orbit, causing it to twist out of the equatorial plane. There are other perturbing forces on geostationary satellites, but the two identified above are the most significant.[3] The orbit determination software uses models of the natural forces acting on a satellite to account for their effects on satellite motion.

Given a set of satellite observations and the force models, the orbit determination software follows a least-squares method to select the orbit (as given by its orbital elements) that best matches the observations. One common scheme employs an iterative differential correction algorithm to improve an initial estimate of the orbit.

In the differential correction algorithm, the initial estimate of the orbit is propagated over the data collection interval, and the observations that would have been obtained at the tracking stations are computed. These computed observations are compared with actual observations, and the initial orbit estimate is corrected to reduce the sum of the squared observation differences (residuals). This procedure is repeated iteratively until the residuals are minimized and the orbit estimate converges.

In addition to providing the orbit estimate itself, the least-squares method can also generate improved estimates of the tracking model parameters and force-model coefficients used in the orbit determination process. The new orbit estimate, along with updated model coefficients, is stored for use in future orbit determination computations.

Orbit determination tells where the satellite was at a certain time (epoch). But a more important issue for orbit control is how this orbit (and thus the satellite's position) will vary over a subsequent time interval—the next 30 days, for example. The prediction of a satellite's future orbit is commonly called *ephemeris generation.*

Ephemeris generation software uses the results and natural force models of the orbit determination procedure. The best estimate of the current orbit is extrapolated forward in time by means of the force models; orbit values are output at selected times during the extrapolation (for example, every four hours). Manual or automated review of these data permits identification of the time at which the satellite's position will approach the edge of the tolerance band around the assigned orbital locations. This time defines when stationkeeping maneuvers will be required. The maximum period for which a valid satellite ephemeris can be generated is limited by the uncertainties associated with the force models. Figure 7 is an overview of orbit determination and ephemeris generation processing.

The satellite's ephemeris can aid system operations in other ways. For example, it can reveal the timing and duration of satellite eclipses to facilitate appropriate onboard power management. Also, it can identify periods when the sun or moon will interfere with proper operation of the attitude control system's earth sensor.

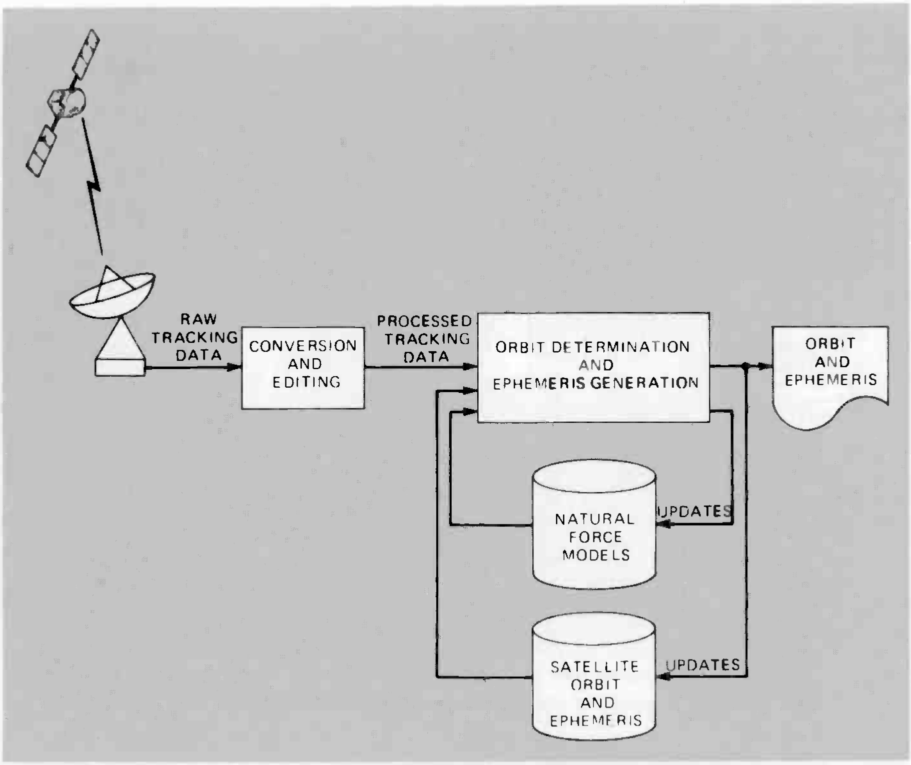


Fig. 7. *Orbit determination processing employs tracking observations and natural force models to estimate the current satellite orbit and to generate a satellite ephemeris.*

Stationkeeping software. The principal tasks of stationkeeping software are to assist in planning the maneuver and to update satellite status data after maneuver completion.[3, 4]

Generally, stationkeeping maneuvers are designed to minimize the amount of fuel expended and maximize the time interval between orbit corrections. However, special circumstances might dictate a different strategy for any given maneuver. The stationkeeping software typically possesses a preliminary planning mode. In this mode, simplified models of satellite dynamics and thruster performance provide rapid, approximate solutions, so that an analyst can explore the impact of various options and constraints. For instance, an analyst might wish to investigate

- one-burn versus two-burn orbit correction maneuvers;
- the time required for a minimum-fuel orbit correction; or
- the amount of correction achievable with a given maximum burn time.

On the basis of this preliminary analysis, the analyst selects a stationkeeping strategy. He can use detailed models available in the software to determine the exact velocity change (ΔV) required for implementing this strategy and the time at which the correction should be made. Furthermore, the stationkeeping software uses detailed thruster performance models and data on thruster geometry to determine which thrusters must be fired and the time phasing of the firings needed for the required ΔV. This thruster firing data must be translated into the appropriate satellite commands for uplink at the required times. Figure 8 is an overview of stationkeeping processing.

Because the satellite's attitude can be altered by thruster firings that do not pass through the satellite's center of mass, system designers use symmetrical configuration for storing expendables and use an on-board attitude control function to minimize residual torques, which would rotate the spacecraft.

After a stationkeeping maneuver, the stationkeeping software generates updated estimates of remaining fuel on the basis of data available in the satellite telemetry stream. Also, a set of tracking data is collected and used to determine the new orbit. A comparison of the new and old orbits allows estimation of the actual ΔV imparted to the satellite and thus provides data for refining thruster models.

Telemetry, tracking, and control

Telemetry, tracking, and control, or TT&C, software functions are best described within the system concept (Fig. 9). A communi-

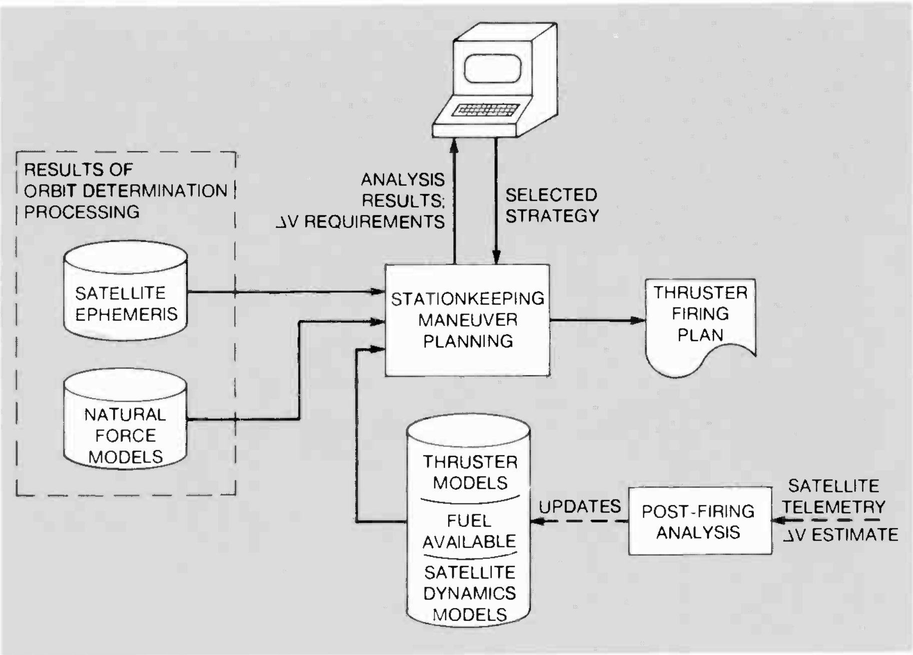


Fig. 8. *Stationkeeping maneuver planning uses satellite ephemeris, natural force models, and thruster models to determine the required thruster firing plan.*

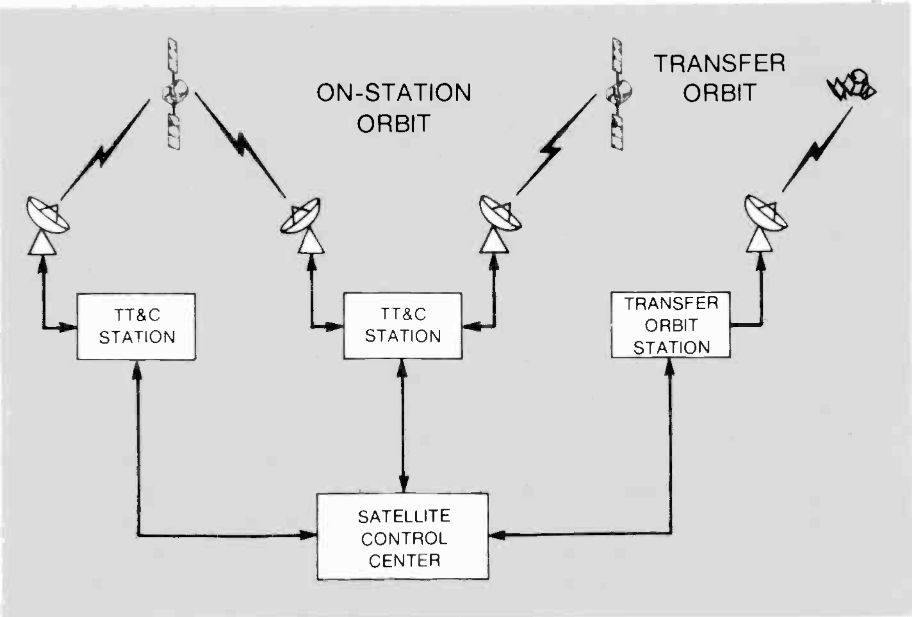


Fig. 9. *A configuration showing telemetry, tracking, and command stations.*

Table II. *Data exchanged between the off-line and on-line software.*

On-Line to Off-Line	*Off-Line to On-Line*
Tracking and range data	Stationkeeping parameters
Thruster performance parameters	Tracking antenna predicted pointing data
Attitude data	

cations satellite, whether in transfer or geosynchronous orbit, must be tracked, monitored, and controlled. These functions must be performed both when injecting a satellite into geosynchronous orbit and when controlling the communications satellite operations without service disruption for a seven- to ten-year mission lifetime.

Tracking and attitude data are collected to determine and predict a spacecraft's orbit and orientation. This information is used in maneuver planning, anticipating solar and lunar interference with on-board sensors, and predicting battery reconditioning opportunities. Real-time telemetry monitoring provides rapid detection of anomalous conditions and aids problem analysis and recovery. In addition, long-term telemetry trend analysis assists in pinpointing environmentally induced degradations and in predicting spacecraft life expectancy. Using the TT&C spacecraft command capability, ground controllers can diagnose problems, work around failures, collect range and attitude data, and execute maneuvers.

The ground segments of TT&C systems have historically been composed of semi-autonomous TT&C stations, satellite control centers, and transfer-orbit stations, or TOSs. Various combinations of these are used for transfer-orbit and on-orbit operations. Sometimes, a satellite control center, or SCC, and a TT&C station are collocated.

On-line software processing functions. Processing functions within a satellite ground control system are classified as either on-line or off-line.[5] Some TT&C configurations host these functions on separate computers. Others integrate them in the same computer and delegate on-line tasks to the foreground and off-line tasks to the background. Off-line functions typically require 32-bit accuracy; 16-bit accuracy suffices for on-line, real-time processing.

Off-line functions executing in a batch mode or as a background task include

- orbit determination;
- ephemeris generation;
- transfer-orbit attitude determination;
- maneuver planning; and
- fuel management.

The coupling of off-line and on-line software functions is perhaps best illustrated in examples of the types of information they exchange (Table II).

Thruster performance is modeled by parameters that include the number of times each thruster is fired, duration of pulses, temperatures during thrusting, and fuel pressure. Attitude data, typically collected via telemetry during transfer orbit only, are processed on the ground to determine spacecraft orientation. On the basis of tracking, range, and attitude data, off-line software generates stationkeeping maneuver parameters.

On-line software processing functions include

- conversion of spacecraft and ground equipment telemetry to engineering units;
- telemetry limit-checking and alarm processing;
- computer networking;
- backup attitude control;
- redundancy management; and
- operator-initiated tasks.

Raw digital telemetry is received as binary "counts," converted to engineering units by use of a fifth-order polynomial, and then multiplied by a scaling factor (for those data items to be displayed on a strip chart). Converted telemetry is compared with preset red and yellow alarm limits. Red limits define failure boundaries (minimum and maximum); yellow limits define caution boundaries. In addition to informational messages displayed on terminals, a limit violation can cause station or console alarms to sound. Telemetry-unique polynomial coefficients, scale factors, limit values, and descriptive text are kept in online databases.

Computer networking is the pulse of a satellite ground control system. Although this is primarily an input/output function, networking requires certain processing for error detection, correction, and recovery. The software monitors this processing so that an operator can respond quickly to data transmission problems between stations.

Although attitude control is performed on board a spacecraft, the TT&C software provides a backup capability. When activated, this software samples earth sensor and gyro data in telemetry, computes attitude errors, and automatically sends spacecraft control commands to correct the errors. The software must accommodate normal spacecraft commanding, as well as time-critical attitude control processing.

Multisatellite ground control systems are typically configured with backup hardware including computers, computer peripherals, networking equipment, and some RF equipment. The software handles redundancy management by detecting failures in the primary device and switching to its backup. Systems with hot standby computers require intercomputer watchdog circuits. The standby software monitors the primary computer via this circuit and switches all command and control functions to the standby system when failures are detected.

Satellite operation and control procedures initiated by an operator typically include

- range and tracking data collection;
- command creation and release for execution;
- intercomputer file transfer;
- ground equipment reconfiguration;
- telemetry page display assignments; and
- strip chart pen assignments.

These tasks invoke software that interacts with the operator, performs the task, and returns control to the operator. The software's interactive, or man-machine, interface is critical to effective, error-free operations. Predefined procedures or auto-answer files for menu-based systems simplify routine tasks such as ranging, which require several operator-supplied parameters. Hardware features such as reverse video, flashing values, and color CRTs enhance communication of failures and out-of-limit conditions. At the same time, the software reduces the probability of command errors by alerting the operator to potentially hazardous operations before executing them.

On-line software products. Commands are the primary ground control system output to the spacecraft. Software sends commands via programmed I/O to special-purpose command generation hardware. This baseband device encodes each command and interfaces with the RF uplink equipment used to transmit the command signal.

An integral part of commanding is verifying reception of a transmitted command. Command verification, or CV, is achieved on the ground by comparing the sent command with the telemetered contents of the spacecraft's command register. After successful CV, an execute command or tone is sent to the spacecraft to activate the previously transmitted command.

Typically, commands for each satellite are kept in an on-line database. This contains the binary command code, a mnemonic, descriptive text, and a classification of "hazardous" or "not hazardous."

Commands that fire thrusters or turn off a transponder are considered potentially hazardous to satellite operations. Therefore, TT&C software and hardware provide a series of interlocks that prevent their accidental use.

Antenna control is another TT&C output. Satellite tracking requires that the system direct the tracking antenna to a predefined position or direct the antenna to continuously track a predicted orbit. Antenna predict files are generated by the orbit determination and ephemeris generation functions of the off-line software. Antenna control software primarily supports acquisition of a satellite in transfer orbit, but this software can also back up continuous tracking operations, even in the event of a tracking subsystem failure. The software typically effects antenna control through some standard I/O interface, such as an EIA RS-232C or IEEE 488 bus.

Ground equipment control is generally achieved by a combination of hardware and software under operator direction. The ground equipment control software interfaces with the equipment through serial, parallel, and special I/O buses or ports. Control and monitoring functions can be stand-alone or be integrated with the real-time satellite control software. The on-line databases that support control operations are highly dependent on configuration and operational philosophy. Ground control systems run the gamut from mostly manual to highly automated.

The display products of on-line software include CRT display of telemetry pages in color or black and white; CRT, printer, and strip chart traces of selected spacecraft or ground telemetry; and engineering-unit or hexadecimal printouts of telemetry frames, command lists, and on-line databases. The operator initiates requests for generation of these products, which are employed during routine operations as well as anomaly diagnoses. The software retrieves information for display from current-value tables and short-term historical databases. Short-term data, typically kept on disk, include telemetry and commands for the previous 12 to 36 hours. Long-term data are kept on digital magnetic tape.

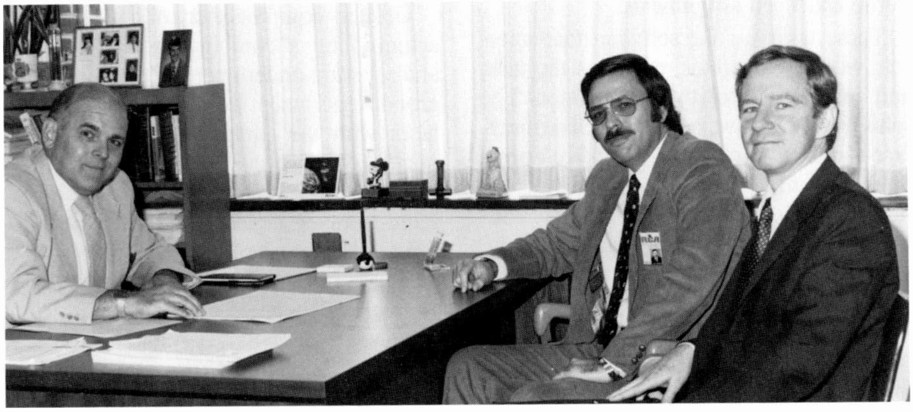
Archive data are used as a time-tagged record of spacecraft telemetry and commands. Typically, these data are retained for 30 to 60 days and then compressed for permanent storage. Compression software can record only changes (deltas) in a telemetry value or the minimum, maximum, and mean of a telemetry value over a one-hour or 24-hour period. The archive tapes are used for long-term trend analysis of satellite thermal, power, and attitude control subsystems, as well as communications payload performance.

Conclusion

Software has become an essential ingredient in the design, implementation, and operation of communications satellite systems. Starting from a very modest base, the number and complexity of software applications have increased dramatically over the last decade. These new applications have reduced the amount of manual effort required to manage a communications satellite system. They have improved satellite design and allowed microprocessors to replace many satellite hardwired logic assemblies.

References

1. U. Domb, A.J. Grise, and C. Kes, "The Orbit Determination System of TELESAT Canada," *Proceedings, AAS/AIAA Astrodynamics Specialist Conf.*, AIAA paper no. 75-057 (1975).
2. J. Lewin, "Ground-Control System for SATCOM Satellites," *J. Spacecraft*, Vol. 16, No. 4, pp. 273-279 (July/August 1979).
3. M.J. Neufeld, "Orbit Correction," *Space/Aeronautics*, pp. 48-55 (February 1965).
4. W.H. Wright and B.M. Anzel, "TELESAT Station-keeping Methods and Performance," *Proceedings, AIAA Fifth Communications Satellite Systems Conf.*, AIAA paper no. 74-452 (1974).
5. D.E. Ekman and C.W. Hanson, "DOMSAT Operations Support with a Single Computer," *Proceedings Int'l Telemetering Conf.* (1981).



Authors (left to right) Dusio, Cashman, and Murphy.

Thomas P. Murphy, Manager of Software Design and Development at RCA Astro-Electronics, is responsible for flight, ground test, and satellite control software. Prior to joining RCA, he worked at Control Data Corporation as a senior systems specialist.

Murphy received his MS in computer science from New York University's Courant Institute and his BS in mathematics from St. Francis College.

Contact him at:
RCA Astro-Electronics
Princeton, N.J.
TACNET: 229-2038

William F. Cashman is Manager of Series 3000 Engineering for RCA Astro-Electronics. Prior to joining RCA, he was a member of the technical staff at the Analytic Sciences Corp., an associate professor of computer science at the State University of New York College at Potsdam, and a control systems engineer at NASA/Electronics Research Center.

Cashman received a BS in electrical engineering from Tufts University, an MS in applied science from Yale University, and a PhD in electrical engineering from Northeastern University.

Contact him at:
RCA Astro-Electronics
Princeton, N.J.
TACNET: 229-3165

Emilio W. Dusio, Manager of Electronics Systems at RCA Astro-Electronics, is responsible for the design of flight electronics and ground control and test equipment, and for software development for both flight and ground equipment. His current interests are software engineering, management, and the use of computers to increase productivity of engineers.

Dusio received his BEE and MEE in 1958 and 1959, respectively, and an MS in computer science in 1982. He has been with RCA since 1961 and has completed a number of design and management projects.

Contact him at:
RCA Astro-Electronics
Princeton, N.J.
TACNET: 229-2212

G.W. Suhy

Applying software tools to system development

The effective use of software tools is becoming a major factor in controlling cost and reliability in computer system development.

Almost every branch of engineering uses automated design aids, measurement tools, and analysis tools in the system development process. Electrical engineers, for example, use computers to facilitate the design of circuits, and operate special-purpose equipment for component, assembly, and system testing levels as well as for long-term maintenance. Similarly, software tools aid the design effort; verify correctness of software operation at the subprogram, program, and system levels; and support many aspects of long-term field support. This article surveys some of the more common types of software tools and their application to the various stages of the software development process.

Types of software tools

A new software system develops in a series of characteristic stages: the requirements definition stage, in which the system's desired behavior is described; the design stage, which defines the internal structure of the system and each of its elements; the coding stage, in which the detailed design description is converted to computer programming language source code statements; unit and integration testing stages; and the software maintenance stage. Many specific types of software tools support software development; these tools are important not only to their utility in supporting the process, but also because they represent a significant part of the system development costs.

Specification and design languages, for instance, simplify the analysis and modification of requirements and design descriptions, and static analysis tools improve the productivity and quality of the coding effort. Dynamic analysis tools support unit and integration testing, and configuration management tools simplify long-term maintenance.

Software tools are essential to system development, but unnecessary complexity of these tools should be avoided. Factors in software tool selection are expected system lifetime, delivery schedules and personnel availability. However, the most important single factor in determining the appropriate level of support capability is system size (Fig. 1).

Requirements definition

A performance specification, the product of the requirements definition activity, formally describes the function to be performed by a new software system and the external interfaces and constraints with which it will be required to operate. While performance specifications may define certain critical algorithms and assumptions (for instance, encryption algorithms, earth curvature models, coordinate systems), they generally do not address the internal design of the system, but rather set the performance criteria the system will be expected to meet.

A new system is defined iteratively, a process requiring creativity and judgment. No mechanized system can create new ideas, but many software tools can reduce the time and effort involved in refining the system. These tools, sometimes called specification language processors, combine the editing features of commercial word processors with some of the syntax analysis capabilities of structured compilers. The editing features allow the user to alter text; change page and paragraph numbers; relocate, insert, or delete large sections of text; and receive a fresh draft, complete with an updated table of contents and index, within minutes.

A specification language requires the user to observe certain conventions of syntax when entering new text. For example, the description of a low-level function might require a preceding comment defining the higher-level function which controls it and any applicable military or industrial standards affecting it. The specification language processor can recognize these comments and generate "trees" showing the interde-

Abstract: *The development of computer systems is a multistage process requiring detailed design, testing, and maintenance. Software tools, if carefully selected for the appropriate tasks, can decrease development expenses besides enhancing the quality of the system's design. Various software tools are used throughout each stage of the development process; the cost versus utility tradeoffs considered in choosing these tools are discussed in this paper.*

Final manuscript received October 12, 1983.
Reprint RE-28-6-3

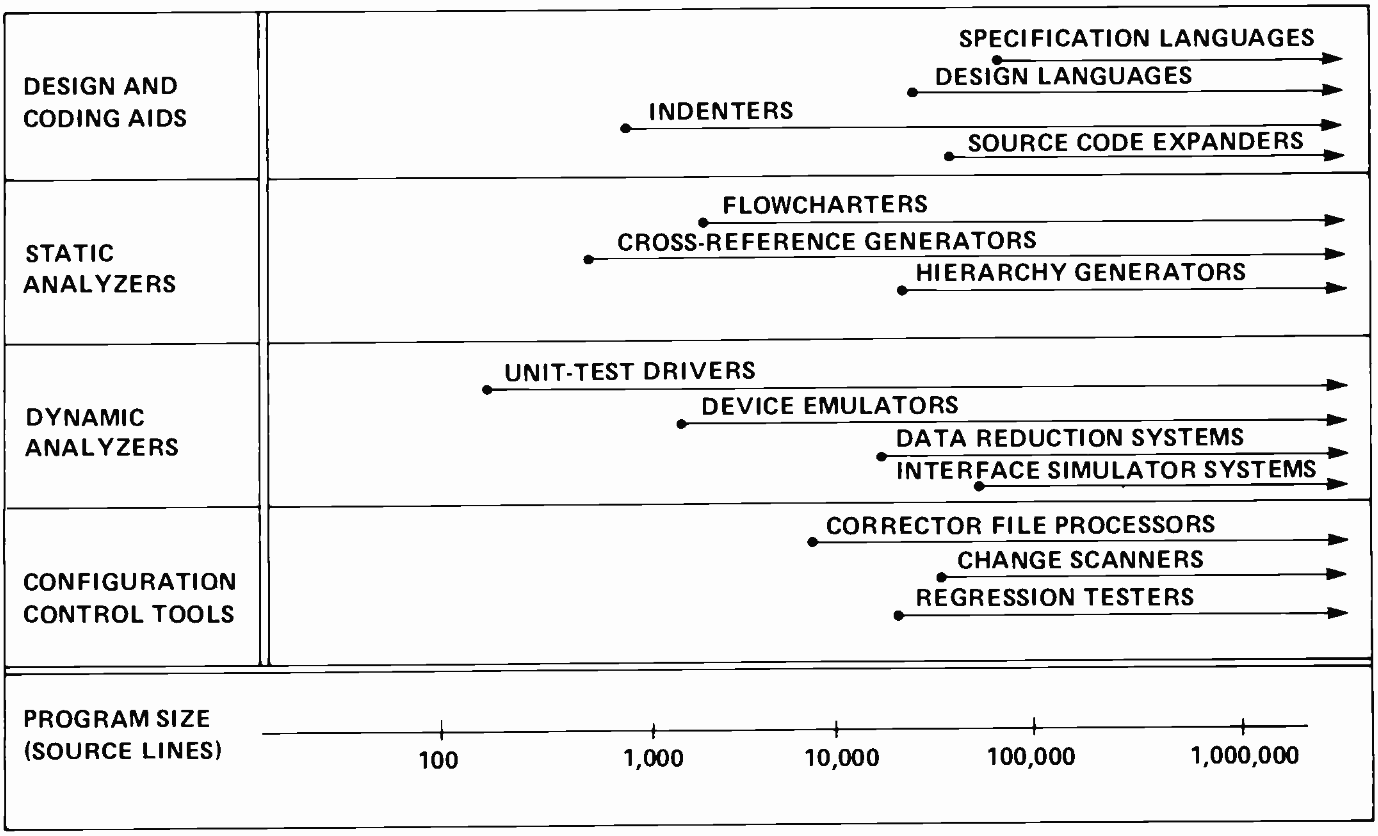


Fig. 1. *Types of software development tools and the sizes of the systems for which they are most frequently used.*

pendence of functions and the applicability of standards. If at a later iteration in the process a function or standard is changed, the processor can identify immediately all sections of the specification affected by the change.

The powerful cross-referencing capabilities of a specification language help detect subtle but potentially costly oversights in a requirements document. In an industrial process control system, for example, each major function may be required to sense certain emergency conditions. A cross reference generated to show each section and paragraph where "Emergency State" is referenced can readily identify those sections in which it is missing. If an "inverted tree" diagram is generated for each reference to "Emergency State," it can be confirmed that for each instance of "Emergency State" a control path exists back to the executive module. An exception would indicate that an emergency condition can be detected, but that no subsequent response action has been specified.

Design

The design stage of a software development effort has two overlapping phases: high-level design and detailed design. The product of the high-level design phase is the software system architecture, comprising the high-level algorithms that control the system, the specific functions to be controlled by the high-level algorithms, groups of functions allocated to software modules, and data base elements to be used by the modules. Detailed design, which begins at the module level, reduces the functions to be performed by a module to algorithms and data item definitions to be implemented in a computer programming language.

Once the software engineer conceives an initial design, appropriate software tools can be used to develop specific control and data structures. These tools, called program design language processors, reduce the time and effort required to define a top-level design. More importantly, they can improve the quality and completeness of the design. They have word processing capabilities similar to specification language processors, but have much more detailed syntactic requirements resembling that of structured higher-order computer programming languages (Fig. 2).

Program design language processors can manipulate and format the input statements as well as detect certain types of logical inconsistencies. They can also analyze a design description in this language, producing block diagrams and inverted trees that show all control paths leading to a module. Some processors may require that the data used by each module be listed; they can then use this information to generate data flow diagrams and identify inadvertently broken or duplicated paths for each data item. This feature is particularly valuable in identifying the implications of modifications to the original design.

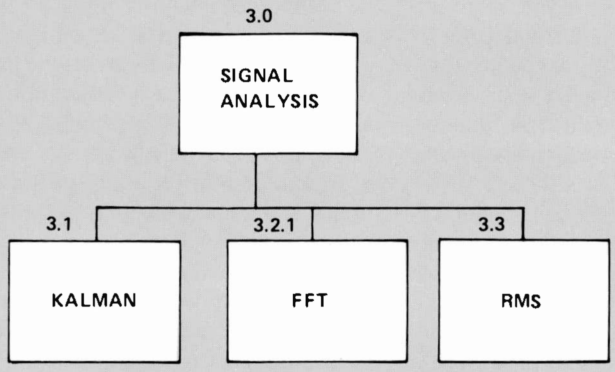
Unlike the manual approach of drawing block diagrams, where each functional block is the same size, the use of a program design language produces text representations which are roughly proportional to the complexity of each function. Thus, the designer has an early indication that the allocation of functions to modules is unbalanced and that the complex modules should be divided or the simple modules should be combined. Generated data flow diagrams also can help optimize module functional definitions by reducing the size of their data interfaces.

Program design languages can provide the rapid feedback during iterative refinement at the high-level design phase. The detailed design phase is less iterative and is closely related to the coding activity; in fact, the minimum element of a detailed design typically corresponds to five to ten lines of source code. For these reasons, detailed design can be implemented as source code, and software tools can be used for coding analysis to provide the feedback for making corrections to the detailed design.

Coding

Because design changes and corrections become considerably more expensive after a source code baseline has been established, software tools used during the coding phase can be very valuable in reducing that cost. For smaller systems, standard tools such as compilers and editors are usually sufficient. For a larger project, such tools as indenters, flowcharters, cross-reference generators, and data base analyzers should be considered.

Indenters, which are most useful for such languages as JOVIAL, CMS-2, and

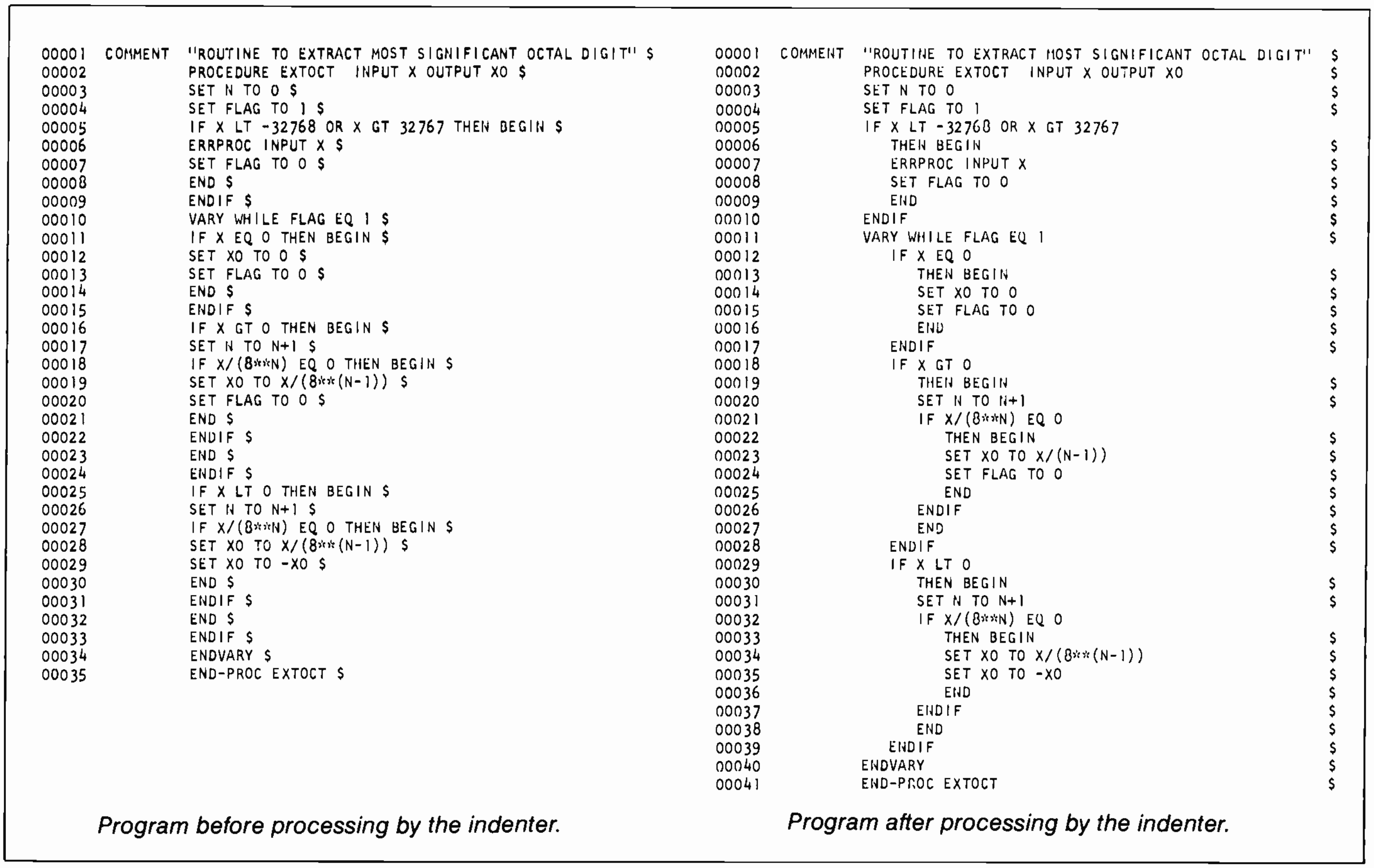


(a) Conventional Block Diagram

```
PROCESS 3.0;   "signal evaluation"

   STEP:  "CLASSIFY SIGNAL TYPE"

   FOR CASE OF:   "select analysis type"

      'PREDICTION':   EXECUTE PROCESS 3.1 "Kalman filter"
      'UNKNOWN':      EXECUTE PROCESS 3.2 "Fast-Form Transfer"
      'MONITOR':      EXECUTE PROCESS 3.3 "RMS value"

   ENDCASE;

   RETURN;

ENDPROCESS 3.0;
```

(b) Program Design Language Description

Fig. 2. *A hypothetical representation of program design language.*



```
COMMENT  "ROUTINE TO EXTRACT MOST SIGNIFICANT OCTAL DIGIT" $
         PROCEDURE EXTOCT  INPUT X OUTPUT XO $
         SET N TO 0 $
         SET FLAG TO 1 $
         IF X LT -32768 OR X GT 32767 THEN BEGIN $
         ERRPROC INPUT X $
         SET FLAG TO 0 $
         END $
         ENDIF $
         VARY WHILE FLAG EQ 1 $
         IF X EQ 0 THEN BEGIN $
         SET XO TO 0 $
         SET FLAG TO 0 $
         END $
         ENDIF $
         IF X GT 0 THEN BEGIN $
         SET N TO N+1 $
         IF X/(8**N) EQ 0 THEN BEGIN $
         SET XO TO X/(8**(N-1)) $
         SET FLAG TO 0 $
         END $
         ENDIF $
         END $
         ENDIF $
         IF X LT 0 THEN BEGIN $
         SET N TO N+1 $
         IF X/(8**N) EQ 0 THEN BEGIN $
         SET XO TO X/(8**(N-1)) $
         SET XO TO -XO $
         END $
         ENDIF $
         END $
         ENDIF $
         ENDVARY $
         END-PROC EXTOCT $
```

Program before processing by the indenter.

```
COMMENT  "ROUTINE TO EXTRACT MOST SIGNIFICANT OCTAL DIGIT"  $
         PROCEDURE EXTOCT  INPUT X OUTPUT XO                $
         SET N TO 0                                         $
         SET FLAG TO 1                                      $
         IF X LT -32768 OR X GT 32767
            THEN BEGIN                                      $
            ERRPROC INPUT X                                 $
            SET FLAG TO 0                                   $
            END                                             $
         ENDIF                                              $
         VARY WHILE FLAG EQ 1                               $
            IF X EQ 0
               THEN BEGIN                                   $
               SET XO TO 0                                  $
               SET FLAG TO 0                                $
               END                                          $
            ENDIF                                           $
            IF X GT 0
               THEN BEGIN                                   $
               SET N TO N+1                                 $
               IF X/(8**N) EQ 0
                  THEN BEGIN                                $
                  SET XO TO X/(N-1))                        $
                  SET FLAG TO 0                             $
                  END                                       $
               ENDIF                                        $
               END                                          $
            ENDIF                                           $
            IF X LT 0
               THEN BEGIN                                   $
               SET N TO N+1                                 $
               IF X/(8**N) EQ 0
                  THEN BEGIN                                $
                  SET XO TO X/(8**(N-1))                    $
                  SET XO TO -XO                             $
                  END                                       $
               ENDIF                                        $
               END                                          $
            ENDIF                                           $
         ENDVARY                                            $
         END-PROC EXTOCT                                    $
```

Program after processing by the indenter.

Fig. 3. *A simple CMS-2 program before and after being processed by an indenter. A simple review of the block for positive X shows that the SET N TO N+1 statement is common to two paths and could be relocated at the top of the block, simplifying the program and reducing memory requirements. For 2 negative X, FLAG is never reset and the VARY becomes an infinite loop.*

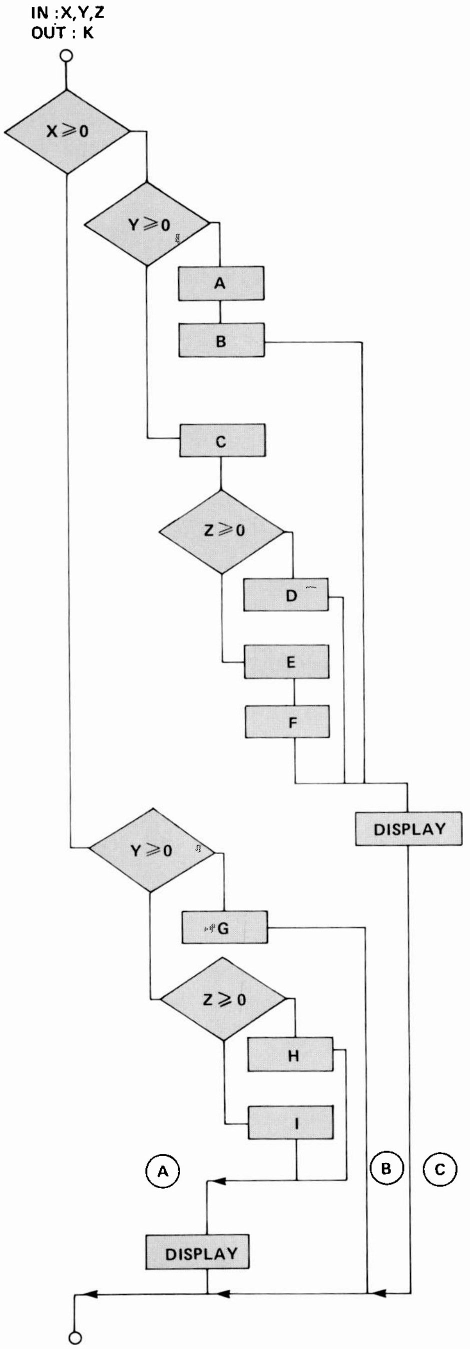


Fig. 4. *Example of a program as it might appear after being processed by an automatic structured flowcharter.*

FORTRAN.77, format computer programs in a way that shows hierarchical control relationships. A clear and consistent representation of these relationships can suggest improvements in efficiency and reveal subtle design errors (Fig. 3).

Structured flowcharters provide another valuable means for analyzing source code. Consider a simple program shown in Fig. 4, for computing a filter constant K from three sampled input values X, Y and Z. A logical error in the program is that for some input values, the output is undefined. The source of this error would be difficult to isolate from the program listing, but the flowcharter, because all possible paths are shown as solid lines, makes this task much simpler. Only two locations in the program generate output. By drawing a horizontal line through the last output statement (at "I"), it is clear there are only three possible ways to leave the program. Path A results in an output and path C can be reached only after an output, but path B provides a complete path from program entry to program exit without ever passing through an output statement, and is the source of the error.

A cross-reference generator is useful for identifying another class of design errors. After processing a program, a cross-reference generator lists each data item and references every point in the program where that item is written (set) and read (used). The two most frequent design errors are "item set but not used" and "used but never set," the latter being the software equivalent of an open circuit.

For very large programs, where it is not possible for any one individual to be familiar with the entire program, standardization of source code becomes increasingly important and sometimes is one of the criteria for acceptance during program delivery. If the source code is not standardized during the coding phase, it is very difficult to change it after the program is placed under formal configuration control. Although the strict observance of standards may appear to be a nuisance to the original designer, who is intimately familiar with the program, it is extremely helpful to the maintenance programmer who must modify the program many years later.

Source code expanders and standards enforcers are useful tools in standardizing source code. Source code expanders are similar to assembly language macro-instructions, but apply to higher order languages and result in modified source code rather than object code. In cases where extensive repetitive coding is required, they allow a programmer to insert a key word in the original source code file and expand it to many lines (Fig. 5). Standards enforcers can read large files of source code and report any deviations from format-related standards.

Unit testing

Extensive testing of modules before integration into a system provides a practical way of isolating and correcting most of the errors not detected during the design and coding phases. At the unit-test level, it is possible to control module inputs in order to evaluate each logical path over a representative range of conditions.

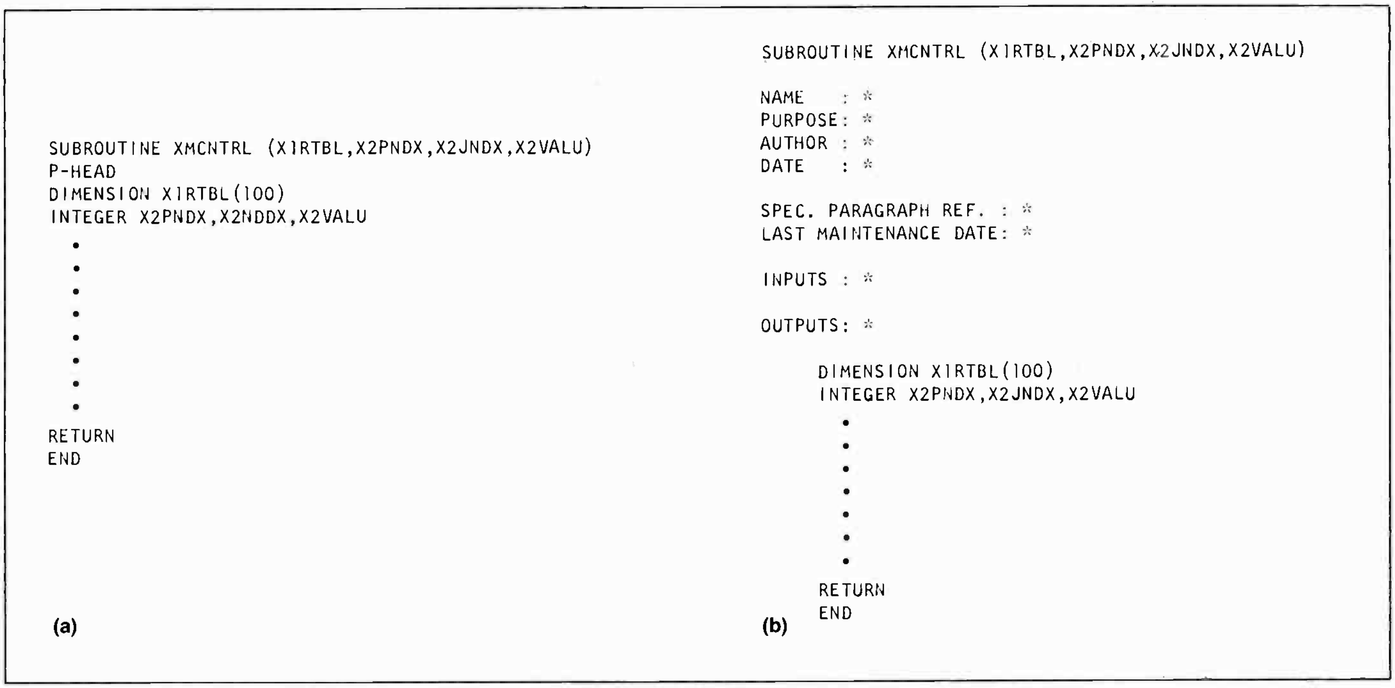
More sophisticated software tools are required with more complex modules, larger systems, and systems with greater life expectancies. Some small programs can be tested adequately by using breakpoint and input registers, available on most computers, and would not warrant the expense of developing a software-based test driver. Most larger computer programs, however, can be unit-tested only by another program.

A unit test driver must provide controlled and repeatable inputs to the module being tested. For simple modules or programs having only a few modules, it may be sufficient to develop a test driver that generates only a fixed set of test inputs. This is essentially a "throwaway," single-purpose test driver: if its outputs need to be changed, then the driver can be modified or a new one can be written.

For larger modules, for systems with many modules having similar input characteristics, or for programs expected to be in use for a long period of time, it is more practical to develop a single, more general test driver that can read from disk or tape a file of control commands and test data. Such a "programmable" test driver can be re-used simply by changing its input file. For very large systems, where the manual generation of a test data file is impractical, the development of an event-oriented test language may be justified.

Except for cases where a module output is easily observable (for example, a symbol generator that drives a display), a unit test driver must be able to accept and to some extent process the module's outputs. In a simple module, the output can be displayed on the computer's indicator lights; for a larger system, a hexadecimal printout may be adequate; and for a very complex module, a sophisticated data reduction capability may be required to interpret the module's outputs adequately.

The last factor to be considered is internal monitoring capability, which involves inspecting intermediate data and status inside the module before its outputs are produced. In simple modules, output data or the computer's own breakpoint registers can provide sufficient insight into module operation to identify errors. For larger modules, some means of capturing internal data without disrupting module operation is necessary. Although special-purpose equipment, wired to specific instruction and memory



```
SUBROUTINE XMCNTRL (X1RTBL,X2PNDX,X2JNDX,X2VALU)
P-HEAD
DIMENSION X1RTBL(100)
INTEGER X2PNDX,X2NDDX,X2VALU
   •
   •
   •
   •
   •
   •
   •
   •
RETURN
END
```

(a)

```
SUBROUTINE XMCNTRL (X1RTBL,X2PNDX,X2JNDX,X2VALU)

NAME    : *
PURPOSE: *
AUTHOR : *
DATE    : *

SPEC. PARAGRAPH REF. : *
LAST MAINTENANCE DATE: *

INPUTS : *

OUTPUTS: *

     DIMENSION X1RTBL(100)
     INTEGER X2PNDX,X2JNDX,X2VALU
        •
        •
        •
        •
        •
        •
        •
     RETURN
     END
```

(b)

Fig. 5. *A program shown before source code expansion* **(a)**, *in which comment is originally coded as P-HEAD, and the program shown after source code expansion* **(b)**. *It is much easier and more accurate for a programmer to edit each of the asterisks than to generate repeatedly all of the non-varying text manually.*

buses, is sometimes used, a more common approach is to insert diagnostic statements in the module's source code that call a subprogram in the test driver and indicate the data items to be recorded. The unit test driver copies these data items, writes them to a tape or disk file, and returns control to the module being tested. Except for the most critical real-time applications, this technique does not significantly disrupt the module's timing and provides a comprehensive internal "history" of how the module processed its inputs.

In many cases, a simple binary printout of the recorded data is sufficient and has a cost advantage in that the same unit test driver subprogram can be called from many diagnostic test points in the module. For particularly complex modules, specialized formatting and data reduction capabilities for certain critical test points may need to be developed.

Integration testing

After all of the modules have been unit-tested individually, they are integrated into a complete program and tested as a system. Integration testing, usually more costly than unit testing, is often a formal requirement for product acceptance. Integration testing identifies the residual design errors not detected in unit testing, identifies subtle specification errors, and provides the first reliable measurements of system-level performance parameters.

Integration testing is usually performed on the computer in which the program will operate as a finished system, while unit testing often can be performed on a time-sharing system. A typical integration testing approach would be to execute a small number of simple test cases and correct any obvious errors such as incorrect memory assignments, wrong channel numbers, or missing modules and then to perform a detailed series of tests consisting of an item-by-item verification of each function in the performance specification for the system.

Many of the software tools available for integration testing are the same as those for unit testing and the tradeoffs for selecting them are similar in nature. The optimum point, however, is shifted upward somewhat; for example, a system for which an off-line data reduction capability and an input data script language are not cost-effective in the unit testing stage might require such tools for the integration phase.

The complex analysis tasks involved in integration testing are also a factor in tool selection. In general, a single module performs a well-defined function, will always produce the same outputs from the same series of inputs, and predict analytically its output for any particular input. When a system of modules is integrated, data feedback paths may be established, requiring the outputs of many modules to be collected and correlated simultaneously. If the time constants of these intermodule feedback loops are within an order of magnitude of the computer's basic cycle period (which is not unusual in real-time systems), and if more than one computer is involved or if electromechanical input/output devices form part of the feedback loops, then the system begins to take on non-deterministic properties. As a result, many tests may be needed to determine the statistical properties of the system.

An integration testing problem typical in embedded computer systems (where the computer is an integral part of a much larger electrical and mechanical sysem) is the need to support the computer system's external interfaces. A computer program that, for instance, controls frequency selection in a special-purpose radio receiver could be tested with the actual equipment under laboratory conditions. If the program's peripheral devices are the wing flaps of a new aircraft or the steering mechanism of a guided missile, integration testing with the real equipment is not feasible, at least not until a very high level of confidence has been established. Simulation must therefore be used to provide the program with realistic response data at its external equipment interfaces. Interface simulators can consist of dedicated hardware, software/hardware hybrid combinations, or software. Software simulators become most cost effective as the complexity or the probability of modification to the simulated interface increases.

Maintenance

Although software does not wear out, characteristics analogous to aging can occur in large software systems. One major factor is that computer programs are very sensitive to "drift" in the external environment. For instance, an electronic subsystem can be expected to show little change in performance if its 115-volt power supply is replaced by an improved 120-volt model. If this change is not explicitly anticipated in its design, a 257-bit serial message sent to a program expecting 256 bits could cause a complete failure. Almost any system interface change requires corresponding maintenance changes in the software.

After a new system has been in operation for some time, functions are often added that were not anticipated in the original performance specification. Frequently, software changes rather than hardware field modifications offer the most economical way of adding functions. Whenever functional enhancements are added or when interfaces are changed, it is desirable to verify that none of the original capabilities of the system has been adversely affected. This verification is particularly important for older systems in which the person making the changes is not the original designer and therefore may overlook subtle interactions between the new function and the original design. The software tools to support such regression testing are similar in nature to, but more formal than, those used in integration testing. A selected group of input test data sets may be retained from integration testing, catalogued, and used whenever system changes are made.

Undetected errors in the original design are another source of software maintenance. It is almost never possible, particularly with larger systems, to test every permutation of input and output conditions. When an error resulting from an extremely rare combination of inputs is detected, the external conditions that induced it need to be reproduced in order to isolate the location of the design fault and then to verify that the corrective action was effective. A programmable test driver can be used in cases where the specific external conditions are difficult to reproduce. An economical approach is to add new test data sets to those already developed for unit and integration testing.

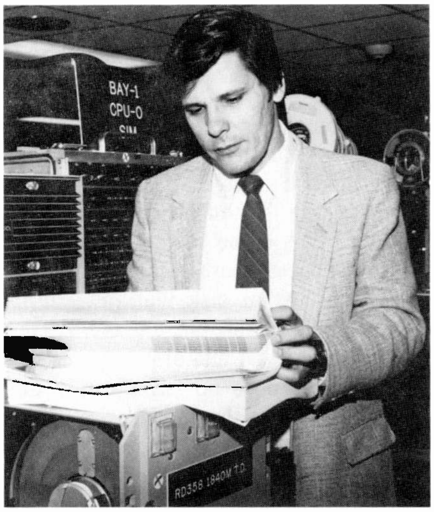
Configuration management ensures the reliability of large software systems over long periods of time. It provides formal, well-documented procedures for making permanent changes to a software system and maintains cumulative historical records of all preceding changes. Configuration management is especially necessary for multiple-site systems, where the accumulation of site-specific changes to a once-common software baseline can lead to an unmanageable array of diverging and ultimately incompatible baselines.

Software configuration control is often implemented by controlling the files that contain source code statements from which the executable program is generated. Specialized file-management programs are often used to ensure compliance with approval, documentation, and formatting policies. Generally, such programs are based on a master copy of the original baseline, which is maintained in a read-only file. When an approved change is to be made, the master copy is not edited, but rather a "corrector file" is prepared that specifies the lines to be changed, inserted, or deleted. The configuration management program then makes a "work copy" of the master file, reads the corrector file (or files), and executes the specified editing changes, all of which results in a "new version file."

Usually, each line of the corrector file has a field reserved for an official change number that is traceable to a document describing the problem, explaining the proposed solution, and containing the necessary review and approval signatures. This change number is preserved in the text of the new version file. A utility program scans these numbers and produces a summary list of all changes that are present in a particular version. Such a list allows a maintenance programmer to analyze a particular version at a particular site and determine which changes have or have not been implemented. It also provides a concise reference to more detailed documentation of these changes so that earlier changes, which were subsequently shown to be incorrect or for which better solutions are available, can be easily located and removed.

Conclusion

Software tools can reduce cost and improve quality at every stage of the software development cycle. Their selection and use should be an early, integral part of the project planning process. The application and evolution of software tools is becoming one of the key issues in software engineering because the maximum practical size and complexity of any system is limited by the reliability of its components. Software tools extend that limit by providing uniformity and reliability for the development of larger and more sophisticated systems.



George Suhy is Unit Manager, Simulator Systems, in the Software Design and Development Activity at Moorestown; he is currently responsible for the AEGIS Interface Simulator System. After joining RCA in 1973 he worked on a variety of system software projects before his current assignment on the AEGIS Program. He received the BSEE degree (1970) and the MS in Systems Engineering (1975) from the University of Pennsylvania. He is active in the Computer Society of the IEEE and the Association of Old Crows.
Contact him at:
RCA Missile and Surface Radar
Moorestown, N.J.
TACNET: 224-2542

H.L. Resnick

Experience with life-cycle software development

What is the ideal way to manage large-scale software development projects? The author details a two-phase plan that RCA Automated Systems has employed with success.

Continued success in a software development effort depends upon effective coordination and instruction of the programmers. To achieve success, the team must use a blend of development methodology, software tools, and progress-tracking techniques. This paper discusses the software-development approach used in the Vehicle Test Systems Engineering Section of RCA Automated Systems.

Experience dictates that the software development effort should be divided into two phases. The first phase, or the planning phase, establishes the software system requirements, the high-level system design, and the subsystem software requirements and program team sizes. The second phase, the design phase, invokes the life-cycle development methodology to guide programmers in design, code, and test. A strong reliance on several techniques—such as top-down design-concurrent documentation, thorough design reviews, compiler-verifiable interface specifications, and progress tracking—directs and gives structure to the programming effort.

Abstract: *The software development approach used in the Vehicle Test Systems Engineering Section of RCA Automated Systems forms the basis for this article. A blend of development methodology, software tools, and progress-tracking techniques go into the effort, which happens in two phases—a planning phase and a design phase. This article examines the various aspects of these two phases, giving a clear overview of the process of software development.*

Final manuscript received October 3, 1983.
Reprint RE-28-6-4

The planning phase

The first step in software development is the establishment of the planning phase as shown in Fig. 1. This phase begins with a definition of system requirements and concludes with a structure that allows the effective addition of programming teams to the project.

System definition is the first step, and a complete job must be done to avoid frustrating iterations. Within system definition, attention is focused on the system as a whole. Based on the perceived system requirements, a high-level design is performed. This begins with a description of the system functions, interfaces, and performance. High-level block diagrams must be included. System definition concludes with an allocation of the specified functions to hardware and software. The description of software functions then becomes the input to the next step, software planning.

The first software-planning task is a determination of software requirements. Requirements analysis forms the foundation of software development. It provides a basis for costing, a guide to design, and a reference for testing. In this life-cycle context, it is extremely important that each of the generated requirements be testable, unambiguous, quantitative, and bounded. Examples include statements of the total number of simultaneous users, the maximum response times to possible conditions, and the restrictions based on hardware limitations.

A useful technique for augmenting requirements analysis is the construction of data-flow diagrams. As illustrated in Fig. 2, these diagrams show the flow of data from sources to sinks through the system. Software modules are generally shown in bubbles, and each bubble can be broken down to a lower, more detailed level. These diagrams lead to an increased understand-

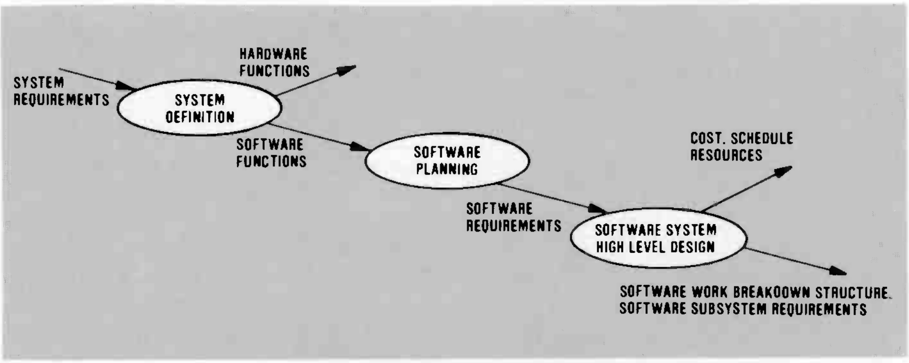


Fig. 1. *The planning phase begins with a definition of system requirements and concludes with a structure that allows the effective addition of programming teams.*

ing of what the software is required to do within the total system environment.

Once the software requirements have been generated, one must start listing the actual software components that will satisfy the requirements. This is most effectively done in the form of a work-breakdown structure.

The software portion of the work-breakdown structure is a hierarchical decomposition of the development effort. Its purpose is to organize, define and graphically portray all the software tasks to be accomplished. This becomes the cornerstone of the development effort. The work-breakdown structure forms the parameters of what is to be developed, and it will remain relatively static for the remainder of the project. An example of the first level of a software work-breakdown structure is shown in Fig. 3.

The best rule to follow in developing a work-breakdown structure is to develop as much detail as possible (thus increasing the accuracy of size estimates). Detailed structures can increase understanding, lessen the impact of a faulty estimate of any individual items, and make it less likely that an item can be missed.

Another rule is to generate a list of hardware and software design decisions that may independently change. Each of these design decisions will then be encompassed by a work-breakdown element, and a change of any individual design decision will only then affect a specific software element. For example, given the implementation shown in Figs. 2 and 3, any module that needs to reference the data base will use the file handler. Thus, only the file handler need be concerned with the structure of the data base, and any change to the data-base format will involve a corresponding change only to the file-handling subsystem.

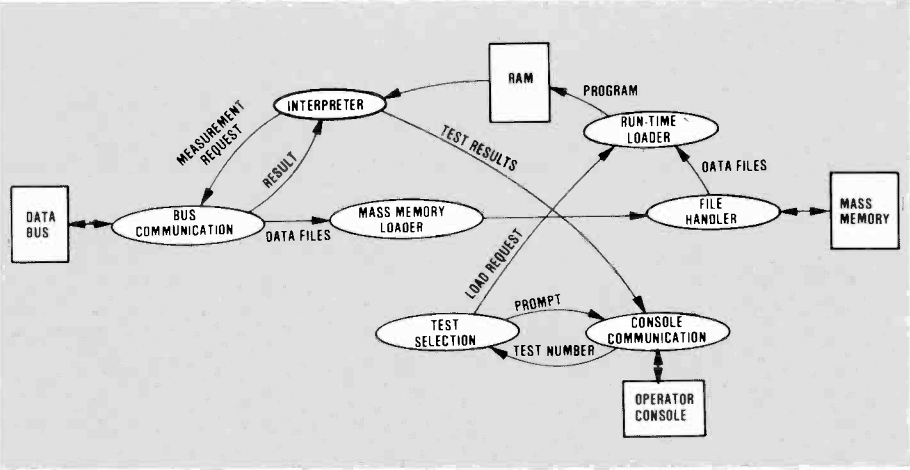


Fig. 2. *Data flow diagrams lead to an increased understanding of what the software is required to do within the total system environment.*

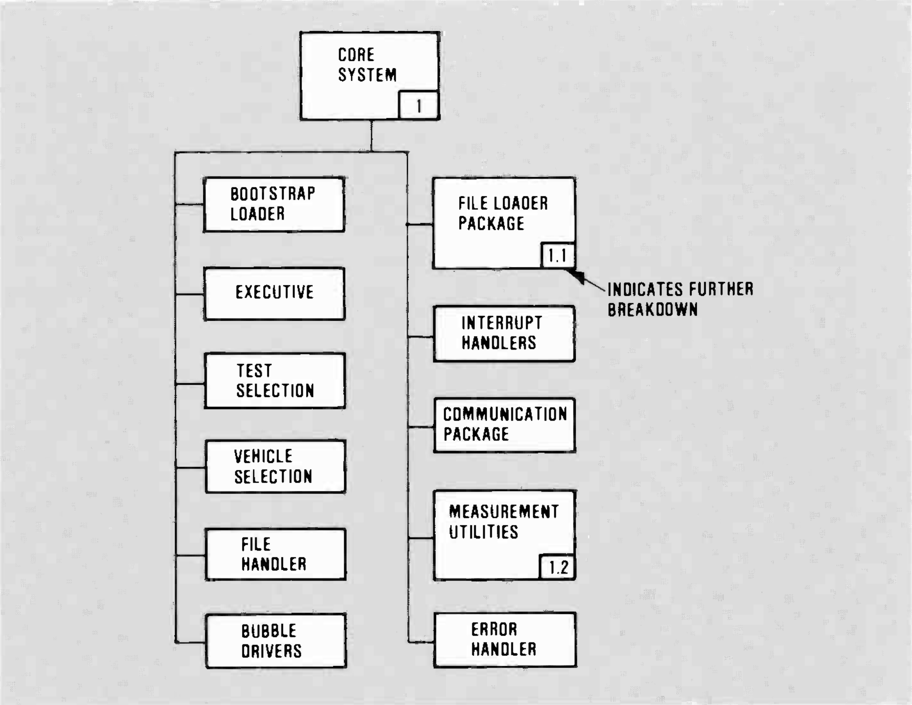


Fig. 3. *The software work breakdown structure organizes, defines, and graphically portrays all the software tasks to be accomplished.*

The work-breakdown structure is tied to the requirements through a cross-matrix that lists individual requirements in the vertical dimension and work-breakdown structure elements in the horizontal dimension. For each requirement, a mark is put under each work-breakdown element that is needed to satisfy the requirement.

Once the work-breakdown structure is complete (through a combination of actions that include satisfying the system requirements and performing segmentation based on design decisions), the next step is to size each element within the structure. Sizing must blend information—from available literature, past experience and expert opinion—into an estimate for each module identified in the work-breakdown structure.

Once the program size is estimated, the scheduling and manpower requirements for each subsystem can be planned. An outline of the requirements of each subsystem and a preliminary specification of how it interfaces with the rest of the system must be provided to each programming team in an efficient and organized manner. This concludes the planning phase of the project.

The design phase

The design phase demands guidance for each programming team in the following areas: requirements refinement, high-level design, detailed design, code, and unit test. The final step is integration test, which may be performed within a subsystem by the associated programming team. However, integration between subsystems is performed by an independent team.

The keys to success in this phase are the production of concurrent documentation and the monitoring of progress. The documentation described in Fig. 4 is a recommended requirement for each programming team. All documentation should be maintained on a computer for easy access, update and control.

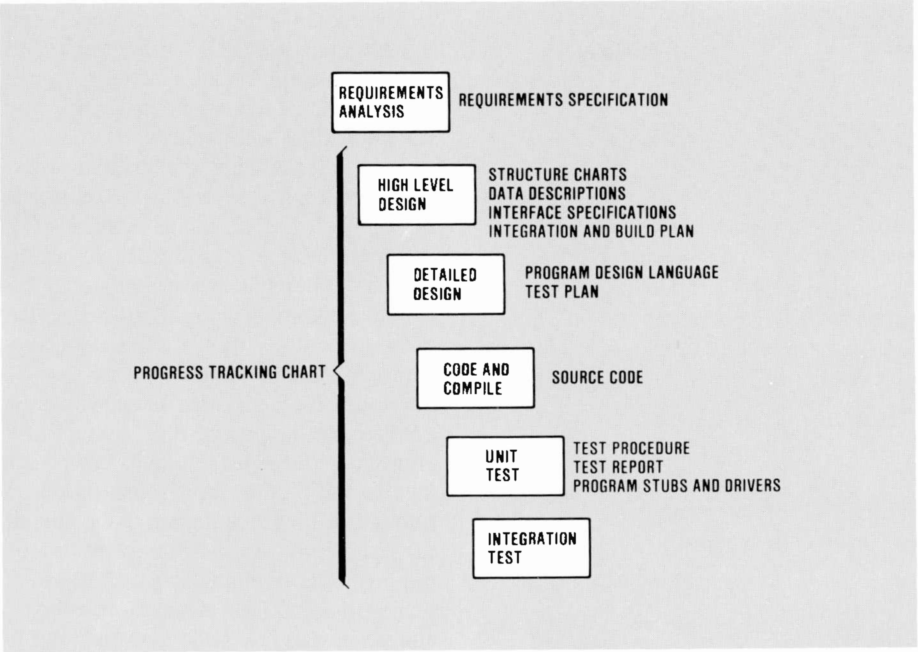


Fig. 4. *A key to success in the design phase is the production of concurrent documentation.*

The documentation produced during each portion of the design phase is discussed using specific examples. An important tool for the software manager is the progress-tracking chart, which encompasses the entire design phase. An example of this chart is shown in Fig. 5.

This chart is created during the high-level design step and has an entry for each program element (procedure, function, or subroutine). It follows a top-down order, with indication given if more development effort is required for any element. Once all the indicators are removed, the identification of program elements for this subsystem is complete.

Each element has an entry for the following milestones and items:

1. *PDL (Program Design Language) Complete:* The date that the designer completes the Program Design Language description of the element.
2. *PDL Accepted:* The date that the PDL is approved in a design review.
3. *EST/ACT Lines and Bytes:* The estimated (before coding) and actual (after coding) lines and bytes of code in the element.
4. *Code Complete:* The date that the code is compiled without errors.
5. *Unit Test Complete:* The date that the coder completes a unit test on the element.
6. *Integration Test Complete:* The date the integration team completes integrating this element to all other relevant system elements, inside or outside the subsystem.

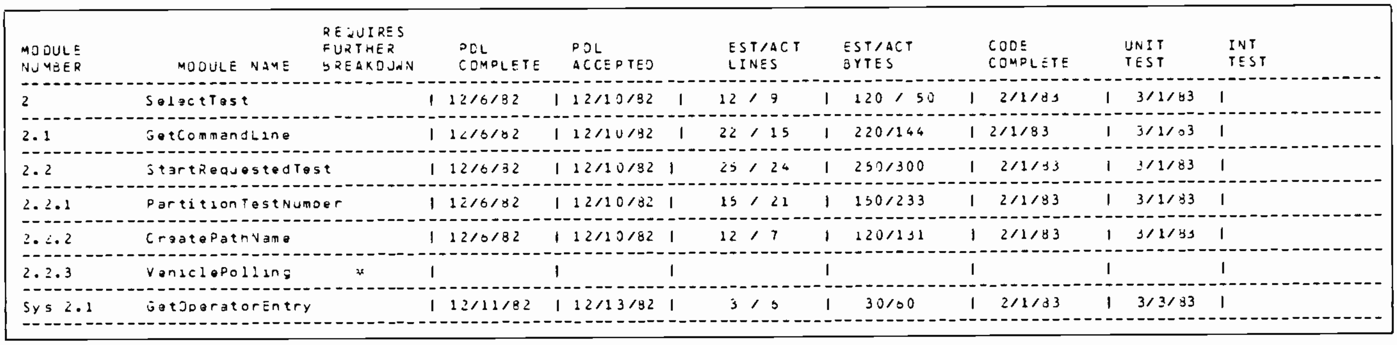
The complete set of progress-tracking charts is the software manager's roadmap to the system and the status of each subsystem. Examples of documentation associated with each of the steps in the design phase follows.

The high-level design step provides four forms of output: structure charts, data descriptions, interface specification files, and an integration and build plan. A sample structure chart is shown in Fig. 6.

A unique number should be assigned to each subsystem, thereby beginning the element numbering, and the structure charts should be developed in a top-down manner. These charts contain the calling hierarchy and data passing. They provide a high-level view of the subsystem. It is recommended that each element be named with a highly descriptive label rather than a short mnemonic. This may entail extra typing effort, but it clarifies the function of each software module.

Any data elements that show on the structure charts are described in detail, preferably using a non-ambiguous method (such as a definition that conforms to the Pascal-type syntax).

The interface specification provides controlled entry into the subsystem and "read-only" access to global information. The specification is helpful if the software is being written in a language such as ADA or certain Pascal implementations that allow the separation of programs into specification parts and body parts. The specification file defines all possible entry points into the software subsystem and the exact types and order of parameters that are passed. In this way, all programmers know how to call procedures within that subsystem, and they can then use the language's INCLUDE statement to add the specification file to their compilations. The compiler will then automatically check to assure that each external procedure is being called correctly. The specification developer also compiles with the specification file, allowing the compiler to assure that the interface is correctly specified. This compile-time error checking virtually eliminates interface problems during integration testing. Interface specification files are used by the programming teams even if the compiler



MODULE NUMBER	MODULE NAME	REQUIRES FURTHER BREAKDOWN	PDL COMPLETE	PDL ACCEPTED	EST/ACT LINES	EST/ACT BYTES	CODE COMPLETE	UNIT TEST	INT TEST
2	SelectTest		12/6/82	12/10/82	12 / 9	120 / 50	2/1/83	3/1/83	
2.1	GetCommandLine		12/6/82	12/10/82	22 / 15	220/144	2/1/83	3/1/83	
2.2	StartRequestedTest		12/6/82	12/10/82	25 / 24	250/300	2/1/83	3/1/83	
2.2.1	PartitionTestNumber		12/6/82	12/10/82	15 / 21	150/233	2/1/83	3/1/83	
2.2.2	CreatePathName		12/6/82	12/10/82	12 / 7	120/131	2/1/83	3/1/83	
2.2.3	VehiclePolling	*							
Sys 2.1	GetOperatorEntry		12/11/82	12/13/82	3 / 6	30/60	2/1/83	3/3/83	

Fig. 5. *The progress tracking chart has milestone entries for each program element.*

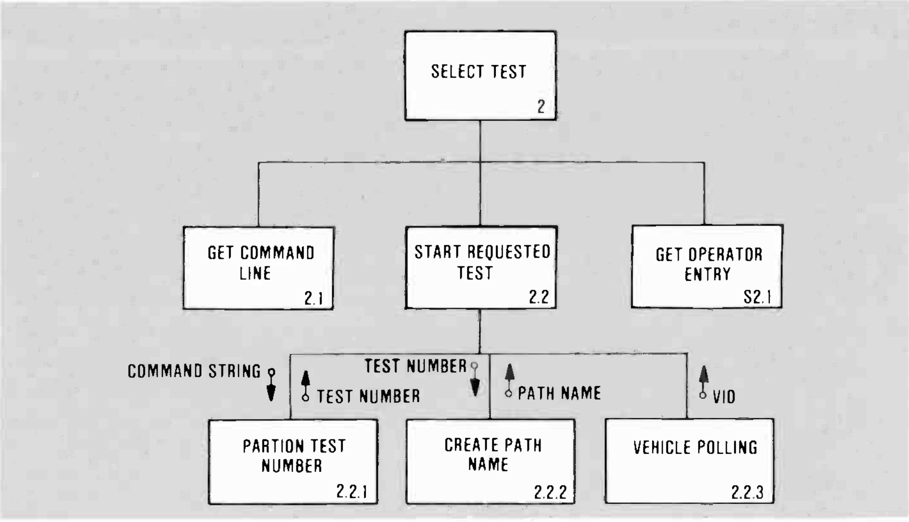


Fig. 6. *Structure charts illustrate calling hierarchy and data passing in a top-down manner.*

```
{Interface Specification for Test Selection Subsystem}
  PUBLIC TestSelection;
      {Public type definitions for users of test selection routines:}
         TYPE      VehicleRange = 0..999;
                   RString = Record
                           L:     Integer;
                           C:     Packed array 41 of char;
                   END;
      PROCEDURE SelectTest;
         {This is the invocation of the test selection subsystem. It will prompt the
         operator, accept a command line input, and initiate the requested test.}
      PROCEDURE GetOperatorEntry (VAR CommandLineCopyBuffer: RString);
         {This procedure returns a copy of the command line string and its length.
                   INPUTS:     None
                   OUTPUTS:  Command line record}
      PROCEDURE ReadVehicle (VAR VID: VehicleRange)
         {This procedure returns the value of the most recently entered vehicle ID.
                   INPUTS:     None
                   OUTPUTS:  Vehicle ID}
```

Fig. 7. *The specification file defines all possible entry points into the software subsystem and the exact types and order of passed parameters.*

does not accommodate them. A sample specification file for a subsystem is shown in Fig. 7.

The integration-and-build plan lays out the order in which system modules will be built and integrated. This plan must assure an effective system integration through a build-a-little test-a-little strategy, keeping down the potential for simultaneously adding many new problems. Test criteria accompany this integration plan. A sample portion of an integration and build plan is shown in Fig. 8.

The detailed design step uses an abstraction tool that is easily maintained on a computer. Program Design Language (PDL) seems to be the most useful tool. Our experience has shown that a two-step PDL design is most useful (high and low level). The high-level PDL is an English-like easily-readable structured description of the program element. After review, it should be refined into a low-level PDL that clears any ambiguities and is closer to the actual code. The high-level PDL is retained as permanent documentation while the low-level PDL is reviewed and further refined to actual code. Figure 9 shows examples of high-and low-level PDL descriptions.

The coding is the easiest step because it is built upon the PDL and specification files. When compilations are complete, sizes can be checked to assure compliance with size estimations and budgets.

Unit test examines the internal workings of each software element as completely as possible. Test cases are established to exercise important logical paths within the program, concentrating specifically on boundary conditions of control structures. Because a module is not a stand-alone program, driver and/or stub software must be developed for a unit test. In most cases, a driver is nothing more than a main program that establishes test data, passes such data to the module, and prints results. A stub is a dummy subprogram of the module that may print verification of entry, do minimal data manipulation and return. A menu-driven driver program is effective and allows easy selection of which unit to test and what sort of inputs to provide.

Integrating subsystems

Integration is the process of putting the modules together into subsystems and the subsystems together into the system. The integration-and-build plan produced during the high-level design step is the guide for the build-and-integration strategy and test criteria. The programming team performs the integration within the subsystem, but integration between subsystems is performed by a team with no direct involvement with the subsystems. This integration team treats each system as a black box with no "conflict-of-interest" problem. Any discrepancies found are returned to the design team for correction.

The integration test strategy must be based on key functional "threads of execution." These threads are derived from a review of system structure charts. Top-level pieces, which provide a view of the preliminary version of the system, should be integrated through to the most critical low-level modules. For example, based on the integration and build plan of Fig. 8, the test-selection module can be incrementally integrated with the file handler, loader, and interpreter through to the mass-memory driver. Additional pieces can be integrated incrementally in whatever manner seems most meaningful. High-level pieces are added for user visibility, while low-level pieces are added for test of hardware-critical features.

Experience has proven that a computer or development system, using special dummy software to simulate hardware components, is the best tool for implementing the integration strategy. The result is an integrated system that loads easily into the hardware and continues to exist on the computer to provide a test environment for further software development and maintenance.

Summary

The use of state-of-the-art software development methodology combined with well-structured documentation and effective progress tracking allows a software manager to retain effective control of a large software project. By dividing the project into two phases—planning and design—the appropriate number of programmers are added in a well-coordinated manner, minimizing resource waste. Concepts such as these, formulated and refined through project experience, must form the foundation for software-development practices in the future.

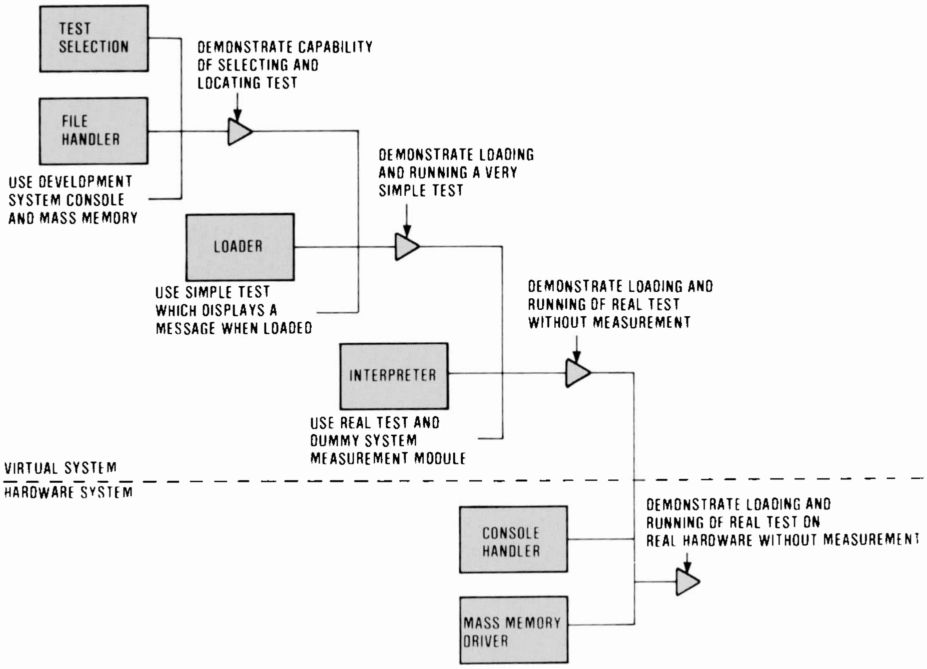
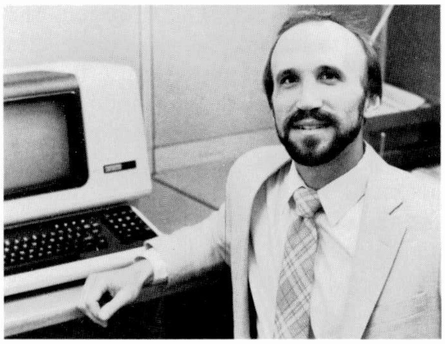

TEST SELECTION
DEMONSTRATE CAPABILITY OF SELECTING AND LOCATING TEST
FILE HANDLER
USE DEVELOPMENT SYSTEM CONSOLE AND MASS MEMORY
DEMONSTRATE LOADING AND RUNNING A VERY SIMPLE TEST
LOADER
USE SIMPLE TEST WHICH DISPLAYS A MESSAGE WHEN LOADED
DEMONSTRATE LOADING AND RUNNING OF REAL TEST WITHOUT MEASUREMENT
INTERPRETER
USE REAL TEST AND DUMMY SYSTEM MEASUREMENT MODULE
VIRTUAL SYSTEM
HARDWARE SYSTEM
DEMONSTRATE LOADING AND RUNNING OF REAL TEST ON REAL HARDWARE WITHOUT MEASUREMENT
CONSOLE HANDLER
MASS MEMORY DRIVER

Fig. 8. *The integration and build plan lays out a build-a-little test-a-little strategy.*

```
MODULE NAME: GetCommandLine
  High-Level PDL:
    REPEAT Forming the Command Line
      Clear the buffer
      Prompt the operator
      REPEAT getting legal keys and loading buffer
      UNTIL key is a terminator
    UNTIL key is not a 'CLEAR'
  Low-Level PDL:
    REPEAT Forming the Command Line
      Initialize the command line buffer length to 1
      Display 'ENTER TEST NUMBER' on the Communicator
      REPEAT getting command line keys
        IF the key is a digit and the maximum number of digits has not been
        received
          THEN Place the digit into the command line buffer
             Echo the digit
             Increment the count of total digits received
        ENDIF
      UNTIL the key received is a terminator key
    UNTIL the key received is not 'CLEAR'
    Place the terminator key into the command line buffer
```

Fig. 9. *Program design language is an abstraction tool for detailed design that is easily maintained on a computer.*



Herb Resnick is an engineering scientist at Automated Systems, Burlington, Massachusetts. He received a BSEE and MEE from Cornell University and is pursuing a MS in Software Engineering at Boston University. Since joining RCA in 1974, he has been involved in hardware and software design of automatic test equipment. As a project leader, he currently is involved with software architecture and methodology in Vehicle Test Systems.

Contact him at:
RCA Automated Systems
Burlington, Mass.
TACNET: 326-5422

P.G. Goodwin

Autonomous redundancy management on the DMSP spacecraft

On-board computer control permeates most aspects of the DMSP's operations.

The Defense Meteorological Satellite Program (DMSP) instituted a major program to increase reliability and prolong on-orbit life of the new Block 5D-2 spacecraft. The goal was to eliminate single-point failures and to maximize fault tolerance through means of an on-board autonomous redundancy control strategy. A major new software function was developed to perform automatic fault detection and to manage the reconfiguration of redundant hardware and software.

Because ground-station contact time is limited, autonomy from ground intervention is a primary consideration for mission success. Due to the complexity of the redundant hardware, the control system must use a symptomatic approach to fault isolation and also must allow a manual override capability. This redundancy management software design and implementation has been subjected to a rigorous test program. Examples of the on-orbit experience show that hardware redundancy and software monitoring have allowed faults while extending the overall life of the satellites.

Abstract: *The author describes the DMSP spacecraft and its mission. Software control permeates the operations, because the DMSP's low-earth orbit allows little time for ground communication and because software control minimizes problems due to ground interruptions. The author introduces redundancy management schemes after describing the subsystems and the software. On-orbit experience has shown the value of the software controL*

Final manuscript received September 28, 1983.
Reprint RE-28-6-5

Mission and system description

The primary mission of DMSP is to provide visual and infrared global weather data to tri-service users, and to give tactical weather data to selected areas around the globe. Special meteorological and atmospheric sensors are carried on various missions as well. The satellite is three-axis stabilized, in a sun-synchronous, 450-nautical-mile orbit, with a 98.7° inclination. The ground segment consists of a Command and Control Center at Offutt Air Force Base in Omaha, Nebraska, with Command Readout Stations at Loring Air Force Base in Maine and Fairchild Air Force Base in Washington state, all operated by the 1000th Satellite Operations Group of the Air Force Space Command. The Global Weather Central (AFGWC) in Omaha receives and processes orbital stored data and provides data to the Fleet Numerical Oceanography Center in Monterey. Mobile ground vans and aircraft carrier terminals receive direct readout of data when the spacecraft is within their line of sight. Figure 1 is an artist's rendition of the satellite and the continental ground facilities.

The DMSP spacecraft weighs 1650 pounds on orbit. Most aspects of operations are controlled through on-board computers. A 128-square-foot solar-array panel is driven at an average rate of once per orbit with the computer providing the tracking-control signals. Attitude determination and control are provided through a complement of equipment including the earth sensor assembly, inertial measurement unit, solar aspect sensor, celestial sensor assembly, reaction wheel assemblies, and momentum torquing coils. Primary system pointing accuracy is 0.01 degree per axis with backup system accuracies better than 0.1 degree. All communications links are S-band with a space-ground link subsystem (SGLS) compatible command link. The primary payload is government-furnished equipment, an Operational Line Scanner, which works in both the visible and infrared bands and provides complete coverage of the earth each day.

The first DMSP Block 5D-1 satellite was launched in September 1976. The design lifetime was 18 months; however, the third spacecraft in that series completed 5 years of operation on May 1, 1983. The first of the new Block 5D-2 series was launched December 21, 1982. With expanded redundancy over the Block 5D-1 design, the design life of this series is 3 years.

Need for autonomy

To obtain an acceptable probability of survival over the required lifetime, on-board redundancy is required. This on-board redundancy requires autonomous management for several reasons.

1. The low-earth orbit time of 101 minutes provides only about 5 to 15 minutes of communications contact with ground stations. Even with the addition of a data-readout station at Kaena Point, Hawaii, up to two orbit-contact "blinds" can occur. Thus, to provide the uninterrupted operational mission, continu-

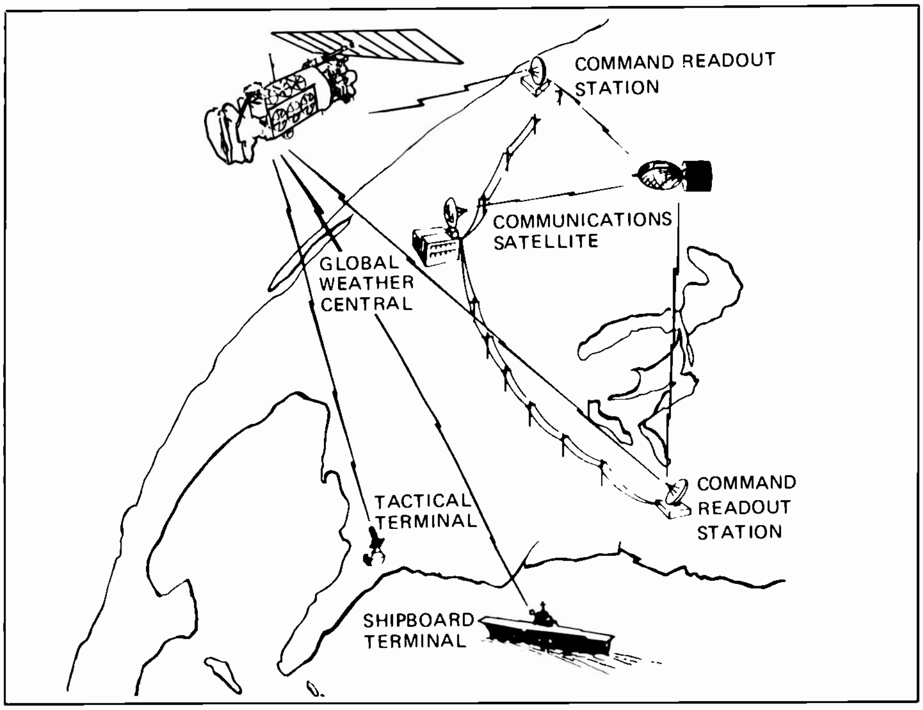


Fig. 1. *The DMSP system. Real-time weather data are transmitted to mobile vans and shipboard terminals. Stored orbital data are collected and processed by Global Weather Central.*

ous on-board monitoring of satellite health is required.

2. Autonomous redundancy management also simplifies ground-station operations. The need for high engineering skills among around-the-clock crew members is overcome. More time is allowed for data retrieval and sensor management rather than spacecraft bus monitoring. Engineering personnel are required for analysis after an anomaly has occurred.
3. The keystone for spacecraft survivability must be autonomous control and redundancy management. In a crisis environment, ground contact and control could be interrupted for long periods, while continued spacecraft operational support would be required more than ever.

Subsystem redundancy and selection

The descriptions below are of those spacecraft subsystems and hardware units that provide protection against at least single-point failures. These units fall into the categories of command and control, attitude determination and control, and power. In some cases, redundancy switching is done automatically with failure-detection circuits; in others, switching is done by failure-symptom detection in the flight software. All units may be selected by ground command, bypassing the internal detection techniques.

Command and control subsystem

Standard Controls Processor. The DMSP spacecraft is controlled through a redundant pair of Standard Controls Processors (SCPs). Each SCP is a general-purpose data processor operating in fractional, fixed-point, 2's complement arithmetic. The SCP with its resident software is a major component of both the Command and Control and the Attitude Determination and Control Subsystems. Each SCP contains 28K of 16-bit-word read/write memory with a 256-word bootstrap ROM.

Each SCP is loaded with identical flight software and is capable of providing full mission-operational capability. The first two 8K boards are switchable by ground command to provide redundancy for those locations used by microprogram control ROM. The selection of an SCP as the primary computer can be done by ground command or by fault detection in the software. Full processing must cycle through every half second and report to a hardware watchdog timer. If, for some reason, it does not do so for four consecutive cycles, the detector marks the SCP as potentially bad and allows the backup unit to take over.

Controls Interface Unit. The Controls Interface Unit (CIU) contains all the input/output (I/O) circuitry for the SCP. The I/O bus is fully redundant and may be cross-strapped with the two SCPs. The selection of a bus as primary can be done by ground command or by redundancy management software, described below.

Although there are redundant buffers for external devices, such as the earth sensor, data can usually only be transferred on the bus designated as the Control Bus. The SCP on the Control Bus is designated the Control SCP.

The CIU also contains special circuitry that converts reaction-wheel input data to digital words and SCP-torque output words to analog drive voltages for the wheels. These circuits are redundant and are operational on both buses. In addition, the CIU houses a pair of ±10-volt power converters and two uplink command processors that decode uplink commands and pass them on to the SCPs. These units are permanently connected to a given SCP.

Attitude Control Subsystem

Attitude Determination and Control Hardware. Redundancy is provided in all areas of subsystem units used for attitude determination and control. The Inertial Measurement Unit (IMU) has separate redundant units for its ac power supply, dc power supply, and logic electronics. Four rate-integrating gyros are provided, including three orthogonal units and a skew gyro that may be substituted for any one of the three orthogonals. An Earth Sensor Assembly (ESA) has fully redundant electronics. The ESA is a static infrared horizon sensor with four independent quadrant detector sets, only three of which are required.

Three-axis stabilized control is affected by orthogonal Reaction Wheel Assemblies (RWAs). A fourth, skew RWA may be substituted for any one of the three orthogonals. Momentum buildup in the RWAs is unloaded through magnetic torquing coils interacting with the earth's field. Separate coils are provided for the pitch axis and the roll/yaw axis, and each contains a redundant set.

Attitude Control Modes. The primary attitude determination and control mode is a system using the strap-down gyros and a celestial sensor to give control to 0.01 degree. This software system executes in one SCP while monitoring software runs in the backup SCP using the ESA for backup attitude determination. Should the primary system fail and an error exceed 0.4 degree, the backup SCP takes over and controls the spacecraft in the backup attitude mode to better than 0.1 degree per axis. Should one SCP fail, the single remaining SCP can run the primary system and the backup monitoring software together.

The celestial sensor has two sets of star transit slits. Upon failure, either set may be turned off, allowing the second set to determine star sightings. Should the sensor fail altogether, the backup system can control the satellite continuously. In the backup mode, roll and pitch attitude are controlled using the ESA. There are then also redundant modes for yaw control. Initially, yaw is controlled in a gyrocompassing mode using a derivative of the roll error and the normal orbital pitchover rate. After filter convergence, a nominal mode can be selected whereby yaw is controlled by the gyros and once-per-orbit sun sensor updates. If the sun sensor should fail, the software automatically reverts to the gyrocompassing mode. Thus, redundant computers, redundant attitude determination and control modes in the software, as well as redundant hardware, provide highly reliable and autonomous attitude control.

Power Subsystem

Solar-array orientation and power transfer are accomplished by the Solar Array Drive (SAD) and the Array Drive Electronics (ADE). The ADE contains a fully redundant set of electronics under software control. The array is driven about the pitch axis at a nominal rate of 360 degrees per orbit. Shaft position information from a two-pole synchro is passed to an SCP, which then computes positional errors and derives a rate-control signal. The power system also contains a switchable redundant 5-volt power converter required for operation of the transmitters.

Error detection and switching by the software

The basic design goal throughout the spacecraft was to avoid single-point failures. That is, no single failure of any part should cause a mission failure. However, there was no hard requirement to survive double failures (although there are many multiple failures that are survivable). In addition, in the design of the redundancy switching, a decision was made to automatically switch-in redundant units upon detection of a failure, but to try not to switch back to the primary unit later. It would be left to the ground-station operations personnel to test, determine if the original side was in working order, and reselect it if so desired. In many cases, there was the option of declaring the original side as backup and the newly selected side as the primary. This designation is strictly for purposes of redundancy management; the two sides of a redundant subsystem are usually identical. The caution in doing this is to ensure that the original side really is working and to be convinced that the switchover was really caused by some other anomaly not related to that subsystem.

Status flags are provided in software telemetry to indicate when a unit has been automatically switched. The structure of the design is such that these flags also inhibit these paths from being taken again if a failure symptom remains. Thus, to ensure redundancy-switching protection after the original side is reselected by ground operators, the failure-status flags must also be cleared.

When a unit is believed to have failed, the software selects the designated backup unit for active use. This could be the powering on of redundant electronics and turning off the indicated failed unit (IMU, ESA, ADE, RWA). In other cases the redundancy action is to direct the software and CIU to functionally use the redundant unit (magnetic coils, gyros, SCPs, I/O bus, ESA detector quads, attitude control modes).

If an anomalous symptom persists after the unit switch, the software then makes the assumption that the fault may not be in the subsystem unit, but could be in the I/O bus. Therefore, a control-bus transfer command is issued that places the control SCP on the new bus. Should the fault still exist, the software concludes that the software itself or its SCP is in error and takes that SCP off-line.

When the redundant SCP takes control, it starts the entire checking and switching process again, from the beginning. The hardware units that have been switched remain switched. The functional usage as directed by software will go back to the original side. This latter feature is due to the fact that the two SCPs do not communicate with each other, and the second cannot know that the first had taken any redundancy-switching actions.

The switching sequence concludes when the anomaly is corrected. It is left to the ground controller to command all good units switched during the sequence to their original sides and to reset the software status flags. If the anomaly is not corrected, the software will continue to attempt operation in the SCP last selected by ground command as primary.

Switching criteria

Given that the spacecraft design contained a full redundancy capability, a design study was undertaken to determine how best to use these features and on what basis the software should do switching. The hardware units were treated individually because of the uniqueness of their interfaces and data content. The rules for each were formalized, and a new redundancy management software module (REDMN) was designed to centralize decisions and control. Experience from the Block 5D-1 flights and ongoing test programs allowed updates and refinements. Modifications have been made based on the flight performance of the first Block 5D-2 satellite. Situations detected in the following areas are some of the fault indications that may cause redundancy changes.

Inertial Measurement Unit. The IMU interrupts the SCP at a 10-Hz rate, while the flight software executive (including REDMN) cycles at a 2-Hz rate. If there have not been exactly five interrupts per cycle for five consecutive cycles, the backup IMU electronics and power supplies are commanded on. If the data transfer buffer times out (data late) for five consecutive cycles, the electronics is switched or, if the SCP is in Monitor mode, the backup attitude mode is selected.

The attitude-control software requires data from any set of three of the four gyros. If at least three are not indicating that they are on, REDMN cycles through a process that commands them on, switching the IMU into backup, switching the control bus, and so on. After each Gyro-On command, a 60-second delay is inserted to allow the gyros to reach operating speed.

Earth Sensor Assembly. The ESA interrupts the SCP at an 8-Hz rate. The software checks that there have been exactly four interrupts per 2-Hz cycle and that all 12 detectors have been read. If either does not occur for five consecutive cycles, including buffer-data late timeouts, the ESA is commanded to use its backup electronics.

If a three-detector quad has unreasonable input data, that quad is marked bad. An alternate attitude algorithm is then called using only the data from the three good quads. Should a second quad look bad for five consecutive cycles, the backup ESA is selected. Further fault indications will cause the bus switch, and so on. However, after an ESA switch, a test-time delay of 150 seconds is invoked to allow the backup to warm up and provide valid data.

On-board ephemeris computations allow erroneous inputs caused by the sun or moon in the field of view to be disregarded.

Reaction Wheel Assembly. The software normally controls spacecraft attitude by torquing the three orthogonal RWAs. The wheels are continuously being tested to see if, for a given torque value, the change in speed is within bounds. If a wheel should fail this test twice in a row, the skew wheel is turned on and the failed wheel turned off. If less than three wheels are found on, the software turns all wheels on again.

Momentum Unloading. The RWA speeds are used as criteria for verifying the functioning of the magnetic unloading. If a wheel speed is too high after the spacecraft has just passed through a magnetic field unloading zone, then the corresponding magnetic coil is marked bad. At the next unloading opportunity, the backup coil is used by the software.

If the magnetic system is unable to maintain the momentum below threshold (normally 20 in-lb-sec on an axis), then the nitrogen thruster system is used to reduce and stabilize the momentum to less than threshold. The magnetics will continue to be used as long as it can maintain the momentum below threshold.

Array Drive Electronics. The software reads the ADE status and issues a speed/direction command every 0.5 second. If the SAD is in slew mode, but the error does not decrease for two sets of 60 times each, the ADE is commanded to backup by REDMN. If the input buffer reading times out, or if the input status is not the same as the output command for 10 consecutive cycles, a switch is requested.

Standard Controls Processor. As stated earlier, if a fault occurs after a subsystem unit has been switched, the software will switch the control bus and then the control SCP. In addition, the SCP and the flight software have self checks to determine their state of health. The software must fully cycle through all tasks each 0.5 second and report to the CIU watchdog timer. If it fails to report, the CIU hardware will switch that SCP out of control (off-line).

The SCP has hardware fault alarms for parity errors, transfer errors, addressing errors, and arithmetic overflow. Fault-recovery software tries to deal with these problems and take corrective actions. If it is unable to, the cycle reporting will then be withheld, allowing the SCP to be taken off-line.

Algorithm implementation

Figure 2 is the overall flowchart for the REDMN module. The individual unit tests are handled by appropriately named subroutines. The ESA and IMU tests are only run if the SCP is in control. The ADE and RWA tests are run if this SCP has access to the attitude-control functions. The bus switching and watchdog timer inhibit is common. Any of the functions can be disabled from the ground by clearing a bit in an enable flag. If there is an SCP hardware error alarm (such as a parity error), this code is bypassed. A special error handler saves SCP status and invokes various recovery routines depending upon the priority level of the error.

An example of a unit subroutine is shown in Fig. 3. The ESA has two tests: QCNTR is a counter of interrupts and QFFR is an indicator of bad quad data. These will then trigger a call to the subroutine ESACM shown in Fig. 4. Here the ESA is commanded to backup, or if ESASWINH is set indicating a prior switch, then a flag is set requesting that REDMN switch the control I/O bus. After the ESA is switched, the bad quad data counter limit is modified to check for 300 consecutive errors.

Similar type switching and requests for bus switching occur for the IMU, ADE, and RWA. Functional usage of backup devices such as magnetic coils and gyros is selected by the software in the corresponding unloading and attitude-determination modules.

In addition to the above functions in REDMN, a nitrogen-gas momentum-unloading module has been added for large disturbances, in which case thrusters are activated immediately if the total momentum of the spacecraft exceeds a limit. If need be (for example, if there's a leak in the nitrogen system itself) this unloading module will stay active until the gas is depleted. Except in an extreme worst case, system momentum should be low at the time of depletion.

Of the present 28K memory, less than 1K total is devoted to redundancy management functions. Normal execution time takes less than 0.5 millisecond of the 0.5-second processing cycle.

On-orbit experience

The experience of the DMSP program has shown the value not only of on-board redundancy management processing but of a reprogrammable computer itself. Four Block 5D-1 series spacecraft have been launched. Table I is a listing of some of the on-orbit faults (since September 1976) that were treated through automatic redundancy switching. While extending spacecraft life, the autonomous features also prevented any loss of mission data. As one might expect, there were a few cases of switching on indicated faults that later turned out to be either temporary failures or erroneous fault indications.

The computer functions and experience for the first two spacecraft are discussed in Reference 1. While not a part of autonomy, the computers have been reprogrammed many times. Usually modifications are made to enhance certain features or to avoid memory parity errors. Complete new attitude-control functions have been derived and implemented, including the recovery and despin of a previously totally dead spacecraft.[2]

The first Block 5D-2 spacecraft was launched in December 1982. There have been no permanent failures, but the redundancy management has been exercised sev-

Table I. *On-orbit faults corrected by redundancy switching on four DMSP Block 5D-1 spacecraft.*

Automatic redundancy switching and alternate attitude control modes that extended spacecraft life

- Roll-yaw coil relay
- SCP Arithmetic Logic Unit function
- Pitch reaction wheel
- Yaw gyro
- SCP parity errors
- SCP memory address error
- Receiver demodulator side A
- Celestial sensor south Atlantic anomaly (protons)
- Earth sensor glint

Indicated failures that caused precautionary redundancy switching

- Charge controller
- Yaw reaction wheel
- Array drive electronics
- Roll-yaw coil
- Pitch reaction wheel
- ESA side A

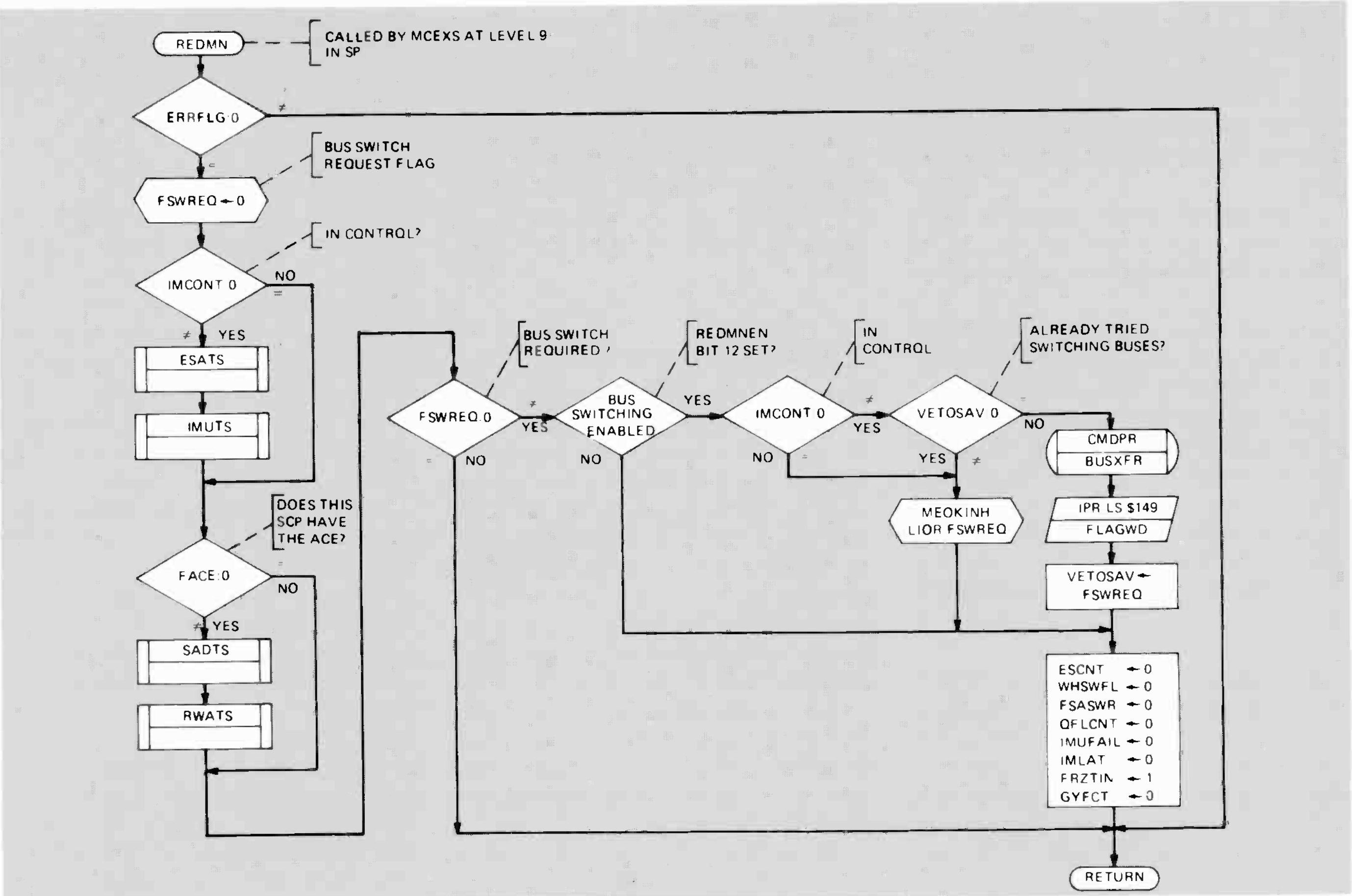


Fig. 2. *REDMN flowchart. Top-level module that provides subroutines to test and switch units and to switch buses or SCPs based on input flags.*

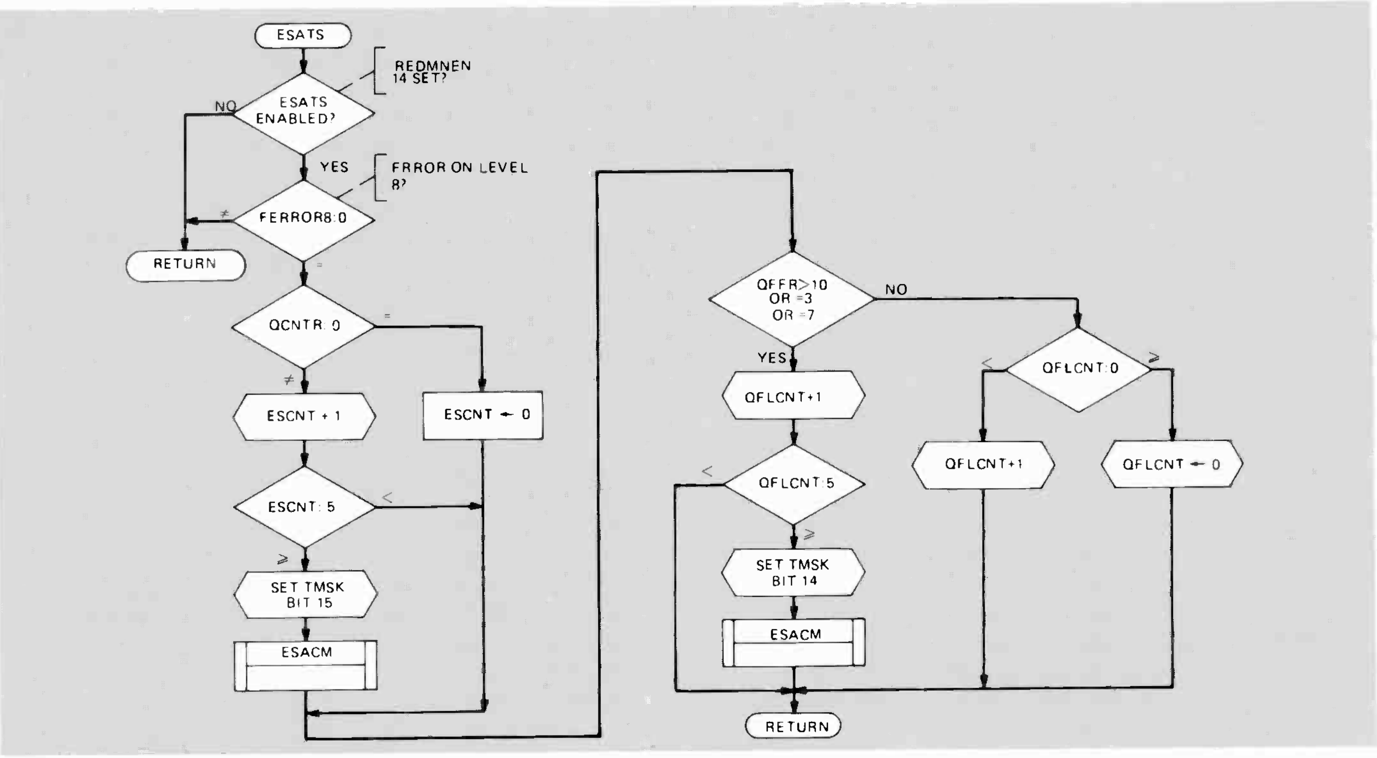


Fig. 3. *ESATS flowchart. A REDMN subroutine that tests for proper operation of, and inputs from, the Earth Sensor.*

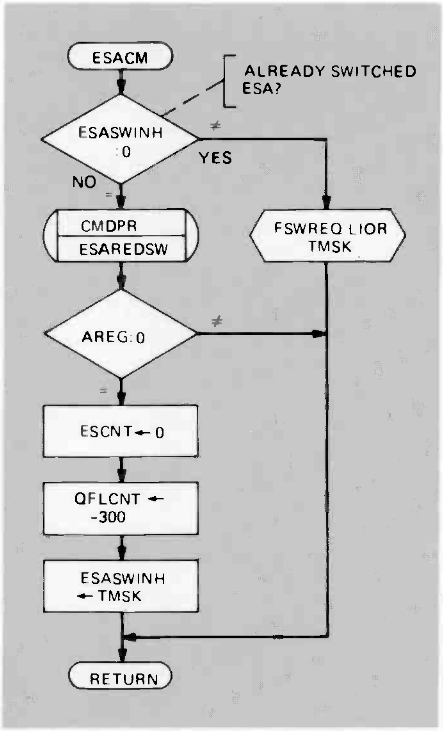


Fig. 4. *ESACM flowchart. An ESATS subroutine that issues the ESA-To-Backup command, or sets a flag to request a bus switch.*

eral times. The GN_2 unloading function was used during revolution 1 when there was a large disturbance probably caused by residual effects from the apogee kick motor. Precautionary switchouts of a magnetic coil, the ADE, and the roll gyro were made; these units have subsequently been put back into service. Continuing problems with the celestial sensor have caused the backup Attitude Control subsystem to take over. The problems are caused by proton bombardment and normally occur in the South Atlantic region. These proton hits appear as star transits and cause the primary system to become lost. The software has now been programmed to ignore all transits in this geographic region, thus allowing the primary system to retain control. Two occurrences of soft parity errors have appeared in SCP1 and, in one case, that SCP was caused to go off-line. These were repaired by a reload of memory while the redundant SCP maintained control.

Follow-on efforts

Various studies and design improvements are being undertaken to increase the spacecraft's lifetime and to upgrade the continuity of operational support. Increased autonomy is one of the keys to endurance of DMSP mission support. Therefore, the Air Force is authorizing full-autonomy studies for follow-on buys of Block 5D-2 satellites.

At present, additional autonomous functions are being developed. Power Management Software (PMS) has been implemented and will be uplinked to all the on-orbit satellites. This software will continuously compute the state of charge of the batteries, allowing a charge rate program to be used that will extend the batteries' life. In addition, it can power down the spacecraft in phases if the state of charge becomes too low.

Two reprogrammable computers make it possible to reprogram around a memory parity error in any one machine. If, after a total failure of one computer, a parity error should occur in the remaining SCP, that satellite would be lost. To circumvent even this double failure, software has been developed to provide redundant functional capabilities within a single machine. Called Redundant Attitude Determination and Control Software (RADACS), this subpackage actually contains a minimum command and control, sensor interface, and solar array drive, as well as a minimum attitude-control function. A parity error within the normal code area will cause an automatic jump into RADACS. The mission can continue uninterrupted while the first section of memory is under diagnosis and repair. This then, in effect, is equivalent to adding a third redundant SCP on the satellite.

Starting with the sixth Block 5D-2 satellite, the SCPs will have error-correcting memory using highly reliable CMOS/SOS LSI 4096X1 RAMs.[3] A modified Hamming code provides single-bit error correction and double-bit error detection. All on-orbit experience with parity errors have been single-event, single-bit errors. With this new memory, the on-board computers themselves should no longer be of any concern in the goal for total autonomy.

With increased memory size in the follow-on spacecraft, more can be done with the redundancy manager. The present design is a series sequence of switches—unit, bus, then SCP. An improved design could include a matrix operation whereby all combinations of equipment could be tried. For instance, after a bus switch, all hardware subsystems could be put back on their primary sides (except for those known to have truly failed). This type of management will be required for a truly autonomous spacecraft.

Conclusion

Use of redundant hardware and programmable computers with redundancy management software has improved the reliability and lifetime of the DMSP spacecraft. Present designs have shown the value of autonomous operations. Even more autonomy and reliability can be implemented to enhance survivability and ensure operational mission success.

References

1. G. A. Beck, R. F. Buntschuh, A. J. Aukstikalnis, "Adaptable Spacecraft Computer Systems Experience," presented at AIAA Computers in Aerospace Conference II, Los Angeles, Calif. (October 1979).
2. J. R. Staniszewski, "Lazarus Sleeps No More—Satellite Recovered from Tumble," *RCA Engineer* (August/September 1978).
3. K. K. Oey, S. Teitelbaum, "Highly Reliable Spaceborne Memory Subsystem," presented at AIAA Computers in Aerospace III Conference, San Diego, Calif. (October 1981).

Paul G. Goodwin, Manager, Software Systems Engineering at RCA Astro-Electronics, directs the activities of an engineering group responsible for the design, implementation, and test of spacecraft flight software. Under his direction, software has been developed for DMSP, the TIROS weather satellites, and the Direct Broadcast Satellite for the Satellite Television Corporation. Goodwin received a BSEE from Purdue University and an MSE from the University of Pennsylvania. He joined RCA in Camden in 1962 and transferred to Astro-Electronics in 1964.
Contact him at:
RCA Astro-Electronics
Princeton, N.J.
TACNET: 229-3352

R.L. Prenoveau

An ATLAS environment for testing units with embedded processors

Wide-ranging and necessary augmentation of an automatic test system depended on interface hardware, certain ATLAS language extensions, and a multitasking executive routine that simplified the test programming.

Typical automatic test systems for avionics and military electronics maintenance (for example, the RCA-manufactured EQUATE, which is Electronic Quality Assurance Test Equipment) use a minicomputer and are configured to enable analog and digital test capability. Test programs are usually written in an adapted version of the standard ATLAS test language. Tests that involve detailed signal timing and handshaking between a UUT (unit under test) containing an embedded processor and an ATE (automatic test equipment) often do not lend themselves to straightforward description in existing standard ATLAS statements. RCA recently modified its ATE and added ATLAS statements that in conjunction with an 8086 microprocessor augmentation multitasking routine, are well suited for handling the special needs of testing units that contain one or more embedded microprocessors. Tests that involve detailed signal timing and handshaking between the UUT and ATE are handled without problems. The software modifications to ATLAS are highlighted.

System software background

The RCA-manufactured ATE systems contain an extremely powerful software system for test program development, debugging, and validation. The software system consists of the master software system, RDOS; the ATLAS test system; and other user-oriented support programs and software library routines. The basic components of the ATLAS test system are the ATLAS compiler and the ATLAS run-time system (Fig. 1). The ATLAS compiler, when initiated, takes an ATLAS source file and analyzes it line by line for syntax and format. If this source file is correct, the compiler translates the file into ATLAS intermediate code.

The intermediate code file consists of opcodes and operands. Opcodes are of two general types. The first type specifies the operations performed on the stack, which primarily involve mathematical calculations. The second type specifies the control of the stimulus, measurement, and switching devices. When the ATLAS run-time system is initiated, the intermediate code file is brought in blocks from disk to main memory, where it is interpreted. Instructions are executed from an entry point. Each opcode is decoded and the proper procedure is called to interpret the instruction. Also included in the run-time system are error-handler routines and analytic routines that analyze raw

Abstract: *RCA has recently modified its automatic test equipment and added ATLAS language statements which, in conjunction with an 8086 augmentation multitasking routine, can handle special needs for embedded processor testing. The necessary software modifications are highlighted in this article. By executing 8086 code in parallel with an ATLAS program, two main constraints of the ATLAS environment—slow execution time and task limitations—are alleviated.*

Final manuscript received October 3, 1983.
Reprint RE-28-6-6

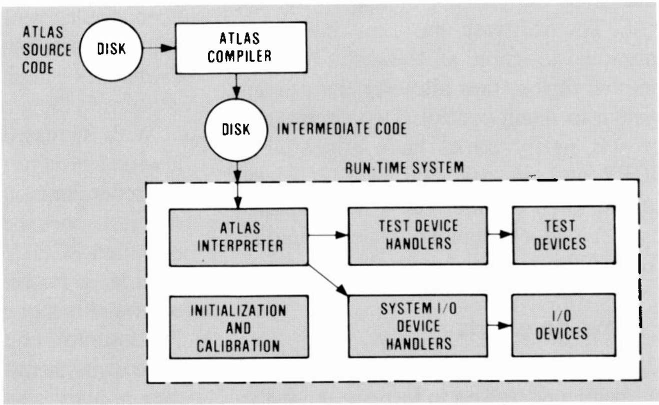


Fig. 1. *The EQUATE test station employs the ATLAS test system. The ATLAS compiler is invoked to obtain intermediate code for a test program. The test program is executed when its intermediate code is interpreted by the run-time system.*

measurement data. Also provided are input/output routines so the user can communicate with an ATLAS program.

Current test requirements

In addition to the many standard analog and digital circuits that are directly testable by the RCA system using the ATLAS test programming language, new user systems can contain a distributed processing system with embedded microprocessors. The processors communicate over serial digital links using different formats and protocols and use parallel busses to communicate internally between the embedded microprocessors and their support devices.

An analysis was made to identify specific current test requirements. This analysis identified tests that could not be performed with the existing ATLAS language since the language was designed to support stimulus/response testing predicated on predefined blocks of stimulus and expected responses. The analysis also identified requirements such as tests with large numbers of patterns (memory tests) and tests that require intelligent interactions between the tester and a processor under test (communication links). Algorithmic generation of patterns and responses was considered necessary to provide the large number of pattern/responses required. The test requirements analysis also established the need to access the memory space of processors under test and to supply instruction codes, via an external memory, to exercise the processors.

System augmentation

Since the existing ATE did not have the hardware or the test language required to support all of the new test requirements, a trade-off analysis (pitting augmentation versus adapters) was made. The decision was made to augment the existing system for the testing of parallel bus structure and serial digital communication links. This augmentation is accomplished by designing the necessary interface hardware and adding the required ATLAS language extensions.

The augmentation hardware consisted of both interface and digital test electronics. The interface electronics used existing hardware elements to support communication between the core system and the digital test electronics. The digital test electronics consists of a CPU board containing an 8086 processor and support chips, a 64K memory board, a parallel input/output interface board and three serial digital interface boards. The digital test electronics section supports the testing of RS232 serial digital devices and other devices that operate on a DATA, READY CLOCK format. The section also supports the testing of parallel devices with 8- or 16-bit data busses and less than 20 bits of address.

In conjunction with the development of the augmentation hardware, it was necessary to develop appropriate ATLAS language extensions and hardware driver routines to perform the following functions:

- Downloading a program to the augmentation processor;
- Execution of a downloaded program;
- Communication/synchronization between the downloaded program and the ATLAS program;
- Writing/reading to the augmentation memory space;
- Performing memory tests;
- Transmitting/receiving serial messages; and
- Transmitting/receiving USART messages.

8086 software

In addition to developing the required ATLAS extensions, it was necessary to consider a means of simplifying the test programmers' tasks of developing tests and generating the appropriate code. To accomplish this goal, a multitasking executive routine (EXEC) was developed that resides in the augmentation processor's ROM. The EXEC routine provides preprogrammed tests executed from ATLAS test statements and support routines to simplify and reduce the amount of code to be generated by the test programmer.

The EXEC program is entered when the augmentation hardware is reset with the start of the execution of an ATLAS program. The EXEC program provides the following functions:

- Device and memory initialization;
- Interrupt vector initialization;
- Register initialization for a downloaded program;
- Transfer of program execution to a downloaded program;
- Bidirectional data transfer between the downloaded program and the ATLAS program;
- Interrupt handling;
- Task synchronization;
- Execution of a dedicated task for memory testing;
- Execution of a dedicated task for transmitting/receiving serial messages; and
- Execution of a dedicated task for transmitting/receiving USART messages.

Communication between the ATE and the augmentation processor takes place on an interrupt-driven basis via two 16-bit data buses, one a control data bus, the other a measurement data bus (Fig. 2). The ATLAS run-time system transfers commands or data to the EXEC program by sending a 16-bit value via the control data bus to a register (ODATA) in the augmentation CPU. It then interrupts the EXEC program and the EXEC reads the contents of the ODATA register. The EXEC program transfers commands or data to the ATLAS run-time system by writing a 16-bit value into a control register (IDATA) on the augmentation CPU. It then interrupts the run-time system and the run-time system receives the contents of the IDATA register via the measurement data bus. There are defined handshake protocols using interrupts between the two processors for transfer-

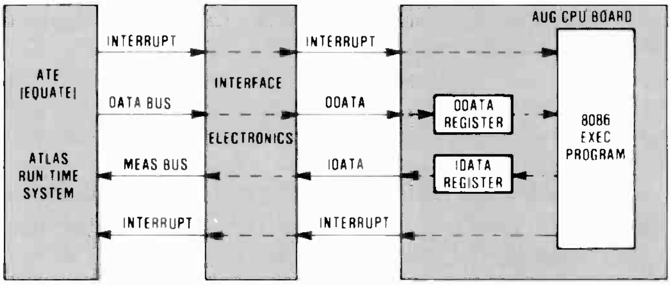


Fig. 2. *The ATLAS run-time system and the 8086 EXEC program communicate on an interrupt-driven basis via two 16-bit data buses.*

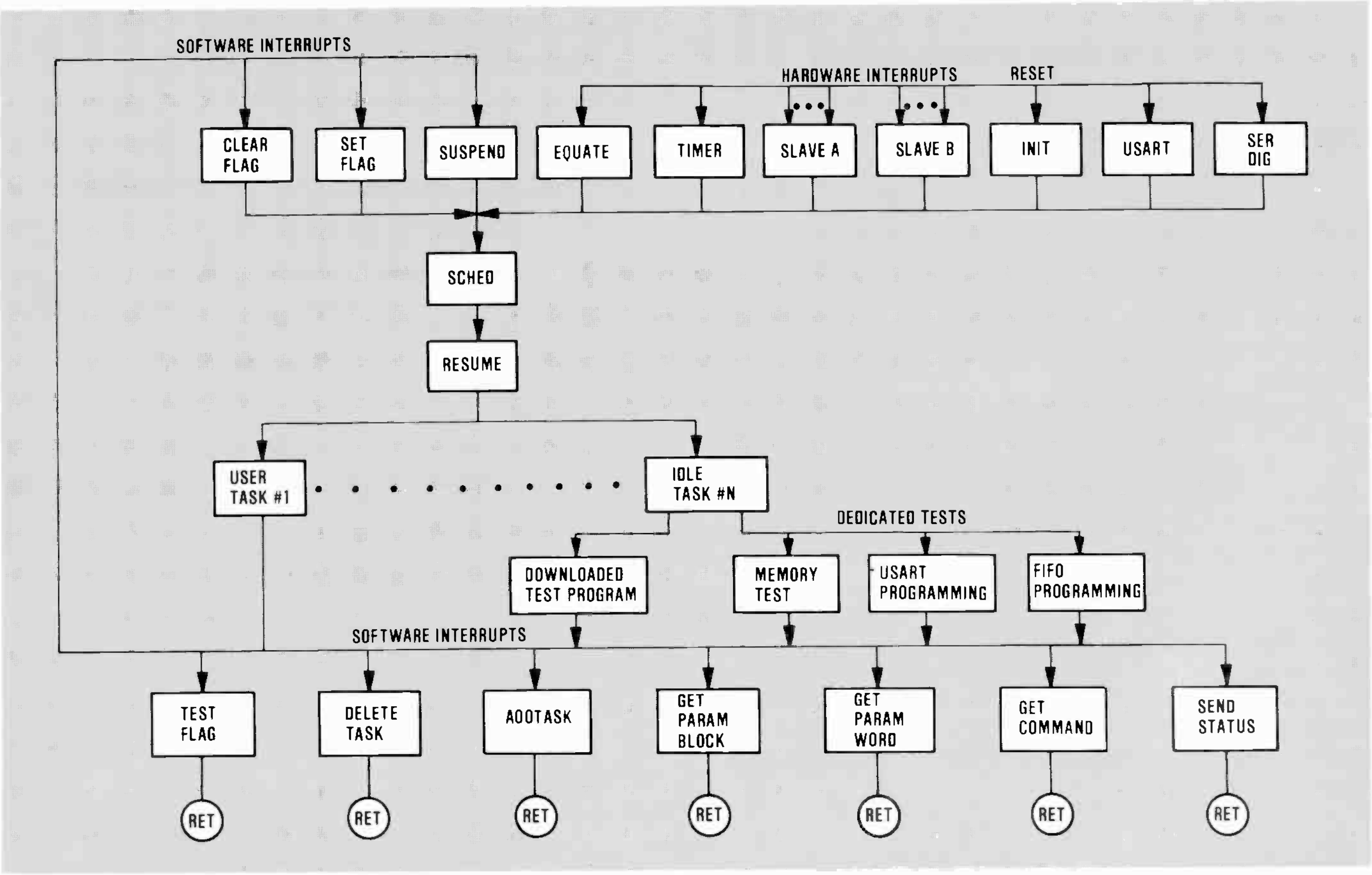


Fig. 3. *The EXEC provides multi-task scheduling. When an interrupt occurs (the timer interrupts every 0.1 seconds), the SCHED routine will pass control to the highest-priority non-blocked task.*

ring commands and data between the ATLAS run-time system and the EXEC program. We have highlighted the applications of these protocols below:

- Transferring initial 8086 register values from the ATLAS run-time system to the EXEC program before the start of a downloaded program.
- Starting one of three dedicated tasks or a downloaded user task.
- Transferring data parameters to a downloaded 8086 program from the ATLAS statement that starts the program execution.
- Transferring a 16-bit message from a running 8086 task to a running ATLAS program.
- Transferring a 16-bit message from a running ATLAS program to a running 8086 task.

The EXEC program is capable of providing elementary multitask scheduling. Figure 3 contains an overview of the flow for this program. Eight tasks may be queued for execution. The EXEC allocates the CPU to the highest priority non-blocked task. Tasks are blocked when they are waiting for an event flag or in a time-out. Upon reset, the EXEC initializes the PICs (programmable interrupt controllers) and PITs (programmable interval timers), clears RAM, and initializes the interrupt vector table. It then adds the IDLE task to the task queue. Control is then passed to the scheduler, which allocates the CPU to the highest priority task. The IDLE task is then entered. This task is suspended or waiting for an interrupt from the ATLAS run-time system. The run-time system will interrupt the 8086 only when it is processing an ATLAS statement that requires a downloaded task or a dedicated task to be executed.

The EXEC routine provides several support routines, available to both dedicated and downloaded tasks. These routines are executed by invoking internal (software) interrupts. By using these routines, the 8086 test programs will be more efficient and may be designed and validated in a shorter amount of time. Each routine's application is described below.

SUSPEND: Allows the user to stop execution or block a task until either a time out has occurred or a flag has been set due to a hardware interrupt.

SET-FLAG and CLEAR-FLAG: These routines allow the user to set or clear bits in the software-interrupt flag. These are the same bits used by the hardware-interrupt service routines.

TEST-FLAG: Allows the user to check the software-interrupt flag to see if a hardware interrupt has occurred.

ADD-TASK: This routine provides a mechanism for the user to spawn tasks.

DEL-TASK: Allows the user to remove a task from the active task list.

The following support routines handle data transfer between an 8086 task and the ATLAS run-time system. The ATLAS statements that have been created for data transfers between the two processors will be described later. The routines described below will handle a defined, interrupt-driven handshake protocol between the EXEC routine and the ATLAS run-time system.

GET-CMND: To receive a 16-bit value from the running ATLAS program.

SEND-STATUS: To send a 16-bit value to the running ATLAS program.

GET-PARAM-WORD and GET-PARAM-BLOCK: To receive one 16-bit value or a block of 16-bit values from the ATLAS program at the time that the downloaded task is executed.

In addition to the executable routines mentioned above, the EXEC program also provides interrupt-handling routines for two slave programmable-interrupt controllers, which are available for UUT testing. When one of these routines is called, it will clear the interrupt and set a bit in the software interrupt flag. The software-interrupt flag can be checked by calling the TEST-FLAG routine mentioned above. The current task may be blocked while waiting for one of these interrupts by invoking the SUSPEND routine.

Test program execution

The most powerful level of access to the capabilities of the digital augmentation is by executing a downloaded test program in the augmentation processor. These programs are run in parallel with the ATLAS test program. Downloaded test programs are written in 8086 assembly language and incorporate many support routines from the 8086 multitasking program. The following sequence of ATLAS statements will enable an 8086 test program to be transferred to the augmentation memory space:

```
OPEN-CHANNEL n, FILE "filename"$
DOWNLOAD DIGITAL-AUG, CHANNEL n$
CLOSE CHANNEL n$
```

When the ATLAS run-time system processes the DOWNLOAD statement, it will fetch data from the disk file (filename) through the ATLAS channel that has already been opened. The disk file has been created on a VAX development system by invoking software tools supplied by both Intel and RCA. This file contains 8086 absolute code and a header table. The header table contains one or more entries of the following groups of variables: the initial 8086 address to start downloading the absolute code, the amount of code to be transferred, and a pointer in the disk file to where he code begins. As each table entry is examined by the run-time system, the appropriate 8086 code is transferred to the augmentation memory space.

When data is transferred to the augmentation memory space, the run-time system will put the 8086 in HOLD mode, and "direct-memory-access" the data to the augmentation memory space at the address specified from the header table. When all data has been transferred to the augmentation memory space, the run-time system will release HOLD of the 8086. The disk file header will also contain initial values for the CS (code segment) and IP (instruction pointer) registers. The values placed in these registers will determine where the downloaded task will begin execution. The header table may also contain initial values for the DS (data segment) and ES (extra segment) registers. The run-time system will store these values in its memory space until the ATLAS statement for starting the downloaded code is processed.

Actual 8086 test program execution will begin when the following ATLAS statement is executed:

```
START DIGITAL-AUG      <,initial register values>
                       <,parameter list> $
```

The ATLAS run-time system will process the START statement in the following manner. First, it will interrupt the 8086 multitasking program (it should be in the IDLE task) and begin a defined handshake protocol for initializing 8086 registers. The run-time system will transfer to the EXEC program the initial values of the CS and IP registers that it has stored in memory from the execution of the DOWNLOAD statement. The START ATLAS statement also has optional noun modifiers for the following 8086 general-purpose registers: AX (accumulator), BX (base), CX (count), DX (data), BP (base pointer), SI (source index), and DI (destination index). If any of these noun modifiers and their values are present on the run-time stack, then these values will also be transferred to the EXEC program for placement in the appropriate register or registers.

After register initialization has been completed the run-time system will interrupt the IDLE task again and tell it via a command to begin execution of the downloaded task. The IDLE task will now transfer control of the 8086 to the downloaded test program. The START ATLAS statement also has optional noun modifiers for passing parameters (16-bit values) between the ATLAS program and the downloaded 8086 program. This option is available to initialize a variable or a table of variables so that a single downloaded test program may be more versatile. If this option has been chosen, the run-time system will then wait for an interrupt from the 8086 to signal the beginning of another interrupt-driven handshake protocol. The downloaded 8086 program does not need to know the details of this protocol, because it simply can invoke either the GET-PARAM-WORD or GET-PARAM-BLOCK support routines. After the completion of parameter passing (if any) the run-time system will go ahead and process the next ATLAS statement so the ATLAS program and the 8086 program are now running in parallel.

While the ATLAS and 8086 programs are running in parallel, they may synchronize their operations by exchanging 16-bit messages. The ATLAS program has no control or knowledge of the state of the 8086 program unless the 8086 program sends it a message and it is listening (waiting for an interrupt) for the message. The 8086 test program may send the ATLAS program a message by invoking the support routine SEND-STATUS. This routine will handle the interrupt-driven handshake protocol for this data transfer. The ATLAS program may receive this message when the following ATLAS statement is executed:

```
READ (STATUS × VALUE), DIGITAL-AUG
  <, MAX-TIME n SEC> $
```

When the ATLAS run-time system processes this statement, it will immediately go into a wait-for-interrupt routine that will suspend execution of the ATLAS program until an interrupt has been received from the 8086. This routine may "time out" after 1 second, or the value of the optional noun modifier MAX-TIME, has been reached. If a time-out occurs, then a run-time error will be posted and the program will be terminated. When the run-time system does detect an interrupt from the 8086, it will enter another protocol routine to obtain the message from the 8086 support routine SEND-STATUS.

If the ATLAS program desires to send a 16-bit message to the running 8086 program, then the following statement would apply:

SEND DIGITAL-AUG, COMMAND × $

The 8086 program should be looking for this interrupt by invoking the 8086 support routine GET-CMND. When the GET-CMND routine acknowledges the interrupt, it and the ATLAS run-time system will again enter another defined protocol for this data transfer. If the 8086 program fails to acknowledge the initial interrupt within 1 second, then a run-time error will be posted and the ATLAS program will be terminated. If the two programs are in sync, then the GET-CMND routine will return a 16-bit value to the downloaded routine.

To terminate a downloaded task, the following ATLAS statement is executed:

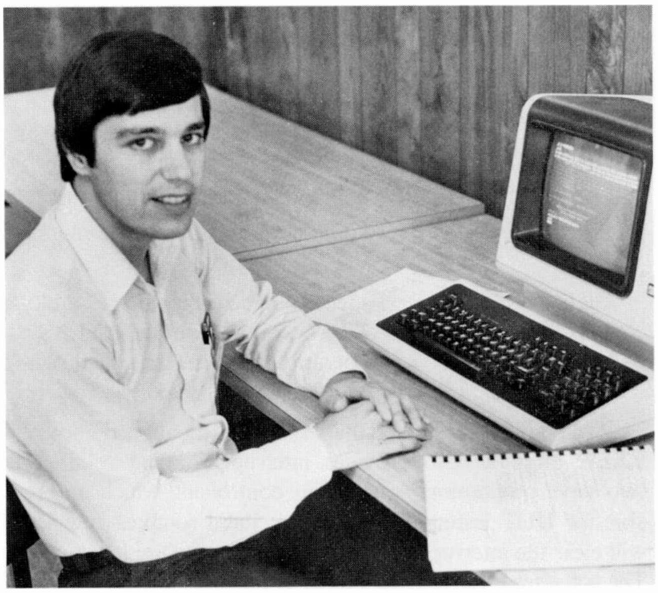
REMOVE DIGITAL-AUG $

This statement will cause the run-time system to reset the augmentation processor. The EXEC program will then go through its initialization code and return to the IDLE task.

Unique ATLAS statements have been created for three specific functions of the digital augmentation. These three functions are: UUT memory testing, transmitting or receiving serial messages with the UUT, and transmitting or receiving USART messages with the UUT. Interaction between the ATLAS run-time system and the EXEC program is similar for these test functions. For each function there exists an 8086 routine that resides with the 8086 multitasking program in the augmentation ROM space. These routines are callable from the IDLE task.

The ATLAS run-time system will process one of the above unique ATLAS statements in the following manner. The run-time system will place a command in an 8086 register and interrupt the EXEC routine while it is in the IDLE task. The EXEC will read the command and then transfer control to a specific routine to handle the function of the particular ATLAS statement. The 8086 routine will then call the support routine, GET-PARAM-BLOCK, to obtain a list of values from the run-time system. The run-time system will have already prepared this list of values based upon the values of the individual noun modifiers in the ATLAS statement. The ATLAS run-time system will now wait for the specific test function to be completed by the 8086. When the 8086 routine has finished its dedicated function it will return a minimum of one word (status) to the run-time system. When receiving serial or USART messages, it will be returning several values to the run-time system. The 8086 support routine, SEND-STATUS, will handle the defined handshake protocol for the above data transfers from the 8086 routine to the run-time system. Once the dedicated 8086 routine has finished its execution it will return control to the IDLE task.

The designs of the ATLAS run-time system and the EXEC program allow for easy addition of modules. Modular design is important to reduce the amount of non-ATLAS code that is generated for special test requirements that may be identified in the future.



Richard Prenoveau, Senior Member Technical Staff, Automatic Test Systems at RCA Automated Systems, received his BSEE from Rochester Institute of Technology in 1978. Since joining RCA Automated Systems in 1980, he has worked on many projects encompassing the design and upgrading of the system software package for the AN/USM 410 ATE. Specific components that he has worked on include: ATLAS run-time modules, alignment modules for stimulus and measurement devices, and various user-oriented support routines.

Contact him at:
RCA Automated Systems
Burlington, Mass.
TACNET: 326-2714

Conclusion

The software concept we have described for implementing the augmentation capability provides a unique means of testing UUTs with embedded processors or other real-time functions. By executing 8086 code in parallel with an ATLAS program, two main constraints of the ATLAS environment—that is, slow execution time and task limitations—are alleviated.

The augmentation design that was developed enables a method of implementing additional functions compatible with the existing RCA ATE or similar systems using ATLAS. This methodology will provide expansion capability to satisfy the new and sophisticated test requirements of advanced weapon systems.

R.J. Felbinger | W.R. Leahy

Computer-aided functional analysis—A new concept in system engineering

Written in an enhanced BASIC language, this set of Functional Information Control System programs automatically checks for content errors in functional flow diagrams, and provides a record of AEGIS Combat System interrelationships.

Functional analysis—the analysis of system functional requirements from top level to implementation level—is regarded in industry as a useful and versatile engineering technique in the design of complex military systems. At Moorestown, a functional analysis tool known as Functional Flow Diagrams and Descriptions (F^2D^2)* has been developed and used over the past 13 years in the design and auditing of the AEGIS Weapon System and more recently the AEGIS Combat System of *Ticonderoga*-class cruisers.

The original AEGIS Combat System, generally considered the most technologically advanced, most automated Navy combat system in the world, requires a complex assortment of equipment, computer programs, and personnel. These requirements are increasing as the system evolves. The level of detail needed to design the combat system provides numerous opportunities for introducing human error and necessitates extensive bookkeeping and consistency checks. As a result, a computer-aided version of F^2D^2 has been developed, using desktop microprocessors and specially tailored software packages. This approach, known as the Functional Information Control System (FICS), has improved the speed, efficiency, and accuracy of information generation.

This paper discusses the capabilities of the Functional Information Control System in the functional analysis of the AEGIS Combat System.

F^2D^2—Functional Flow Diagrams and Descriptions

The objective of F^2D^2 is to define, interrelate, and integrate the functions of electronic equipment, computer programs, personnel and facilities that constitute a system. The fundamental tiering process of F^2D^2 (see Fig. 1) ties into MIL-STD-490, which sets forth practices for the preparation and interpretation of specifications for military programs.

In general, F^2D^2 describes a system in a logical sequence, showing all basic functional interactions and interrelationships; it provides a structure that bounds the system it describes. It employs feedback loops, has a logical ordering scheme, identifies messages generically, and describes functions at each level. The number of tiers required depends on the complexity of the system being defined and the degree of specificity needed, although funding and time constraints also play a part. In the functional analysis performed on the AEGIS Combat System, for example, as few as two and as many as five tiers were necessary for a complete functional definition. Functions at each tier can be traced to the system specification, enabling the functions to be tracked through the system.

In addition, F^2D^2 diagrams indicate the element/subsystem, the major equipment, the computer programs, and the personnel involved. For example, the Tier 1 functions are assigned to specific elements or subsystems of the overall system, and Tier 2 functions are assigned to individual major equipment, computer programs, and manned operating stations. Tier 3 functions are allocated to cabinet drawers, operator tasks, and computer program modules.

At the final step of the F^2D^2 process, a narrative is generated for each functional tier, summarizing what the function does and how it is used in the system. These narratives also list incoming and outgoing data or signals, provide a subfunction summary, and include an allocation section that provides a complete cross-reference between the specification and the diagrams.

*F^2D^2 is a derivative of an earlier Air Force approach to functional analysis, Functional Flow Block Diagrams (see AFSCM-375-5).

Abstract: *The Functional Information Control System (FICS), a computer-aided data base management and graphics-generation control system, has been developed for use in the analysis of functional requirements for the U.S. Navy's AEGIS combatant ships. The functional analysis process relates requirements to hardware, software, and personnel, and identifies errors and gaps (for instance, missing functional components) in the system design. FICS is easy to use, requires minimal training, and ensures the accuracy, completeness, and consistency of the functional analysis products, including Functional Flow Diagrams and Descriptions (F^2D^2). Moreover, the speed and flexibility of FICS enables functional analysis to be responsive to user needs and ensures functional traceability of the functional analysis products to other engineering documentation, including system specifications.*

Final manuscript received October 17, 1983.
Reprint RE-28-6-7

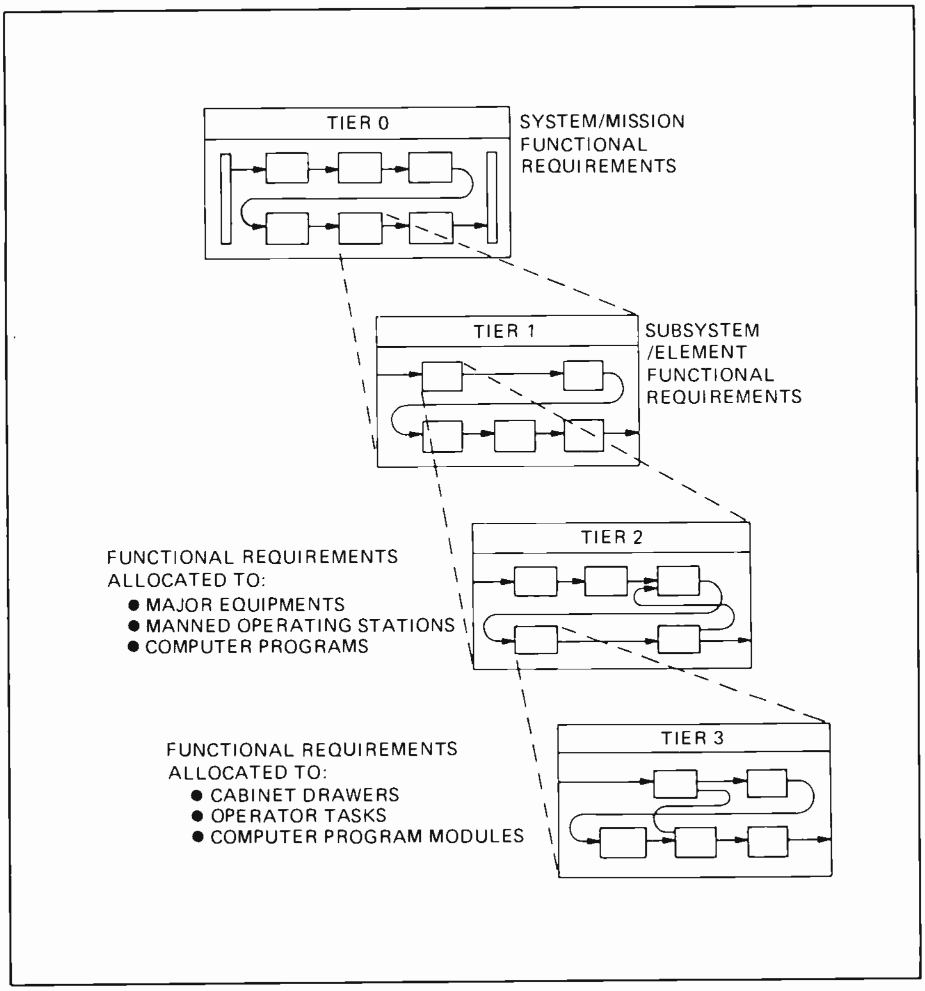

TIER 0
SYSTEM/MISSION FUNCTIONAL REQUIREMENTS
TIER 1
SUBSYSTEM /ELEMENT FUNCTIONAL REQUIREMENTS
TIER 2
FUNCTIONAL REQUIREMENTS ALLOCATED TO:
● MAJOR EQUIPMENTS
● MANNED OPERATING STATIONS
● COMPUTER PROGRAMS
TIER 3
FUNCTIONAL REQUIREMENTS ALLOCATED TO:
● CABINET DRAWERS
● OPERATOR TASKS
● COMPUTER PROGRAM MODULES

Fig. 1. F^2D^2 partitioning process. Functional analysis employs a top-down approach to system definition, beginning with a carefully selected set of top-level (Tier 0) functions, based on primary requirements of the system. The functional requirements of the system are then successively partitioned into subfunctions to the level of detail essential for performance, design, and procurement specifications.

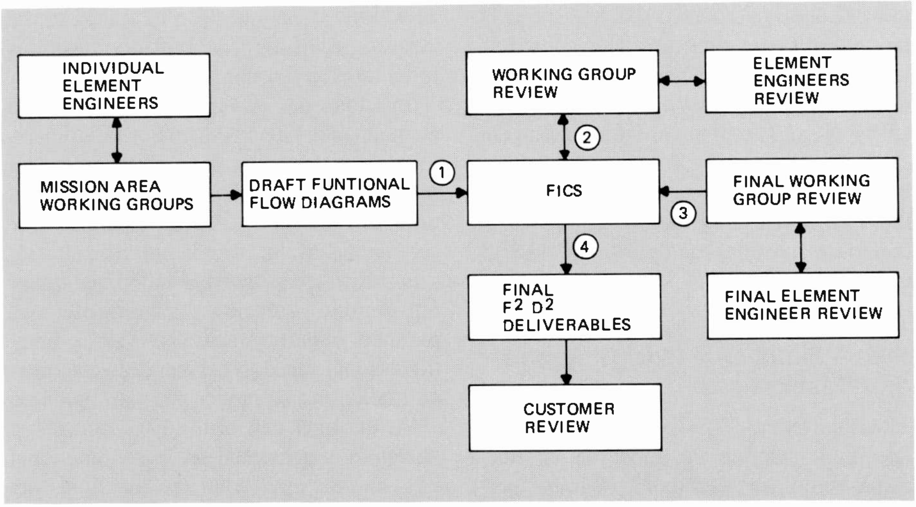

INDIVIDUAL ELEMENT ENGINEERS
MISSION AREA WORKING GROUPS
DRAFT FUNTIONAL FLOW DIAGRAMS
1
WORKING GROUP REVIEW
ELEMENT ENGINEERS REVIEW
2
FICS
3
FINAL WORKING GROUP REVIEW
4
FINAL F2 D2 DELIVERABLES
FINAL ELEMENT ENGINEER REVIEW
CUSTOMER REVIEW

Fig. 2. F^2D^2 development process. Although system and design engineers involved with various elements of the AEGIS Combat System do not work directly with FICS, they review products generated by FICS. Technical working groups composed of these engineers as well as personnel involved with F^2D^2 operation meet to resolve any problems or discrepancies that may arise. This review is an iterative process, culminating with the delivery of a final product to the customer.

Functional information control system

The requirement—to develop a computer-based information processing and data management system for analyzing, defining, and documenting Navy Ship Combat Systems functions and interfaces—was basically twofold. First, the statement of work and contract deliverable item required the development and implementation of a methodology to automate functional analysis and to deliver F^2D^2 products on mass storage media. Second, the clerical and administrative work needed to maintain an up-to-date functional analysis data base became excessive from the Tier 1 level down. To alleviate the burden of paperwork and to automate the system, the Functional Information Control System (FICS) was created.

FICS is a simple, easy-to-use tool that is tailored specifically to F^2D^2 requirements. The objective in developing FICS is manifold. Clean, reproducible copies of the diagrams and descriptions are required by the customer, and the methodology should be easily understood and communicated. A common store of information that supports simple, rapid operator data entry and modification throughout the entire life cycle of the ship must be available, and the relation between functions and the MIL STD-490 specification tree should be readily traceable. The system should provide automatic detection, location, and identification of errors in the functional analysis data base; use existing, low-cost hardware; and require minimal skill and training to operate. To date, FICS has met, even exceeded, all these objectives. The basic F^2D^2 development process using FICS is shown in Fig. 2.

The FICS programs are written in an enhanced BASIC language. They are designed to run on a Hewlett-Packard 9845B desktop microprocessor having at least 160 Kbytes of random access memory (RAM) and 500 Kbytes of disk memory. Figure 3 illustrates the basic hardware and software components constituting the Functional Information Control System (FICS); Fig. 4 shows present and planned uses of FICS.

FICS comprises a set of software packages that include error-checking programs that examine each input in order to verify internal consistency with existing functions, elements, and input/output messages. Error messages identifying a discrepancy (for example, "This function number does not exist," or "This message is a duplicate of message number 703") are printed; thus, the computer operator is prevented from entering incorrect or redundant information into the system.

FICS has a graphics or plotting program that automatically produces updated functional flow diagrams (Fig. 5). This program uses the functional analysis data base to determine the number of messages flowing into and out of each function block and the number of lines of text required for each message. Then, after computing the size of each function block and by using certain sequencing rules, the system automatically allocates function blocks to specific locations on a functional flow block diagram and plots the diagrams using a multi-pen plotting device.

Sorting programs in the FICS perform various data sorts and print information in tabular format, including function, element, and function block listings (for example, all incoming and outgoing messages involving a specific function). These lists serve as current, effective working tools. Another program automatically generates element interface diagrams (Fig. 6) from the same functional analysis data base used to produce the functional flow diagrams. These diagrams portray the interfaces and information transfer between one element and all other elements in the system; the diagrams are normally contained in the performance characteristics section of both the system-level and the element-level specifications. Functional traceability and internal consistency are assured because the functional flow diagrams and element interface diagrams are derived from the same data base.

In the Development Phase (Phase II) of a program, information related to function sequencing, personnel, equipment, and software allocations will be included in the data base. This expansion of the data base will increase its versatility and complexity, extending FICS capability to other types of documentation including Operational Sequence and Timing Diagrams, personnel planning, and the use and interaction of equipment and software.

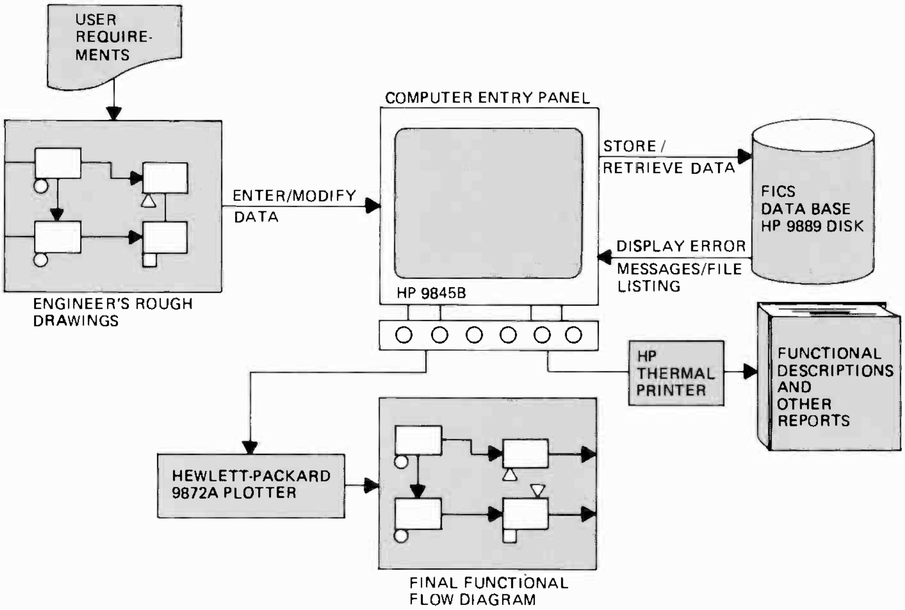

USER REQUIRE-MENTS
ENGINEER'S ROUGH DRAWINGS
ENTER/MODIFY DATA
COMPUTER ENTRY PANEL
HP 9845B
STORE / RETRIEVE DATA
FICS DATA BASE HP 9889 DISK
DISPLAY ERROR MESSAGES/FILE LISTING
HP THERMAL PRINTER
FUNCTIONAL DESCRIPTIONS AND OTHER REPORTS
HEWLETT-PACKARD 9872A PLOTTER
FINAL FUNCTIONAL FLOW DIAGRAM

Fig. 3. Basic components of FICS. The equipment configuration used for FICS consists of a monochromatic graphics CRT display, a typewriter-like keyboard, a thermal printer, and two magnetic tape drives built into a single chassis. A Hewlett-Packard 9872A model flatbed graphics plotter and a 9889 disk drive round out the array of equipment used.

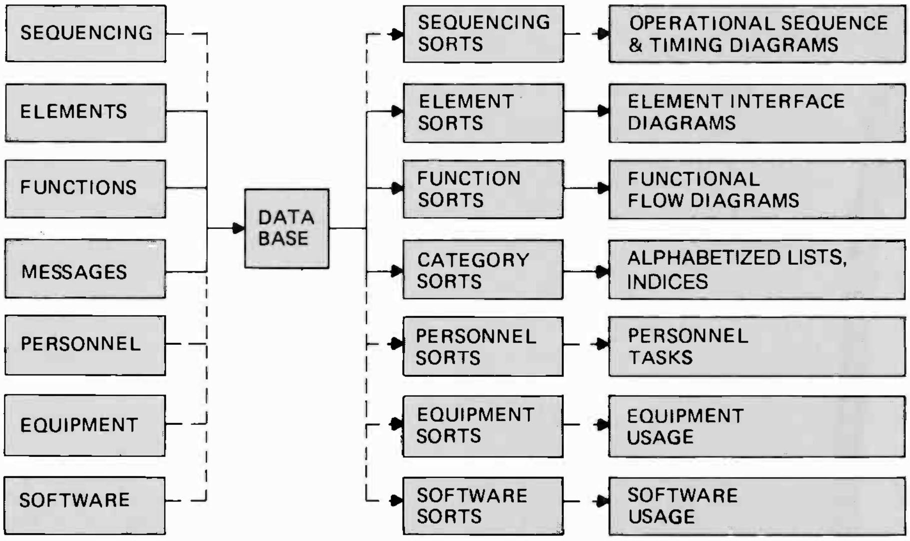


Fig. 4. Present and planned uses of FICS. This diagram shows the present and planned uses of FICS in terms of input, analysis, and output. In the DDG 51 Definition Phase Contract, three aspects of the data base are currently implemented: elements, functions, and messages. These aspects culminate in the production of functional flow diagrams and element interface diagrams. In the Development Phase, tasks will be further subdivided and then allocated to personnel, equipment, or software. In addition, a single-thread pass will be made through the functions, producing representative mission scenarios, for example, Operational Sequence and Timing Diagrams (OSATDs).

Experience to date

The value of FICS in functional analysis has been demonstrated in RCA; in particular, the formerly routine clerical task of checking for content errors in the functional flow diagrams is no longer required. Automatic checking is now done by FICS more rapidly and with greater accuracy than was possible manually. More important, the functional analysts, formerly required to manually check for design correctness, now have more time to devote to the actual analysis task. FICS can also quickly provide an updated set of functional diagrams.

FICS automatically renumbers functions when it becomes necessary to change an element number or the sequence in which functions are described. Thus, after functions are deleted or new functions are added to the flow, FICS will adjust the serial listing, eliminating any gaps in the numerical sequence. FICS also renumbers the message source and destination throughout the data base to reflect the new function-num-

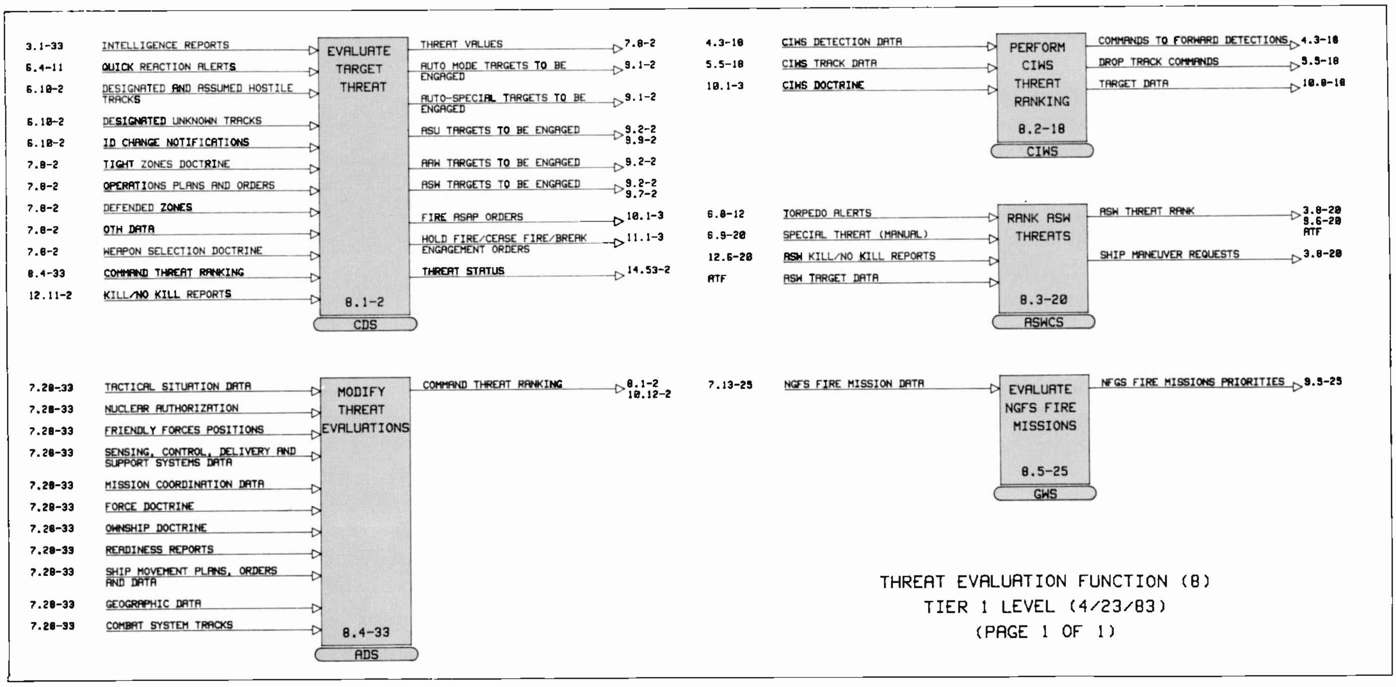


Fig. 5. Sample functional flow, diagram. This diagram is an example of one of 63 Tier 1 functional flow diagrams that was produced for the DDG 51 program. It was plotted by using Hewlett-Packard equipment and FICS software.

bering sequence. This capability represents a substantial savings in labor costs and also provides better control over the accuracy of the end product.

Functional flow diagrams indicate the incoming and outgoing message flow between functions, and the element interface diagrams indicate the message flow between a given element and other interfacing elements. The information contained in the two types of diagrams is the same but is presented in different formats, an important traceability feature (Fig. 7). There is a direct derivation of the messages appearing on the element interface diagrams from those shown on the functional flow diagrams. Also, the input/output section of the functional narratives for both types of diagrams is generated from the same centralized data base used to produce the message flow information, resulting in savings in manpower and time and providing a more consistent functional analysis product.

Growth plans

Plans for increasing FICS capabilities are already under way and several expanded features have been proposed. This potential growth of the system includes a program that automatically repositions blocks to generate the lines that connect various functional blocks on a single diagram. With the current capability, numbers are used at the end of the lines to identify the source or destination function for the message data; the analyst instructs the computer where to plot connecting lines and arrowheads and also where to insert labels. In the proposed modification, additions and changes will automatically become part of the diagram before it is plotted.

Additional programs are being developed for producing functional flow diagrams classified by mission area (for example, anti-air warfare and anti-submarine warfare); cross-indexing messages by message title; and sorting/listing functions by operators, equipment, and computer programs at the Tier 2 level. Another function planned for FICS is computer-aided generation of Operational Sequence and Timing Diagrams (OSATDs). The OSATD provides a "needle-and-thread" pass through the F²D² that is scenario oriented. Because OSATD links functions in serial form for any scenario, system response times can also be predicted. Finally, programs to crosslink the text material, for example, functional narrative descriptions to the F²D² data base diagrams, are being developed. This text is currently stored on a separate word processing system.

The speed and memory of FICS are being upgraded. Currently, several of the FICS programs approach the upper memory limits of the HP 9845. One possible modification is the use of a large, mainframe computer system with interactive terminal and large drafting plotter. The DEC-20 computer at Moorestown, one of the systems presently under consideration, has extensive memory capacity, several graphics terminals, and a CALCOMP 1051 plotter that produces engineering-scale, high-quality graphics. Using the DEC-20 and peripherals in lieu of the HP 9845 system can significantly reduce the overall time for data processing and substantially decrease the plotting time. The FICS software, written in BASIC, can be rewritten in FORTRAN to be compatible with the DEC-20 computer.

FICS is being considered as a support to the development of functional sequence flow diagrams (FSFDs) and of configuration definition documents (CCDs). FSFDs and CDDs are similar to the functional flow diagrams except that instead of depicting message or information flow between functions, they illustrate the signals and the physical connections among individual element equipment.

Another potential application is in support of the FAA's Advanced Automation System (AAS) for which RCA is currently the system engineering technical advisor. In this program, F²D² is being used to audit a computer replacement system design for enroute air traffic control centers, providing visibility required to ensure that all functions and interfaces have been correctly incorporated into the specification.

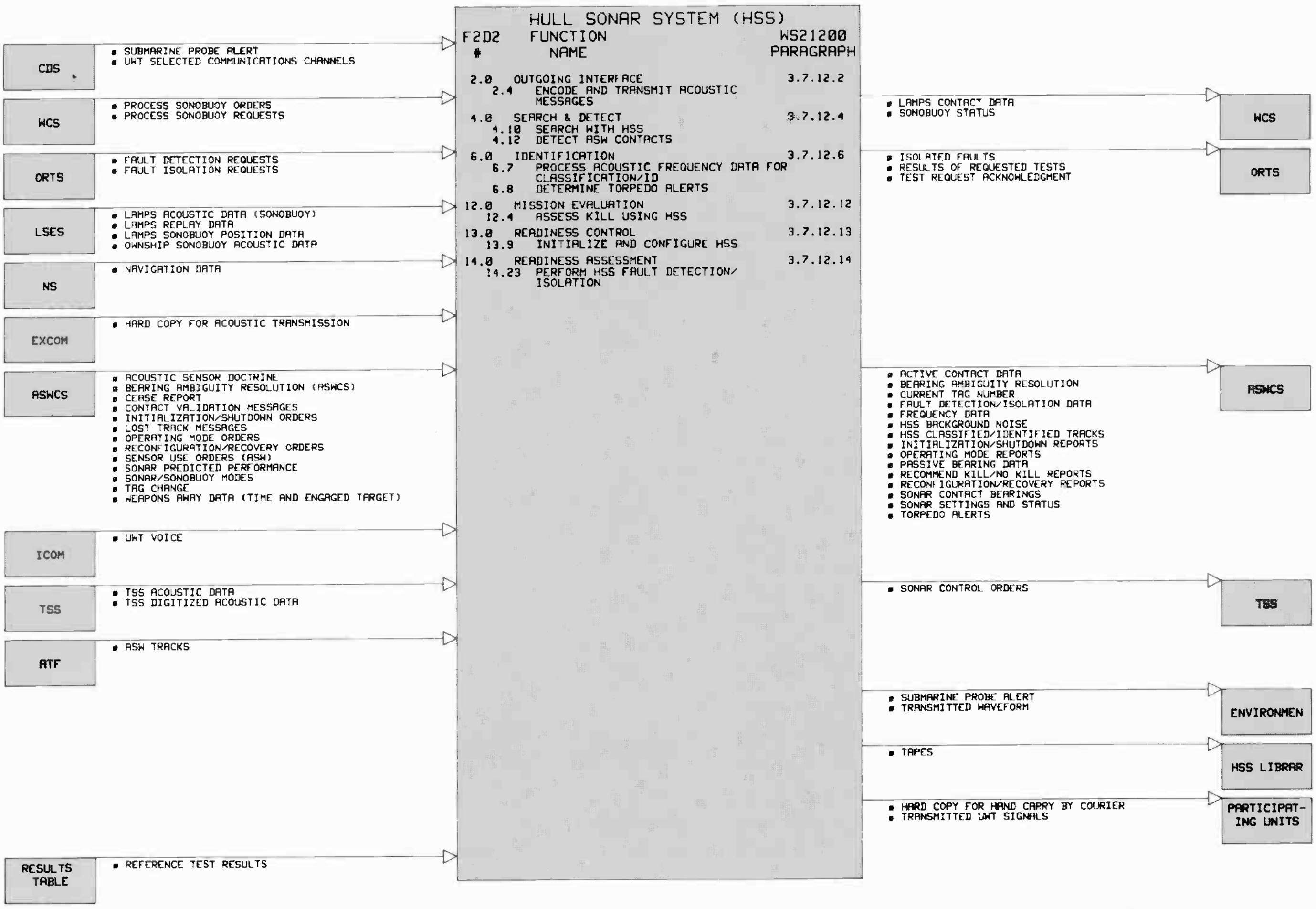

HULL SONAR SYSTEM (HSS)
F2D2 #
FUNCTION NAME
WS21200 PARAGRAPH
2.0 OUTGOING INTERFACE 3.7.12.2
2.4 ENCODE AND TRANSMIT ACOUSTIC MESSAGES
4.0 SEARCH & DETECT 3.7.12.4
4.10 SEARCH WITH HSS
4.12 DETECT ASW CONTACTS
6.0 IDENTIFICATION 3.7.12.6
6.7 PROCESS ACOUSTIC FREQUENCY DATA FOR CLASSIFICATION/ID
6.8 DETERMINE TORPEDO ALERTS
12.0 MISSION EVALUATION 3.7.12.12
12.4 ASSESS KILL USING HSS
13.0 READINESS CONTROL 3.7.12.13
13.9 INITIALIZE AND CONFIGURE HSS
14.0 READINESS ASSESSMENT 3.7.12.14
14.23 PERFORM HSS FAULT DETECTION/ ISOLATION
CDS
• SUBMARINE PROBE ALERT
• UWT SELECTED COMMUNICATIONS CHANNELS
WCS
• PROCESS SONOBUOY ORDERS
• PROCESS SONOBUOY REQUESTS
ORTS
• FAULT DETECTION REQUESTS
• FAULT ISOLATION REQUESTS
LSES
• LAMPS ACOUSTIC DATA (SONOBUOY)
• LAMPS REPLAY DATA
• LAMPS SONOBUOY POSITION DATA
• OWNSHIP SONOBUOY ACOUSTIC DATA
NS
• NAVIGATION DATA
EXCOM
• HARD COPY FOR ACOUSTIC TRANSMISSION
ASWCS
• ACOUSTIC SENSOR DOCTRINE
• BEARING AMBIGUITY RESOLUTION (ASWCS)
• CEASE REPORT
• CONTACT VALIDATION MESSAGES
• INITIALIZATION/SHUTDOWN ORDERS
• LOST TRACK MESSAGES
• OPERATING MODE ORDERS
• RECONFIGURATION/RECOVERY ORDERS
• SENSOR USE ORDERS (ASW)
• SONAR PREDICTED PERFORMANCE
• SONAR/SONOBUOY MODES
• TAG CHANGE
• WEAPONS AWAY DATA (TIME AND ENGAGED TARGET)
ICOM
• UWT VOICE
TSS
• TSS ACOUSTIC DATA
• TSS DIGITIZED ACOUSTIC DATA
ATF
• ASW TRACKS
RESULTS TABLE
• REFERENCE TEST RESULTS
• LAMPS CONTACT DATA
• SONOBUOY STATUS
WCS
• ISOLATED FAULTS
• RESULTS OF REQUESTED TESTS
• TEST REQUEST ACKNOWLEDGMENT
ORTS
• ACTIVE CONTACT DATA
• BEARING AMBIGUITY RESOLUTION
• CURRENT TAG NUMBER
• FAULT DETECTION/ISOLATION DATA
• FREQUENCY DATA
• HSS BACKGROUND NOISE
• HSS CLASSIFIED/IDENTIFIED TRACKS
• INITIALIZATION/SHUTDOWN REPORTS
• OPERATING MODE REPORTS
• PASSIVE BEARING DATA
• RECOMMEND KILL/NO KILL REPORTS
• RECONFIGURATION/RECOVERY REPORTS
• SONAR CONTACT BEARINGS
• SONAR SETTINGS AND STATUS
• TORPEDO ALERTS
ASWCS
• SONAR CONTROL ORDERS
TSS
• SUBMARINE PROBE ALERT
• TRANSMITTED WAVEFORM
ENVIRONMEN
• TAPES
HSS LIBRAR
• HARD COPY FOR HAND CARRY BY COURIER
• TRANSMITTED UWT SIGNALS
PARTICIPAT- ING UNITS

Fig. 6. Sample element interface diagram. Illustrated above is an example of one of many element interface diagrams produced using the Hewlett-Packard equipment and FICS software. The element interface diagrams are particularly useful in the design process because they permit comparison of functional allocations to elements with design parameters established by element engineers, thereby facilitating the identification of inconsistencies. Auditing of the element interfaces provides rapid feedback and the opportunity to re-evaluate the design.

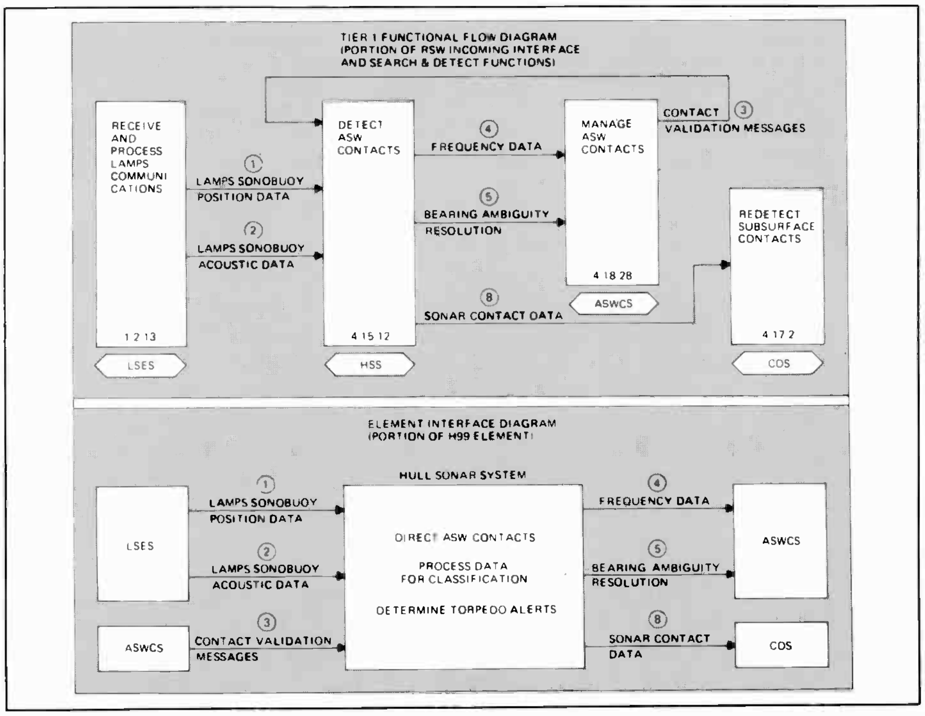

TIER 1 FUNCTIONAL FLOW DIAGRAM
(PORTION OF ASW INCOMING INTERFACE AND SEARCH & DETECT FUNCTIONS)
RECEIVE AND PROCESS LAMPS COMMUNICATIONS
1 2 13
LSES
1
LAMPS SONOBUOY POSITION DATA
2
LAMPS SONOBUOY ACOUSTIC DATA
DETECT ASW CONTACTS
4 15 12
HSS
4
FREQUENCY DATA
5
BEARING AMBIGUITY RESOLUTION
8
SONAR CONTACT DATA
MANAGE ASW CONTACTS
4 18 28
ASWCS
CONTACT VALIDATION MESSAGES
3
REDETECT SUBSURFACE CONTACTS
4 17 2
CDS
ELEMENT INTERFACE DIAGRAM
(PORTION OF H99 ELEMENT)
HULL SONAR SYSTEM
LSES
1
LAMPS SONOBUOY POSITION DATA
2
LAMPS SONOBUOY ACOUSTIC DATA
ASWCS
3
CONTACT VALIDATION MESSAGES
DIRECT ASW CONTACTS
PROCESS DATA FOR CLASSIFICATION
DETERMINE TORPEDO ALERTS
4
FREQUENCY DATA
5
BEARING AMBIGUITY RESOLUTION
8
SONAR CONTACT DATA
ASWCS
CDS

Fig. 7. A point-by-point correspondence between functional flow diagrams and element interface diagrams. The upper half of the figure is a functional flow diagram that depicts one of several Hull Sonar System (HSS) functions, "DETECT ASW CONTACTS," and its relationship to other Tier 1 element functions. Also indicated is the message between these functions. The lower portion of the figure contains an element interface diagram showing the relationship of the same HSS element to other interfacing elements of a combat system. The same "DETECT ASW CONTACTS" function (shaded area), indicated in the functional flow block diagram, is included among the HSS functions shown in the center block of the element interface diagram.

Conclusion

Because functional analysis has traditionally been a "paper and pencil" process, albeit an effective means of handling complex design issues, it has also been limited in precision and growth; particularly, when program schedules or budget constraints exist, the accuracy, comprehensiveness and internal consistency of F^2D^2 and OSATDs have suffered. The Functional Information Control System (FICS) is a powerful technique for improving the F^2D^2 process. It assures greater accuracy, consistency, and completeness in the functional information being developed.

Additionally, FICS permits facile data modification, enhanced cross referencing, and indexing. It is fast and responsive to user requirements, and offers significant growth potential. System engineers, for example, are able to obtain prompt, up-to-date functional information tailored to their particular needs. Because FICS can automatically detect, locate, and identify errors in the functional analysis data base, the resulting products are more accurate, consistent, and comprehensive than can be produced by non-computerized methods. Finally, traceability between system functions, top-level requirements, and specifications is improved by using FICS.

Authors Leahy, seated, and Felbinger.

References

1. E.G. Lurcott, "Functional Flow Diagrams and Descriptions for AEGIS—A Systems Engineering Approach," *RCA Engineer,* Vol. 19, No. 1 (July 1973).
2. E.G. Lurcott, "F^2D^2, a System Management Tool," RCA, *Defense Systems Management Review,* Vol. 1, No. 4 (1977).
3. J.T. Threston, "System Functional Analysis and Functional Allocation," Combat Systems Engineering and Ship Design Seminar, MIT, Cambridge, Massachusetts (August 1982).
4. J.T. Threston, "Managing the Future—The Expanding Role of Systems Engineering in a High-Technology Society," *RCA Engineer,* Vol. 28, No. 4 (July/August 1983).

Rick Leahy, Senior Member Engineering Staff, Functional Analysis, has conducted research in modeling and computer simulation of man-machine systems using a wide variety of computer systems over the past 15 years. He has over 30 technical reports, primarily in the area of simulation, to his credit. Currently, at RCA, Mr. Leahy is working on expansion of the capability of the Function Information Control System (FICS) in the area of computer generated graphics. This work is being performed in support of the U.S. Navy's AEGIS project.
Contact him at:
RCA Missile and Surface Radar
Moorestown, N.J.
TACNET: 224-2663

Robert Felbinger, Unit Manager, Functional Analysis, has over 20 years experience in system engineering and man-machine interface. He has worked with data processing systems in a variety of government and private industry applications and settings. His most recent efforts at RCA Corporation have involved the use of desktop microprocessors and interactive graphics programs as automated aids to the system analysis process on the U.S. Navy's AEGIS cruiser and destroyer combatant ship projects.
Contact him at:
RCA Missile and Surface Radar
Moorestown, N.J.
TACNET: 224-2674

R.H. McEachern | K.J. Prost | D. Hampel | D.D. Mawhinney

A personnel-tracking system

RCA is developing a microwave-based system for general-purpose physical security applications.

In 1981, Signal Processing Engineering in Somerville, part of Government Communications Systems in Camden, demonstrated the feasibility of a personnel-tracking system to control access to various critical areas in a building, or yard, for example. Since prior trade-off studies indicated that microwave sensing would be more effective than ultrasonic, infrared, or other sensing, such a system was designed, built, and demonstrated with the help of the Microwave Technology Center, RCA Laboratories.

In operation, a uniquely coded, badge-type transponder (a credential) is issued to each "user." Sensors or interrogators strategically deployed in given rooms or areas will monitor for the presence of these credentials (Fig. 1). The system will be used in conjunction with a second detection system that senses all intrusions into the controlled areas. If the tracking system detects a valid credential, authorized for access to the area, it will prevent the second system from sounding an alarm. When the system detects intrusions into controlled areas, and no valid credential is present, the alarm is sounded.

The sensors continually emit one of sixteen frequency-modulated tones, pulsed on a 10.5-GHz carrier. Each credential is set to monitor one of these tones and, upon detection, transmits a return pulse in one of sixty-four time slots on one of eight preassigned frequencies in the 200- to 300-MHz range. The whereabouts and movements of personnel can thus be automatically recorded and appropriate alarms or other measures can restrict unauthorized entry. By employing a multifrequency, time–division coding scheme, the system has a capacity of 8000 codes.

The credentials use a diode detector, amplifier, threshold detector for receiving, an oscillator for transmitting, and digital logic for timing, control, and filtering. The sensors use a Z-80 microprocessor controlling a Gunn-diode oscillator (microwave

Abstract: *A microwave-based tracking system was designed to control access to various areas. A uniquely coded, badge-type transponder is issued to each "user," and sensors or interrogators deployed in given areas sense and record the whereabouts of personnel. Standard microwave technology and a uniquely designed system are the key here. The combination of high-efficiency, low-power-dissipation circuits, application of a single state-of-the-art LSI chip, and the system's relatively low-duty cycle makes the active credential approach highly reliable.*

©1983 RCA Corporation
Final manuscript received October 31, 1983.
Reprint RE-28-6-8

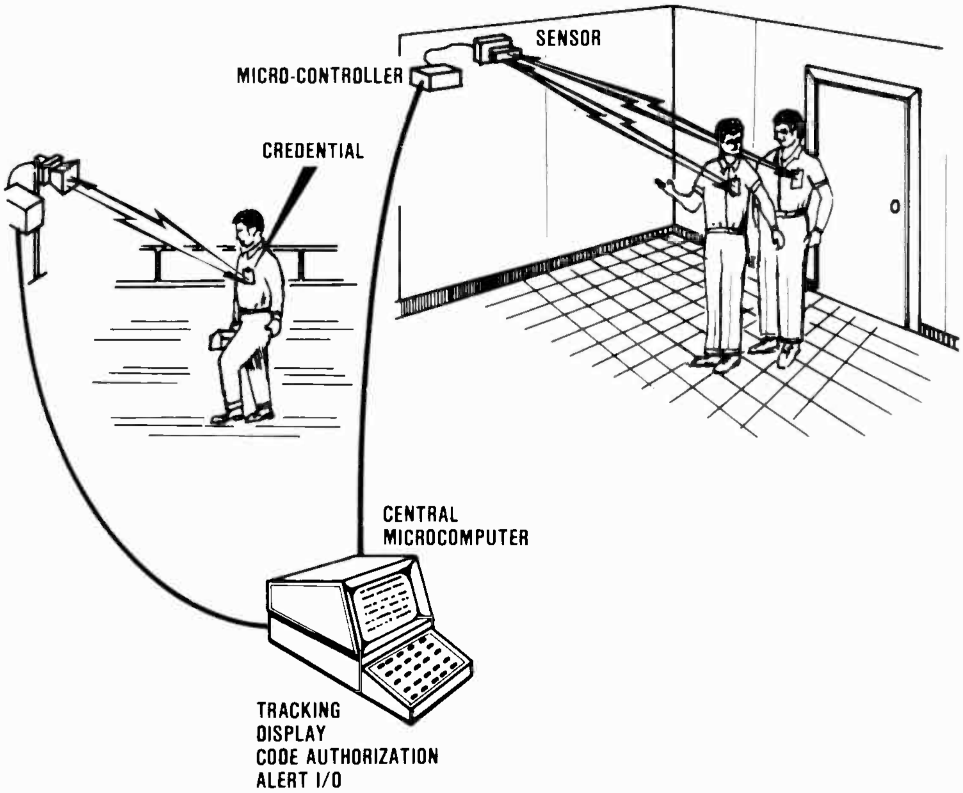


Fig. 1. *Personnel tracking system for use in enclosed and open areas with appropriately placed interrogators connected to a central microcomputer.*

source) and frequency synthesizer for interrogation, and a surface-acoustical-wave (SAW) filter front-end for receiving the credential's responses. Another Z-80 microprocessor is used as an overall system controller to monitor up to eight sensors and relay information to a central control point.

The goals of the tracking system (Fig. 2) were as follows:

- Allow no ambiguity and handle multiple credentials in the field of regard;
- Reliably detect a large population with a minimum of 8000 separate credentials;
- Allow flexibility of use and be able to work in corridors as well as open areas;
- Provide an expandable system with minimum "breakage" required to add areas of coverage;
- Minimize technical risk and avoid component or technology development; and
- Create no health hazard.

In addition to these goals, the final system would have to work in a very harsh environment. This environment includes:

- Effects such as reflections due to totally metallic surroundings;
- A very severe electromagnetic environment that could be produced by many varieties of communication and other electronic equipment, and by machinery;
- A very severe mechanical and acoustical vibration environment with many high-frequency resonances and overtones;
- A highly compartmented physical structure, with isolation between many spaces provided by doors and passageways following complicated zigzag patterns; and
- A very damaging environment.

To ensure that the system would not need major revisions caused by unanticipated conditions, a flexible system design was desired. For example, instead of uniquely identifying each credential by means of a single complex code (8000 or more combinations), the code was broken into three simple subcodes of sixteen, eight and sixty-four combinations apiece. Each of these subcodes is incorporated into a different system parameter. In this case, the total number of unique code combinations will be equal to the product of the number of subcode combinations (8192) so that large numbers of credentials can be uniquely identified. However, since several different parameters of the system are used to encode this information, trade-offs can be made to optimize performance.

Furthermore, since the number of subcode combinations is small, there are few critical constraints or tight tolerances on the system. As a result, modifications, necessitated by unanticipated problems resulting from the harsh environment, may be made without difficulty. For the system prototype, the three subcodes were chosen to be:

1. Sixteen possible modulating frequencies on a 10.5-GHz carrier, broadcast one at a time by the sensor.
2. Eight possible frequencies in the 200-to 300-MHz region, to "acknowledge" reception of the transmitter's signal.
3. Sixty-four possible time slots during which the "acknowledge" can occur.

In operation, a detector on each of the credentials would be activated by one and only one of the sixteen possible frequencies being sequentially broadcast by the transmitter. Once activated, the credential could listen for the end of the transmission (the point at which the transmitter is turned off to await an acknowledge signal). At the end of transmission, a timer is started and after waiting one of sixty-four possible timed delays, the credential broadcasts one of eight possible frequencies to acknowledge its presence. The cycle is then repeated.

Identification is indeed unambiguous, because credentials that respond to the *same* sensor code, one of sixteen frequencies, are polled at *different* times so that they never respond at the same time. Similarly, credentials that respond to the same sensor code, but in different time slots, are also never on simultaneously. Finally, credentials responding to the same sensor code and in the same time slot do so by transmitting simultaneously, but at different frequencies, so that all credentials can be uniquely identified.

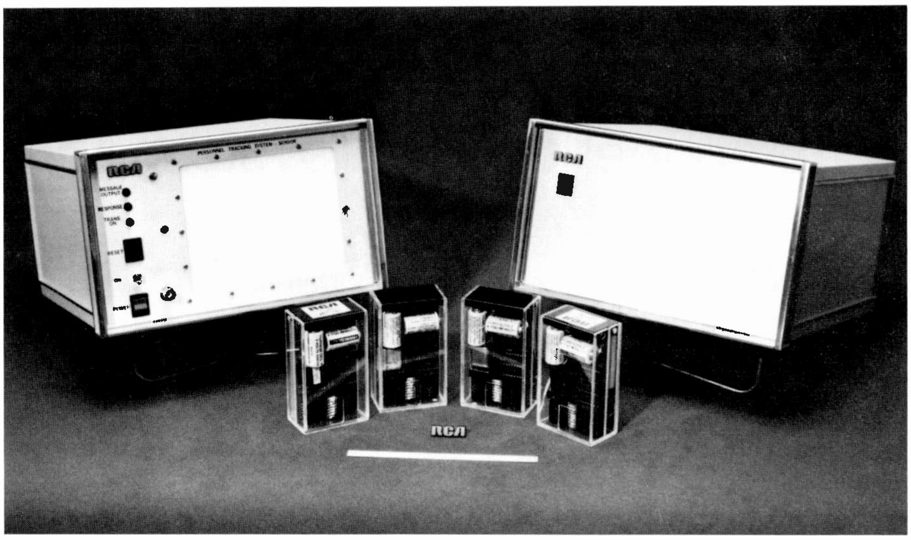


Fig. 2. *Proof-of-principle demonstration model showing all major system components.*

This system has several advantages over other methods such as time-division multiplexing. The polling cycle is shorter—sixteen times sixty-four total time delays (since acknowledgments of up to eight credentials are allowed to be simultaneous). This, in turn, allows longer "listening" times which, in turn, result in better rejection of random and multipath interference (produced by multiple reflections, prior to reception) and noise as well as simpler battery technology.* The latter occurs because peak power drain, as well as average drain can be reduced by allowing weaker credential responses integrated over longer periods of time to obtain the same or better signal-to-noise ratio. The longer transmissions also simplify the synchronization of multiple sensors since the greater time delays reduce the need for accurate timing. Also, digital circuits on the credential can be run at lower speeds, thus requiring less power. Most important of all, this concept allows engineers to make numerous trade-offs, such as increasing the number of combinations for one subcode while reducing another, or reducing the length of the time slots while increasing the polling frequency. The number of "acknowledge" frequencies used by the credentials can also be reduced so that wider frequency spacing can be obtained within a specified band. This would reduce the requirement for accurately maintaining stable frequency control, to ensure minimal overlap of adjacent signals.

* Credentials which are powered by radiation absorbed from the sensor, rather than a battery, were deemed impractical due to the low sensor-power output, dictated by safety considerations.

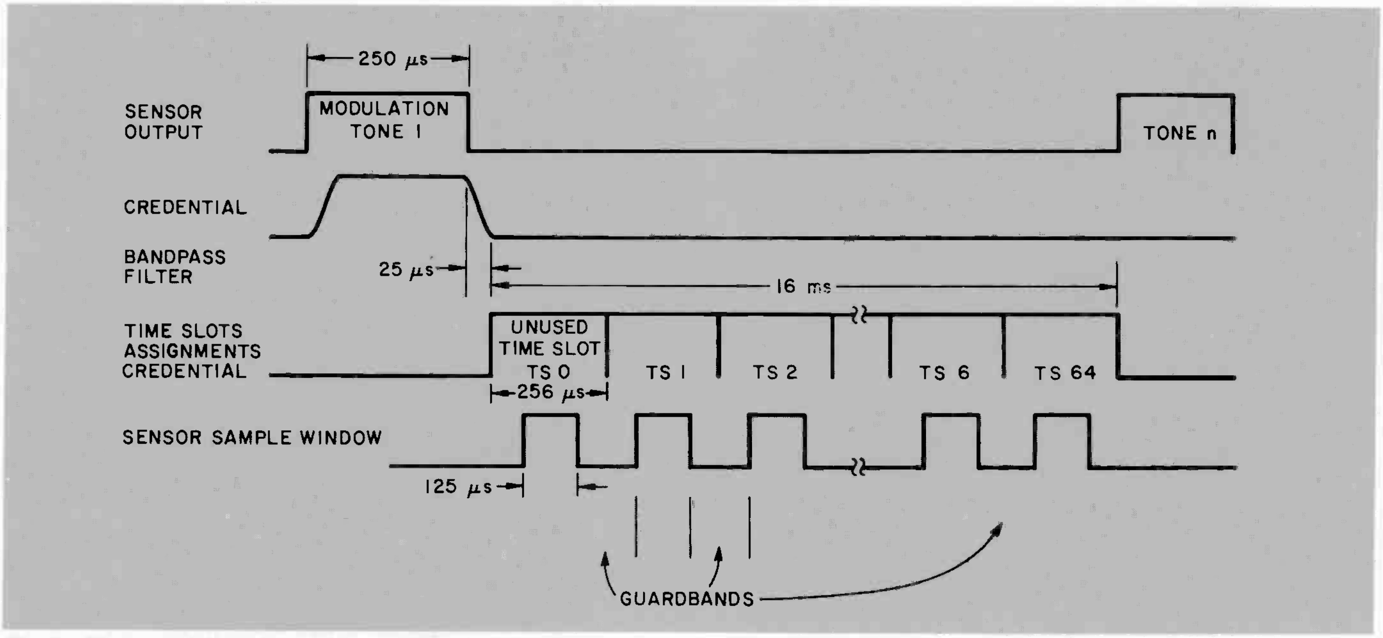


Fig. 3. *Timing relationships of control tones.*

The system described above has several additional features and capabilities that are interesting to note. First, since there are no critical timing or synchronization requirements, low-cost serial communication links can be used for all interconnections. Second, the carrier frequencies broadcast by the transmitters could also function as Doppler-motion detectors (to detect personnel without credentials) at very little additional cost.

Third, the three-level coding of identities provides a natural and orderly way of establishing access priorities. Thus, if only a small portion of the population is to be authorized to enter a particular area, this group can be assigned to a special frequency or time slot, and so on. The notions of keys, sub-master keys, and master keys can be transcribed in a natural way into the proposed coding. In fact, this could be done at the level of local controllers instead of at the central computer.

Fourth, as checks for counterfeit credentials, the central computer that tracks the personnel can look for invalid or duplicate codes as well as test for impossibly fast motion of a credential from one point to another. Furthermore, since the proposed polling cycle can be made quite fast (by shortening the time slots), this "excess" polling speed can be used to greatly increase the number of allowable codes. Then, in an application where only a very small fraction of the possible codes are actually used, a counterfeiter is not likely to discover a valid code by simple trial-and-error methods, particularly if the errors result in a computer-generated warning.

Timing

Each credential is programmed to monitor one of sixteen modulated tones on a 10.5-GHz carrier and transmit pulses on one of eight preassigned frequencies in the 200- to 300-MHz band. The pulse duration that can occur in one of sixty-four time slots is 256 µs. Hence, the total time for a polling sequence (including one dummy time slot) is 16.64 ms. This is repeated once for each of the sixteen polling frequencies for a total time of 270.336 ms, or about four scans per second.

To maintain synchronization between all of the system components, the timing logic within the credentials is slaved to the interrogating sensor's timing. This is accomplished by using the modulated tone burst generated by the sensor as an enable signal in the credentials. The clock circuits in the credential are held inactive until the trailing edge of the tone burst is detected. The transition enables the counters and the timing sequence is started. Since the timing circuits in the sensor are also enabled at the end of the tone generation, all units will be started simultaneously. To compensate for possible errors due to filter time constants, clock phasing and so forth, each time slot as decoded by the sensor is divided into three segments (Fig. 3). The first and last segments are used to guard bands. During this interval, no tests are made. The center segment is the only portion used to test for a valid response. This segment is sampled at a 1-MHz rate by the sensor. For each valid sample, a counter is incremented. At the end of the assigned time slot, the counter's output is examined and if it exceeds a threshold, then a valid response has been detected. Since each polling sequence is 16.64 ms in duration, clock drift as high as one part in a thousand can be tolerated.

Implementation

As mentioned, the system consists of three major components: a credential, or badge, that is worn by each user; sensors or interrogators, the number of which is determined by the platform that is used; and the controller (one controller can service up to eight sensors). We will describe each of these components, with the major emphasis being placed on the credential.

Credential hardware

The configuration of the credential is fairly simple. This subsystem consists of a diode detector, and an amplifier followed by a threshold detector, which then feeds the digital logic blocks (filters, counters). The combination of the detector and digital filter determines if a response is required. If it is, the logic enables the transmitter at the proper time. The hardware requirements for the credential are minimal and can be realized with a single LSI circuit.

Credential microwave receiver

The function of the credential receiver is to detect the tone-modulated microwave signal from the sensor transmitter and provide a drive signal to the credential logic. To accomplish this function, the credential

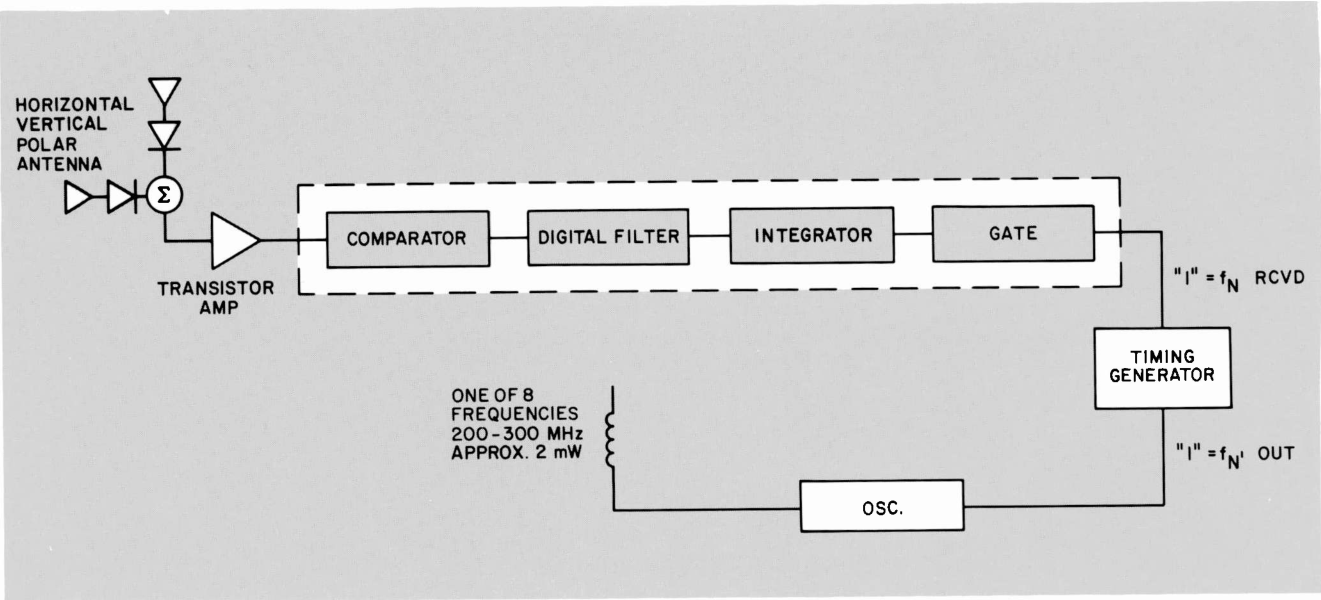


Fig. 4. *The digital filter used in the front end of the credential consists of three one-shots (the outputs of which form a digital bandpass filter) and miscellaneous logic circuits. Two of the one-shots are used to set the bandpass of the filter. Normally, this is set for ±4 kHz around the nominal with a transition frequency of about 500 Hz. The other unit is used as a time out to reset the "filter."*

Each credential can generate a response on any one of sixty-four time slots, programmed via a set of switches mounted on the credential. The switches are binary coded from zero to sixty-three. Ultimately, this assignment would be made in a ROM. Each time slot is 256-μs wide and is generated whenever an inband signal is detected by the passband filter.

When an input signal is within the passband of the filter, the clock to the time slot counters is enabled, jamming the counters to their assigned values. When the filter output goes low, the transmitting tone from the sensor has been turned off, and two counters start to count. They count down until the programmed interval is reached, at which time the output transmitter is enabled for 256 μs. On the falling edge of the transmitter enable, the counters are reset and the system returns to its standby state.

receiver consists of a receiving antenna, microwave diode detectors, and a stage of transistor amplification. A schematic representation of the credential receiver is shown by Fig. 4.

The credential antenna is a dual-element microstrip patch antenna in which the two receiving elements are cross-polarized. With this dual-polarization arrangement, the total detected signal is relatively independent of the orientation of the transmitter antenna with respect to the credential. The credential antenna is less directive than the sensor transmitter antenna but the two must be generally facing each other for maximum reception range. Each element is connected to a separate detector diode but these diodes are connected in series so as to sum the detected voltages from the rf signals incident upon the respective antenna element. A plot of output signal versus distance is shown in Fig. 5.

Credential transmitter

The function of the credential transmitter is to send one of eight VHF frequencies back to the sensor at a time dictated by the timing of the credential's logic circuitry. At the proper time slot after the credential is interrogated with the tone for which it is set, the logic circuit "gates" the credential transmitter "on" for 256 μs. During this time, the transmitter generates a cw VHF signal at a preset frequency between 200 and 300 MHz and transmits it to the sensor using the meander line antenna, which is on the same side of the credential PC board as are the two receiving patch antennas. This particular frequency

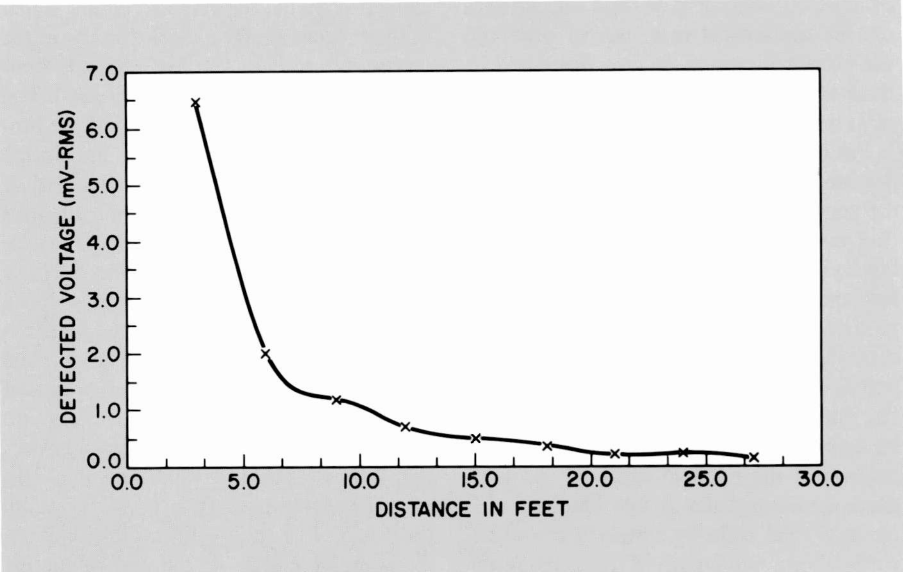


Fig. 5. *Credential characteristics describing system sensitivity.*

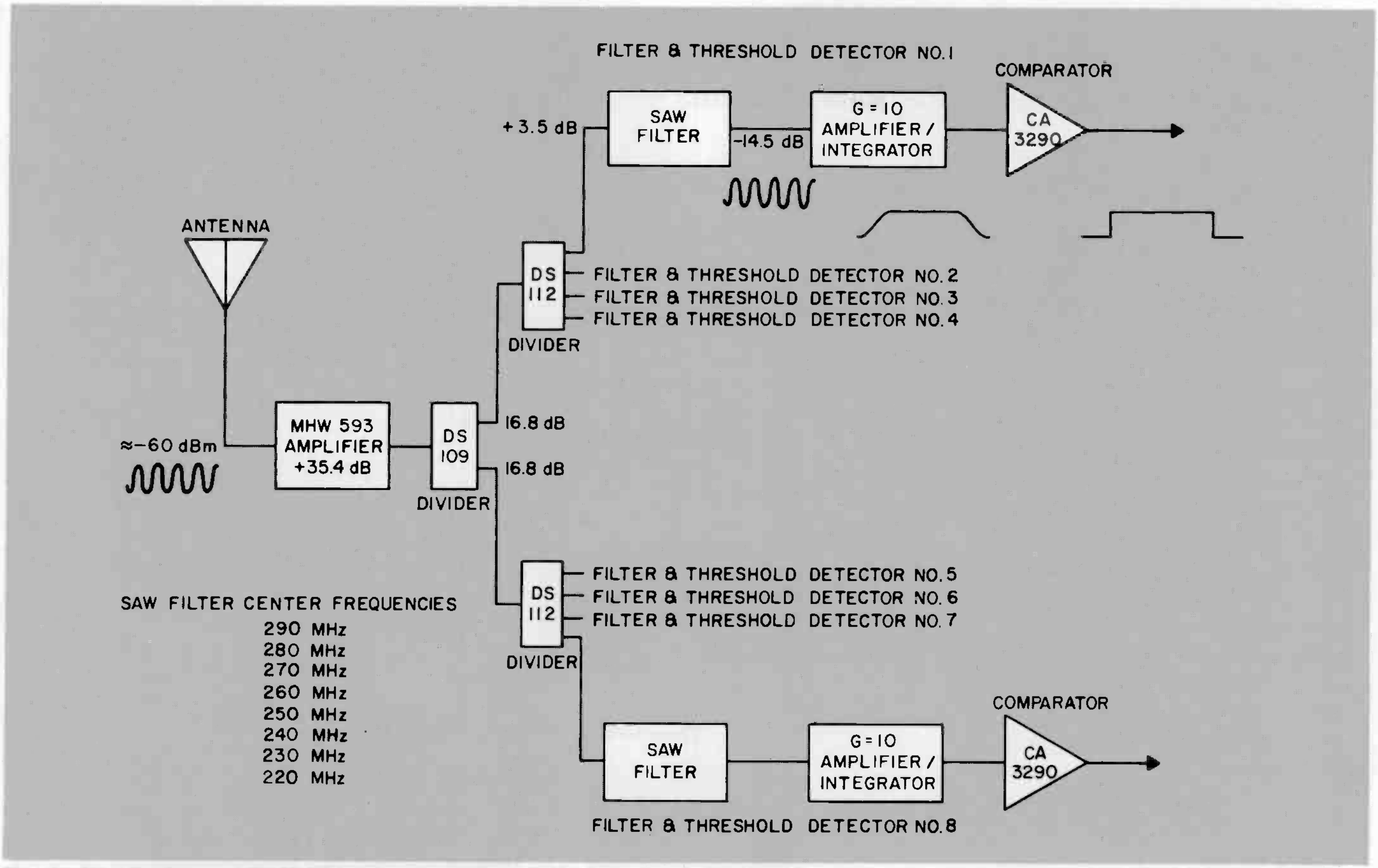

FILTER & THRESHOLD DETECTOR NO. 1
COMPARATOR
+3.5 dB
SAW FILTER
-14.5 dB
G = 10 AMPLIFIER / INTEGRATOR
CA 3290
ANTENNA
DS 112
FILTER & THRESHOLD DETECTOR NO. 2
FILTER & THRESHOLD DETECTOR NO. 3
FILTER & THRESHOLD DETECTOR NO. 4
DIVIDER
≈-60 dBm
MHW 593 AMPLIFIER +35.4 dB
DS 109
16.8 dB
16.8 dB
DIVIDER
DS 112
FILTER & THRESHOLD DETECTOR NO. 5
FILTER & THRESHOLD DETECTOR NO. 6
FILTER & THRESHOLD DETECTOR NO. 7
DIVIDER
SAW FILTER CENTER FREQUENCIES
290 MHz
280 MHz
270 MHz
260 MHz
250 MHz
240 MHz
230 MHz
220 MHz
COMPARATOR
SAW FILTER
G = 10 AMPLIFIER / INTEGRATOR
CA 3290
FILTER & THRESHOLD DETECTOR NO. 8

Fig. 6. *The 8-channel SAW filter receiver subsystem.*

range was selected for the credential's return transmission as a reasonable trade-off between simplicity and antenna size.

Power requirements

The credential operates from a 5.6-V power source that is provided by two lithium cells with a capacity of 1000 milli-amp hours. The detector circuits, amplifier and threshold comparator, operate directly from the 5.6-V supply. The digital logic circuits and output transmitter operate from a diode drop below or a 4.9-V supply. The average drain of the credential, when operating at four scans per second, is 1.2 mA, with the analog front end requiring 75 percent of the total.

By customizing the analog front-end, particularly the comparator, the power requirements can be reduced to the point whereby a credential can be operated for about three months before recharging or replacing the battery. Also, the final size of a credential can be reduced to that of a badge 3 by 2 inches and ¼-inch thick.

Sensor

The sensor (transmitter/receiver) is the control element for each area of coverage. The sensor is comprised of a 10.5-GHz transmitter, an eight-channel VHF receiver, and a Z80 microprocessor.

Sensor transmitter

The sensor transmitter consists of a coaxial-cavity transferred electron-effect oscillator, a ferrite isolator, and a printed-circuit antenna array.

The transferred-electron-effect oscillator uses a bulk-effect GaAs diode, commonly known as a Gunn diode, which when suitably coupled to a resonant circuit, oscillates at a microwave frequency with the application of a relatively low dc voltage. A simple coaxial cavity provides the resonant circuit, and frequency adjustment is obtained by a tuning screw in which is mounted a rutile rod for temperature compensation of the operating frequency. The output of the oscillator is probe coupled and connected to the transmitting antenna through a ferrite isolator to improve load stability. The antenna is a printed-circuit dipole array that provides a directive radiation pattern with a 3-dB beamwidth of approximately 20 degrees. It was considered that this pattern was a reasonable compromise for most installations which require looking down corridors or passageways. The printed-circuit material used for the antenna is a copper-clad fiberglass-reinforced teflon board held securely in place by a low-loss potting material.

The sensor transmitter is mounted behind the front panel of the sensor and the antenna is covered by a ¼-inch thick piece of white polystyrene for protection and appearance purposes. The sensor's transmitter was adjusted to produce approximately 50-mW cw output at 10.5 GHz with +10 VDC applied to the oscillator. In actual operation, the oscillator is driven by a square wave at the various modulation rates between the +5 V and +10 V levels to avoid pulsing through the Gunn-diode threshold voltage.

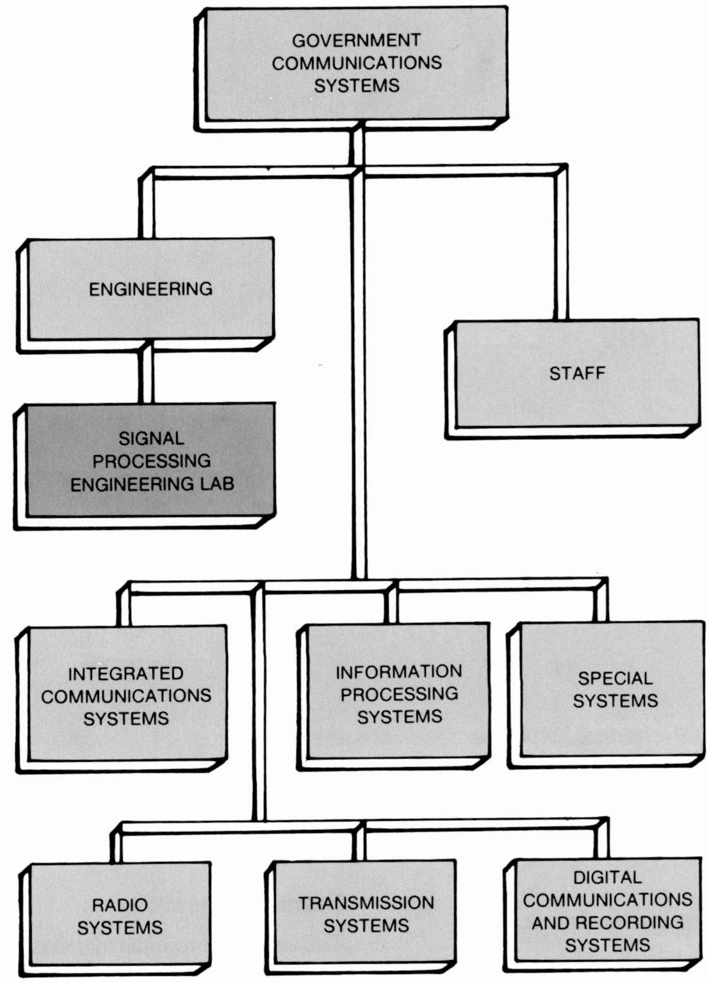
Eight-channel receiver subsystem

A signal from a credential device 15-feet away measures −60 dBm at the antenna (Fig. 6). Two power amplifiers (power gain of 35.4 dB each) amplify the signal. The signal is then evenly split into eight signals. Each of these outputs drives a surface-acoustical-wave (SAW) filter. Each filter has 18.2-dB insertion loss, a −55-dB bandwidth of 4.3 MHz, and a −3-dB bandwidth of 1.3 MHz. The center frequency of channel eight is 300 MHz, and each

Signal Processing Engineering Laboratory of Government Communications Systems

The personnel-tracking system described in this paper is only part of the story at the 14-year-old Signal Processing Engineering Laboratory in Somerville. The Lab supports technology development in several Government Communications Systems (GCS) business areas.

The group, perhaps best known for developing low-power microsignal processors, shows feasibility and approach in a number of programs, including signal intelligence applications, classifiers for intelligence/fuzing, advanced low-power CMOS implementation for "smart sensors," electronic-support-measures signal processors, and more. Their engineering work is then transferred to Government Communications Systems skill centers in Camden where engineers make the system, equipment, or component producible (see chart).


GOVERNMENT COMMUNICATIONS SYSTEMS
ENGINEERING
STAFF
SIGNAL PROCESSING ENGINEERING LAB
INTEGRATED COMMUNICATIONS SYSTEMS
INFORMATION PROCESSING SYSTEMS
SPECIAL SYSTEMS
RADIO SYSTEMS
TRANSMISSION SYSTEMS
DIGITAL COMMUNICATIONS AND RECORDING SYSTEMS

Initiated in 1969, the Lab's 15-member team headed by Kal Prost who reports to Dan Hampel's Skill Center, which is part of Jim Fayer's Camden GCS Engineering Department, concentrates on applying the rapid advances being made in commercial LSI components at the Solid State Division in Somerville and the microwave advances coming from the Microwave Technology Center in Princeton. The team maintains signal-processing skills, signal-analysis and other analytical skills, and computer hardware-and software-development expertise.

Staff members interface with both device-research and advanced-systems groups to assure successful and creative solutions to military systems problems. Signal Processing Engineering has successfully accomplished signal analysis and algorithm development for the detection, identification, and classification of patterns applicable to seismic, acoustic, and rf systems. These engineers specialize in low-cost, low-power, expendable systems, very much like the one described in this paper. "Smart sensors" are based on a microsignal processor technique that can identify, detect, and/or classify particular target signatures in real time, with low-power dissipation.

—*MRS*

channel thereafter is separated by 10 MHz down to 230 MHz for channel one. Each channel has a shielded detector formed by a resistor network and a Schottky-barrier diode. The detected output drives a high-impedance operational amplifier followed by a comparator circuit.

Thus, a short burst of high-frequency signal is detected by the appropriate SAW filter and diode detector amplifier, and results in a logic-level signal of duration and time-slot location identifying the transmitting credential.

Physically, all of the circuitry described above (Fig. 6) is on a shielded circuit card in the rack next to the power supply, with the exception of the two power amplifiers and the first power splitter. These devices are mounted with a good heat-sink arrangement near the receiver antenna and transmitter.

Sensor-control microprocessor

The sensor controller consists of a microprocessor, random-access memory (RAM) for data storage, read-only memory (ROM)

Dan Mawhinney joined RCA in 1952 working in the Microwave Equipment Development group in Harrison, N.J. He later became involved with the design and development of various solid-state microwave oscillators and subsystems and served as Engineering Manager of the Microwave Solid State Design group before transferring to RCA Laboratories, Princeton, N.J., in 1975. As a Member of Technical Staff of the Microwave Technology Center, he continues to work primarily on the design, development, and prototype manufacture of small military and commercial microwave subsystems. Mr. Mawhinney received the BEE degree from the Polytechnic Institute of Brooklyn and the MSEE from Newark College of Engineering.

Contact him at:
RCA Laboratories
Princeton, N.J.
TACNET: 226-2802

for program storage, and input/output (I/O) devices. The I/O is used to select the required output tone, monitor the receiver outputs and form messages for transmittal to the controller.

Controller

The controller (which processes information from a number of sensors) uses the same type of processor (Z80) as the sensor. Its main function is to display ID numbers, status, entering or leaving area, and time. In the final system, the controller will interface with a host computer that is part of the overall physical security system.

Conclusion

The combination of high-efficiency, low-power-dissipation circuits, application of a single state-of-the-art LSI chip, and the system's relatively low-duty cycle, makes the active-rf credential approach optimum from a reliability and performance viewpoint. Coupled with a detection-mode capability in a compatible sensor-radar configuration, the system will detect intrusions, as well as track personnel.

Authors (left to right) Hampel, McEachern, and Prost.

Daniel Hampel, Manager, Computer-Controlled Equipment Engineering, Government Communications Systems holds a BSEE and MSEE from Newark College of Engineering. Mr. Hampel has been a Section Manager in the Engineering Department since 1981, and is responsible for the operation of four Skill Centers, as well as the Signal Processing Engineering group collocated with the Solid State Division in Somerville, N.J. These Skill Centers, with over 65 engineers and unit managers, provide the design and development engineering for the digital equipment for classified programs, communications and EW systems, and applications of signal processing technology.

Prior to this assignment, he was the manager of the Signal Processing Engineering group in Somerville. There, he was responsible for providing new technology for the major business of Government Communications Systems. He managed a number of techniques contracts and contributed to advancing the state-of-the-art in custom LSI, low-power dissipation micro-signal processing, and special-purpose recognition processors. Earlier work included R&D in the areas of threshold logic and radiation-hardened circuit development.

Contact him at:
Government Communications Systems
Camden, N.J.
TACNET: 222-5918

Robert McEachern, Unit Manager, Signal Processing Engineering (SPE), received an M.S. degree in Physics from Michigan State University in 1974. Prior to joining RCA, Mr. McEachern was employed as a geophysicist in the oil-exploration industry.

Since joining Signal Processing Engineering of GCS in 1979, Mr. McEachern has been involved in a variety of signal processing projects, particularly in the areas of signal intelligence and smart sensors. He has been responsible for algorithm development for munitions sensors and communications emitter identification for both IR&D and techniques contracts.

Contact him at:
Government Communications Systems
Somerville, N.J.
TACNET: 325-6943

Kal J. Prost, Manager, Signal Processing Engineering (SPE), attended RCA Institutes, 1957 to 1958, and studied engineering at Hofstra University, City University of New York and Newark College of Engineering.

Since joining RCA in 1966, he has been involved in design and testing of integrated threshold gates, digital systems, and radiation-hardened circuits. He has been the lead engineer on special purpose logic modules for signal processing and has designed high-speed arithmetic units for computer interface, and the hardware for microprocessor-based classifiers for mine fuzing and sensor systems. He has applied both custom LSI and standard parts to micro-signal processing equipment.

Mr. Prost has been the Manager of the Signal Processing Engineering group since 1981. He has the responsibility for providing advanced signal processing technology to the major business areas of Government Communications Systems.

Contact him at:
Government Communications Systems
Somerville, N.J.
TACNET: 325-6365

Homing in on the home computer—practical applications

Lee E. Kitchens
RCA Missile and Surface Radar
Moorestown, N.J.

A revolution is changing the lives of everyone—the computer has entered the home. Engineers, especially, are welcoming the "information era" by equipping themselves with a new tool. The personal computer cuts the umbilical cord binding users to the mainframe and brings the processing power of the laboratory and the office right into the home. How can practical engineers use the home computer?

First, a definition. The personal computer is a microprocessor-based, general purpose, digital computer dedicated to the individual user. It must be able to read a program or data from self-contained storage, modify the program or data communicating with the user, and store the program or data for later use. The effective personal computer, then, must include at least some form of mass long-term storage, such as magnetic tape or disk. As a whole, the serviceable home computer consists of the following:

- a keyboard for entry of information,
- a screen for visual display of information to the user,
- a magnetic tape or disk memory system, and
- a microprocessor, internal memory, and input/output provisions—the computer itself.

The operator can apply this machine to a great many technical and business problems, but the most widely used applications are entertainment, education purposes, and word processing. Balancing a checkbook is not a practical application for a personal computer. And keeping a list of names in a data base is more difficult to do than keeping them on a few index cards. The personal computer becomes useful in cases where a large number of items need updating frequently, for example, a personal accounting system designed to give the user an end-of-the-year report on where all his money went (tax information).

Programs

A core library of application programs for the personal computer consists of the following:

Word Processing—for writing, editing, changing, and storing letters, reports, journal articles, and symposium papers.

Electronic Spread Sheet—for performing multiple calculations on a matrix. Such an application might be a table of bowling league scores.

Data Base Management—for storing, filing, sorting, and retrieving data much as one would in a file cabinet. Applications might include address lists, investment records, or an article index.

Financial Analysis—for tracking personal income: expenditures, credit card purchases and payments, and savings accounts; then producing useful analysis reports on the information.

Entertainment—for just having fun. Some programs, demonstrating the ability of the computer, pit the machine against the user in chess, poker, blackjack, bridge or rummy. Arcade-type games are, of course, also available.

Application examples

Business expense reports

Filling out an expense report at the end of a long business trip is hardly enjoyable—this is especially true after an international trip, with expenses and receipts mixed in U.S. and foreign currencies. An "Electronic BER" alleviates the headache. The RCA Business Expense Report, shown in Fig. 1, consists of two interrelated matrices: the summary matrix on the front page and the meals matrix on the reverse.

To complete the BER electronically, the engineer establishes a BER data base consisting of these two matrices and defines the equations that relate the items in each matrix. The traveler includes two entries for each item to allow input of the expense item value in either U.S. dollars or foreign currency.

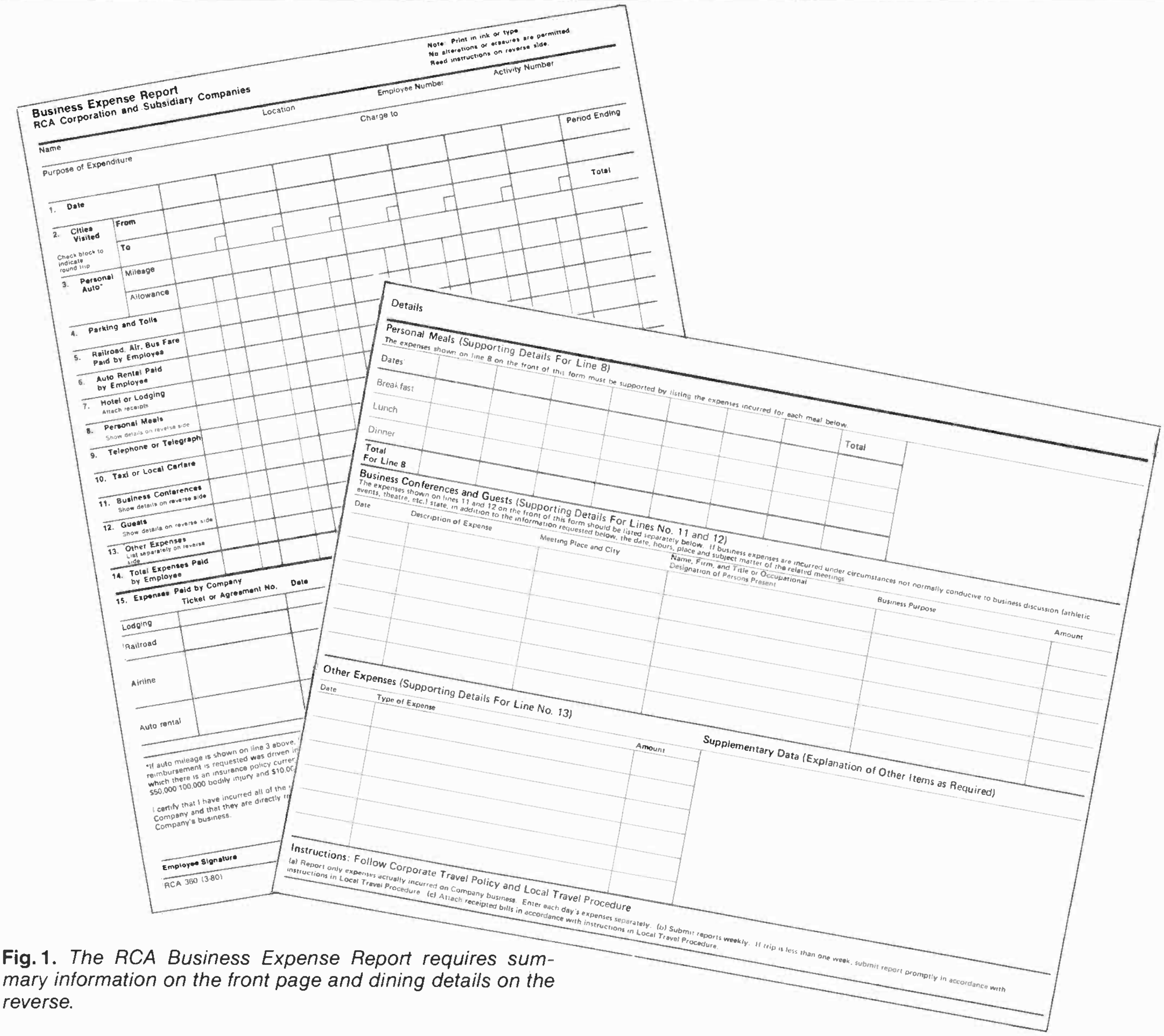
Figure 2 is a printout of the report generated by the program. This same format is displayed on the computer screen during input of the information on the keyboard. Notice the mileage rate, $0.20 per mile, in the upper left corner. The foreign currency conversion factor and the cash advance are supplied in the lower left and right corners, respectively.

With the trip receipts and expense data in hand, the engineer enters meal information into the meals matrix. The computer converts foreign currency inputs and displays them immediately in U.S. dollars. The meal inputs are computed, summarized, and automatically entered into the MEALS slot on the summary matrix. When all entries have been made, the user prints out a hard copy and transfers the information to the standard RCA form.

Word processor

Using the personal computer as a word processor (better termed "text processor") is its most universal and practical application. A word processor is, in effect, an electronic typewriter.

Final manuscript received October 19, 1983.
Reprint RE-28-6-Technical Note



Note: Print in ink or type
No alterations or erasures are permitted
Read instructions on reverse side.

Business Expense Report
RCA Corporation and Subsidiary Companies

Employee Number
Activity Number
Location
Charge to
Period Ending
Name
Purpose of Expenditure
Total

1. Date
2. Cities Visited
From
To
Check block to indicate round trip
3. Personal Auto*
Mileage
Allowance
4. Parking and Tolls
5. Railroad, Air, Bus Fare Paid by Employee
6. Auto Rental Paid by Employee
7. Hotel or Lodging
Attach receipts
8. Personal Meals
Show details on reverse side
9. Telephone or Telegraph
10. Taxi or Local Carfare
11. Business Conferences
Show details on reverse side
12. Guests
Show details on reverse side
13. Other Expenses
List separately on reverse side
14. Total Expenses Paid by Employee
15. Expenses Paid by Company
Ticket or Agreement No.
Date
Lodging
Railroad
Airline
Auto rental

*If auto mileage is shown on line 3 above, reimbursement is requested was driven in which there is an insurance policy curren $50,000 100,000 bodily injury and $10.00

I certify that I have incurred all of the Company and that they are directly r Company's business.

Employee Signature
RCA 360 (3-80)

Details

Personal Meals (Supporting Details For Line 8)
The expenses shown on line 8 on the front of this form must be supported by listing the expenses incurred for each meal below.
Dates
Breakfast
Lunch
Dinner
Total For Line 8
Total

Business Conferences and Guests (Supporting Details For Lines No. 11 and 12)
The expenses shown on lines 11 and 12 on the front of this form should be listed separately below. If business expenses are incurred under circumstances not normally conducive to business discussion (athletic events, theatre, etc.) state, in addition to the information requested below, the date, hours, place and subject matter of the related meetings.
Date
Description of Expense
Meeting Place and City
Name, Firm, and Title or Occupational Designation of Persons Present
Business Purpose
Amount

Other Expenses (Supporting Details For Line No. 13)
Date
Type of Expense
Amount

Supplementary Data (Explanation of Other Items as Required)

Instructions: Follow Corporate Travel Policy and Local Travel Procedure
(a) Report only expenses actually incurred on Company business. Enter each day's expenses separately. (b) Submit reports weekly. If trip is less than one week, submit report promptly in accordance with instructions in Local Travel Procedure. (c) Attach receipted bills in accordance with instructions in Local Travel Procedure.

Fig. 1. *The RCA Business Expense Report requires summary information on the front page and dining details on the reverse.*

Numerous word processor programs are available for every personal computer, with catalog names such as "Electric Pencil," "Lazy Writer," "Word Star," "New Script," and "Scripsit." Although features and command instructions vary, they all take keyboard text entry, display the text on the computer display, allow editing of the text (corrections, additions, deletions, block text moves), and print a hard copy with formatting and margin justification.

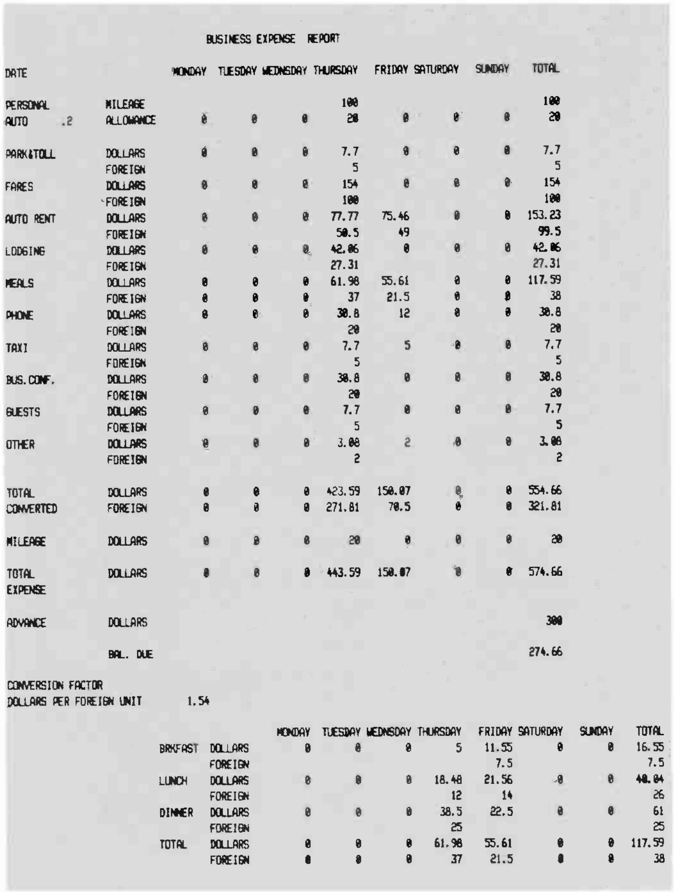
This writer uses two different word processor programs. The first is a simple, easy-to-use, and "natural-typing" program for most personal uses. It is ideal for letters and for draft text that will be formatted and typeset elsewhere.

The second program, a full-featured one, makes type size and style changes, centers titles, generates an automatic index, paginates, and formats variable pages. Figure 3(a) is a simple example of the raw input for a personal letter. Note the embedded, invisible-to-the-printer commands (*.sp 2* tells the printer to skip two lines). Figure 3(b) is the result of the processor's formatting function. Notice the resultant line spacing and the neat justified right margin.

Engineers are proficient at many things, but spelling isn't often one of them. A spelling checker adds a helpful dimension to a personal computer word processor. Among the several that operate on personal computers are: the Electric Webster, HEXSPELL, CHEX TEXT, and Perfect Speller.

The spelling checker operates by comparing the words in the working document against the words contained in its built-in dictionary. All words that are not found are listed out on the screen or the printer. This list of words becomes a list of "potential errors"—only "potential" because a word will appear if it is spelled correctly but not contained in the dictionary. The original dictionary does not contain proper names, irregular plural forms of nouns, and many technical words.

The Electric Webster spelling checker, contained on one 5¼-



```
                          BUSINESS EXPENSE  REPORT

DATE                   MONDAY  TUESDAY WEDNSDAY THURSDAY  FRIDAY SATURDAY  SUNDAY   TOTAL

PERSONAL     MILEAGE                                100                              100
AUTO      .2 ALLOWANCE      0       0       0        20        0       0       0      20

PARK&TOLL    DOLLARS        0       0       0       7.7        0       0       0     7.7
             FOREIGN                                  5                                5
FARES        DOLLARS        0       0       0       154        0       0       0     154
             FOREIGN                                100                              100
AUTO RENT    DOLLARS        0       0       0     77.77    75.46       0       0  153.23
             FOREIGN                               50.5       49                    99.5
LODGING      DOLLARS        0       0       0     42.06        0       0       0   42.06
             FOREIGN                              27.31                            27.31
MEALS        DOLLARS        0       0       0     61.98    55.61       0       0  117.59
             FOREIGN        0       0       0        37     21.5       0       0      38
PHONE        DOLLARS        0       0       0      30.8       12       0       0    30.8
             FOREIGN                                 20                               20
TAXI         DOLLARS        0       0       0       7.7        5       0       0     7.7
             FOREIGN                                  5                                5
BUS.CONF.    DOLLARS        0       0       0      30.8        0       0       0    30.8
             FOREIGN                                 20                               20
GUESTS       DOLLARS        0       0       0       7.7        0       0       0     7.7
             FOREIGN                                  5                                5
OTHER        DOLLARS        0       0       0      3.08        2       0       0    3.08
             FOREIGN                                  2                                2

TOTAL        DOLLARS        0       0       0    423.59   150.07       0       0  554.66
CONVERTED    FOREIGN        0       0       0    271.81     70.5       0       0  321.81

MILEAGE      DOLLARS        0       0       0        20        0       0       0      20

TOTAL        DOLLARS        0       0       0    443.59   150.07       0       0  574.66
EXPENSE

ADVANCE      DOLLARS                                                                 300

             BAL. DUE                                                             274.66

CONVERSION FACTOR
DOLLARS PER FOREIGN UNIT      1.54

                      MONDAY  TUESDAY WEDNSDAY THURSDAY   FRIDAY SATURDAY  SUNDAY   TOTAL
         BRKFAST DOLLARS   0        0        0        5    11.55        0       0   16.55
                 FOREIGN                                     7.5                      7.5
         LUNCH   DOLLARS   0        0        0    18.48    21.56        0       0   40.04
                 FOREIGN                             12       14                       26
         DINNER  DOLLARS   0        0        0     38.5     22.5        0       0      61
                 FOREIGN                             25                                25
         TOTAL   DOLLARS   0        0        0    61.98    55.61        0       0  117.59
                 FOREIGN   0        0        0       37     21.5        0       0      38
```

Fig. 2. *The output of the "Electronic BER" program lists expenditures in U.S. dollars and foreign currency.*

inch floppy disk, will proof a typical two-page document in 30 seconds.

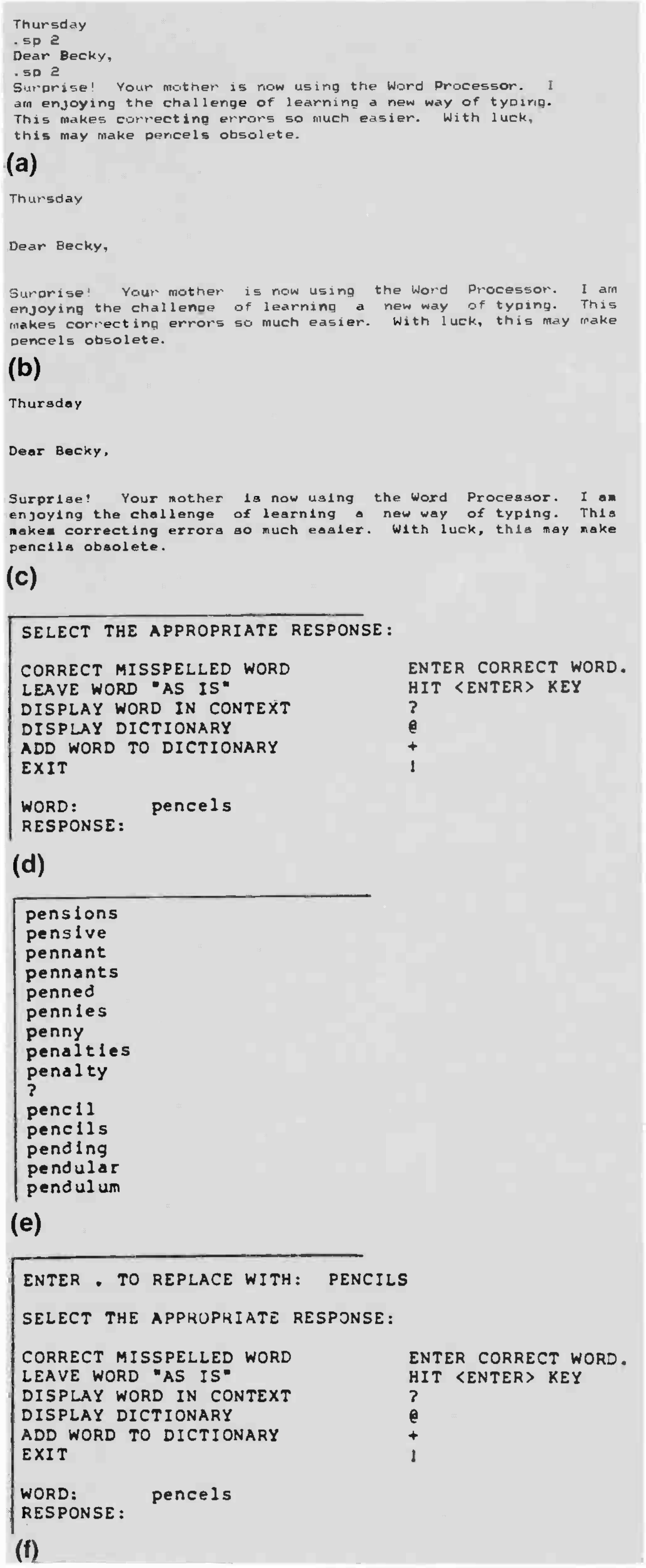
Having identified the potential spelling errors in the text, the program gives the user five options for each identified word (see Fig. 3(d)):

1. Type in the correct spelling.
2. Leave the word "as is," if it is a "new" word.
3. Display the word within the text where it is used.
4. Display the dictionary around the word (see Fig. 3(e)); a "?" symbol indicates where the word would be located in the dictionary.
5. Add the word to the dictionary.

In the example above, "pencils" is spelled incorrectly as "pencels."

The spelling checker found this error and displayed it in Fig. 3(d). The operator entered the @ symbol to call up the dictionary; the computer displayed Fig. 3(e) with the "?" symbol where "pencels" would be in the list. The writer then selected "pencils" as the correct spelling, producing the display in Fig. 3(f). Lastly, the computer replaced "pencels" with "pencils" in the text. Figure 3(c) shows the final printing with the error automatically corrected.

Other automatic features are available for use with word pro-



```
Thursday
.sp 2
Dear Becky,
.sp 2
Surprise!  Your mother is now using the Word Processor.  I
am enjoying the challenge of learning a new way of typing.
This makes correcting errors so much easier.  With luck,
this may make pencels obsolete.
```

(a)

```
Thursday


Dear Becky,


Surprise!   Your mother  is now using  the Word  Processor.  I am
enjoying the challenge  of learning  a  new way  of typing.  This
makes correcting errors so much easier.  With luck, this may make
pencels obsolete.
```

(b)

```
Thursday


Dear Becky,


Surprise!   Your mother  is now using  the Word  Processor.  I am
enjoying the challenge  of learning  a  new way  of typing.  This
makes correcting errors so much easier.  With luck, this may make
pencils obsolete.
```

(c)

```
SELECT THE APPROPRIATE RESPONSE:

CORRECT MISSPELLED WORD            ENTER CORRECT WORD.
LEAVE WORD "AS IS"                 HIT <ENTER> KEY
DISPLAY WORD IN CONTEXT            ?
DISPLAY DICTIONARY                 @
ADD WORD TO DICTIONARY             +
EXIT                               !

WORD:       pencels
RESPONSE:
```

(d)

```
pensions
pensive
pennant
pennants
penned
pennies
penny
penalties
penalty
?
pencil
pencils
pending
pendular
pendulum
```

(e)

```
ENTER . TO REPLACE WITH:  PENCILS

SELECT THE APPROPRIATE RESPONSE:

CORRECT MISSPELLED WORD            ENTER CORRECT WORD.
LEAVE WORD "AS IS"                 HIT <ENTER> KEY
DISPLAY WORD IN CONTEXT            ?
DISPLAY DICTIONARY                 @
ADD WORD TO DICTIONARY             +
EXIT                               !

WORD:       pencels
RESPONSE:
```

(f)

Fig. 3. *A paragraph evolves. The operator inputs commands to achieve the required spacing and uses the spelling checker to clean up the copy.*

cessors that are equally impressive—grammar checkers look for the 15 most prevalent grammatical errors, and hyphenation insertion programs adjust line justification while correctly hyphenating words.

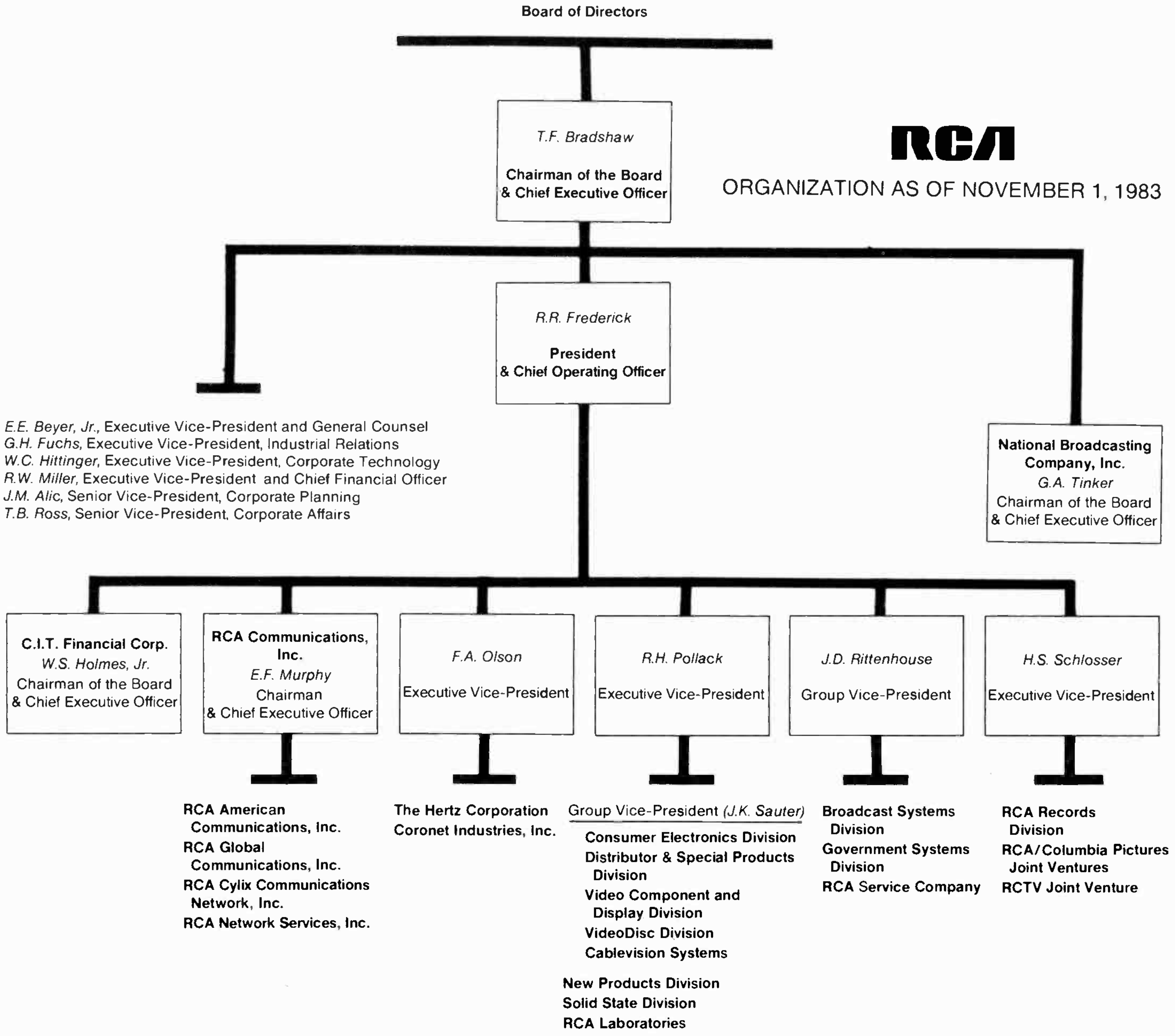
Conclusion

Owners of personal computers should tailor their programs to satisfy their own specific requirements. For example, by using a data management program, the user can maintain a cross-reference index of a phonograph record collection. The computer will search for and display the title of the particular album that contains a desired composition.

Similarly, the operator can create an index of favorite recipes by programming the computer to search for and display the desired cookbook page. For a work-related application, a radar design engineer may wish to enter random part failure data from several radars in the field into another simple data management program. Then, it becomes a matter of merely sorting the data and extracting printouts of failure trends of a particular part or a certain radar.

To make the computer investment pay off, buyers should program the personal computer to work for them. The most practical owner uses it for the complex and repetitive tasks encountered in the personal or business environment.

Contact: ***L.E. Kitchens*** **TACNET: 224-2096**



RCA

ORGANIZATION AS OF NOVEMBER 1, 1983

Board of Directors
T.F. Bradshaw Chairman of the Board & Chief Executive Officer
R.R. Frederick President & Chief Operating Officer
E.E. Beyer, Jr., Executive Vice-President and General Counsel
G.H. Fuchs, Executive Vice-President, Industrial Relations
W.C. Hittinger, Executive Vice-President, Corporate Technology
R.W. Miller, Executive Vice-President and Chief Financial Officer
J.M. Alic, Senior Vice-President, Corporate Planning
T.B. Ross, Senior Vice-President, Corporate Affairs
National Broadcasting Company, Inc. *G.A. Tinker* Chairman of the Board & Chief Executive Officer
C.I.T. Financial Corp. *W.S. Holmes, Jr.* Chairman of the Board & Chief Executive Officer
RCA Communications, Inc. *E.F. Murphy* Chairman & Chief Executive Officer
F.A. Olson Executive Vice-President
R.H. Pollack Executive Vice-President
J.D. Rittenhouse Group Vice-President
H.S. Schlosser Executive Vice-President
RCA American Communications, Inc.
RCA Global Communications, Inc.
RCA Cylix Communications Network, Inc.
RCA Network Services, Inc.
The Hertz Corporation
Coronet Industries, Inc.
Group Vice-President (*J.K. Sauter*)
Consumer Electronics Division
Distributor & Special Products Division
Video Component and Display Division
VideoDisc Division
Cablevision Systems
New Products Division
Solid State Division
RCA Laboratories
Broadcast Systems Division
Government Systems Division
RCA Service Company
RCA Records Division
RCA/Columbia Pictures Joint Ventures
RCTV Joint Venture

Patents

Astro-Electronics

Ganssle, E.R. | Samhammer, N.F.
Modular spacecraft structures—4395004

Ganssle, E.R.
Apparatus for remotely indicating alignment of male and female members—4395005

Gounder, R.N.
Solar cell array with lightweight support structure—4394529

Automated Systems

Bosselaers, R.J.
High speed sampling head—4399413

Dion, D.F. | Cantella, M.J.
Quantized video signal level interpolator—4399411

Broadcast Systems Division

Hacke, J.F. | Bazin, L.J.
Balanced modulator with feedback stabilization of carrier balance—4393395

Hedlund, L.V. | Herzog, D.G.
Helical scan tape recording and/or replay apparatus—4395738

Hedlund, L.V.
Overcoming drum stall in record and/or replay systems—4396956

Hurst, R.N.
Time changing system for VTR—4393415

Zorbalas, G.S.
Rapid stepping of a moving recorded medium—4393421

Consumer Electronics

Amery, J.G. | Jorgenson, R.W.
Signal translating apparatus for composite signal subject to jitter—31326

Bingham, J.P. | Benford, J.F.
Split phase stereophonic sound synthesizer—4394535

Carlson, D.J.
Television remote control system for selectively controlling a plurality of external apparatus—4392022

Fernsler, R.E. | Steckler, S.A. | Balaban, A.R.
SCR regulator control circuit—4396873

Fernsler, R.E. | Steckler, S.A. | Balaban, A.R.
Dual mode horizontal deflection circuit—4396948

Holmes, D.D.
FM counter detector—4395735

Nero, L.W.
Television receiver high voltage generator—4394722

Yost, T.D.
Synchronizing pulse separator—4400733

Laboratories

Avery, L.R.
Integrated circuit protection device—4400711

Ban, V.S.
Radiation heated reactor process for chemical vapor deposition on substrates—4401689

Belohoubek, E.F.
Adjustable phase shifter—4395687

Bogner, B.F.
Non-contacting RF shielding gasket with folded stub members—4396795

Botez, D.
High power semiconductor laser—4393504

Chin, D. | Henderson, J.G. | Maturo, R.J.
Signal-seeking tuning system with automatic bandswitching for a television receiver—4398303

Desai, N.V. | Poliniak, E.S.
Method of forming pattern in positive resist media—4396702

Desai, N.V. | Gavalchin, E.J.
Method of forming resist patterns using x-rays or electron beam—4397938

Dieterich, C.B.
Signal expander/compressor with adaptive control circuit—4398157

Dischert, R.A.
Controlled RAM signal processor—4396938

Goldstein, B. | Dresner, J. | Szostak, D.J.
Method and apparatus for determining minority carrier diffusion length in semiconductors—4393348

Goodman, A.M.
Method of making CCD imagers—4396438

Guarracini, J.A. | Walentine, J.P.
Video disc mastering using a gimballed air puck—4402072

Hawrylo, F.Z.
Light emitting device with a continuous layer of copper covering the entire header—4394679

Hinn, W.
Dynamic de-emphasis compensation system—4400720

Holmes, D.D.
Television ghost signal detector with color burst phase delay control—4393397

Kaganowicz, G. | Robinson, J.W.
Adherent perfluorinated layers—4391843

Kern, W. | Schnable, G.L.
Selective etching of phosphosilicate glass—4395304

Kilichowski, K.B.
Method of using a positive electron beam resist medium—4397939

Knight, S.P. | Henderson, J.G.
Television tuning system with electronic frequency adjustment apparatus—4402089

Kumar, M. | Menna, R.J. | Huang, H.
Hybrid power divider/combiner circuit—4394629

Lewis, H.G., Jr.
Digital video signal processing filters with signal-to-noise enhancement—4395729

Lewis, H.G., Jr.
Clock generator for a digital color television signal receiver—4402005

Liu, F.C. | Kao, Y.
Impulse noise detection circuit for TV signals—4398210

Osaka, S. | Toda, M.
Rotative motor using a triangular piezoelectric element—4399385

Osaka, S. | Toda, M.
Rotative motor using plural arrays of piezoelectric elements—4399386

Pampalone, T.R.
Aqueous developable poly(olefin sulfone) terpolymers—4393160

Pampalone, T.R. | Miller, A.Z.
Method of mounting electronic components—4398660

Pampalone, T.R. | Seffren, S.S.
Epoxy encapsulating formulation—4401775

Pankove, J.I. | Wu, C.P.
Solar cell structure incorporating a novel single crystal silicon material—4392011

Powers, K.H.
Television display system with reduced line-scan artifacts—4400719

Prabhu, A.N. | Hang, K.W.
Conductor inks—4399320

Prabhu, A.N. | Hang, K.W.
Overglaze inks—4401709

Rangachar, H.V. | Powell, K.D.
System and method for controlling an etch line—4400233

Reitmeier, G.A.
Horizontal-rate phase-change of TV pixel distribution among multiple recorder tracks for dropout concealment—4393414

Reitmeier, G.A. | Dischert, R.A.
Use of a single set of steering bits in a plural channel system—4396937

Rinehart, J.K. | Willis, D.H. | Luz, D.W.
Remote responsive television receiver ferroresonant power supply—4392090

Ritt, P.M.
Method for removing insolubilized PVA from the surface of a body—4401508

Shaw, J.M.
Method of manufacturing low resistance gates and interconnections—4392299

Smollin, C.F.
Cascaded digital broadcast transmitter—4399558

Southgate, P.D. | Gorog, I.
Defect detection system—4395122

Steigmeier, E.F. | Auderset, H.
Method for determining the quality of light scattering material—4391524

Taylor, G.C.
Method of forming metal lines—4400257

Theriault, G.E.
Diplexer for television tuning systems—4397037

Theriault, G.E.
Voltage controlled tuner with voltage variable frequency selective arrangements controlled in response to a control voltage generated independently of the tuning voltage—4399559

Upadhyayula, L.C.
Signal switching matrix—4399439

Vossen, J.L., Jr. | Zelez, J.
Transparent conductive film having areas of high and low resistivity—4395467

Wang, C.C. | Bates, R.F.
Bis(hydroxyalkyl) disiloxanes and lubricant compositions thereof—4391720

Wargo, R.A.
Video signal analyzer—4402013

Woodward, D.M.
Non-contacting RF shielding device—4399316

Yamada, A. | Makino, J.
Shutter control system—4396831

Zanzucchi, P.J. | Frenchu, W.R.
Device for multisample infrared analysis of materials in microgram quantity—4399361

Zelez, J. | Vossen, J.L., Jr.
Transparent conductive film—4399194

Missile and Surface Radar

Schelhorn, R.L.
Porcelain coated metal boards having interconnections between the face and reverse surfaces thereof—4393438

Weinberg, L.
Memory addressing circuit for converting sequential input data to interleaved output data sequence using multiple memories—4393444

Yorinks, L.H. | Milton, C.E., Jr.
Broadband, high power, coaxial transmission line coupling structure—4401955

Young, W.C.
Filtered connector—4401355

RCA Records

Ruda, J.C. | Wartzok, R.E. | Wedekind, R.J.
Active tape storage bin—4394951

"SelectaVision" VideoDisc Operations

Cowden, J.
Method and apparatus for holding styli—4391696

Helm, J.H.
Apparatus for molding a recorded disc—4397627

Hughes, L.M.
Caddy loading video disc player—4398281

Kelleher, K.C.
Bidirectional pulse generator for video disc stylus deflector transducer—4397014

Manson, E.T. | Taylor, B.K.
Pickup arm suspension for video disc cartridge—4400806

Mehrotra, G.N.
Audio processor for single, channel, matrixed two-channel, and un-matrixed two-channel signals—4393489

Prusak, J.J.
Matrixing process for the manufacture of molded records—4400245

Sickert, P.
Preparation of video disc molding composition—4399061

Straub, P.J., Jr.
Stylus cartridge—4393487

Wharton, J.H. | James, J.E.
Video disc slow down processor with reverse color detector/corrector—4395737

Wharton, J.H.
Stereophonic bilingual signal processor—4399329

Wierschke, D.
Method to center and separate electroformed replicas from a matrix—4394341

Ziegel, D.H.
Technique for optical alignment of a workpiece—4391520

Solid State Division

Butterwick, G.N.
Photomultiplier assembly having universal alignment means—4396859

Harwood, L.A. | Shanley, R.L., 2nd
Hettiger, J.
Video signal peaking control system with provision for automatic and manual control—4399460

Mazgy, J.D.
Wide temperature range switching circuit—4400635

Neilson, J.M.
Transistor with integrated diode and resistor—4398206

Policastro, S.G. | Woo, D.S.
Method of making low leakage *n*-channel SOS transistors utilizing positive photoresist masking techniques—4393572

Schade, O.H., Jr.
Low drift amplifiers—4392112

Smith, E.M.
Method for aging a cathode of a cathode-ray tube—4392834

Stewart, R.G.
Gated parallel decoder—4398102

Wittlinger, H.A.
Voltage reference circuit with feedback circuit—4399398

Video Component and Display Division

Chen, H.
Color picture tube having an improved expanded focus lens type inline electron gun—4400649

Hale, J.R.
Self-indexing insulating support rods for an electron gun assembly—4400644

Hughes, R.H.
Color picture tube with means for affecting magnetic deflection fields in electron gun area—4396862

Liller, P.R.|Shahan, D.J.
Method of electrically processing a CRT mount assembly to reduce afterglow—4395242

Nubani, J.I.|Sawicki, F.S.
Method of processing a cathode-ray tube for eliminating blocked apertures caused by charged particles—4398897

Pen and Podium

Recent RCA technical papers and presentations

To obtain copies of papers, check your library or contact the author or his divisional Technical Publications Administrator (listed on back cover) for a reprint.

Advanced Technology Laboratories

G. Ammon
An Optical Disk Jukebox Mass Memory System—Presented at SPIE 2nd International Symposium on Optical Mass Data Storage, Arlington, Virginia, and published in the *Proceedings* (6/6-10/83)

G. Caracciolo|W. Gehweiler|W. Heagerty
G. Brucker (Laboratories)
Design and Performance of Two 1K CMOS/SOS Hardened RAMs—*IEEE Transactions on Nuclear Science* (6/83)

A. Feller|R. Noto|D. Smith
B. Wagner|R. Putatunda
CADDAS, A Total Capability CAD VLSI Design System Based on Fully Automatic Design Techniques—*Electronics* magazine (6/16/83)

P. Hess
Implementation of the 20 (1,7) Code for Improved Packing Density on Optical Disk Recorders—Presented at SPIE 2nd International Symposium on Optical Mass Data Storage, Arlington, Virginia, and published in the *Proceedings* (6/6-10/83)

R. Kenville
Optical Mass Storage at 300 Mb/s—*Prceedings of CLEO '83* (5/83)

M. Levene
Optical Disk Media Parameters and their Relationship to Equipment Design—Presented at SPIE 2nd International Symposium on Optical Mass Data Storage, Arlington, Virginia, and published in the *Proceedings* (6/6-10/83)

H. Li
A VLSI Modular Architecture Methodology for Realtime Signal Processing Applications—Presented at the 1983 International Conference on Parallel Processing, Bellaire, Michigan, and was published in the Conference *Proceedings*, pp. 319-324 (8/23-26/83)

D. O'Rourke (Government Communications Systems)|A. Bramble (ERADCOM)
Agile Frequency Synthesizer—Presented at Frequency Control Symposium, Ft. Monmouth, New Jersey, and published in the*Proceedings* (5/31/83)

C. Reno
Optical Disk Recording Techniques for Data Rates Beyond 100 Mb/s—Presented at SPIE 2nd International Symposium on Optical Mass Data Storage, Arlington, Virginia, and published in the *Proceedings* (6/6-10/83)

B. Siryj|A. Litwak
Evaluation of Erasable Optical Recorder Materials—Presented at SPIE 2nd International Symposium on Optical Mass Data Storage, Arlington, Virginia, and presented in the *Proceedings* (6/6-10/83)

D. Smith|J. Werbickas|R. Noto|F. Borgini
VGAUA: The Variable Geometry Automated Universal Array Layout System—Presented at the ACM IEEE 20th Design Automation Conf., and published in the *Proceedings* (6/83)

M. Stebnisky|M. McGinnis|J. Werbickas
APSS: An Automatic PLA Synthesis System—Presented at ACM IEEE 20th Design Automation Conf., and published in the *Proceedings* (6/83)

M. Stebnisky|A. Smith
F. Borgini|S. Sharma
Approaches and Tradeoffs in Optimal Standard Cell Design—Presented at the Custom Integrated Circuits Conf., and published in the *Proceedings* 5/23/83

R. Tarzaiski
Selection of 30 (1,7) Code for Improving Packing Densities on Optical Disk Recorders—Presented at SPIE 2nd International Symposium on Optical Mass Date Storage, Arlington, Virginia, and published in the *Proceedings* 6/6-10/83)

Astro-Electronics

W. Altman|H. Gurk|P. Kaskiewicz
Feasibility of a Windsat Free Flyer Using an Advanced TIROS N Satellite—Presented at the 2nd Topical Mtg. on Coherent Radar, Aspen, Colorado (8/1/83)

G.J. Brucker
Ground Testing Devices for Ionization Damage in Space Society for Testing & Failure Analysis (ISTFA)—1983 Symposium of the Intn'l. Soc. of Testing & Failure Analysis (10/17/83)

L. Bulwer
Performance Measurement Using Value of Work—Presented at Artemis Users Association, Denver, Colorado (11/15/83)

F. Chu
Heuristic Optimization in Balancing High Speed Rotors—Presented at the ASME Winter Annual Meeting, Boston, Massachusetts (11/14/83)

J.A. Collins
Development of a Dielectric, High Voltage Module for Space—Presented at '83 High Voltage Workshop (IEEE), Adelphi, Maryland (10/4-6/83)

R. Devaux|M. Plesher|S. Schiffer
Predicting the Behavior of Parallel Connected Batteries—Presented at the 18th EICEC, Orlando, Florida (8/21/83)

J. Frohbieter
RCA's Approach to Increased Communication Satellite Performance—Presented at the 1st Canadian Domestic & Intn'l. Satellite Communications Conference, Ottawa, Canada (6/15/83)

A. Garfinkel
APT Evolution As Seen By A S/C Designer—Presented at the Intn'l. Direct Broadcast Services User Conference, New Carrollton, Maryland (6/7/83)

P. Goodwin
Autonomous Redundance Management on DMSP S/C—Presented at the AIAA Computers in Aerospace IV Conference, Hartford, Connecticut (10/24/83)

R. Gounder
Advanced Composite Antenna Reflectors for Communications Satellites—*SAMPE Journal* (7/83)

H. Gurk (Astro-Electronics)
E. Stetina (NASA) | G. Comeyne (NASA)
R. Foley (Ball Aerospace)
Ocean Observations Using NOAA Satellites—Presented Coastal Zone 1983, San Diego, California (6/1/83)

C. Hubert | N. Goodzeit
Effects of Propellant Motion on A Spinning S/C with Vaned Tanks—Presented at the AIAA Guidance & Control Conference, Gatlinburg, Tennessee (8/15/83)

J.N. LaPrade
A Solid State C-Band Power Amp for Communications Satellites—Intn'l. Telemetering Conference, San Diego, California (10/27/83)

K. Lebsock
Magnetic Desaturation of a Momentum Bias System—Presented at the AIAA/AAS Astrodynamics Conference, San Diego, California (8/9/83)

J. Swale | S. Marona
Stationkeeping of a Constellation of Geostationary Satellites—Presented at the AIAA/AAS Astrodynamics Conference, Lake Placid, New York (8/22/83)

F. Thiessen
Design of Memory Subsystem for Advanced Spaceborne Computer—Presented at the Massachusetts Institute of Technology (MIT) Library, Cambridge, Massachusetts (9/83)

A. Weinrich
RCA Satcom—Long Mission Life Spacecraft—Presented at the NSIA Mission Assurance Conference, Los Angeles, California (6/7/83)

D. Zion
Broadband Performance of Branch Guide Couplers—Presented at the Massachusetts Institute of Technology (MIT) Library, Cambridge, Massachusetts (9/83)

Automated Systems

E.M. Stockton
TCAC, A Tactical C^{3}1 System—Presented at EASCON '83, Washington, D.C. (9/19/83)

Broadcast Systems Division

R.K. Baldock
An Introduction for the TCR-10—Presented at the Washington Executive Broadcast Engineers, Ocean City, Maryland (10/14/83)

T.M. Gurley | C.J. Haslett
Development of a Prototype Studio Quality Component Analog VTR—Presented at SMPTE, Los Angeles, California (10/30/83)

R.N. Hurst
New RCA CCD Imager Demonstrated at 1983 NAB—Published in *RCA Broadcast News*, Vol. 173 (8/83)

E.H. Mayberry
Expectations vs Realities of Circularly Polarized Panel Antennas for TV Broadcasting—Presented at IEEE Broadcast Symposium, Washington, D.C. (9/22/83)

J.F. Monahan
A Fully Automatic Portable Production Camera—Presented at IRECON International, Sydney, Australia (9/83)

Corporate Technology

W.C. Hittinger
Information Technology Management in the Private Sector—Presented at "Coping with the Information Age," New Jersey State Government, Hyatt Regency Hotel, Princeton, New Jersey (10/14/83)

J. Vollmer
Military and Civilian Satellites: Today, Tomorrow, and The Day After Tomorrow—Presented at the International Aerospace Forum, Rome, Italy (10/5/83)

Government Communications Systems

A. Bramble | D. O'Rourke
Agile Frequency Synthesizer—Presented at the Frequency Control Symposium, Philadelphia, Pennsylvania, and published in the *Proceedings* (6/1-3/83)

A.M. Earman
A High Performance Optical Disc Drive for Use in Non-Benevolent Environments—Presented at the SPIE Optical Mass Data Storage Symposium, Arlington, Virginia, and published in the *Transactions* (6/6/83)

D.G. Herzog
Optical Disc Utilized as a Data Storage System for Reconnaissance Application—Presented at SPIE's 27th Annual Intn'l. Tech. Symposium, San Diego, California, and published in the *Proceedings* (8/21-26/83)

J.H. Hoover
Intelligent Optical Disc Controller Architecture—Presented at the SPIE Optical Mass Data Storage Symposium, Arlington, Virginia, and published in the *Transactions* (6/6/83)

G. Ludwig
Local Area Radio Intercom Network—Presented at MILCOM '83, Washington, D.C., and published in the *Proceedings* (10/31/83)

J. Lynerd
Tactical Military Equipment Parts Control Programs—Presented at the 18th Int'l. Logistics Symposium, Atlanta, Georgia, and published in the *Proceedings* (8/83)

J. Moroney
Emitter Location Concepts—Presented at Military Uses of the High Frequency Spectrum, AFCEA Course 104, Washington, D.C. (9/29)

B. Patrusky | J. Salerno (RADC)
A Unified Packet/Voice Data Switch—Published in *IEEE Transactions on Communications* (10/83)

S.M. Ravner
A Semiconductor Laser Optical Disc Image Data Storage and Retrieval System—Presented at the SPIE Optical Mass Data Storage Symposium, Arlington, Virginia, and published in *Transactions* (6/6/83)

M. Rosenblatt
Low Rate Multiplexer—Presented at NATO Integrated Communication System Management Agency, Brussels, Belgium (9/29)

J.B. Sergi | J.P. Viola
Very High Speed Fiber Optic Data Distribution Network—Presented at MILCOM '83, Washington, D.C., and published in the *Proceedings* (10/31/83)

Laboratories

C.H. Anderson | C.R. Carlson
R.W. Klopfenstein
Spatial-Frequency Representations of Images with Scale Invariant Properties—*Proceedings of SPIE*, Vol. 360 (8/24-27/83)

B.L. Bentz | P.J. Gale
Characterization of Primary Beams in Fast Atom Bombardment and Liquid Secondary Ion Mass Spectrometry Sources—*Analytical Chemistry*, Vol. 55, No. 8, p. 1434 (7/83)

D. Bodson | D. Botez
Electro-Optical Communications Dictionary—Hayden Book Company (4/83)

D. Botez
Optical Data Storage—*SPIE Proceedings*, Vol. 382 (1/17-20/83)

D. Botez
Fiber-Optics Sources and Detectors—Presented at the SPIE Conference, Washington, D.C. (4/4/83)

D. Botez
High-Power Diode Lasers for Optical Recording—*SPIE Proceedings,* Vol. 382, pp. 208-210 (1/83)

D. Botez
High-Power Diode Lasers for Optical Recording—Presented at the Optical Society of America Winter '83 Optical Data Storage Conference, Incline Village, Nevada (1/17-20/83)

D. Botez
High-Power Phase-Locked Arrays of Index-Guide CSP-LOC Lasers—Presented at the 4th IOOC Conference, Tokyo, Japan (6/27-30/83)

D. Botez
Liquid Phase Epitaxial Growth over Nonplanar Substrates—Presented at the American Association of Crystal Growers Meeting, New Jersey Section (5/18/83)

D. Botez
Single-Mode Diode Lasers for Optical Communications—Presented at the TMSA/LACES Conference, San Francisco and Newport Beach, California (2/14-17/83)

D. Botez | J.C. Connolly
Crystal Growth of Mode-Stabilized Semiconductor Diode Lasers by Liquid-Phase Epitaxy—*RCA Review* (3/83)

D. Botez | J.C. Connolly
Phase-Locked Array of CSP-LOC Lasers—Presented at the Device Research Conference, Burlington, Vermont (6/20-22/83)

D. Botez | J.C. Connolly | M. Ettenberg
D.B. Gilbert | J.J. Hughes
Reliability of Constricted Double-Heterojunction AlGaAs Diode Lasers—*Appl. Phys. Lett.,* Vol. 43, No. 2 (7/15/83)

J.K. Butler | D. Botez
Spatial Mode Discrimination and Control in High-Power Single-Mode Large-Optical-Cavity Diode Lasers—Presented at the 6th Topical Meeting on Optical Fiber Communication, Paper WJ3, New Orleans, Louisiana (2-28/3-2/83)

R.L. Camisa | G. Taylor | W. Reichert
F. Cuomo | R. Brown
A Flip-Chip GaAs Power FET with Gate and Drain Via Connections (Paper) (PRRL-83-TR-156)—*Electron Device Letters*

D.B. Carlin
Spot-Forming Efficiency of Optical Recording Systems using High-Power Diode Lasers—*SPIE,* Vol. 382, Optical Data Storage

J.F. Corboy | R. Pagliaro, Jr.
An Investigation of the Factors that Influence the Deposit/Etch Balance in a Radiant-Heated Silicon Epitaxial Reactor—*RCA Review,* Vol. 44 (6/83)

G.W. Cullen | J.F. Corboy | R. Metzi
Epitaxial Reactor Systems: Characteristics, Operation, and Epitaxy Costs—*RCA Review,* Vol. 44 (6/83)

A. Dreeben | A. Schujko
On a Relationship between Substrate Perfection and Stacking Faults in Homoepitaxial Silicon—*RCA Review,* Vol. 44 (6/83)

M.T. Duffy | J.T. McGinn | J.M. Shaw
R.A. Soltis | G. Harbeke
LPCVD Polycrystalline Silicon; Growth and Physical Properties of Diffusion-Doped, Ion-Implanted, and Undoped Films—*RCA Review,* Vol. 44 (6/83)

T.J. Faith | R.S. Irven | S.K. Plante
J.J. O'Neill, Jr.
Contact Resistance: Al and Al-Si to Diffused N$^+$ and P$^+$ silicon—*J. Vac. Sci. Technol.* A 1 (2) (4-6/83)

A.M. Goodman | L.A. Goodman
H.F. Gossenberger
Silicon-Wafer Process Evaluation Using Minority-Carrier Diffusion-Length Measurement by the SPV Method—*RCA Review,* Vol. 44 (6/83)

G. Harbeke | L. Krausbauer | E.F. Steigmeier
A.E. Widmer | H.F. Kappert | G. Neugebauer
LPCVD Polycrystalline Silicon: Growth and Physical Properties of In-Situ Phosphorus Doped and Undoped Films—*RCA Review* Vol. 44 (6/83)

E.F. Hockings | S. Bloom | C.A. Catanese
A Focus Mask for Color Picture Tubes Using Dipolar and Quadrupolar Lensing—*IEEE Transaction on Electron Devices,* Vol. ED-30, No. 8 (8/83)

L. Jastrzebski
Comparison of Different SOI Technologies: Assets and Liabilities—*RCA Review,* Vol. 44 (6/83)

L. Jastrzebski | J.F. Corboy | J.T. McGinn
R. Pagliaro, Jr.
Growth Process of Silicon Over SiO_2 by CVD: Epitaxial Lateral Overgrowth Technique—*Journal of the Electrochemical Society,* Vol. 130, No. 7 (7/83)

H. Kiess | G. Harbeke | B.J. Curtis | R. Keller
Near IR-Optical Absorption of Undoped and Doped $(CH)_x$—*Journal De Physique,* Colloque C3, supplement au n°6, Tome 44 (6/83)

J. Lagowski | L. Jastrzebski | G.W. Cullen
Optical Probing of Silicon-Sapphire Interface of Heteroepitaxial SOS Films—*Journal of the Electrochemical Society,* Vol. 130, No. 3 (8/83)

R.U. Martinelli | R. Ford (Solid State Div.)
Design Guidelines for Power Switching Transistors—*RCA Review,* Vol. 44 (6/83)

G.H. Olsen | T.J. Zamerowski
Double-Barrel III-V Compound Vapor-Phase Epitaxy Systems—*RCA Review,* Vol. 44 (6/83)

D. Raychaudhuri | L. Schiff
Unauthorized Descrambling of a Random Line Inversion Scrambled TV Signal—*IEEE Transactions on Communications,* Vol. Com-31, No. 6 (6/83)

W. Rehwald | A. Vonlanthen
Static and Dynamical Elastic Behaviour of Squaric Acid Around Its Phase Transition—*Z. Phys. B* — Condensed Matter, Vol. 52, pp. 139-144 (1983)

J.R. Sandercock
Film Thickness Monitor Based on White Light Interference—*J. Phys. E: Sci. Instrum.,* Vol. 16 (1983)

J.H. Thomas, III
Backscattered Electron Imaging with Scanning Auger Electron Spectroscopy—*Rev. Sci. Instrum.,* Vol. 54, No. 10 (10/83)

J.H. Thomas, III | A. Catalano
Auger Electron and X-Ray Photoelectron Spectroscopy Analysis of the Hydrogenated Amorphous Silicon-Tin Oxide Interface: Evidence of a Plasma-Induced Reaction—*Appl. Phys. Lett.,* Vol. 43, No. 1 (7/1/83)

J.L. Vossen
Safety Aspects of Pumping and Maintenance in Plasma Etching and Deposition—Presented at the Workshop on Industrial Plasma Developments; International Union of Pure and Applied Chemistry, Sherbrooke, Quebec, Canada (7/23/83)

J.L. Vossen | J.H. Thomas, III | J.S. Maa
O.R. Mesker | G.O. Fowler
A System for *In-Situ* Studies of Plasma-Surface Interactions Using X-Ray Photo-Electron Spectroscopy—*J. Vac. Sci. Technol.,* A 1, 1452 (1983)

J.L. Vossen | F.J. Thomas | O.R. Mesker
J.D. Smith | R. Vaughn | R.A. Denton
Design of a Versatile In-Line Vacuum Deposition System—Presented at the 9th International Vacuum Congress, Madrid, Spain (9/29/83)

J.L. Vossen | N.L. Hammond
The American Vacuum Society—1972-1983—*J. Vac. Sci. Technol.,* A 1, 1351 (1983)

J.L. Vossen
Size Effects in Thin Films (Book Review)—*J. Vac. Sci. Technol.,* A 1, 1575 (1983)

J.L. Vossen
The American Vacuum Society—1973 to 1983—*J. Vac. Sci Technol.,* A 1, (3) (7-9/83)

J.L. Vossen | J.H. Thomas, III | J.S. Maa
O.R. Mesker | G.O. Fowler
A System for *In-Situ* Studies of Plasma-Surface Interactions Using X-Ray Photoelectron Spectroscopy—*J. Vac. Sci. Technol.,* A 1 (3) (7-9/83)

L.K. White
Planarization Properties of Resist and

Polyimide Coatings—*Journal of the Electrochemical Society,* Vol. 130, No. 7 (7/83)

B.S. Yarman
A Dynamic CAD Technique for Designing Broadband Microwave Amplifiers (Paper) (PRRL-83-TR-167)—Presented at Int. Microwave Antenna-Propagation and Electro-Magnetic Wave Theory Symposium, Ankara, Turkey, and in *RCA Review* (10/18-24-83)

Missile and Surface Radar

J.A. Bauer
Design Considerations for Surface Mounting Technology—Published in *Proceedings,* Lake Publishing Company, Arlington Heights, Illinois (9/9/83)

J.A. Bauer
Overview and Applications of Surface Mounting Technology—Published in *Proceedings*, Lake Publishing Company, Arlington Heights, Illinois (9/8/83)

M.W. Buckley, Jr.
Future Trends Lecture Segment, Chairman—Presented at the International Radar Symposium, India (10/9/83)

M.W. Buckley, Jr.
Introduction to Project Management—Presented at the IEEE Industry Applications Society, Bangalore Chapter, Bangalore, India (10/13/83)

R.J. Camlin, |D.R. Higgs |J.K. McCann
Automating High-Volume Engineering Documentation—Presented at the 1983 IEEE Professional Communications Society Conference, Atlanta, Georgia, Conference *Proceedings* (10/83)

A.S. Cooper
Computer-Aided Engineering for Thick Film Multilayer Hyrids—Presented at ISHM, Philadelphia, Pennsylvania (11/2/83)

J.W. Douglas
Applicon Disk RSX File Structure—Applicon Fall Users Meeting, Detroit, Michigan (11/83)

T.J. Forquer
Software Technology and Architecture Issues for the Future (Panel Co-Chairman)—Presented at the EIA Seventeenth Annual Computer Resources Data and Configuration Management Workshop, Phoenix, Arizona (9/19-23/83)

W.J. Graham
Analysis and Synthesis of Axial Field Patterns of Focused Apertures—Published in *IEEE Transactions on Antennas and Propagation,* Vol. AP-31, No. 4 (7/83)

D.M. Jones
Connecting Drafting to CAME Analysis—Presented at the Mechanical Engineering Symposium, MSR, Moorestown, New Jersey (10/19-83)

F.E. Oliveto
System Availability—Presented at the Productivity & Technological Innovation Symposium, The Engineers' Club of Philadelphia (10/14/83)

E.J. Podell
Mathematics Must Be Effective in Technical Communications—Presented at the 1983 IEEE Professional Communications Society Conference, Atlanta, Georgia, and published in the *Proceedings* (10/20/83)

R.J. Pschunder
Experience with Tektronix-4114 Graphics Work Station in Finite Element Modeling—Presented at the Mechanical Engineering Symposium, Moorestown, New Jersey (10/19/83)

R.J. Pschunder
Experience with Tektronix Graphics Workstation—Presented at the Mechanical Engineering Symposium, Moorestown, New Jersey (10/19/83)

W. Siegel
Heuristic Decision Aids to Find Waypoints for an Anti-Ship Cruise Missile—Presented at the 51st Military Operations Research, Wright-Patterson AFB, Ohio (9/27/83)

J.W. Smiley
Automated Wirewrap Backplane Design System—Presented at the Applicon User Meeting, Detroit, Michigan (11/6-11/83)

S.A. Steele
Issues Affecting Software Standards to Ensure Quality and Productivity—Presented at the Computers in Aerospace IV Conference, Hartford, Connecticut, and published in the *Proceedings* (10/24/83)

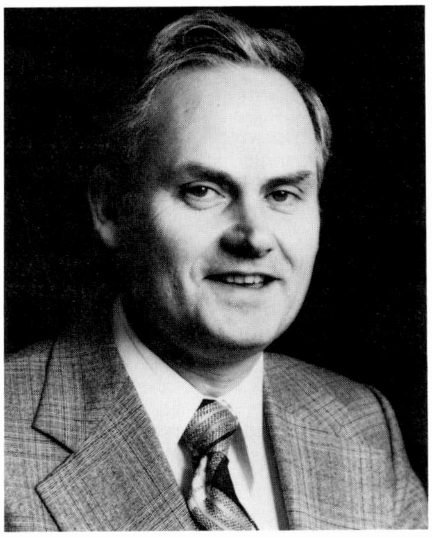
H. Urkowitz
Signal Theory and Random Processes—Artech House, Dedham, Massachusetts (6/83)

T.O. Vinson|T.P. Speas
AEGIS Logistics Analysis—Presented at the 18th Annual Logistics Symposium, Atlanta, Georgia, and published in Symposium *Proceedings* (8/23-26/83)

M. Weiss
Modal Testing as Part of the Overall Structural Analysis/Vibration Test Cycle—Presented at the Mechanical Engineering Symposium, MSR, Moorestown, New Jersey (10/9/83)

Engineering News and Highlights

Turner is RCA Staff Vice-President, Planning



Appointment of **Carl R. Turner** as an RCA Staff Vice-President in the company's corporate offices to coordinate the planning activities for several RCA divisions and the RCA Laboratories, was announced by **Roy H. Pollack,** Executive Vice-President.

In this post, Mr. Turner will have the title of Staff Vice-President, Planning, and will review, evaluate and coordinate the planning activities of the following RCA divisions: Consumer Electronics Division; Distributor & Special Products Division; New Products Division; Video Component and Display Division; Solid State Division; and VideoDisc Division. He also will have the planning responsibility for the RCA Laboratories in Princeton, N.J., where he will have his office.

Previously, Mr. Turner had been Division Vice-President, Product Assurance and Planning, Solid State Division in Somerville, N.J. Mr. Turner, who has been with RCA for more than 25 years, has held several key management posts within the Corporation. He joined RCA in 1957 at its Solid State Division, and subsequently held a series of posts of increasing responsibility. In 1974 he was named Division Vice-President, Solid State Power Devices. From 1978 to 1983 Mr. Turner held several posts as a Division Vice-President within the RCA Solid State Division.

Mr. Turner was graduated from Rutgers University with a Bachelor of Science degree in Electrical Engineering. He was awarded a Masters Degree in Electrical Engineering from Stevens Institute in 1960. He has authored a number of articles on solid state devices and systems for technical publications.

Bianculli is Manager, Engineering Information

Anthony J. Bianculli has been appointed Manager, Engineering Information. His responsibilities include RCA's Engineering Communication activities, Engineering Publication and Technical Information Systems, as well as Technical Excellence Programs, Corporate Technology Symposia, and the Minorities in Engineering Program. Engineering Information is part of RCA's Technical Excellence Center located at 13 Roszel Road, Princeton, N.J.

Mr. Bianculli is a mechanical engineer who began his RCA career with the Microwave Tube Operation in Harrison, N.J. in 1952. He was engaged in various engineering assignments involving product design, equipment development, and the development of new techniques and processes.

At RCA Laboratories, from 1963 to 1970, Mr. Bianculli directed the assembly, processing, and packaging of newly invented vacuum tube and solid state devices. In 1970, he transferred to the Solid State Division at Somerville where his responsibilities have included manufacturing equipment development, engineering standards, and strategic planning. Most recently, he has managed engineering publications and office services, including the responsibility for office automation throughout the Division.

Concurrent with other assignments, he was chairman of the Somerville Minorities in Engineering Program, and management representative to the Somerville Technical Excellence Committee. He has been active in professional societies and industry associations as well as several non-job-related activities including being Mayor of his community, Rocky Hill, N.J. He has written numerous technical articles and presentations.

Mr. Bianculli will assume his new responsibilities at the Technical Excellence Center in mid-November. He replaces **Hans K. Jenny,** who is retiring.

Staff announcements

Jack K. Sauter, Group Vice-President, announced that **Arnold T. Valencia** is appointed Division Vice-President and General Manager of the newly established VideoDisc Division. The following individuals will report to Mr. Valencia: **David M. Arganbright,** Division Vice-President, Business Planning; **Bruce G. Babcock,** Division Vice-President, Special Marketing-VideoDisc; **Jay J. Brandinger,** Division Vice-President and General Manager, "SelectaVision" VideoDisc Operations; and **Thomas G. Kuhn,** Division Vice-President, "SelectaVision" Video Discs.

Roy H. Pollack, Executive Vice-President, and **John D. Rittenhouse,** Group Vice-President, announced that the Solid State Technology Center will become the responsibility of the Government Systems Division. **Dr. Larry J. French,** Division Vice-President, Solid State Technology Center, will report to **James B. Feller,** Division Vice-President, Engineering. Mr. Feller will continue to report to **Paul E. Wright,** Division Vice-President and General Manager, Government Systems Division.

Roy H. Pollack, Executive Vice-President, announced that **Erich Burlefinger** is appointed Division Vice-President and General Manager of the newly established New Products Division. The Electro-Optics and Devices organization activities will become the responsibility of the New Products Division. Dr. Burlefinger will report to the Executive Vice-President.

John D. Rittenhouse, Group Vice-President, announced that the name of the Commercial Communications Systems Division is changed to the Broadcast Systems Division. **Joseph B. Howe,** Division Vice-President and General Manager, will continue to report to the Group Vice-President.

Broadcast Systems Division

Joseph B. Howe, Division Vice-President and General Manager, Broadcast Systems Division, announced that the Broadcast Systems Division organization is as follows: **Robert B. Alleger, Jr.,** Division Vice-President, Finance; **Arthur J. Barrett,** Division Vice-President, Manufacturing; **Arch C. Luther,** Division Vice-President, Engineering; **Leo A. VanLingen,** Manager, Business Planning; **Joseph C. Volpe,** Division Vice-President, Operations; and **Dennis J. Woywood,** Division Vice-President, Marketing.

Consumer Electronics

Stephen L. Golliher, Manager, Purchasing, announced that **Lawrence J. Sahm** is appointed Manager, Component Sourcing. Mr. Sahm will report to the Manager, Purchasing.

Global Communications, Inc.

Donald R. Stackhouse, Vice-President, Leased Facilities and Systems Operations, announced that **Edgar G. Hammons** is appointed to the newly created position of Manager, Marine Station Operations. In this capacity, Mr. Hammons will be responsible for directing the operation of coastal stations handling marine message traffic in addition to his present responsibility for managing the Chatham, Massachusetts coastal station. Mr. Hammons will report to the Vice-President, Leased Facilities and Systems Operations.

Joe Terry Swaim, Vice-President, Switched Services Engineering and Operations, announced that **R. Lawrence Chory** is appointed Director, Software Engineering. Mr. Chory will report to the Vice-President, Switched Services Engineering and Operations.

M. Glen Looney, Director, Computer Programs, announced that **Paul Rubin** is appointed Program Manager, Message Switching Systems. Mr. Rubin will report to the Director, Computer Programs.

Government Communications Systems

Lawrence J. Schipper, Division Vice-President and General Manager announced that **Dr. Ulrich L. Rohde** has been retained by RCA Government Communications Systems to direct its future development of advanced radio systems.

Lawrence J. Schipper, Division Vice-President and General Manager announced the appointment of **Guy H.B. Shaffer** as Director, Information Processing Systems.

Patent Operations

Paul J. Rasmussen, Director, Television Equipments, announced that **Eric P. Herrmann** is appointed Managing Patent Attorney. Mr. Herrmann will report to the Director, Television Equipments.

Paul J. Rasmussen, Director, Television Equipments, announced the organization of Television Equipments as follows: **Peter M. Emanuel,** Managing Patent Attorney; **Eric P. Herrmann,** Managing Patent Attorney; **Joseph J. Laks,** Managing Patent Attorney; and **William H. Meagher,** Senior Staff Patent Counsel.

A. Russinoff, Staff Vice-President, Interparty Patent Matters, announced the appointment of **Clement A. Berard, Jr.** as Staff Patent Counsel.

Joseph S. Tripoli, Director, Electronic Systems, announced that **George E. Haas** is appointed Managing Patent Attorney.

Professional activities

Sterzer cited for service

New Jersey Governor Thomas Kean presented a plaque for "distinguished service to the State of New Jersey" to **Fred Sterzer,** Director, Microwave Technology Center, who is a member of the New Jersey Commission on Radiation Protection, which recently observed its 25th anniversary. Dr. Sterzer is Chairman of the Commission's Advisory Committee on Non-Ionizing Radiation.

Krittman elected

Irwin M. Krittman, Senior Patent Counsel, Patent Operations (Princeton),has been elected President of the International Patent Club (New York) for the 1983-85 term. The Club was established in 1961 to provide a forum for discussion by patent attorneys and other professionals involved in industrial property matters throughout the world. Krittman served as Secretary of the organization from 1981 to 1983.

Powers elected SMPTE Fellow

Kerns H. Powers, Staff Vice-President, Communications Research, has been elected a Fellow of the Society of Motion Picture and Television Engineers. Dr. Powers received a Certificate of Fellowship on Nov. 1 at the Society's 125th Technical Conference and Equipment Exhibit in Los Angeles. SMPTE Fellowships are awarded to "members of the Society who, because of their proficiency and contributions, are considered to have attained a superior rank among engineers or executives in the Motion-Picture, Television or related industries."

NBC staffers' professional activities

Mr. Peter Smith, Senior Staff Engineer, Technical Development, Operations and Technical Services, NBC, was a member of the "Network Experience with Teletext" panel, at the IEEE Broadcast Symposium, in Washington, D.C., on Sept. 22, 1983.

Mr. Jim Gibbings, Director, Strategic Planning and Quality Control, Operations and Technical Services, NBC, presented a paper entitled, "Production Options in Prime Time Stereophonic Television," at the SMPTE Fall Conference, Los Angeles. The conference was held from Oct. 31 through Nov. 4.

Mr. Robert J. Butler, Director Engineering Planning, Broadcast Operations, Operations and Technical Services, NBC, was a member of the "TV Network Distribution By Satellite" panel, at the IEEE Broadcast Symposium, in Washington, D.C., on Sept. 22, 1983.

Mr. Martin Meaney, Director Allocation Engineering, Operations and Technical Services, NBC, attended final CCIR meetings in Geneva, Switzerland, Sept./Oct. 1983. Mr. Meaney is involved with Study Groups 10 and 11, and has recently been appointed as Chairman of WG11E.

RCA in Burlington

On 21 October 1983, RCA Automated Systems celebrated its twenty-fifth anniversary by rededicating the Burlington, Massachusetts facility to "continued service to our nation." The rededication ceremony began with the Burlington High School band and chorus performing a variety of musical renditions, followed by the Fort Devens color guard presenting the colors. The Fort Devens drill team then performed precision close order drills to the delight and appreciation of the assemblage.

The first guest speaker was the Honorable Edward J. Markey, United States Representative from the Seventh Congressional District, who praised RCA Automated Systems for the great contributions made to the Commonwealth of Massachusetts and the Nation, as a leader and innovator among high technology manufacturers. Representative Markey presented Automated Systems with a commemorative copy of the Congressional Record declaring 21 October 1983 be designated RCA Burlington Day throughout the state.

Dr. Evelyn Murphy, Massachusetts Secretary of Economic Affairs, read a proclamation from Governor Dukakis that singled out Automated Systems as a catalyst in the high technology revolution in Massachusetts and America. Albert Kelly, Chairman of the Board of Selectmen of Burlington, praised Automated Systems for making Burlington a more desirable place to work and live.

On October 21, 1983, RCA Automated Systems, the New England Business Unit of Government Systems Division, commemorated 25 years of providing electronics in support of national goals. In a ceremony reminiscent of the facility's original dedication, RCA executives unveiled a plaque and recommitted the Burlington, Massachusetts plant to the service of the Nation. From left to right they are: ***John D. Rittenhouse,*** *RCA Group Vice-President;* ***Andrew T. Hospodor,*** *Automated Systems Vice-President and General Manager;* ***Dr. Robert C. Seamans, Jr.,*** *the original Chief Engineer at the Burlington plant;* ***Paul E. Wright,*** *Government Systems Division Vice-President and General Manager; and* ***James R. Foran,*** *Staff Vice-President and Financial Planning.*

Itamar Lubetzky *leads the Burlington High School Band in a musical accompaniment of the day's ceremonies.*

The Fort Devens color guard paraded the colors and performed in the bright autumn sunshine.

Rededicated to Continued Service to Our Nation
21 October, 1983

Andrew T. Hospodor, Division Vice-President and General Manager, Automated Systems gave the keynote address, citing the record of past achievements and challenging his employees to surpass this illustrious record over the next quarter century.

In attendance and of special interest was Dr. Robert C. Seamans, former NASA Administrator, Secretary of the Air Force and Burlington's first Chief Engineer. The ceremony closed with the unveiling and display of a new plaque signifying RCA Automated Systems' dedication to the continued service to our nation.

25th year on 128 AMERICA'S TECHNOLOGY HIGHWAY

From left to right: ***Mr. Albert Kelley,*** *Town of Burlington, Chairman of the Board of Selectmen;* ***Dr. Evelyn Murphy,*** *Massachusetts Secretary of Economic Affairs;* ***Honorable Edward J. Markey,*** *Congressman, Massachusetts Seventh District;* ***Mr. Andrew T. Hospodor,*** *Division Vice-President and General Manager, RCA Automated Systems;* ***Mrs. Hospodor; Mr. Murray Radlo,*** *Manager, Materials, RCA Automated Systems;* ***Col. Delmar Corbin,*** *Commander, Fort Devens;* ***Mr. John Rittenhouse,*** *Group Vice-President, RCA Corporation;* ***Mr. Paul Wright,*** *Vice-President and General Manager, RCA Government Systems Division;* ***Mr. James Foran,*** *RCA Staff Vice-President, Financial Planning.*

Honorable Edward J. Markey, *United States Representative from the Seventh Congressional District in the State of Massachusetts delivering the opening remarks.*

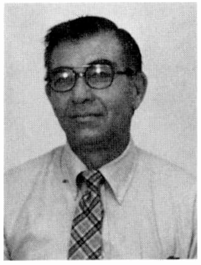
Dr. Evelyn Murphy, *Massachusetts Secretary of Economic Affairs, reading Governor Michael S. Dukakis' proclamation declaring October 21, 1983, RCA Burlington Day in Massachusetts.*

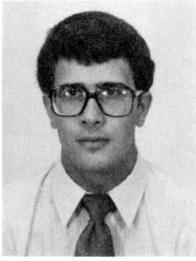
Andrew T. Hospodor, *Division Vice-President and General Manager, RCA Automated Systems, giving the keynote address.*

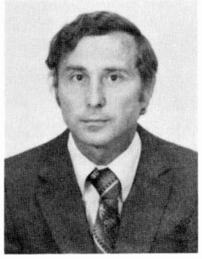
Technical excellence

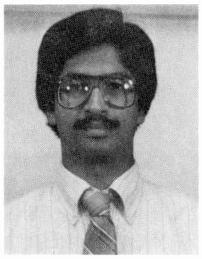
Consumer Electronics 2nd-quarter award winners

Dr. J.E. Carnes announced the second-quarter, 1983, Technical Excellence Award winners at the Aug. 17th T.E.C. lecture. Each recipient receives an appropriately engraved plaque and a reference book of his choice. The T.E.C. congratulates the following four individuals:


Becker


Fling


Modesitt


Naimpally

Hollis Becker—Awarded for the development of a graphics printing process combining inks and a topcoat that produces a highly reliable product for use on the unified remote for J-line television and video disc products. The wear performance of this process is presently considered superior to that of the CE competition.

Russ Fling—Awarded for the development of new methods of analyzing and understanding the performance of the analog-to-digital conversion process in the ITT chip set. His work led to improvements in the performance of the A/D conversion circuitry.

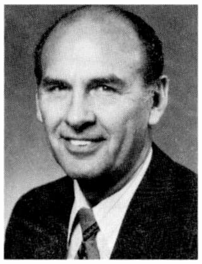
Mark Modesitt—Awarded for the design and development of a computer-aided test system for deflection ICs, which not only improved testing efficiency in Engineering, but also helped resolve correlation and testing issues with manufacturing automatic test equipment.

Sai Naimpally—Awarded for the adaptation of analysis methods and the innovative design of circuits leading to a better understanding of the video interface requirements of the ITT digital TV chip set.

Missile and Surface Radar 2nd-quarter awards

Cohen

DiMaria


Laible

Smargiassi

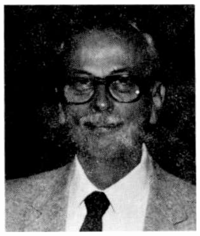
Albert Cohen—for his special contributions to the software design and testing of radar modules for the Crossbow-S Generic Radar. Mr. Cohen's innovative use of a computer-based emulator with early module testing in conjunction with a system simulator resulted in early elimination of errors in the functional software modules prior to either software or hardware integration. The success of this testing approach makes it attractive for application on future system developments.

Charles DiMaria—for innovative technical achievement in the conception, development, and test of the Robotic Assembly System for automated assembly of windows in phased-array antennas. His personal mastery of such diverse technologies as adhesives, bonding techniques, specialized tool design, kinematics and mechanics of robotic devices, and system integration was a critical element in the successful introduction of robotics to the assembly processes at MSR.

Charles W. Laible—for his personal dedication and professional guidance as an RCA team member supporting the Orbit '81 Ant Colony Experiment flown aboard the Space Shuttle Challenger. As the representative of MSR Design Engineering, he served as the engineering advisor for equipment packaging as well as the technical advisor during installation of the canister in Challenger. The leadership and stimulus he provided were a major element in the success of the project.

Robert Smargiassi—for outstanding performance in logic design and hardware implementation of the Military Computer Family Special Function Unit. Mr. Smargiassi undertook this complex, schedule-critical development task as an additional assignment to his support of module integration in the MCF CPU, successfully implementing an IEEE Floating Point Unit and a 32-bit double precision multiplier in two logic boards involving 659 integrated circuit chips.

"SelectaVision" VideoDisc Technical Excellence Award



Larry M. Hughes has been the Mechanical Engineering Project Leader during the entire J-line development cycle. The major goals of the J-line player were reduced cost, reduced size, and additional features. Through Larry's technical leadership, the cost was reduced by 30 percent, the volume was reduced by 15 percent, and competitive high-technology styling with automatic caddy loading was added. Larry personally developed the major mechanical philosophy of the player and acted as a major contributor to several mechanisms, including the disc transfer mechanism, the mechanism shut-off logic, the function-gear cam system, and the single pivot disc lifter.

As a result of his design effort and attention to detail, the J-line player started production with minimal problems and set new standards for VideoDisc player mechanism reliability. With his accomplishments as J-line Mechanical Engineering Project Leader and his innovative mechanical contribution, Larry has demonstrated his technical excellence in the design of a Consumer Electronics product.

Index to Volume 28

Subject Index

In this Subject Index, we asked the authors to classify their own articles under three principal subject headings. Under these headings you will find appropriate article titles, each followed by author name(s), volume and issue numbers, issue date, and initial page number.

Computer-aided manufacturing (CAM)

CAD/CAM at RCA Laboratories: Tools and applications, present and future. Reid-Green, K. S., Hatter, W. A. and Quardt, D. C., v28n1, Jan/Feb 1983, p29.
Producibility: The road to increased productivity. Gimigliano, R. F., v28n1, Jan/Feb 1983, p25.

Computer-aided testing (CAT)

Gun parts measurement, A computerized system for. Hook, H. O., Beltz, J. P., Bortfeld, D. P., Kuczer, P., Saunders, W. C. and Mavrodin, A., v28n1, Jan/Feb 1983, p43.
Microprocessor architecture and mass-storage techniques for field-portable automatic test equipment. Courcy, S. E. and DelPriore, G. A., v28n3, May/June 1983, p25.

Computerized control

CAD/CAM at RCA Laboratories: Tools and applications, present and future. Reid-Green, K. S., Hatter, W. A. and Quardt, D. C., v28n1, Jan/Feb 1983, p29.
Communications satellite software: a tutorial. Dusio, E. W., Murphy, T. P. and Cashman, W. F., v28n6, Nov/Dec 1983, p20.
Intelligent telephone applications of the CDP1800 microcomputer family. Nizet, J. and Shand, D., v28n1, Jan/Feb 1983, p72.

Computerized instrumentation

Gun parts measurement, A computerized system for. Hook, H. O., Beltz, J. P., Bortfeld, D. P., Kuczer, P., Saunders, W. C. and Mavrodin, A., v28n1, Jan/Feb 1983, p43.

Computers, home

Home computer used for engineering management tasks. Kitchens, L., v28n6, Nov/Dec 1983, p64.
Microcomputer Club at Missile and Surface Radar. Poletti, G., v28n4, July/Aug 1983, p106.

Data communications

Ground-control processing for communications satellites: Trends for the next decade. Cashman, W. F., v28n2, Mar/Apr 1983, p30.

Data management

Computer as an engineer's toolbox. Dreher, R. P., v28n1, Jan/Feb 1983, p11.
Home computer used for engineering management tasks. Kitchens, L., v28n6, Nov/Dec 1983, p64.
SAS (Statistical Applications System) applications within RCA (sidebars to article by R. P. Dreher). Sockwell, C., v28n1, Jan/Feb 1983, p14.

Digital electronics

Digital television, Technology for. Marlowe, F. J., v28n5, Sept/Oct 1983, p26.
Noise-reduction in video signals by a digital fine-structure process. Acampora, A. A., Bunting, R. M. and Petri, R. J., v28n2, Mar/Apr 1983, p48.
Wideband picture detail restoration in a digital NTSC comb-filter system. Acampora, A. A., v28n5, Sept/Oct 1983, p44.

Direct broadcasting satellites (DBS)

Direct-broadcast satellite considerations. Buntschuh, R. F., v28n2, Mar/Apr 1983, p54.

Editing

Guide for RCA Engineer authors. Sweeny, M. R., v28n4, July/Aug 1983, p33.

Electron guns

COTY-29 (Combined Optimum Tube and Yoke) color picture tube system. Morrell, A. M., v28n4, July/Aug 1983, p28.
Gun parts measurement, A computerized system for. Hook, H. O., Beltz, J. P., Bortfeld, D. P., Kuczer, P., Saunders, W. C. and Mavrodin, A., v28n1, Jan/Feb 1983, p43.

Energy conversion

Diffusion length and surface photovoltage. Goldstein, B., Dresner, J., Moore, A. R. and Szostak, D., v28n3, May/June 1983, p45.
Silane discharges, Chemistry of. Catalano, A., Longeway, P. A. and Weakliem, H. A., v28n3, May/June 1983, p50.
Solar energy: Sun sets on RCA energy research. Williams, B. F., v28n3, May/June 1983, p39.
Sunshine in the Garden State: Solar-energy applications. Stranix, A. J. and Firester, A. H., v28n3, May/June 1983, p56.

Engineering management

Life-cycle software development, Experience with. Resnick, H. L., v28n6, Nov/Dec 1983, p35.
RCA Technical Excellence Center established in Princeton. Jenny, H. K., v28n4, July/Aug 1983, p24.
Space Technology Roundtable: An Astro-Electronics Engineering Management Review and Forecast. Astro-Electronics' Staff, v28n2, Mar/Apr 1983, p11.

Engineering profession

RCA Technical Excellence Center established in Princeton. Jenny, H. K., v28n4, July/Aug 1983, p24.

Engineering tools and techniques

APL and engineering productivity. Solomon, J. G., v28n1, Jan/Feb 1983, p18.
Enhancing engineering productivity in GSD. Moore, E. E., v28n1, Jan/Feb 1983, p4.

Heat transfer

Steady-state NOdal Thermal Analysis for printed wiring assemblies. Nguyen, T. N. and Wolfe, D. B., v28n1, Jan/Feb 1983, p56.

Image encoding

Image Processing Facility at Advanced Technology Laboratories. Schaming, W. B. and Zapriala, J. S., v28n1, Jan/Feb 1983, p62.

Image enhancement

Image Processing Facility at Advanced Technology Laboratories. Schaming, W. B. and Zapriala, J. S., v28n1, Jan/Feb 1983, p62.

Imaging systems

The Technology Transfer Laboratory: A view from the middle. Duschl, R. A., v28n3, May/June 1983, p6.

Information management

Computer-aided functional analysis: A new concept in system engineering. Felbinger, R. J. and Leahy, W. R., v28n6, Nov/Dec 1983, p51.
Factory data versus factory information. Stein, P. G. and Rayl, M., v28n4, July/Aug 1983, p81.
Information technology management. Freedman, H. A., v28n4, July/Aug 1983, p87.

Information theory

Artificial intelligence research at ATL's Software Technology Laboratory. Zapriala, J. S., Spool, P. R., Whitehead, J. M., Gaev, J. A., Kleinosky, P. and McAdams, J., v28n6, Nov/Dec 1983, p4.

Infrared equipment

Passive and active sounding from satellites for atmospheric temperatures and constituents. Rosenberg, A. and Hogan, D. B., v28n2, Mar/Apr 1983, p83.

Integrated circuits

Gate arrays or standard cells: RCA Solid State's semicustom device operation. Hively, J. W., v28n3, May/June 1983, p31.

Introductions (to issues)

Productivity: Where it begins and ends. Vollmer, J., v28n1, Jan/Feb 1983, p1.
RCA Astro-Electronics leads twenty-five years of excellence. Schmidt, C. A., v28n2, Mar/Apr 1983, p1.
Software engineering: Growing our new roots. Hospodor, A. T., v28n6, Nov/Dec 1983, p1.
Systems thinking: A commitment to creativity and cooperation. Frederick, R. R., v28n4, July/Aug 1983, p1.
Technology transfer for industrial survival. Rosenthal, H., v28n3, May/June 1983, p1.
The changing environment for technical standards. Powers, K. H., v28n5, Sept/Oct 1983, p1.

Laser applications

Passive and active sounding from satellites for atmospheric temperatures and constituents. Rosenberg, A. and Hogan, D. B., v28n2, Mar/Apr 1983, p83.

Magnetic tape recording

Component analog video recording for studio applications. Gurley, T. M. and Haslett, C. J., v28n5, Sept/Oct 1983, p30.

Management

Intracorporate technology transfer: Theories and practice. Matulis, B. J., v28n3, May/June 1983, p20.
Manufacturing Technology Center: Key to advanced consumer-electronics manufacturing methods. Limberg, C. J., Stephens, D. T. and McCorkle, D. P., v28n3, May/June 1983, p13.
Systems thinking: A commitment to creativity and cooperation. Frederick, R. R., v28n4, July/Aug 1983, p1.
Technology transfer. Webster, W. M., v28n3, May/June 1983, p4.

Manufacturing

Factory data versus factory information. Stein, P. G. and Rayl, M., v28n4, July/Aug 1983, p81.

Manufacturing engineering

Producibility: The road to increased productivity. Gimigliano, R. F., v28n1, Jan/Feb 1983, p25.

Manufacturing equipment

Hydrogen peroxide solutions for silicon wafer cleaning. Kern, W., v28n4, July/Aug 1983, p99.

Manufacturing processes

Post-molding processing is a unique requirement for VideoDisc. Hakala, D. F., v28n5, Sept/Oct 1983, p57.

Materials, electronic

Diffusion length and surface photovoltage. Goldstein, B., Dresner, J., Moore, A. R. and Szostak, D., v28n3, May/June 1983, p45.

Memory devices

Microprocessor architecture and mass-storage techniques for field-portable automatic test equipment. Courcy, S. E. and DelPriore, G. A., v28n3, May/June 1983, p25.

Microcomputers

Microcomputer Club at Missile and Surface Radar. Poletti, G., v28n4, July/Aug 1983, p106.

Microprocessor applications

ATLAS environment for testing units with embedded processors. Prenoveau, R. L., v28n6, Nov/Dec 1983, p46.

Intelligent telephone applications of the CDP1800 microcomputer family. Nizet, J. and Shand, D., v28n1, Jan/Feb 1983, p72.

Microprocessor architecture and mass-storage techniques for field-portable automatic test equipment. Courcy, S. E. and DelPriore, G. A., v28n3, May/June 1983, p25.

Microwave technology

Personnel-tracking system. McEachern, R. M., Prost, K. J., Hampel, D. and Mawhinney, D. D., v28n6, Nov/Dec 1983, p57.

Military communications

Personnel-tracking system. McEachern, R. M., Prost, K. J., Hampel, D. and Mawhinney, D. D., v28n6, Nov/Dec 1983, p57.

Noise suppression

Noise-reduction in video signals by a digital fine-structure process. Acampora, A. A., Bunting, R. M. and Petri, R. J., v28n2, Mar/Apr 1983, p48.

Photovoltaic technology

Amorphous-silicon photovoltaics, Harnessing the sun with. Firester, A. H. and Carlson, D. E., v28n3, May/June 1983, p40.

Solar energy: Sun sets on RCA energy research. Williams, B. F., v28n3, May/June 1983, p39.

Sunshine in the Garden State: Solar-energy applications. Stranix, A. J. and Firester, A. H., v28n3, May/June 1983, p56.

Physical chemistry

Adherent dust particles. Williams, R. and Nosker, R. W., v28n1, Jan/Feb 1983, p76.

Physics

Diffusion length and surface photovoltage. Goldstein, B., Dresner, J., Moore, A. R. and Szostak, D., v28n3, May/June 1983, p45.

Picture tubes

COTY-29 (Combined Optimum Tube and Yoke) color picture tube system. Morrell, A. M., v28n4, July/Aug 1983, p28.

Printed circuit boards

Steady-state NOdal Thermal Analysis for printed wiring assemblies. Nguyen, T. N. and Wolfe, D. B., v28n1, Jan/Feb 1983, p56.

Process control

Process control charts to monitor and guide device production. Morey, R. F. and Kleppinger, R., v28n1, Jan/Feb 1983, p49.

Product engineering

Intracorporate technology transfer: Theories and practice. Matulis, B. J., v28n3, May/June 1983, p20.

Manufacturing Technology Center: Key to advanced consumer-electronics manufacturing methods. Limberg, C. J., Stephens, D. T. and McCorkle, D. P., v28n3, May/June 1983, p13.

Technology transfer. Webster, W. M., v28n3, May/June 1983, p4.

Productivity

APL and engineering productivity. Solomon, J. G., v28n1, Jan/Feb 1983, p18.

Enhancing engineering productivity in GSD. Moore, E. E., v28n1, Jan/Feb 1983, p4.

Information technology management. Freedman, H. A., v28n4, July/Aug 1983, p87.

Productivity: Where it begins and ends. Vollmer, J., v28n1, Jan/Feb 1983, p1.

Systems thinking: A commitment to creativity and cooperation. Frederick, R. R., v28n4, July/Aug 1983, p1.

Technology transfer for industrial survival. Rosenthal, H., v28n3, May/June 1983, p1.

Quality control

Process control charts to monitor and guide device production. Morey, R. F. and Kleppinger, R., v28n1, Jan/Feb 1983, p49.

Radar antennas

Producibility: The road to increased productivity. Gimigliano, R. F., v28n1, Jan/Feb 1983, p25.

Radar tracking

Missile Test Project support functions (sidebars to article by J. Simpson). Mertens, L., v28n2, Mar/Apr 1983, p67.

Space Shuttle operations, RCA's Missile Test Project support. Simpson, J., v28n2, Mar/Apr 1983, p64.

RCA American Communications, Inc

Satellite launching: A combined effort. Schwarze, J. J., v28n2, Mar/Apr 1983, p77.

RCA Astro-Electronics

Astro-Electronics' first quarter century. Astro-Electronics' Staff, v28n2, Mar/Apr 1983, p4.

Communications satellite software: a tutorial. Dusio, E. W., Murphy, T. P. and Cashman, W. F., v28n6, Nov/Dec 1983, p20.

RCA Astro-Electronics leads twenty-five years of excellence. Schmidt, C. A., v28n2, Mar/Apr 1983, p1.

Space Technology Roundtable: An Astro-Electronics Engineering Management Review and Forecast. Astro-Electronics' Staff, v28n2, Mar/Apr 1983, p11.

Thermally stable, precision antenna reflectors for satellite communications. Gounder, R. N., Talley, E. and Ino, S., v28n2, Mar/Apr 1983, p43.

RCA Automated Systems

ATLAS environment for testing units with embedded processors. Prenoveau, R. L., v28n6, Nov/Dec 1983, p46.

RCA Automated Systems: Twenty-five years on America's Technology Highway. Skavicus, A. J., v28n5, Sept/Oct 1983, p4.

RCA Engineer

Guide for RCA Engineer authors. Sweeny, M. R., v28n4, July/Aug 1983, p33.

RCA history

Astro-Electronics' first quarter century. Astro-Electronics' Staff, v28n2, Mar/Apr 1983, p4.

RCA Astro-Electronics leads twenty-five years of excellence. Schmidt, C. A., v28n2, Mar/Apr 1983, p1.

RCA Service Co

Space Shuttle operations, RCA's Missile Test Project support. Simpson, J., v28n2, Mar/Apr 1983, p64.

RCA Technical Excellence Center

RCA Technical Excellence Center established in Princeton. Jenny, H. K., v28n4, July/Aug 1983, p24.

Research and development

Intracorporate technology transfer: Theories and practice. Matulis, B. J., v28n3, May/June 1983, p20.

Manufacturing Technology Center: Key to advanced consumer-electronics manufacturing methods. Limberg, C. J., Stephens, D. T. and McCorkle, D. P., v28n3, May/June 1983, p13.

Technology transfer. Webster, W. M., v28n3, May/June 1983, p4.

Rockets

Launch vehicles for commercial communications. Muller, K. H., v28n2, Mar/Apr 1983, p72.

Satellite communication

Communications satellites, Introduction. Weaver, F. C., v28n2, Mar/Apr 1983, p24.

DBS service in critical system parameters. Klensch, R. J., Knight, S. P. and Staras, H., v28n2, Mar/Apr 1983, p58.

Thermally stable, precision antenna reflectors for satellite communications. Gounder, R. N., Talley, E. and Ino, S., v28n2, Mar/Apr 1983, p43.

Satellite design

Autonomous redundancy management on the DMSP spacecraft. Goodwin, P. G., v28n6, Nov/Dec 1983, p40.

Satellite ground support

Ground-control processing for communications satellites: Trends for the next decade. Cashman, W. F., v28n2, Mar/Apr 1983, p30.

Multi-satellite system management, The challenge of. Lewin, J., v28n4, July/Aug 1983, p93.

Satellite networks

Multi-satellite system management, The challenge of. Lewin, J., v28n4, July/Aug 1983, p93.

Satellite technology

Astro-Electronics' first quarter century. Astro-Electronics' Staff, v28n2, Mar/Apr 1983, p4.

Communications satellites, Introduction. Weaver, F. C., v28n2, Mar/Apr 1983, p24.

Space Technology Roundtable: An Astro-Electronics Engineering Management Review and Forecast. Astro-Electronics' Staff, v28n2, Mar/Apr 1983, p11.

Satellite weather

Passive and active sounding from satellites for atmospheric temperatures and constituents. Rosenberg, A. and Hogan, D. B., v28n2, Mar/Apr 1983, p83.

Satellite, synchronous

Satellite launching: A combined effort. Schwarze, J. J., v28n2, Mar/Apr 1983, p77.

Signal processing

Digital videotape recorders: Tape format and error-concealment considerations. Reitmeier, G. A., v28n5, Sept/Oct 1983, p38.

Image Processing Facility at Advanced Technology Laboratories. Schaming, W. B. and Zapriala, J. S., v28n1, Jan/Feb 1983, p62.

Personnel-tracking system. McEachern, R. M., Prost, K. J., Hampel, D. and Mawhinney, D. D., v28n6, Nov/Dec 1983, p57.

Silicon technology

Hydrogen peroxide solutions for silicon wafer cleaning. Kern, W., v28n4, July/Aug 1983, p99.

Silane discharges, Chemistry of. Catalano, A., Longeway, P. A. and Weakliem, H. A., v28n3, May/June 1983, p50.

Software engineering

Artificial intelligence research at ATL's Software Technology Laboratory. Zapriala, J. S., Spool, P. R., Whitehead, J. M., Gaev, J. A., Kleinosky, P. and McAdams, J., v28n6, Nov/Dec 1983, p4.

Autonomous redundancy management on the DMSP spacecraft. Goodwin, P. G., v28n6, Nov/Dec 1983, p40.

Communications satellite software: a tutorial. Dusio, E. W., Murphy, T. P. and Cashman, W. F., v28n6, Nov/Dec 1983, p20.

Computer-aided functional analysis: A new concept in system engineering. Felbinger, R. J. and Leahy, W. R., v28n6, Nov/Dec 1983, p51.

Software engineering: Growing our new roots. Hospodor, A. T., v28n6, Nov/Dec 1983, p1.

Software tools for system development, Application of. Suhy, G. W., v28n6, Nov/Dec 1983, p29.

Solar cells

Amorphous-silicon photovoltaics, Harnessing the sun with. Firester, A. H. and Carlson, D. E., v28n3, May/June 1983, p40.

Silane discharges, Chemistry of. Catalano, A., Longeway, P. A. and Weakliem, H. A., v28n3, May/June 1983, p50.

Solar energy

Amorphous-silicon photovoltaics, Harnessing the sun with. Firester, A. H. and Carlson, D. E., v28n3, May/June 1983, p40.

Solar energy: Sun sets on RCA energy research. Williams, B. F., v28n3, May/June 1983, p39.

Sunshine in the Garden State: Solar-energy applications. Stranix, A. J. and Firester, A. H., v28n3, May/June 1983, p56.

Solid state processing

Hydrogen peroxide solutions for silicon wafer cleaning. Kern, W., v28n4, July/Aug 1983, p99.

Space sciences

TIROS-based asteroid rendezvous mission. Maehl, R. C., v28n2, Mar/Apr 1983, p91.

Space shuttles

Launch vehicles for commercial communications. Muller, K. H., v28n2, Mar/Apr 1983, p72.

Missile Test Project support functions (sidebars to article by J. Simpson). Mertens, L., v28n2, Mar/Apr 1983, p67.

Space Shuttle operations, RCA's Missile Test Project support. Simpson, J., v28n2, Mar/Apr 1983, p64.

Space systems

Autonomous redundancy management on the DMSP spacecraft. Goodwin, P. G., v28n6, Nov/Dec 1983, p40.

Launch vehicles for commercial communications. Muller, K. H., v28n2, Mar/Apr 1983, p72.

Satellite launching: A combined effort. Schwarze, J. J., v28n2, Mar/Apr 1983, p77.

Spacecraft

TIROS-based asteroid rendezvous mission. Maehl, R. C., v28n2, Mar/Apr 1983, p91.

Statistics

Factory data versus factory information. Stein, P. G. and Rayl, M., v28n4, July/Aug 1983, p81.

Process control charts to monitor and guide device production. Morey, R. F. and Kleppinger, R., v28n1, Jan/Feb 1983, p49.

Synchronous satellite

In-orbit reconfigurable communications satellite antennas. Balcewicz, J. F., v28n2, Mar/Apr 1983, p36.

Systems engineering

Computer-aided functional analysis: A new concept in system engineering. Felbinger, R. J. and Leahy, W. R., v28n6, Nov/Dec 1983, p51.

Direct-broadcast satellite considerations. Buntschuh, R. F., v28n2, Mar/Apr 1983, p54.

Life-cycle software development, Experience with. Resnick, H. L., v28n6, Nov/Dec 1983, p35.

Managing the future: The expanding role of systems engineering in a high-technology society. Threston, J. T., v28n4, July/Aug 1983, p13.

Systems management

Managing the future: The expanding role of systems engineering in a high-technology society. Threston, J. T., v28n4, July/Aug 1983, p13.

Technical writing

Guide for RCA Engineer authors. Sweeny, M. R., v28n4, July/Aug 1983, p33.

Technology transfer

Technology transfer for industrial survival. Rosenthal, H., v28n3, May/June 1983, p1.

Teletext

NBC offers teletext. Watson, B., v28n5, Sept/Oct 1983, p18.

Teletext standards in North America. Astle, B., v28n5, Sept/Oct 1983, p15.

Television equipment

Component analog video recording for studio applications. Gurley, T. M. and Haslett, C. J., v28n5, Sept/Oct 1983, p30.

Digital television, Technology for. Marlowe, F. J., v28n5, Sept/Oct 1983, p26.

Digital videotape recorders: Tape format and error-concealment considerations. Reitmeier, G. A., v28n5, Sept/Oct 1983, p38.

Television standards

Development of international television standards. Hurst, R. N. and Powers, K. H., v28n5, Sept/Oct 1983, p10.

Teletext standards in North America. Astle, B., v28n5, Sept/Oct 1983, p15.

The changing environment for technical standards. Powers, K. H., v28n5, Sept/Oct 1983, p1.

Television transmission

Noise-reduction in video signals by a digital fine-structure process. Acampora, A. A., Bunting, R. M. and Petri, R. J., v28n2, Mar/Apr 1983, p48.

Television, digital

Development of international television standards. Hurst, R. N. and Powers, K. H., v28n5, Sept/Oct 1983, p10.

Digital television, Technology for. Marlowe, F. J., v28n5, Sept/Oct 1983, p26.

NBC offers teletext. Watson, B., v28n5, Sept/Oct 1983, p18.

Teletext standards in North America. Astle, B., v28n5, Sept/Oct 1983, p15.

The changing environment for technical standards. Powers, K. H., v28n5, Sept/Oct 1983, p1.

Television, history

Development of international television standards. Hurst, R. N. and Powers, K. H., v28n5, Sept/Oct 1983, p10.

TIROS satellite

TIROS-based asteroid rendezvous mission. Maehl, R. C., v28n2, Mar/Apr 1983, p91.

Traveling wave tubes

Direct-broadcast satellite considerations. Buntschuh, R. F., v28n2, Mar/Apr 1983, p54.

Author Index

Knight, S. P. with Klensch, R. J. et al., DBS service in critical system parameters. v28n2, Mar/Apr 1983, p58.

Kuczer, P. with Hook, H. O. et al., Gun parts measurement, A computerized system for. v28n1, Jan/Feb 1983, p43.

Leahy, W. R. with Felbinger, R. J., Computer-aided functional analysis: A new concept in system engineering. v28n6, Nov/Dec 1983, p51.

Lewin, J., Multi-satellite system management, The challenge of. v28n4, July/Aug 1983, p93.

Limberg, C. J., Stephens, D. T. and McCorkle, D. P., Manufacturing Technology Center: Key to advanced consumer-electronics manufacturing methods. v28n3, May/June 1983, p13.

Longeway, P. A. with Catalano, A. et al., Silane discharges, Chemistry of. v28n3, May/June 1983, p50.

Maehl, R. C., TIROS-based asteroid rendezvous mission. v28n2, Mar/Apr 1983, p91.

Marlowe, F. J., Digital television, Technology for. v28n5, Sept/Oct 1983, p26.

Matulis, B. J., Intracorporate technology transfer: Theories and practice. v28n3, May/June 1983, p20.

Mavrodin, A. with Hook, H. O. et al., Gun parts measurement, A computerized system for. v28n1, Jan/Feb 1983, p43.

Mawhinney, D. D. with McEachern, R. M. et al., Personnel-tracking system. v28n6, Nov/Dec 1983, p57.

McAdams, J. with Zapriala, J. S. et al., Artificial intelligence research at ATL's Software Technology Laboratory. v28n6, Nov/Dec 1983, p4.

McCorkle, D. P. with Limberg, C. J. et al., Manufacturing Technology Center: Key to advanced consumer-electronics manufacturing methods. v28n3, May/June 1983, p13.

McEachern, R. M., Prost, K. J., Hampel, D. and Mawhinney, D. D., Personnel-tracking system. v28n6, Nov/Dec 1983, p57.

Mertens, L., Missile Test Project support functions (sidebars to article by J. Simpson). v28n2, Mar/Apr 1983, p67.

Moore, A. R. with Goldstein, B. et al., Diffusion length and surface photovoltage. v28n3, May/June 1983, p45.

Moore, E. E., Enhancing engineering productivity in GSD. v28n1, Jan/Feb 1983, p4.

Morey, R. F. and Kleppinger, R., Process control charts to monitor and guide device production. v28n1, Jan/Feb 1983, p49.

Morrell, A. M., COTY-29 (Combined Optimum Tube and Yoke) color picture tube system. v28n4, July/Aug 1983, p28.

Muller, K. H., Launch vehicles for commercial communications. v28n2, Mar/Apr 1983, p72.

Munk, G. E. with Heffernan, J. S., Software tools to enhance engineering productivity, Use of. v28n1, Jan/Feb 1983, p37.

Murphy, T. P. with Dusio, E. W. et al., Communications satellite software: a tutorial. v28n6, Nov/Dec 1983, p20.

Nguyen, T. N. and Wolfe, D. B., Steady-state NOdal Thermal Analysis for printed wiring assemblies. v28n1, Jan/Feb 1983, p56.

Nizet, J. and Shand, D., Intelligent telephone applications of the CDP1800 microcomputer family. v28n1, Jan/Feb 1983, p72.

Nosker, R. W. with Williams, R., Adherent dust particles. v28n1, Jan/Feb 1983, p76.

Petri, R. J. with Acampora, A. A. et al., Noise-reduction in video signals by a digital fine-structure process. v28n2, Mar/Apr 1983, p48.

Poletti, G., Microcomputer Club at Missile and Surface Radar. v28n4, July/Aug 1983, p106.

Powers, K. H., The changing environment for technical standards. v28n5, Sept/Oct 1983, p1.

——— with Hurst, R. N., Development of international television standards. v28n5, Sept/Oct 1983, p10.

Prenoveau, R. L., ATLAS environment for testing units with embedded processors. v28n6, Nov/Dec 1983, p46.

Prost, K. J. with McEachern, R. M. et al., Personnel-tracking system. v28n6, Nov/Dec 1983, p57.

Quardt, D. C. with Reid-Green, K. S. et al., CAD/CAM at RCA Laboratories: Tools and applications, present and future. v28n1, Jan/Feb 1983, p29.

Rayl, M. with Stein, P. G., Factory data versus factory information. v28n4, July/Aug 1983, p81.

RCA Engineer Staff, David Sarnoff Awards for Outstanding Technical Achievement, for 1983. v28n4, July/Aug 1983, p4.

Reid-Green, K. S., Hatter, W. A. and Quardt, D. C., CAD/CAM at RCA Laboratories: Tools and applications, present and future. v28n1, Jan/Feb 1983, p29.

Reitmeier, G. A., Digital videotape recorders: Tape format and error-concealment considerations. v28n5, Sept/Oct 1983, p38.

Resnick, H. L., Life-cycle software development, Experience with. v28n6, Nov/Dec 1983, p35.

Rodda, W. E. with Babcock, W. E. et al., Consumer-type high-definition-television displays, Some power considerations for. v28n5, Sept/Oct 1983, p48.

Rosenberg, A. and Hogan, D. B., Passive and active sounding from satellites for atmospheric temperatures and constituents. v28n2, Mar/Apr 1983, p83.

Rosenthal, H., Technology transfer for industrial survival. v28n3, May/June 1983, p1.

Saunders, W. C. with Hook, H. O. et al., Gun parts measurement, A computerized system for. v28n1, Jan/Feb 1983, p43.

Schaming, W. B. and Zapriala, J. S., Image Processing Facility at Advanced Technology Laboratories. v28n1, Jan/Feb 1983, p62.

Schmidt, C. A., RCA Astro-Electronics leads twenty-five years of excellence. v28n2, Mar/Apr 1983, p1.

Schwarze, J. J., Satellite launching: A combined effort. v28n2, Mar/Apr 1983, p77.

Shand, D. with Nizet, J., Intelligent telephone applications of the CDP1800 microcomputer family. v28n1, Jan/Feb 1983, p72.

Simpson, J., Space Shuttle operations, RCA's Missile Test Project support. v28n2, Mar/Apr 1983, p64.

Skavicus, A. J., RCA Automated Systems: Twenty-five years on America's Technology Highway. v28n5, Sept/Oct 1983, p4.

Sockwell, C., SAS (Statistical Applications System) applications within RCA (sidebars to article by R. P. Dreher). v28n1, Jan/Feb 1983, p14.

Solomon, J. G., APL and engineering productivity. v28n1, Jan/Feb 1983, p18.

Spool, P. R. with Zapriala, J. S. et al., Artificial intelligence research at ATL's Software Technology Laboratory. v28n6, Nov/Dec 1983, p4.

Staras, H. with Klensch, R. J. et al., DBS service in critical system parameters. v28n2, Mar/Apr 1983, p58.

Stein, P. G. and Rayl, M., Factory data versus factory information. v28n4, July/Aug 1983, p81.

Stephens, D. T. with Limberg, C. J. et al., Manufacturing Technology Center: Key to advanced consumer-electronics manufacturing methods. v28n3, May/June 1983, p13.

Stranix, A. J. and Firester, A. H., Sunshine in the Garden State: Solar-energy applications. v28n3, May/June 1983, p56.

Suhy, G. W., Software tools for system development, Application of. v28n6, Nov/Dec 1983, p29.

Sweeny, M. R., Guide for RCA Engineer authors. v28n4, July/Aug 1983, p33.

Szostak, D. with Goldstein, B. et al., Diffusion length and surface photovoltage. v28n3, May/June 1983, p45.

Talley, E. with Gounder, R. N. et al., Thermally stable, precision antenna reflectors for satellite communications. v28n2, Mar/Apr 1983, p43.

Threston, J. T., Managing the future: The expanding role of systems engineering in a high-technology society. v28n4, July/Aug 1983, p13.

Vollmer, J., Productivity: Where it begins and ends. v28n1, Jan/Feb 1983, p1.

Watson, B., NBC offers teletext. v28n5, Sept/Oct 1983, p18.

Weakliem, H. A. with Catalano, A. et al., Silane discharges, Chemistry of. v28n3, May/June 1983, p50.

Weaver, F. C., Communications satellites, Introduction. v28n2, Mar/Apr 1983, p24.

Webster, W. M., Technology transfer. v28n3, May/June 1983, p4.

Wedam, W. F. with Babcock, W. E. et al., Consumer-type high-definition-television displays, Some power considerations for. v28n5, Sept/Oct 1983, p48.

Whitehead, J. M. with Zapriala, J. S. et al., Artificial intelligence research at ATL's Software Technology Laboratory. v28n6, Nov/Dec 1983, p4.

Williams, B. F., Solar energy: Sun sets on RCA energy research. v28n3, May/June 1983, p39.

Williams, R. and Nosker, R. W., Adherent dust particles. v28n1, Jan/Feb 1983, p76.

Wolfe, D. B. with Nguyen, T. N., Steady-state NOdal Thermal Analysis for printed wiring assemblies. v28n1, Jan/Feb 1983, p56.

Zapriala, J. S. with Schaming, W. B., Image Processing Facility at Advanced Technology Laboratories. v28n1, Jan/Feb 1983, p62.

———, Spool, P. R., Whitehead, J. M., Gaev, J. A., Kleinosky, P. and McAdams, J., Artificial intelligence research at ATL's Software Technology Laboratory. v28n6, Nov/Dec 1983, p4.

Editorial Representatives

Contact your Editorial Representative at the TACNET numbers listed here to schedule technical papers and announce your professional activities.

Broadcast Systems Division (BSD)

		TACNET
*Jack Dearing	Camden, New Jersey	222-4688
Clinton Glenn	Meadowlands, Pennsylvania	228-6231

Consumer Electronics (CE)

*Eugene Janson	Indianapolis, Indiana	422-5208
Francis Holt	Indianapolis, Indiana	422-5217
Larry Olson	Indianapolis, Indiana	422-5117
Byron Taylor	Indianapolis, Indiana	426-3247
Don Willis	Indianapolis, Indiana	422-5883

Corporate Technology

*Hans Jenny	Princeton, New Jersey	226-2111

Government Systems Division (GSD)

Advanced Technology Laboratories

*Merle Pietz	Camden, New Jersey	222-2161
Ed Master	Camden, New Jersey	222-2731

Astro-Electronics

*Frank Yannotti	Princeton, New Jersey	229-2544
Carol Coleman	Princeton, New Jersey	229-2919

Automated Systems

*Paul Seeley	Burlington, Massachusetts	326-3095
Dale Sherman	Burlington, Massachusetts	326-3403

Government Communications Systems

*Dan Tannenbaum	Camden, New Jersey	222-3081
Thomas Altgilbers	Camden, New Jersey	222-3351

GSD Staff

*Peter Hahn	Cherry Hill, New Jersey	222-5319

Missile and Surface Radar

*Don Higgs	Moorestown, New Jersey	224-2836
Graham Boose	Moorestown, New Jersey	224-3680
Jack Friedman	Moorestown, New Jersey	224-2112

National Broadcasting Company (NBC)

*Bob Mausler	New York, New York	324-4869

New Products Division

*R.E. Reed	Lancaster, Pennsylvania	227-2485
Bob McIntyre	Ste Anne de Bellevue	514-457-9000

Cablevision Systems

*John Ovnick	Van Nuys, California	534-3011

Patent Operations

		TACNET
George Haas	Princeton, New Jersey	226-2491

RCA Communications

American Communications

*Murray Rosenthal	Princeton, New Jersey	258-4192
Carolyn Powell	Princeton, New Jersey	258-4194

Global Communications

*Dorothy Unger	Piscataway, New Jersey	

RCA Laboratories

Eva Dukes	Princeton, New Jersey	226-2882

RCA Records

*Greg Bogantz	Indianapolis, Indiana	424-6141

RCA Service Company

*Murray Kaminsky	Cherry Hill, New Jersey	222-6247
Dick Dombrosky	Cherry Hill, New Jersey	222-4414
Ray MacWilliams	Cherry Hill, New Jersey	222-5986

"SelectaVision" VideoDisc Operations

*Nelson Crooks	Indianapolis, Indiana	426-3164

Solid State Division (SSD)

*John Schoen	Somerville, New Jersey	325-6467

Power Devices

Harold Ronan	Mountaintop, Pennsylvania	327-1633
		or 327-1827

Integrated Circuits

Dick Morey	Palm Beach Gardens, Florida	722-1262
Sy Silverstein	Somerville, New Jersey	325-6168
John Young	Findlay, Ohio	425-1307

Video Component and Display Division

*Ed Madenford	Lancaster, Pennsylvania	227-3657
Nick Meena	Circleville, Ohio	432-1228
Jack Nubani	Scranton, Pennsylvania	329-1499
J.R. Reece	Marion, Indiana	427-5566

***Technical Publications Administrators, responsible for review and approval of papers and presentations, are indicated here with asterisks before their names.**

RCA Engineer

A technical journal published by
the RCA Technical Excellence Center
"by and for the RCA engineer"

13 Roszel Road, P.O. Box 432, Princeton, NJ 08540
Address correction requested

Printed in the U.S.A.
Form No. RE-28-6