

COMPUTER AIDED DESIGN

UUCS-84-003

March 1984

This report is based on the proposal submitted to the National Science Foundation in September 1981, as part of the Coordinated Experimental Computer Science Research Program. The sections covering the budget and biographical data on the senior research personnel have not been included. Also, the section describing the department facilities at the time of the proposal submission is not included, because it would be only of historical interest.

If anything could be called a five-year plan for the department's research activities, it is this proposal. In reviewing it again after well over two years, it is encouraging to see just how much of the work outlined then has either been successfully completed or is in progress.

Richard F. Riesenfeld, Project Director
Lee A. Hollaar, Co-Principal Investigator

Summary of Proposed Research

The current work on computer aided design (CAD) systems for three-dimensional shape design and integrated circuits being performed by the University of Utah's Department of Computer Science is showing excellent preliminary results. However, even more dramatic improvements are possible by taking full advantage of other ongoing work within the department in portability and software tools, to develop an enhanced software development environment; in database systems, to provide an efficient, uniform means of accessing, modifying, and sharing design data; and in multiprocessor and network operations, to allow the various tasks in the systems to be partitioned and run on processors best suited to their particular requirements.

The current support for these ongoing projects allows satisfactory progress towards their major objectives, but cannot adequately support the inter-project activities necessary to develop state of the art CAD systems. Under this grant we propose to specifically assist these critical inter-project activities by supporting professional programmers and graduate research assistants working in these areas. Limited support will also be given the senior faculty investigators to allow them to broaden their research to include these inter-project activities.

We also propose to enhance the graphics capabilities of the department by the acquisition of a number of designer workstations. These will substantially aid in the development of the necessary CAD system and support software by removing the current bottleneck caused by many projects trying to use the current graphics laboratory. They will also provide more suitable inputs to the VLSI development and simulation packages, and allow more projects to take advantage of the department's VLSI facilities.

Necessary expansion to the department's central computing facility will also be made to accommodate the new research effort. For example, a service node will be added to the local network to provide file storage, including a centralized database management system organized to efficiently support a CAD system, and access to a variety of peripheral devices not cost effective for each workstation.

Introduction

Computer Aided Design (CAD) is a challenging area of engineering and science for many reasons, one of which is its eclectic nature. A successful CAD system requires intimate knowledge of the traditional problems and approaches to its subject area, beginning with conceptual design methods, extending through the preliminary phases of mission and configuration design, and finally passing into the levels of copious and exacting detailing and final design. During this evolution, a product must undergo very different kinds of analytical, consistency, and performance checks. The process is frequently described as the design spiral, rather than cycle, with the allusion that a closer approximation to some "ideal design" is achieved with each pass. Finally the design, to be truly useful and appropriate to the demands of manufacturing, must fit directly into the manufacturing process. In fact, a major difficulty with the traditional process is that the result of the design process, typically large amounts of paper, is not the right kind of information for feeding the manufacturing process. This, of course, has led to the conceptual synthesis of the processes under the term CAD/CAM: Computer Aided Design and Computer Aided Manufacturing.

It has been observed that computers can be useful, indeed in some cases essential, in many aspects of the design process and in many different kinds of product design. It would be unrealistic, for example, to even attempt to design an integrated circuit of the complexity of current microprocessor chips without extensive computer assistance. Similarly, complex manufactured products, of which aircraft and turbine engines are but two examples, are complex assemblies of parts. Each type of part within a product may have considerable technological sophistication expressed in the part shapes, properties, and manufacturing processes. The leverage provided by computer systems has already become essential in the engineering and manufacture of such products. Current research in CAD systems aims to extend the benefits derived from supporting design efforts with the computer system tools now available. The computer-aided design system has to provide quite different kinds of support during the different phases of design, which is part of the difficulty in trying to bring computers into this intrinsically human act of creativity.

In the earliest phases of design, a computer system should be a real sketchpad for ideas, using a terminal and an exceptionally supportive editor as the equivalent of paper and pencil. It should avoid inhibiting the free flow of ideas and approaches from which a more specific solution might follow. Its role is often passive; a primary activity for it during this phase is to record the design concepts for later refinement. These electronically stored designs can also be forwarded to others for comment or elaboration, performing much like electronic mail systems on most time-sharing computer systems. The critical role of a computer system at this time in the design process is one of providing vastly superior visualizations of the model. Design nearly always involves visualizations, and computer graphics techniques have developed to a degree of sophistication that they have become indispensable in many design environments.

In the middle phase of design the computer can offer aid in more constructive and independent ways. This is the phase in which designs must be validated and confirmed. They are subjected to the multifarious set of analytic procedures necessary to explore all aspects of a complicated design. Experts examine every possible consideration during this period of concentrated activity and substantial changes are commonly made to the candidate model as it is discovered that the originally proposed concepts are inconsistent or cannot meet certain analytic requirements. At this point in the design process, communication is of the essence, for it is critically important to maintain a completely consistent current model which reflects all of the changes which the various concerns might dictate. This is usually achieved through a centralized and standardized "Master Model" accessible to all parties responsible for the total product design.

The detailed design is most easily mechanized because it is here that the powers of the computer are most directly invoked. Much of the detail design is quite mechanical, meaning that it primarily reflects and replicates the structure of higher-level design decisions and practices. This tedium lends itself to algorithmic encoding. The issues tend to be localized in the design and the meaning and intention are

clear. These final detailed design tasks are simply a matter of elaborating a level of detail sufficient to assure that the product will meet final specifications and exhibit the needed information for fabrication. For example, in VLSI design, final design steps include design rule checking and generation of the output tapes necessary to drive the desired pattern generators. In machined parts, this level includes the detailed planning of the sequence of operations involved in manufacturing and assembling a finished part or product, as well as the geometric derivation of NC toolpaths, tooling, dies and molds, and such documentation as illustrations for maintenance manuals. This is the lowest level of design and the place where the greatest inroads have been made in the past, but the potential leverage of integrating many of these processes around a common set of design representations and editing and translation processes has not yet been realized.

While discussion of the evolution of design often identifies the conceptual, preliminary, and detailed phases, it must also be observed that product and part designs are frequently modifications of previous designs rather than designs "ab initio". A ship design, for example, most often begins with an existing similar ship which must be modified to meet some new specifications. Computer aided design is especially useful in developing new designs by rapidly and consistently applying alterations to previously archived designs. New VLSI designs are often based on using previously developed cells in a new manner, or updating a previous design based on a new cell library. New machine parts are often quite similar to previously designed parts, especially within the same product line. (A "classification system" for recognizing and retrieving clusters of similar designs is thus an important part of a large-scale CAD system database.) In this important aspect of design, it is critical that all of the implications of a perturbation in a specification are recognized and properly reflected throughout the affected parts of the total design. Without the aid of a computer system to support this function, design modification is an error prone occupation. There are many stories of modifications in one place, which were overlooked in another portion of the design, with costly results. A sufficiently strong associative model for the overall design can be very helpful in developing new but not radically different products, as well as refining existing designs through engineering changes.

Current Departmental CAD Research

There are several areas of current research in our department which specifically relate to the problem of CAD. These include projects directly involved with CAD as well as projects, such as portability and software development systems, whose results can make a substantial contribution to CAD research. Through this grant we hope to unite these activities to address one of the nation's most pressing problems in experimental computing: increasing human productivity through computer aids.

The University of Utah Computer Science department was founded on the basis of excellence in computer graphics and quickly became synonymous with it throughout the world. Much of the research in the department was driven either directly or indirectly by this expertise in computer graphics. This theme remains strong in the department, although during its maturation, the department has become better balanced by including many other areas of computer science.

The computer graphics research will continue to be important in connection with this proposal, since visualization is one of the most important support activities for increasing the human effectiveness in design and analysis. While the methods for producing realistic looking images are rather sophisticated but well understood, it is likely that important advances will come in the use of nonrealistic images which are intended to reveal specifically interesting properties of a model. The figures in this proposal are good examples of images that use nonrealistic color in order to communicate special meaning about objects. For example, color has been used very effectively as an extra dimension in portraying the results of a stress analysis. Many other examples of the utilization of the high information content inherent in this type of image can be cited.

For nearly a decade Utah has been an active center for researchers in the area of Computer Aided Geometric Design, or CAGD. This work has involved the development of mathematical principles and methods for freeform model description, as well as the study of methods for interactive specification and

manipulation. It has resulted in a body of knowledge which is quickly finding its way into use by the many industries that have need for such techniques. Most recently, the development of an experimental system called ALPHA-1 has been undertaken in an effort to combine in a single system both high quality computer graphics and freeform surface representation and design. The system is based on new mathematical theory (discrete splines) and computational algorithms (Oslo Algorithm) for dealing with smooth surface forms [4, 22]. The results of this year-old implementation effort are illustrated in the following figures.

The first figure is a picture of an aircraft bulkhead which was produced using the ALPHA-1 modelling system. It is intended to demonstrate that this approach can successfully model a part involving both complex aerodynamic shapes, like the side which is derived from the actual skin geometry of the airplane engine strut, and simple geometry like the pocketing. It also illustrates that one can derive high quality computer graphics images directly from the spline geometry model itself without using a special permanent and separate graphics model. It is felt that very subtle and faithful illumination models can produce visual cues that are valuable to the designer. Color can be used as an added dimension for communicating manufacturing information, as is portrayed here.

The second picture illustrates that the approach taken in the ALPHA-1 system supports the computations associated with boolean operations like intersections of two arbitrary surfaces. A further conclusion from these pictures is that the normal surface calculations required for a high quality rendering of this part, and the calculations required for intersections provide the essential information for computing direct NC descriptions of the part. The enlargement reveals that the methods used in rendering do not degrade on closeups, since the derived graphics database is ephemeral and employed only to produce the current image. This is an unusual aspect of graphics modelling systems. The third figure shows the intersection curve of the cutting surface traced out on the bulkhead, and indicates that the parts would not readily mate if they were made of rigid materials.

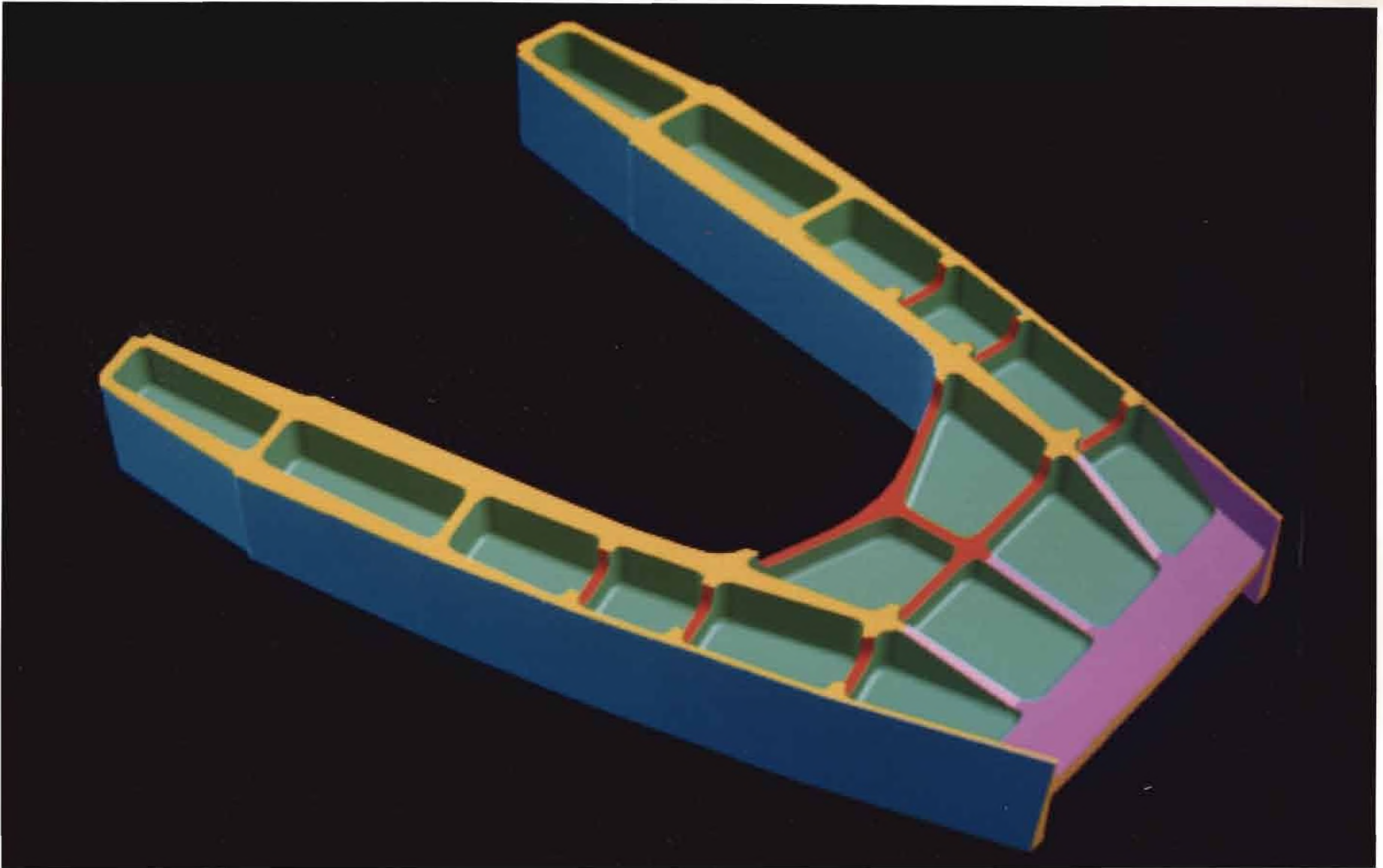
To our best knowledge these pictures of the bulkhead demonstrate a unique capability in a geometric modelling approach, for they combine high quality raster graphics together with both freeform and relatively simple surfaces within the context of a single computer model. In the future these methods will be generalized to generate line drawing images as well.

3D visualization can be enhanced with the development of new hardware devices. At the University of Utah work is being conducted on the design and development of a 3D viewing terminal which employs a vibrating mirror to generate a true 3D (virtual) image [1]. The potential of a practical 3D viewing device applied to design is evident, although considerable research in developing the display appropriate algorithms is necessary to satisfy the needs of computer aided design. This is part of the research which is being proposed under this program: the coordination of the work in computer aided geometric design with the 3D viewing terminal. It is felt that many of the rendering principles and algorithms used in producing the bulkhead pictures can be applied to develop related algorithms for the 3D Viewer.

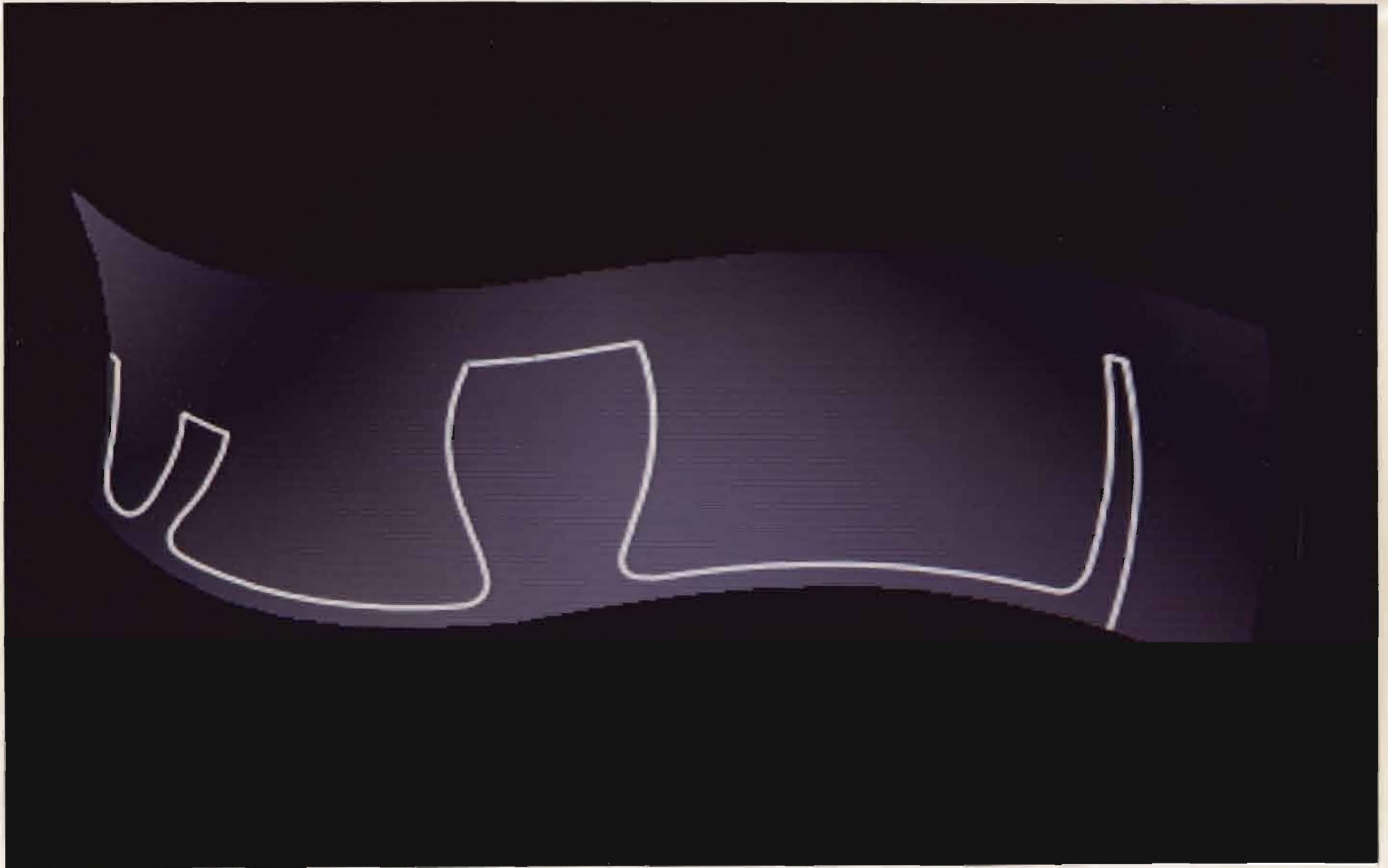
Current Departmental VLSI Circuit Design Research

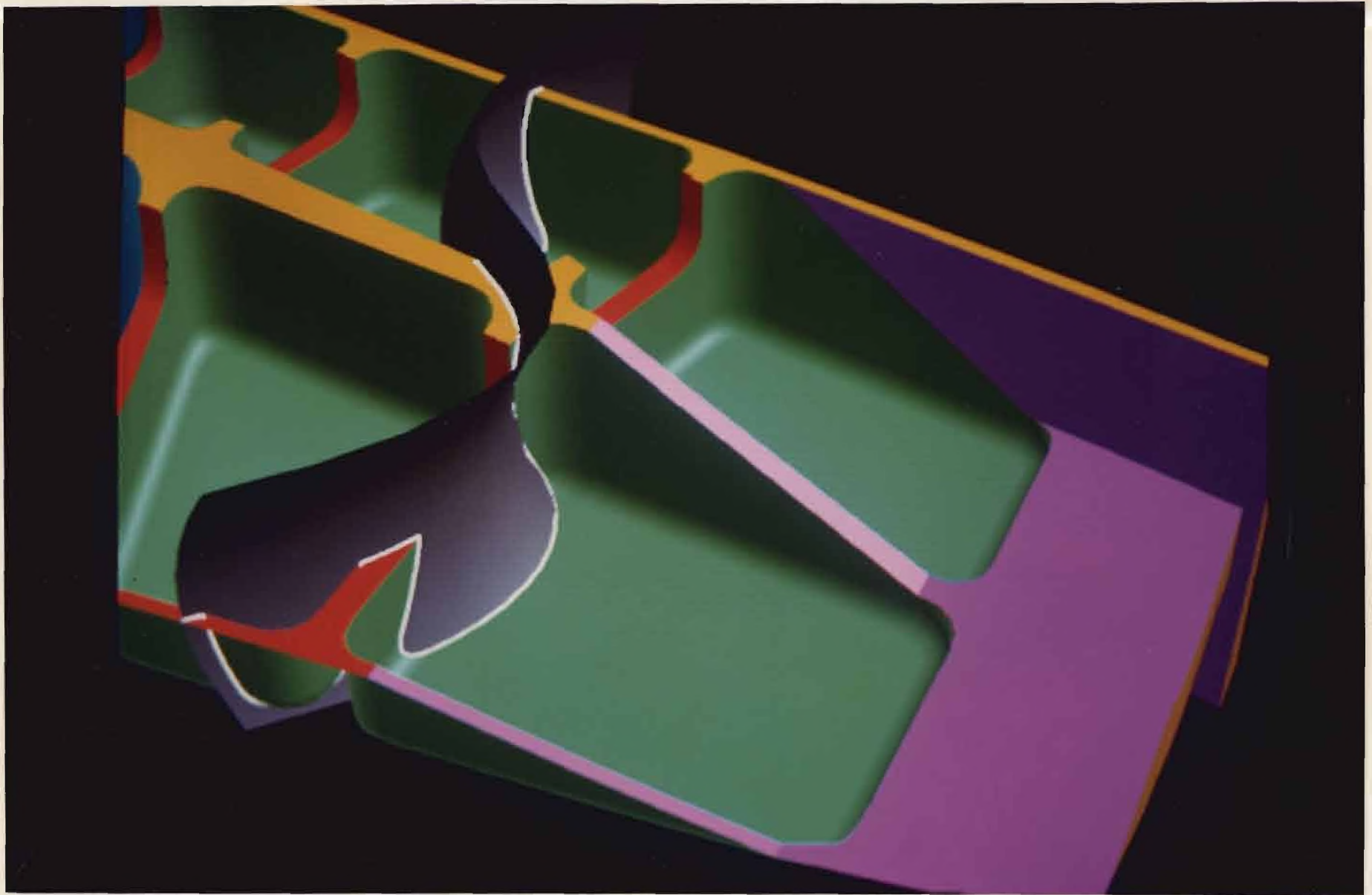
The department's work on the design of very large scale integrated circuits concentrates on two areas: the development of novel circuits suggested by other research projects in the department, and improvements to the tools and techniques for efficiently developing custom integrated circuits. Circuit technologies employed include NMOS, CMOS, and I²L, with many of the circuits designed being fabricated either in the University's Hedco Microelectronics Laboratory (described later in this proposal) or by outside facilities. Circuit design facilities include a ComputerVision designer system, a Hewlett-Packard 3000 computer system, and most of the available IC design software, running on the departmental computer systems.

Circuits under development include a specialized processor running the CORDIC trigonometric algorithm, which a graduate student under the direction of Kent Smith has implemented in a variety of forms, including both clocked and asynchronous versions. A project headed by Alan Davis is implementing a set of self timed [23], bit sliced micro-sequencer parts. These include a soft controller,



Aircraft bulkhead part modelled and rendered using Alpha-1 system. Colors indicate required separate N/C machining tasks.





Bulkhead with intersection cut using Alpha_1 system.

comparable to the Advanced Micro Devices 2911 sequencer, memory interface, and submachine control. These units communicate using a four cycle request-acknowledge protocol and are completely asynchronous with respect to other system parts.

Work is also proceeding on the implementation of a type of text scanning processor, based on the partitioned finite state automaton [12] developed by Lee Hollaar and Roger Haskin, one of his former graduate students. The PFSA provides all the capabilities of a conventional finite state recognizer and has extensions to aid in matching patterns which include tokens indicating that an arbitrary character or string of characters can occur in a given location. It can be readily implemented and employed as a special processor attached to a disk memory system, since it consists of a number of identical character matchers, each comprising about 5 kilobits of memory and a minimal amount of random logic, and capable of performing its comparisons in less than the character delivery time of a normal disk drive. Other circuits have been designed and fabricated as test circuits or as part of our course sequence in integrated circuits.

Most of the research on VLSI circuits within the department is concerned with the development of better tools for the design of structured integrated circuits. Much of it consists of improvements in the methodology and development tools for structured logic arrays, and in particular the stored logic array (SLA). As described by Patil and Welch [21], the SLA is a folded programmable logic array (PLA) which contains flip-flops distributed throughout the array, with arbitrary column and row breaks to give multiple, independent, finite state machines on a single integrated circuit. It can be used to implement both control sections and data handling portions, such as counters, registers, and adders.

SLA circuit design is performed by placing logic symbols on a grid. Each logic symbol corresponds to a predefined cell in the SLA library appropriate for the chosen logic family, and includes flip-flops, AND and OR points in the logic array matrix, pullup resistors, feed-throughs, corners, and inverters. Each symbol's placement becomes both the logic description of the circuit and the cell placement and interconnection necessary to produce the desired circuit. The composite of the circuit can be produced simply by substituting the specified composite description of the cells for their logic symbols. Work is also being done on techniques for mapping the SLA logic symbols onto gate array chips, rather than completely custom circuits.

Several design tools have been constructed to aid in the design of an SLA integrated circuit. These include editors to help the designer create the necessary logic diagram, running on both the ComputerVision system and the department's DECsystem-20. Their utility is limited, however, because only two workstations are available on the ComputerVision system, severely limiting its access for researchers outside the immediate VLSI group. The DECsystem-20 SLA editor uses a standard ASCII CRT display (24 lines of 80 characters), limiting its utility due to the severely restricted graphics available and the limited size of the display. The proposed workstations eliminate both these problems.

A number of simulation programs have been developed or extended by members of the VLSI group. These include a gate-level logic simulator (SIMULOG) which includes not only the normal logic gates, but also PLAs, RAMs and ROMs, and pass transistors. A modified version of the SPICE integrated circuit analysis program is available for checking the real-time performance of a circuit. An SLA simulator has also been developed to aid in checking a design to assure that it matches the circuit's specifications. This simulator does a complete logic simulation of exactly what the designer has specified. No translation errors can occur because the actual mask generation is made from the same specification database as was used for the simulation.

Other work on VLSI design automation has concentrated on higher level tools for mapping logic designs into final IC geometric designs (shapes on various layers which create connections or components such as transistors). Recent technical reports have discussed a layout modeling language for structured arrays [7] and a computer aided design system for placing and connecting very large cells (registers, adders, databuses, and input/output drivers and receivers) [20]. A new technique for automatically mapping asynchronous circuit state diagrams into their final geometric designs has been proposed by Hollaar, based on an implementation which has a flip-flop per state [25]. While it may seem

that this requires considerably more components than a conventionally implemented asynchronous circuit, for sequencers or control sections, the costs are comparable [14]. The technique has fewer restrictions on acceptable input behavior than normal fundamental mode asynchronous circuits, and has been extended to include a FORK/JOIN parallel execution construct and a subroutine capability. Research is concentrating on the theory and operation of the basic circuits, the implementation of the mapping algorithms, and the computer assisted development of appropriate diagnostic sequences.

An effort recently funded by the Defense Advanced Research Projects Agency, with Elliott Organick as principal investigator, is examining the mapping of high order language programs into VLSI structures. The language is a subset of the Ada programming language recently developed for the Department of Defense [5], selected because its concepts of program units, packages, and tasks map easily onto a number of specialized, intercommunicating finite state machines whose structures can be determined from the data declarations and operations being performed. Preliminary, hand-produced mappings of a number of simple programs have been performed, and a more complex program, the Internet Protocol module recently adopted as a DoD standard, is being mapped from its Ada implementation to lower levels of abstraction.

A mapping method from a high level programming language into VLSI circuits offers a number of distinct advantages. The program specification of an algorithm also provides a simulator to assure the algorithm operates correctly, since the program need only be run on a conventional processor. If the programming language is widely used by programmers, it guarantees a broad audience that can understand, criticize, and improve a circuit's specification. Furthermore, languages like Ada offer clear separation between specification and implementation at hierarchically structured levels of abstraction. The modular structure selected by the programmer/designer can serve as a guide (and in many cases, a template) for the intended hardware structure. Thus, models of hardware organization may be related to modules and lesser units of the program, and models of communication between hardware units are related to protocols used by interface modules and lesser units of programs.

Summaries of Individual Proposed Research Projects

We propose to continue and expand on the current work on computer aided design discussed above, taking full advantage of a number of other projects within the department. In addition to the equipment being requested, primarily a number of high quality graphics oriented workstations, support is being requested for senior researchers, visiting faculty, professional programmers and technicians, and graduate students. These staff members will be used primarily to develop the necessary support hardware and software systems for the CAD research, and to provide bridges between the CAD effort and other projects in the department.

The specific projects being proposed can be divided into three distinct groups: projects working directly on computer aided design, extensions to current research activities to benefit CAD, and projects from within and outside of the Department of Computer Science which will use the CAD systems being developed.

Computer Aided Design Projects

These projects represent the major thrust of the proposed research. They represent continuation and expansion of the current departmental research, taking full advantage of the advanced computer science developments of the various support projects.

Computer Aided 3D Shape Design. (Senior Investigator: Richard F. Riesenfeld) The current work on computer aided 3D shape design, which is focused around a project called ALPHA-1, will be one of the primary beneficiaries of this grant. The graphical workstations to be purchased will remove one of the major bottlenecks slowing exploitation of the current work: the limited availability, and consequent scheduling problems, of graphical devices. The current displays (an Evans and Sutherland Multi-Picture System and a Grinnell Frame Buffer, being augmented this summer with a Megatek 9250 color raster system) are certainly capable of performing the operations required for the current work, but must be shared not only among the ALPHA-1 researchers, but VLSI and other researchers, and some classroom work. The use of lower cost graphical workstations will not only provide better access for software development, but will allow "graphical comments," simple pictures and diagrams documenting how a particular algorithm works, to be included in programs and documentation.

The additional facilities will not only further the software development of ALPHA-1, but will also provide a sufficient number of stations so that considerable experimentation and experience can be accumulated in using the design stations on actual design projects. At present, any practical modelling exercises compete with systems development time since there is only one such computing resource for this activity.

We also propose to substantially expand the scope of ALPHA-1 to complete many of the areas of development necessary to make it a viable system in actual practice. At this point it has demonstrated and validated the basic approach of using discrete splines as a central model for geometry and high quality rendering. Now we will extend our work in deriving analytic models from the central geometry model, and to developing direct numeric control programs which are necessary to manufacture many of the parts which might be modelled by ALPHA-1. Although previous attempts to use techniques such as symbolic computation to analytically manipulate and derive advanced surface representations have shown promise, personnel to work on the interfacing of these two projects have been limited. Similarly, work will be done on the development of suitable databases and improved software tools to aid in creating high quality software and transporting it to a variety of systems so that others can benefit from the proposed work. The personnel and systems supported by this grant will make this possible.

A very interesting and useful product of this proposal will be the application of our expertise in software engineering, including interpretive programming environments and symbolic processing, to the development of customizable design systems. In this context a design station is a personal environment

which should be highly tuned to support users in their specialized goals and particular design idiosyncrasies. The station should reflect their needs and style. In order to achieve this, we can imagine a kind of work station compiler which takes into account the special circumstances surrounding a particular effort and then generates an appropriately endowed software system.

Integrated Circuit Design. (Senior Investigators: Kent F. Smith, Alan L. Davis, Lee A. Hollaar) Present design tools will be substantially expanded to take full advantage of the high quality graphics available at each workstation. In addition to cell design, the workstations will be able to transmit sketches and designs along with conventional text in an electronic mail system, eliminating many meetings otherwise necessary in developing a circuit design. This will substantially boost the productivity of the VLSI applications researchers, since it will no longer be necessary to wait for a meeting with all key people present to proceed with a new design.

The new graphics capability will also have a substantial impact on the various simulation techniques. Currently, most gate-level and circuit-level logic simulators have character or text oriented input and output, requiring the translation of a graphic logic diagram to a text input and an inverse translation of the results back to the diagram for all but the most simple circuits. This prevents the designer from achieving a good visualization of what is happening in the circuit, increasing the probability of undiscovered errors. The use of interactive graphics as an input/output medium eliminates this semantic gap.

Work will also continue on high level mapping techniques, eliminating many errors which occur during the manual translation from one level of design to a lower one. A number of LISP based VLSI development tools have been specified and will be refined, including an SLA editor [3] and a layout description language [7]. Graphical techniques will substantially benefit the work being done on direct mapping techniques for asynchronous and clocked control units and the specification and implementation of systems using the self timed bit slice logic being developed. Work will also be done on using these mapping techniques to develop efficient diagnostics for the resulting circuits.

Support Projects

These projects represent ongoing activities in the Department of Computer Science whose results are vital for the successful development of the advanced computer aided design systems being proposed. The following discussion covers only those activities necessary for the CAD research, although the current research activities will continue to examine their areas in more generality and depth, greatly benefiting from the systems purchased under this proposal.

Portability and Software Tools. (Senior Investigators: Martin L. Griss, P. A. Subrahmanyam) A set of advanced systems architecture and implementation techniques has been developed by the Utah Symbolic Computation Group, to support the REDUCE Symbolic Algebra system [13], and the Standard Lisp in which it was implemented. The various research vehicles for portability research have recently been integrated into a software development system of substantial power, called the Portable Standard Lisp (PSL) system [2]. This proposal will support expanded cooperation between the software engineering and portability projects in the Utah Symbolic Computation Group, and the CAD system implementors in the ALPHA-1 and VLSI projects, resulting in a family of flexible and portable CAD systems based on the PSL system and software tools.

Other existing software tools, part of the PSL environment, will be used in the development of the CAD systems, and new tools will be developed as necessary. Of high interest is the development of a new surface language for PSL, much like the current RLISP, but tailored toward datatypes and operations common to CAD systems, allowing improved type analysis for the complex data structures common in graphical and CAD systems. This language will be based on an earlier MODE Analyzing REDUCE, which is LISP-based, but with an Ada-like flavor [9, 5]. A specialized surface language yields a cleaner implementation, improves portability (since algorithms are expressed close to their preferred form, rather than cast to match the requirements of an arbitrary language), and boosts programmer productivity.

Workstation Development. (Senior Investigators: Martin L. Griss, Alan L. Davis) As was discussed previously, the initial workstations will be Apollo DOMAIN systems, Motorola 68000-based systems already in use in the department. However, while we feel that these systems are the best for our applications currently available, they are more expensive and lack many capabilities (such as color or higher resolution) that will be necessary for at least some of the workstations.

Since the development of a totally new workstation, in addition to the research currently underway or proposed for this grant, is beyond the resources of our department, we intend to limit any hardware development to those items absolutely necessary to achieve our goals and not available at reasonable prices from other sources. If complete workstations at the desired cost and with the desired features are not available, subsystems (displays, disks, network interfaces, displays and raster generators, and processors and memories) will be purchased from various sources and integrated by departmental personnel.

Most of the workstation development task will involve writing the necessary support subroutines for common workstation activities, such as graphics, network communications, file transfer, and database and filing system management. These will be written so that all calls from applications programs will be machine and output device independent, allowing both the easy porting of programs from one machine to another and the utilization of new capabilities of a system (color, better resolution, better disk memory) without changes to the applications programs.

The human interface to the workstation will also be explored, including the effects of various input devices, higher resolutions, color, and operator command structure. Care will be taken to assure that the comparisons between different features or attributes yield valid results, rather than the anecdotal reports so common in user interface studies. This requires the use of properly formulated and carefully conducted experiments, as well as careful interpretation of the results. Assistance in designing these experiments and interpreting their results is available from Timothy Maher, a Research Assistant Professor in our department, who recently completed his doctorate in psychology and has a strong interest in the ergonomics of user interfaces, with additional assistance from the Department of Psychology.

Database Systems. (Senior Investigator: Lee A. Hollaar) Design of anything but the most trivial items is a process which requires a number of people working in harmony. Any computer aided design system must assure that changes made by one designer are reflected in all areas with which it interfaces. For example, the redesign of a VLSI memory cell to take advantage of a new implementation technique which results in a smaller cell size, increased speed, or reduced power, can profoundly affect the design of a number of VLSI circuits. Critical changes should be readily identified to their designers, allowing them to take advantage of the new cell, or continue with the previous version if the advantages do not outweigh the costs of altering their designs.

Most current CAD systems do not provide this automatic notification of changes, but rely on external means (word of mouth, memoranda) to notify designers of changes which may affect their work. For any but the most closely knit groups this communication is unreliable, and it is potentially expensive, for a design may be completed based on incorrect information. However, commercially available database management systems are not capable of maintaining the necessary relationships between design components, especially when design activities are distributed across a number of workstations. We propose to apply state of the art database system design methodology to develop a system particularly suited for managing information in a computer aided design system.

The database system will support appropriate data structures, at the several levels of data management within the CAD systems. It appears at present that a hierarchically organized database best reflects the high-level relationships between layers of components in a design, since it reflects the stepwise refinement structure present in good designs. It also appears that the low-level building blocks within a particular component of the overall design are connected nonhierarchically, as in the network of geometrical constraints which is used in a constructive geometry specification, or the network of connections between cells of a VLSI design. In either case, designers must specify how their components

interface with other components, and the database management system must assure that these interfaces remain valid during any modification to the database.

Multiprocessors and Networks. (Senior Investigators: Robert M. Keller, Gary E. Lindstrom, Alan L. Davis) One of the major advances in computer technology has been the development of low cost microprocessor systems with capabilities exceeding most major computer systems of a decade ago. The challenge of this advance is in finding ways to exploit the technology to produce large, cost-effective systems. This presents no difficulty if the application is small enough to fit on an existing microprocessor system, but creates a new set of problems for large systems, such as the CAD systems being proposed, with high computational requirements.

Many of the algorithms and tasks in CAD systems are amenable to distribution across a number of interconnected processors. For example, some processors may be assigned the task of database management, some the manipulation of a particular model (such as performing a cut through a 3D shape), some the rendering or transforming of mathematical models into graphics, and some the task of interacting with the designer. This differs significantly from a general multiprocessor, in that its organization can be optimized for a specific, known application.

The departmental local network will be used as the backbone for a multiprocessor test facility, with the proposed workstations operated during time when they would otherwise be unused, and existing machines forming the various processing nodes. This will allow the easy reconfiguration of the test facility to determine the effect of different interconnection, scheduling, and communications techniques. Prototypes of promising configurations might be constructed to provide a more extensive testbed and to verify the simulated behavior of the test system.

Specialized VLSI Processors. (Senior Investigators: Kent F. Smith, Alan L. Davis, Lee A. Hollaar, Elliott I. Organick) This work takes advantage of the development of computer aided design systems for VLSI circuits to develop specialized processors to enhance the efficiency of the algorithms being developed. It may include specialized data transformation units, geometric algorithm "engines", database machines, and communications modules. VLSI research occupies a unique position in this proposal, since it is one of the major thrusts in the development of CAD systems, acts as a support activity, and will be one of the major users of the resulting systems.

User Projects

There are a number of other research projects which are expected to have a relation to this large scale CAD effort. In the Department of Mechanical Engineering, Professor Stephen Jacobson has been working on prosthetic devices, and in that connection has developed some very effective manipulators which have application in robotic devices. An interplay between the CAD expertise and the robotic device research could produce helpful results to all participants.

Professor Edward Smith of the Department of Architecture has been active in the area of computer aided architecture for many years. Although the particular techniques user in computer aided architecture are somewhat different from those used in 3D shape design for manufactured products, there are many common principles as well. It is hoped that a useful tie will be established between the research efforts in computer aided architecture and this project.

A critical component of a CAD system is analysis. In order to test the hypothesis that a proposed design is valid and meets specifications, it is highly desirable to test the computer model rather than to perform tests on physical models or to perform destructive tests which are expensive, time consuming, and space consuming. They may also be dangerous and not nearly as comprehensive in the degree of analysis. We hope that through the cooperation of Professor Robert Jacobs of the Department of Mechanical Engineering, who has specialized in the research area of finite element analysis, we will be able to integrate analytical procedures with the design procedures in an interactive way. Getting a suitable finite element decomposition is something of an art which requires human refinement of

automated first approximations, so the final process can be thought of as computer aided analysis. We also look forward to drawing on the expertise of Professor Henry Christiansen of the Brigham Young University Department of Civil Engineering. A preliminary discussion indicated that he would like to contribute to this project and advise in the area of computer graphics and finite element analysis, in which he has become very well known for his spectacular color displays of the results of the analysis programs.

The activities discussed above are representative of the interaction we expect with other projects from within and outside the University of Utah which will benefit from, and be beneficial to, the proposed CAD system development. Professor Jacobs, who is Associate Dean for Research in the College of Engineering, has agreed to act as an interface between the project management and other investigators who are potential users of the systems being developed.

Detailed Description of Proposed Research

As mentioned previously, the research projects being proposed can be divided into three classes: the CAD work in 3D shape and VLSI design, research into the computer science areas necessary to support the CAD work (such as portability, multiprocessors, and databases), and activities which will use the CAD systems. In this section, more detailed descriptions of the research to be performed in the first two classes are presented.

Computer Aided 3D Shape Design

The current research into 3D shape CAD has been described previously. The ALPHA-1 development project includes support efforts in both high-quality interactive and shaded raster computer graphics, as well as the development of new curve and surface mathematics, geometric computing algorithms, shape or part representation data structures, and system architectures for implementation of CAD systems. The ALPHA-1 system will serve as a testbed for these ideas, and as an environment capable of supporting a wide range of CAD shape representations, styles of interactive design, and design activities. One result of the ALPHA-1 project will be a family of advanced computer-aided design systems tailored for various types of two- and three-dimensional shape design activity.

Because ALPHA-1 is a new and unique approach to developing a CAD/CAM system based on discrete splines as a modelling method to support representations, renderings, and operations between objects, it presents a plethora of challenging and worthwhile research issues. For example, the problem of developing numerical control (NC) toolpath descriptions can now be attacked in a new and promising way. There has been substantial research into the problem of developing suitable and efficient algorithms for implementing the boolean operations of intersection, union, and difference, both here and elsewhere. Both the theory and algorithms will be significantly extended to bring boolean operations between volumes with curved surfaces into the interactive design environment.

We will extend this new modelling approach and couple it with subdivision techniques to develop (semi-) automatic finite element mesh generation and other analysis interfaces. Subdivision, as supported by the Oslo Algorithm, is also a valuable top-down design aid; a refinement of the knot vector can provide more vertices in areas where greater detail is required. Considerable research and experimentation are needed to incorporate this capability, along with other types of design operations, into a good interactive graphical user interface.

While many of the problems surrounding the design of a flexible, intuitive, and viable user interface are not solved through mathematical analysis, the solutions are tremendously important to the ultimate adoption and success of CAD tools and systems. We will apply locally available expertise in psychology and ergonomics in pursuing these research directions. It is well appreciated at this point in the history of CAD that both analog and textual communications channels are necessary to satisfactorily allow a user to develop and express a complex design. It is important to develop a design command language which integrates the handling of these disparate inputs in a natural and flexible way. There are distinct views within the ALPHA-1 research project on what such a command language might look like, and many approaches will be explored. A set of common tools will be developed which will support a range of different interaction and design styles, as well as supporting customized design environments.

The Workstation Bottleneck. Currently, the work on ALPHA-1 is being supported by the departmental graphics laboratory. While this equipment is certainly capable of performing the operations required for the task, the fact that only a single user can be working with it at a time presents a severe bottleneck. The proposed workstations, while not offering all the capabilities and power of the displays in the graphics laboratory, nevertheless will be sufficient for the development and testing of new algorithms and techniques. Since all programs will be written as machine and output device independent programs, it will be possible to use the more plentiful workstations for development and then transfer the work to higher quality devices. This will substantially eliminate the current bottleneck and dramatically improve

the productivity of the project personnel. It will also contribute to our collective experience gained by actual modelling projects using the ALPHA-1 system.

Coordinated Research Opportunities. The interdisciplinary nature of the ALPHA-1 project leads to opportunities for fruitful cooperation with other projects within the department. Symbolic computation and computational geometry are good examples of previous research collaboration with CAD. Some of the new mathematical representations have become much too complex to be dealt with by traditional hand methods. For this reason and others, it was necessary to invoke the techniques of symbolic computation to advance the mathematical research fronts in CAD. We are currently pursuing different kinds of arithmetic schemes which are more appropriate for geometric calculations. The "classification problem" in computer geometry has evoked a major new algorithm from a faculty member not previously associated with the subject. It is expected that this kind of exchange and challenge will be more fully sustained within the context of this proposed research program.

Unfortunately, many potentially productive opportunities have not been explored or developed due to lack of personnel or facilities. We propose to explore those areas, exploiting the synergy between disciplines and combining sources of well-developed techniques with opportunities to apply, test, and further develop the tools and knowledge base. These will be briefly discussed in this section, and will be covered in more detail individually in the following sections.

The Systems Problem. Development of CAD systems is above all a systems problem. It is partly because the current approaches to building CAD systems are not making adequate use of the great advances in many areas of computer science that there is a substantial and critical lag in bringing new algorithmic concepts into the industrial setting. The requirements on CAD system structure involve many concepts including distributed processing, portability, database integrity, interactive multi-user support, extensible systems, and customizable user environments. The current work on CAD system development in the ALPHA-1 project is implemented within the Unix operating system and the C programming language, but this is not an entirely ideal setting. It is anticipated that the research in the department involving systems ideas based on Lisp environments, and the language ideas being developed for multiprocessor architectures will provide useful tools for implementing newer versions of CAD systems at Utah. It is uncommon for a CAD group to be in the same department as this kind of research, and we feel that many innovative ideas can be applied to the language and systems aspects of the CAD problem.

ALPHA-1 can also benefit from the implementation of algorithms and subsystems as VLSI circuits. The display and geometric computing algorithms in ALPHA-1 are modular and separable into dedicated hardware engines. Many of the algorithms are structured in ways that make implementation in special-purpose processors with parallel, pipelined, or tree-structured machine architectures quite natural and attractive. This provides an additional, valuable source of test cases for the VLSI architecture, design and fabrication methodologies. Moreover, the resulting increase in efficiency in the implementation of the ALPHA-1 display and computing algorithms will be reflected in increased productivity in any of the family of CAD systems supported by the ALPHA-1 tools, including the VLSI CAD system.

Many of the advances in the ALPHA-1 CAD system will come about because of the interaction between it and other departmental projects funded by this grant. Especially important for ALPHA-1 will be the work on portability and software tools, database systems, and multiprocessors. These are discussed in more detail in their respective subsections below.

Workstation Support and Uses. The database system and local network of processors will link host computers with design workstations to allow the design of large and complex products. The workstations will be the heart of the user interface to the design system. A major aspect of ALPHA-1 is the effective display and interactive editing of 3D shapes and the techniques for implementing such CAD systems. A range of workstation types with different capabilities will be supported. The support of workstations includes effective use of specialized display processors and distribution of the high-bandwidth interactive part of the editing session onto general-purpose minicomputers integrated into the design workstations.

High-quality refreshed line drawing displays and color, shaded raster displays are used in the ALPHA-1 prototype workstation in the Graphics Laboratory, along with an 11-inch tablet, analog knobs, switches, and a keyboard. In contrast, the programming of ALPHA-1 has been done exclusively on 24-by-80 character display terminals. The use of high-resolution bitmap display workstations with graphics capability will be explored both in the CAD workstation, and in the CAD-system development workstation, where it will allow describing geometric and graphical algorithms with "graphical comments" embedded in the programs, rather than having to describe the essentially visual parts of the algorithm entirely in English text, with greater ambiguity inherent.

True 3D Display of Surfaces. It is very difficult to visualize 3D objects when they are displayed on a conventional graphical display. Most such displays use techniques such as intensity, perspective, shading, and object motion as cues to 3D depth information. Some specialized displays also produce stereographic pairs to give an illusion of depth. None of these techniques is completely satisfactory, and all of them fail for some people. The recent development of a display device producing a truly 3D image promises to change this. Since this display is so new, there has been almost no work on algorithms to support the 3D display capability. We propose to investigate different algorithms for optimal utilization of the 3D viewer. In particular, we will develop algorithms for display and manipulation of objects modelled by the ALPHA-1 system.

Development of NC-milling Algorithms. An important part of a Computer Aided Geometric Design system is the production and verification of numeric control milling programs. We believe that NC-milling paths can be efficiently generated from the surface representation used in the ALPHA-1 system. We propose to use the NC-milling machine to aid in the development of algorithms to do this.

Further, we propose to investigate simulation of the cutting operation by using the results of ongoing research on volume set operations. For example, the operation of subtracting one volume from another models the action of a milling tool. It should thus be possible to represent a computed numeric control milling path as a surface within the ALPHA-1 system and to check this against the designed surface as a verification of the tool path generation.

Integrated Circuit Design

The establishment of a facility of interconnected design workstations opens the way for a number of valuable changes in the present VLSI design process, as common information can be accessed and messages exchanged among coworkers. Currently, designs for all but the simplest circuits are done primarily on the ComputerVision designer system, with the HP 3000 system used for development of class projects. The workstations purchased either under this proposal or as part of other grants or contracts will substantially increase access to VLSI design automation tools, allowing their use from the very start of the design cycle, rather than using them only in the final steps to prepare the necessary masks, as is presently done.

One of the first software packages to be developed for the workstations will be a cell design system. This package will benefit substantially from the previous work done in the department on graphics systems, since it is primarily a two dimensional system, with the ability to overlay and work with many planes, one for each layer in the cell design. A simplified database format will be employed until the final CAD database management system is available. Final integration of a cell design into a complete circuit, and conversion into the form necessary to drive a given pattern generator, will still be performed on the ComputerVision system. The cell design system will relieve most of the workload on the existing design systems, allow more people to become involved with VLSI design, and provide valuable experience in determining the characteristics necessary for a good VLSI design system.

The graphics system used for the cell design system can also be used to provide a means of electronically interchanging messages containing both text and graphics. Since all of the tools necessary to develop logic diagrams, cell designs, and other pictures will run on the workstation, it will be simple to create a picture, store it in a file, and transfer the file to another workstation, much like current mail

systems transfer files of text between users. This can substantially boost the productivity of the VLSI designers, since it will no longer be necessary to schedule a meeting of all involved parties to discuss a design.

Simulation Systems Improvement. Graphics will also provide a new dimension to the simulation systems necessary in the development of a circuit design. Currently, most gate-level and circuit-level simulators have text or character oriented input and output. Since most logic and circuit designs are visualized graphically, they must be manually translated into the appropriate text form before they can be simulated. For all but the most simple circuits, this introduces a number of errors in translation, many of which are found when obviously incorrect outputs are produced, but others may go undiscovered, and cause incorrect results. Even if the translation of the design to text form could be perfect, the inconvenience of performing it manually prevents the designer from using a potentially powerful computer simulation aid unless it is absolutely necessary.

While the non-graphical input requirement may be considered to be an inconvenience, the character oriented output seriously diminishes the utility of the simulator. Since the users must mentally map the output text onto their graphical visualization of the design, the results are inconvenient to interpret. Moreover, subtle differences from the expected results may be difficult to detect. Interactive use of the simulator would require constant mental mappings between the text inputs and outputs, and the graphical visualization of the design.

We propose to develop complete gate, circuit, and device level simulation packages which takes full advantage of the high quality graphics available at each workstation. Use of these packages will allow designers to enter their appropriate diagrams, define input generators, manually define attribute values at various nodes, and see the effects of various delays. The logic simulator will have special logic values, in addition to 1 and 0, to indicate an undefined state, a high impedance state, an open state, and other special conditions. The circuit and device level simulator will allow the user interactively to vary input voltages and currents and alter device characteristics, while viewing the results either at designated nodes on the diagram or in the form of response curves. The graphical inputs to the simulators will be the normal design system inputs, so existing designs can be readily simulated, and any alterations made during the simulations can be automatically reflected in the design database.

Mapping Techniques. Techniques are now being developed for producing an asynchronous control section design directly from its state diagram. A special variation of the one-hot [25] state assignment, where each state has its own flip-flop in the final design, is being used. Extensions to the technique have been made to allow the parallel execution of two or more strings of asynchronous states, using a FORK/JOIN construct, and to provide a subroutine capability allowing common strings of states to be shared in the state diagram. While these constructs can be described in text form, when represented graphically they are considerably more easily and reliably formulated, as well as being easier to comprehend. The design workstation system proposed here is ideally suited for the development of sequencer state diagram and their direct conversion into the VLSI cells which can become part of a larger design.

The work on self-timed bit sliced processors being done by Alan Davis will also be enhanced by the proposed facility, with the production of design tools and simulators to aid in the specification and interconnection of the standardized parts. Designers will be able to treat these parts much as if they were gates or other logical parts, simulating their behavior to arrive at a satisfactory configuration and then having the CAD system produce the necessary wire lists or printed circuit layouts for the final system.

A different approach to hardware design is being pursued under the direction of Robert Keller. Although it also involves a target implementation of communicating speed-independent asynchronous modules, the top-level specification involves a system of functions intercommunicating via streams. This type of representation fits naturally with the desired hardware realization, but is also very natural in terms of human specification of a desired system. Being pursued is the design of the specification language, a typed version of FGL [17, 18], and several significant implementation problems, including recursion-removal, internal buffer introduction, and verification. The specification language can be presented either graphically [15], textually [16], or in a hybrid fashion [19].

A Graphical Representation for VLSI Design. Practical issues in the specification and translation of nMOS circuits are under examination as part of a doctoral dissertation (in progress) on VLSI design [6]. A design language is under development which exploits the medium of computer graphics for the behavioral, structural and physical specification of an nMOS system. Following the work of Davis and Keller on graph languages and parallel computation, this language uses graphs to decompose the overall design into interconnected subsystems and to describe both control and data operations at the local level within a subsystem.

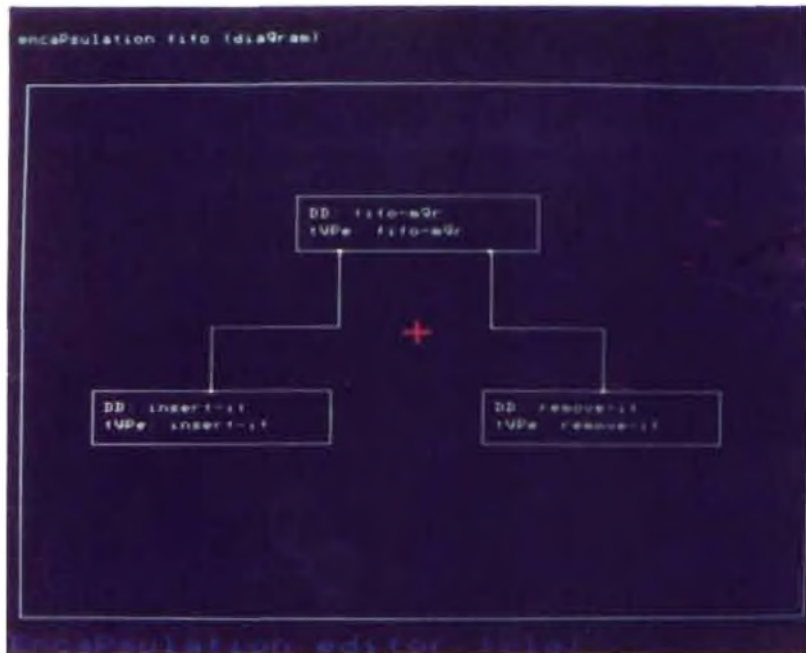
The system hierarchy is captured and decomposed in a set of diagrams called *encapsulations*. Encapsulation diagrams are a more formal version of the black box diagrams that designers typically draw in the early stages of the design process. Subsystems instantiated within an encapsulation may themselves be encapsulations permitting logical substructure to any level. Subsystems that can be translated directly to hardware are called *defined datatypes* or DDs. These units are composed of functions and storage elements and perform logically and operationally as representational datatypes. Defined datatypes are described in diagrams, also. The local structure of a DD is specified in a structural diagram showing the interconnections between the functions and storage elements. Some of these connections extend outside the defined datatype indicating an "up level" connection to the next higher level of the hierarchy. Functions are described by data dependency graphs which specify the data operations to be performed and their temporal order. The nodes in the function graphs are translated into hardware cells and control states through the use of a procedural cell database. The arcs are translated into the wires between cells and into next state information for the design of a local controller. The control state diagram derived in this manner can be translated into a "one-hot" sequencer using the technique of Hollaar [14].

Once a defined datatype has been translated to its geometry, the physical coordinates of the external connections are known. By tracing the logical interconnections in the encapsulation diagrams, a global wiring list can be generated. Unlike other hierarchical VLSI design schemes, global wiring is not performed in a piecemeal fashion as the designer works up or down through the hierarchy. With the global wire list, subsystem interconnections are performed only once at the end of the design process. With careful strategic planning, space utilization should be increased and design time should be reduced.

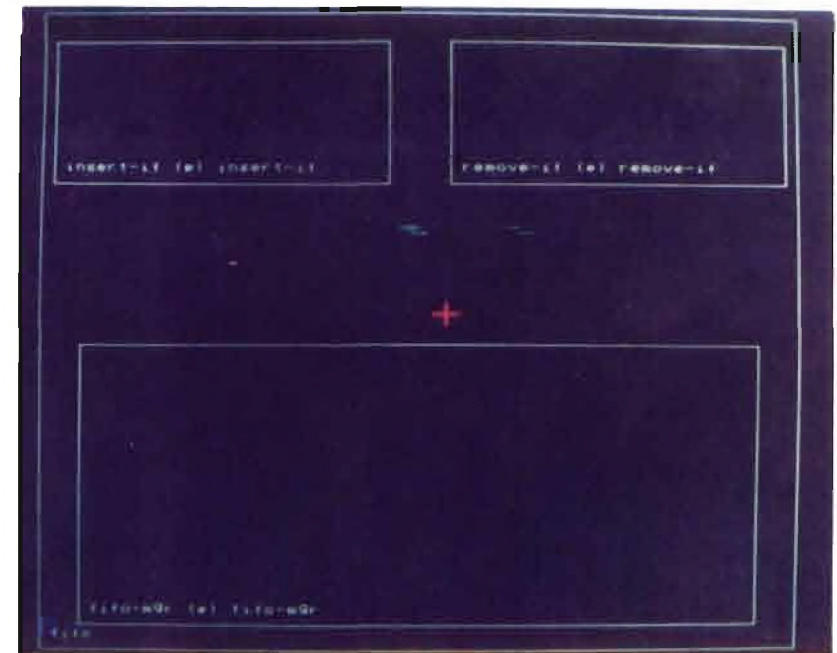
The accompanying photographs show some preliminary results from an experimental design system based on this graphical approach. The pictures in the left-hand column of the first page illustrate the style of structural specification employed in the design system. The photographs show an encapsulation diagram and a defined datatype structural diagram taken from the design of a first-in, first-out (FIFO) memory system. The pictures in the right-hand column are the respective physical representations of the logical elements to their left. On the second page of photographs, the picture in the upper left corner is a function graph - the lowest level of structural and behavioral specification. The other photographs are (clockwise from the upper right) a programmable logic array, a selector circuit and a register stack. These examples demonstrate the degree of component complexity supportable by the procedural cell database.

Much work remains to be accomplished. Tools must still be constructed to generate the local control sequencer and the wire lists. Software aids must be built to support wire routing. Because the language encourages asynchronous design, automatic delay analysis is essential to obtain early feedback about circuit speed. Although the algorithms for delay analysis are known, they must be implemented and their effectiveness tested. Finally, tools to generate electrical and functional simulation files need to be constructed. Although this research has the potential to improve VLSI design, it must be built, applied to some significant design problems, and evaluated. A strong computer graphics facility with several work stations will greatly assist its development.

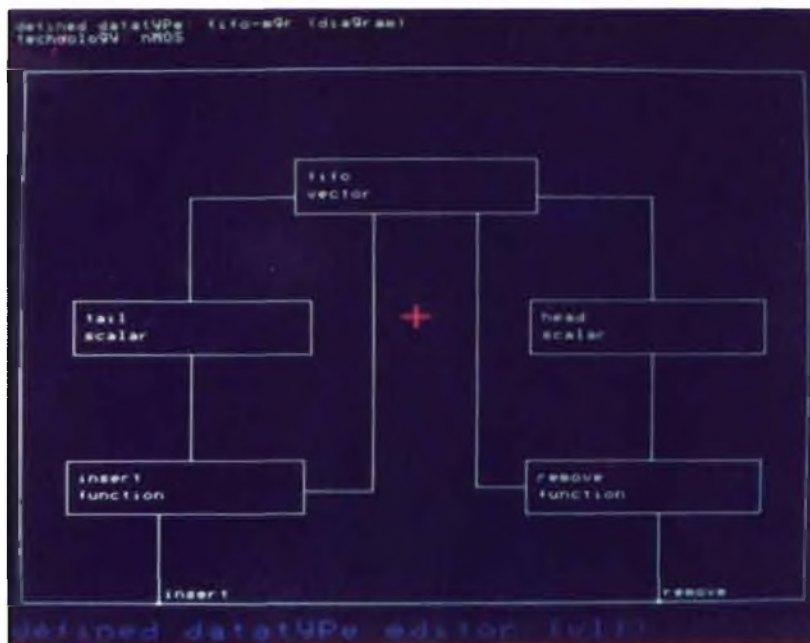
Testing and Diagnosis. The final area of computer design assistance for integrated circuits to be explored under this project is not really design assistance at all, but a vital follow-up to any actual design: the specification of test procedures which assure that the device operates as intended. Since all the design information for a given part has been stored in the database (such as original specifications, logic design, state diagram, electrical design, and interconnections), programs geared toward generating tests



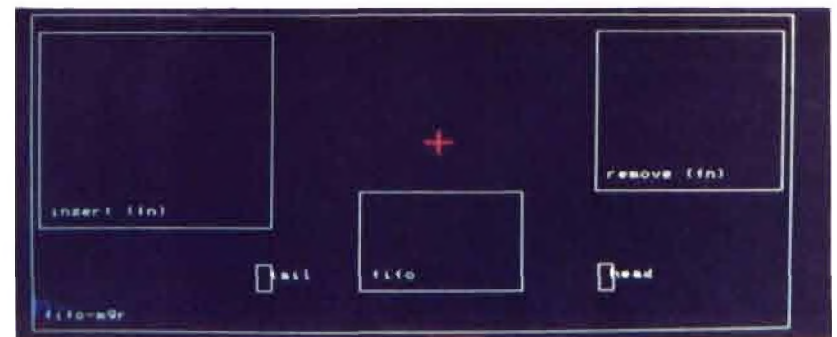
Encapsulation structural diagram.



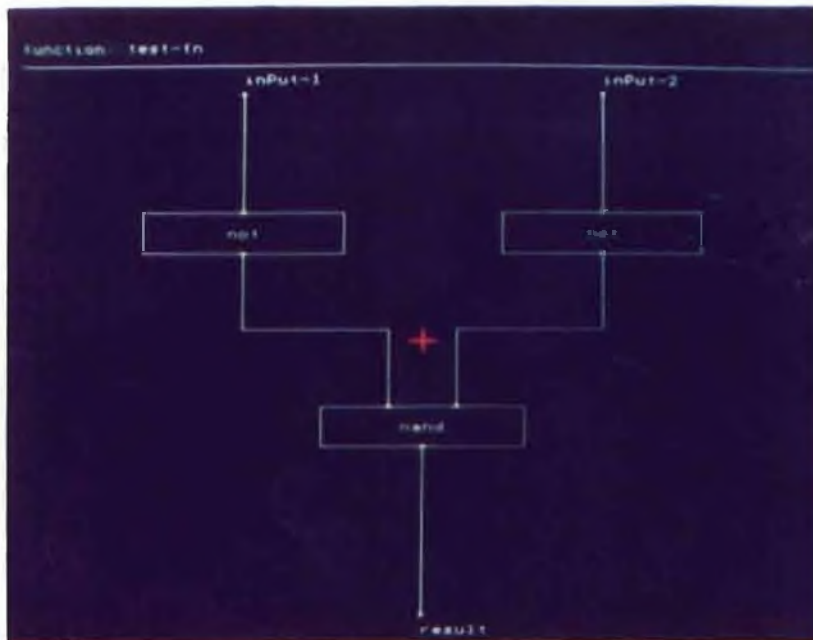
Encapsulation floor plan.



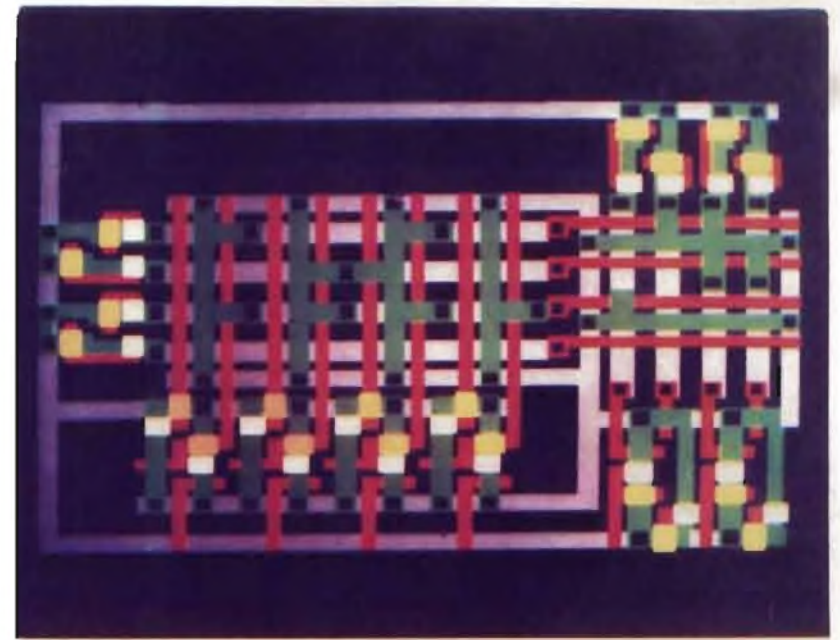
Defined datatype structural diagram.



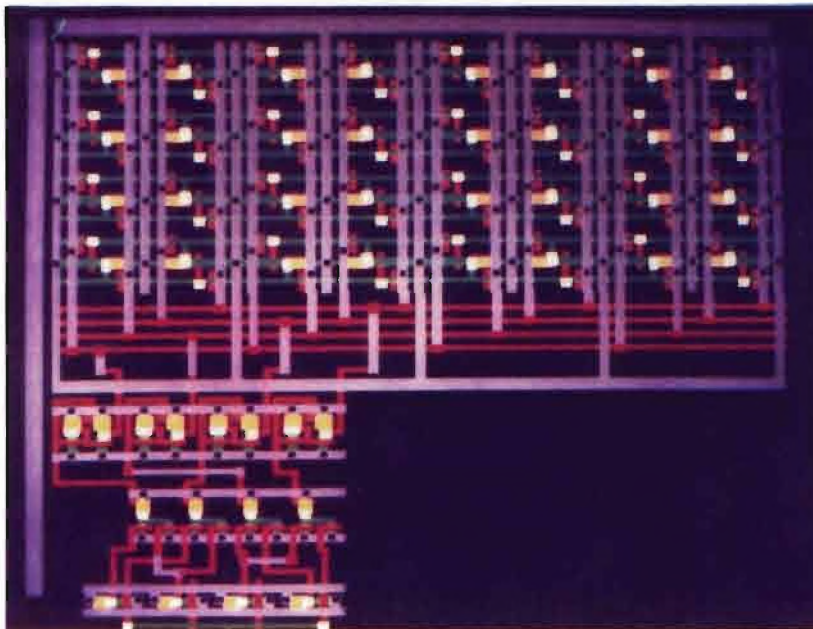
Defined datatype floor plan.



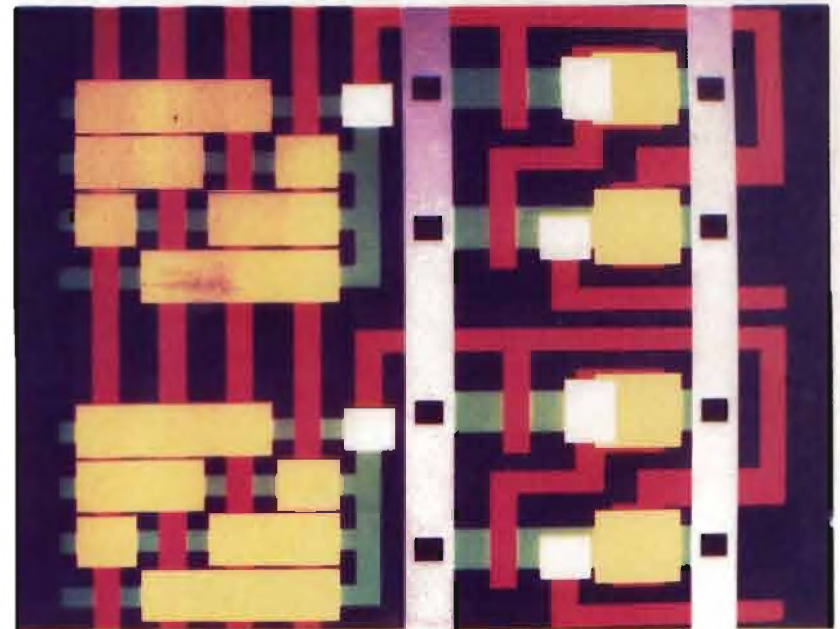
Function data dependency graph.



Programmable logic array.



Variable geometry stack.



Variable geometry one of 4 selector.

which locate specific classes of faults use this information to produce the necessary test sequences. For example, since the state diagram for a sequencer is stored in the database, a minimal path which exercises all possible transitions can be determined. (This will generally be shorter and easier to determine than the minimum path for a nontrivial program.) Information regarding failures can also be compared against information stored in the database, pinpointing the cause of a problem or suggesting other tests which might isolate it.

Packages will be developed to automatically generate test sequences for the circuits developed using the CAD system. These will be used to control the generalized test system described in the equipment section. Circuits will be purposely altered, possibly by micro-scribing to break internal connections or damage components, to demonstrate that the test sequences can determine the cause of the error. The availability of a test facility will also substantially aid in the timely implementation of VLSI circuits for other departmental projects.

The addition of readily-available, high quality graphics, customizable to suit the needs of each designer, will dramatically and positively affect the current work on VLSI circuit design within the department. It will improve productivity, allowing other projects to take advantage of VLSI components and specialized processors, while providing a single focus for the work currently being performed on structured logic, direct mapping procedures, simulation, and diagnostic generation.

Software Tools and Workstation Development

The existing state of the art in software "technology" is, by several accounts, quite abysmal. As far as the productivity and reliability of software systems are concerned, both the human and economic ramifications of that state are enormous. To quote just one statistic (and there are several), the annual cost of software in the US is over \$20 billion [26]. It is our belief that a large portion of this cost (most of which goes towards *maintaining* bad software) can, and invariably must, be improved by computer aided software development systems. We note that the systems that we have in mind are far more sophisticated than the software "aids" in existence.

In order to obtain a quantum jump in the quality and reliability of software, it is imperative to have a coherent theory of program synthesis which can serve as the basis for a sophisticated, interactive program development tool. We have proposed a paradigm for program synthesis that views the problem of program synthesis as that of obtaining implementations of abstract specifications [24]. The theoretical framework is algebraic, and the synthesis relies on the semantics of the functional specifications, performance requirements, and application patterns of the user invokable functions; the programs generated by the synthesis algorithms are provably correct, and the underlying formalism incorporates enough flexibility to allow for various efficiency criteria.

An experimental, extensible, interactive program synthesis system for exploring the theoretical basis has been designed and will be further developed. This system will be designed to allow for the documentation of the development process, and to aid the incremental modifiability of both specifications and implementations. A good "automatic" documentation of the design development process is imperative to alleviate the software maintenance problem. Aspects concerning the specification, synthesis, and verification of software systems will be explored. The ultimate goals are to obtain greater insight into the nature of the program synthesis process, and to incorporate such insight in a computer aided software development system of some sophistication.

The synthesis paradigm, as well as the underlying development system are also of relevance in the context of special purpose VLSI systems synthesis. It has also led to the discovery of some new algorithms: one of these, a clipping algorithm for n-dimensional objects, has been incorporated as the core clipping algorithm in the ALPHA-1 CAGD system.

The Portable Standard Lisp System. Effective portability techniques have been developed, allowing Standard Lisp and thus REDUCE to run in a great variety of computer hardware and software environments. The various research vehicles for portability research have recently been integrated into a software development system of unprecedented power, called the Portable Standard Lisp (PSL) system. This proposal will support expanded cooperation between the software engineering and portability projects in the Utah Symbolic Computation Group, and the CAD system implementors in the ALPHA-1 and VLSI projects, resulting in a family of flexible and portable CAD systems based on the PSL system and software tools.

The PSL system supports parsers for extensible, high-level user or surface languages overlaying the Standard Lisp language. Also supported by the PSL system are flexible, resident compilers which translate the Lisp code into machine or assembly languages [11], medium level languages such as C, Pascal or FORTRAN, or p-codes, to be interpreted by interpreters running on conventional machines or special-purpose micro-coded machines [10]. One of the surface languages is called SysLisp [2], and is used for implementing the data structures and operations which make up the kernel of the PSL system. Hence the term "portable", since the PSL system, and any software based on it, can be moved to another environment by building a SysLisp code generator for the new environment and cross-compiling the system. CAD systems based on the PSL system will benefit from the same portability which REDUCE enjoys now.

Tailored CAD Languages. The key to flexible and powerful CAD systems is that the PSL system can be used to support ideal "tailored" languages for CAD system implementation. Extensions to the standard PSL surface language, known as RLisp, can be made to support the ALPHA-1 and VLSI software development. Strategies of representing, manipulating, and storing objects in a CAD system database have been explored in the first phase of ALPHA-1, using dynamically-allocated, linked data structures of structured data types, as provided by the C environment. Addition of a sophisticated mode (type) analysis ability to the PSL parsers is underway, base upon an earlier Mode analyzing REDUCE [9] (now seen to have a very ADA-like flavor). This will allow support for structured datatypes and operator extensions much improved over that in the C language.

Functions such as the management of database I/O and extended structure reference semantics for maintaining derivative data structures in display processors will be factored into the language operations and datatype definitions. CAD-system specific operations and datatypes, such as ALPHA-1 geometric computing or VLSI state diagram nodes can be embedded in the surface language and invoked as primitive operations and datatype manipulations added to the PSL system. The advantage of such an approach is that the CAD system algorithms are cleanly implemented in terms of their natural datatypes and operations, and the environment-dependent details of the implementation are factored into the PSL system parsers, compilers, and support code.

The use of the PSL system as a software development environment also will provide benefits by making the other high-level systems supported by the PSL system available for integration into the CAD system. Very high-level "specification" languages for designing interactive menu-driven graphical editors in terms of menu and dialog specifications have already been explored in the Unix environment during the first phase of ALPHA-1 development. A meta-language named Yacc (Yet Another Compiler-Compiler) was used, which produces language parsers from language grammar specifications. Similar tools are available in the PSL system, in the META and MINI languages, and pattern matching transformation systems. The REDUCE symbolic algebra system is supported by the PSL system, and will be a valuable addition to the ALPHA-1 development environment, for supporting the mathematical curve and surface formulation research and for transforming the resulting formulae into algorithms for geometric computation and display.

Similarly, the VLSI CAD project will benefit from using the PSL system tailored language and software tools approach. The specialized VLSI tools, such as the simulation and translator subsystems, would be cleanly implemented, portable and extensible, well integrated with the graphics and database support, and use high-level specification languages where appropriate for user interfaces.

The PSL system will be used as the base for the workstation software to be developed. Currently, it is being implemented for the Apollo DOMAIN system being proposed as the initial work station for this project, and will be fully operational by the start of the project. Some experimental LISP based graphics languages and VLSI tools (such as the hierarchical graphics language, PictureBALM [8] and an Emacs-like SLA editor, SLATE [3]) are being converted to PSL, for use on an Apollo DOMAIN system.

Workstation Support Software. Device independent graphics routines for the workstations and the existing graphics devices will be written using the PSL system, providing a uniform calling sequence for any output device on any machine. This will simplify development of a routine on one system, probably one of the highly available workstations, and transportation of it to another system if higher resolution, color, or some other special capability is necessary.

Work will also continue on enhancements to the workstation, both to increase its utility and decrease its cost. The Apollo DOMAIN system, based on the Motorola 68000 microprocessor, will be the initial workstation purchased under this proposal. A number of these systems have already been purchased by the department. However, their current cost is somewhat higher than is desired, and the 1K x 1K monochromatic display may be restrictive for some projects. New workstations will be examined as they become available to determine if they can provide an increase in capability at a cost less than that of transferring the existing software to them.

It is possible that, at least for some applications, no commercially available workstation will provide the capabilities necessary. In that case, a custom workstation may have to be developed using commonly available subassemblies (disks, displays, processor and memory cards) and a limited amount of logic designed especially for the workstation. As more hardware subsystems, particularly for the 68000 and its successors, become available, this may prove to be a reasonable, cost effective approach. For example, Stanford University is about to distribute a 68000-compatible bitmapped display driver printed circuit card, developed for their SUN terminal, at a very attractive price. This is multi-bus based, and could be integrated with other commercially available cards to produce a variety of systems. The use of device independent programming minimizes the difficulties normally encountered when moving existing software from one system to a new one.

Workstation Features and the Human Interface. We are proposing the development of a number of classes of workstations, with the features (such as high resolution, color, input devices) selected to provide the most cost effective solution to a particular designer's needs. Just as it is important that high resolution color displays be available when they are necessary or desirable, it is equally important that the features which exist on a workstation devoted to a particular facet of design be only those which increase the user's efficiency. The commands available to the users must conveniently allow all reasonable operations and be designed to minimize confusion regarding their use.

Unfortunately, the human interface to most computer systems is not developed using experimental evidence of the cost and benefits of its various aspects, but from the guesses of its developers (who often view and use the system in ways different from its eventual users), or brief, unscientific trials made by visitors to the system or friends of its developers. Little effort is made to assure that the "subjects" are sufficiently along the learning curve to accurately judge the value of a given command or feature. These anecdotal experiences generally indicate that features like higher resolution or color are necessary, when actually their benefits may be minimal for a particular application.

We propose to use carefully designed and controlled human factors experiments to determine the effect of various workstation features and command structures. For example, multiple systems will be produced, identical in all but one key feature, such as display resolution. The efficiency of performing a variety of operations will be measured, and each user will be asked to complete a standardized questionnaire rating the various features of the system. After assuring that the results are statistically valid, a determination can be made regarding the effectiveness of each feature or command. Of course, not every command variant or subcommand will be tested in this fashion, but key command structures or type and major workstation attributes will be thoroughly examined.

Database Systems

It is important to the implementation of a unified computer aided design system that the underlying database management system efficiently support the common operations for CAD. The current CAGD and VLSI systems have only primitive database systems: the hierarchically structured filing systems provided by their respective operating systems. While these are sufficient for simply storing the necessary operations, they are inadequate for interactive CAD systems such as those being proposed here.

We propose to construct a database management system tailored to the CAD environment to be constructed under this grant. It will be capable of storing the designs of VLSI cells or mechanical parts, and information giving their interrelationships to other cells or parts. This will allow other parts of a design to be examined, and altered if necessary, when one part has been changed. Most systems do not provide such a capability, leading to problems when one part's design is altered.

It is generally understood that CAD must be interactive and realtime. These requirements can reasonably be met within the confines of a small experimental database system, but real world design applications often result in very large databases. This means that database solutions must not only be capable of storing the necessary data items, but also be capable of supporting realtime interaction.

The work proposed is not general database research, but simply the design and implementation of a system suitable for our particular needs. We intend to evaluate existing database management systems, and, if one can be used with only minor modifications, use that one as our base. If no suitable system can be obtained, a simple system supplying the primitive operations necessary for our work will be constructed by project personnel.

The external view of the database is obviously hierarchical, with the top levels being complete systems, going down through subsystems to individual cells or parts. The various cells, parts, or subsystems can be included in more than one complete system. It is important when a change is made to a part that the old design be preserved until all designers of subsystems which use that part have a chance to evaluate the change in light of their design. The system should provide proper notification to the responsible designers of any pertinent changes. This represents a fundamental difference from most hierarchical database management systems, where a change in any node's data, such as a transaction increasing a salary, is automatically reflected in any later reference to that node.

While the higher level nodes of the database are organized hierarchically, lower level database entities may be connected nonhierarchically, such as information regarding the geometric constraints used in VLSI cell design or a constructive geometry specification. The designers must be able to specify how their components interface with other components, and the database management system must assure that these interfaces remain valid during any alterations to a cell or part design. Access controls must be provided to assure that the database remains consistent at all times, and that simultaneous updates cannot occur or are correctly merged.

Although the use of physical pointers in a hierarchical database system can improve the access time necessary to reach any item, it substantially complicates the task of reorganizing the database, particularly "on the fly" when other tasks are accessing the database. The use of logical pointers can minimize this problem. In fact, the logical pointer can really be the name of the relationship between the various items, allowing the use of a relational database to support the view of a hierarchical organization. This use of a relational model also eliminates many of the problems associated with having one model to represent the various subassemblies, and another to represent connection or geometric rules.

An interesting complication in the database management system design for the systems we are proposing stems from the use of a number of independent workstations connected to a network. Should the database be stored in a central location (such as one of the departmental mainframe computers or a special file server) or be distributed across the various workstations? How should data be addressed and access control be maintained? A possible implementation might have items copied or transferred to workstations only when designers wish to work on them and returned to a central point when work has been completed, with the database management system automatically mapping any intervening accesses

to the appropriate workstation. The effects of available network bandwidth on database performance will be simulated and studied.

Multiprocessors and Networks

Two principle means of employing a multiplicity of processing elements, each a combination of a processor and memory, are as a multiprocessor system, or a local network. The distinction between these modes is ordinarily understood to be one of degree of coupling or cooperation. The processing elements (PEs) of a multiprocessor usually cooperate in centrally-generated, but distributively-executed, computational tasks, whereas the PEs of a local network operate mostly in response to tasks generated independently by the "users" of those PEs. In the local network case, the interconnection of PEs typically serves the role of occasional message communication, whereas for multiprocessors, communication is more frequent and embodies the distribution of the computational load itself.

The research proposed here attempts to employ a network for *both* of these functions, at different times. The motivation stems from effective use of the resources which must be dedicated to such a network. More specifically, we desire to investigate ways of making a local network behave in both its traditional role, as well as a multiprocessor, for the following purposes:

- The mechanics of distributing computational load, particularly regarding distributed management of the underlying object naming and addressing, is a topic which deserves further experimental investigation using the local network context.
- Some, but not all, computational tasks conducted in a given computing environment might well exploit coordinated use of multiple processors. To provide for the heaviest such use at all times would be prohibitively expensive.
- A network provides an inexpensive way of experimenting with the performance of a dedicated multiprocessor system under development. The timing information obtained is likely to be scaled up from that of the corresponding dedicated multiprocessor system being modeled, but this information is obtainable without the dedication of resources required for such a system.

There are thus two reasons for shifting of the load among PEs in a multiprocessor network: to increase execution efficiency by more evenly distributing computational tasks, and to change the mode of a PE being used in multiprocessor mode to use as a design computing station, in the event that it is desired for that purpose.

The following is a description of a tentative approach to establishing the feasibility of load shifting of the type described above. The underlying computational model will be "object-oriented". That is, all data and functions in the entire system will be dynamically allocated, and referenced by the system through a system-wide address. This provides means for the system to refer to all objects stored within it.

The protocol for locating an object given this address is implemented by the nodes of the system through an addressing scheme based upon physical modules. The load shifts described above obviously entail the movement of locally-stored objects and attendant computational activity to other PEs or network buffer storage. Hence it is essential that objects not be "pinned" as would be the case if they were directly referenced by a system-wide address. Instead, some form of mapping must be devised which resides, in part, in the PEs themselves. Load shifting will require that one module be capable of deferring to another module the addressing of an object originally assigned the first module. This scheme seems workable for physical modules which are either active (such as processor/memory units) or passive (like disks).

Within this framework, a number of research problems remain to be solved. Specifically, these are the development of programming language techniques which allow the graceful exploitation of a multiplicity of processor nodes to solve one computational problem, of policy and strategy for shifting network load for balancing purposes, and of a means for insuring the integrity of a local user computation during the transient interval following the shift of the multiprocess load from his node to other parts of the

system. Also, the development of an object scavenging (i.e. storage reclamation) mechanism, to prevent resources local to a given node from being swamped by overhead information (such as segment table entries which represent deferred objects), and of mechanisms through which the system may recover from the failure of a node, by reconstructing essential state information from information contained in other nodes.

Many of the algorithms for geometric and VLSI design lend themselves to partitioning into a number of tasks which can be executed concurrently and have only limited intercommunications, making them ideal candidates for execution on a multiprocessor configuration. The multiprocessor testbed developed using the various workstations and the local network will be used to test and evaluate different partitionings, with promising ones implemented on specially configured workstations containing multiple microprocessors.

References

- [1] B. Baxter.
A 3-D Viewing Device for Interpretation of Multiple Section Images.
In *Proceedings of the National Computer Conference*, pages 437-440. 1980.
- [2] E. Benson and M. L. Griss.
SYSLISP: A portable LISP based systems implementation language.
Utah Symbolic Computation Group, Report UCP-81, University of Utah, February, 1981.
- [3] T. Carter, G. Goates, M. L. Griss, and R. Haslam.
SLATE: A Lisp Based EMACS Like Text Editor for SLA Design.
Utah Symbolic Computation Group, Operating Note No. 55, University of Utah, Computer Science Department, January, 1981.
- [4] E. Cohen, T. Lyche, and R. F. Riesenfeld.
Discrete B-splines and Subdivision Techniques in Computer-Aided Geometric Design and Computer Graphics.
Computer Graphics and Image Processing 14(2):87-111, October, 1980.
- [5] *Reference Manual for the Ada Programming Language, Proposed Standard Document*
U. S. Department of Defense, 1980.
- [6] Paul J. Drongowski.
An engineering aid for the design of nMOS VLSI systems.
1981.
Dissertation in progress. Computer Science Department, University of Utah.
- [7] G. B. Goates.
ABLE: A LISP-Based Layout Modeling Language with User-Definable Procedural Models for Storage/Logic Array Design.
Master's thesis, Department of Computer Science, University of Utah, December, 1980.
- [8] G. B. Goates, M. L. Griss, and G. J. Herron.
PICTUREBALM: A LISP-based Graphics Language with Flexible Syntax and Hierarchical Data Structure.
In *Proceedings of SIGGRAPH-80, Computer Graphics*, pages 93-99. ACM, 1980.
- [9] M. L. Griss.
The Definition and Use of Data-Structures in Reduce.
In *Proceedings of SYMSAC 76*, pages 53-59. SYMSAC, August, 1976.
- [10] M. L. Griss, R. R. Kessler, and G. Q. Maguire, Jr.
TLISP - A Portable LISP Implemented in P-code.
In *Proceedings of EUROSAM 79*, pages 490-502. ACM, June, 1979.
- [11] M. L. Griss, and A. C. Hearn.
A Portable LISP Compiler.
Software - Practice and Experience 11:541-605, June, 1981.
- [12] R. L. Haskin and L. A. Hollaar.
Operational Characteristics of a Hardware-based Pattern Matcher.
Technical Report, University of Utah Department of Computer Science, July, 1981.
Submitted to ACM Transactions on Database Systems.
- [13] A. C. Hearn.
REDUCE 2 Users Manual.
Utah Symbolic Computation Group UCP-19, University of Utah, 1973.

- [14] L. A. Hollaar.
Direct Implementation of Asynchronous Control Units.
Technical Report, University of Utah Department of Computer Science, August, 1981.
Submitted to IEEE Transactions on Computers.
- [15] R. M. Keller.
Semantics and Applications of Function Graphs.
Technical Report UUCS-80-112, University of Utah, Computer Science Department, 1980.
- [16] R. M. Keller, B. Jayaraman, D. Rose, G. Lindstrom.
FGL (Function Graph Language) Programmers' Guide.
Technical Report AMPS Technical Memorandum No. 1, University of Utah, Computer Science Department, July, 1980.
- [17] R. M. Keller, G. Lindstrom, and S. Patil.
A loosely-coupled applicative multi-processing system.
In *AFIPS*, pages 613-622. AFIPS, June, 1979.
- [18] R. M. Keller, G. Lindstrom, and S. Patil.
Data-flow concepts for hardware design.
In *IEEE Compcon '80*, pages 105-111. February, 1980.
- [19] R. M. Keller and W-C. J. Yen.
A graphical approach to software development using function graphs.
In *Proc. Compcon 81*, pages 156-161. IEEE, 1981.
- [20] D. G. Matty.
The Silicon Express: A Step Towards the Automatic Implementation of Very Large Scale Integration Systems.
Master's thesis, University of Utah, August, 1981.
- [21] S S Patil and T A Welch.
A Programmable Logic Approach for VLSI.
IEEE Transactions on Computers C-28(9):594-601, September, 1979.
- [22] R. F. Riesenfeld, et al.
Using the Oslo Algorithm as a Basis for CAD/CAM Geometric Modelling.
In *Proceedings of the Second Annual NCGA National Conference*, pages 14-18. National Computer Graphics Association, Inc., Baltimore, June, 1981.
- [23] C. L. Seitz.
System Timing.
In C. Mead and L. Conway (editors), *Introduction to VLSI Systems*, pages 218-262. Addison-Wesley, Reading MA, 1980.
- [24] P. A. Subrahmanyam.
A Basis for a Theory of Program Synthesis.
In *Proceedings of the First Annual National Conference On Artificial Intelligence*, pages 74-76. AAAI, August, 1980.
- [25] S H Unger.
Asynchronous Sequential Switching Circuits.
Wiley-Interscience, New York, 1969.
- [26] P. Wegner.
Research Direction in Software Technology.
MIT Press, Cambridge, Mass., 1979.