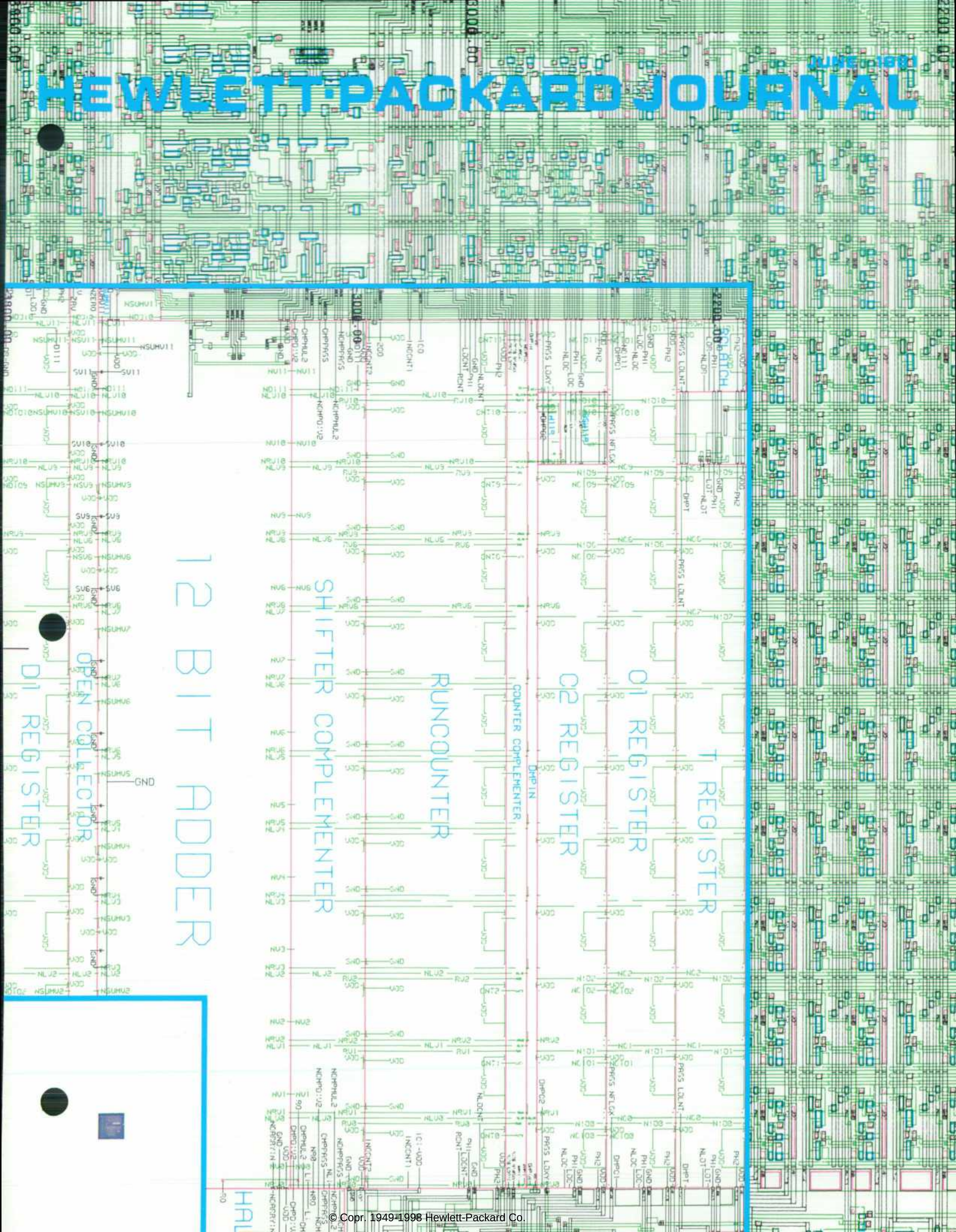
HEWLETT-PACKARD JOURNAL

JUNE 1981



HEWLETT·PACKARD JOURNAL
Technical Information from the Laboratories of Hewlett-Packard Company

JUNE 1981 Volume 32 • Number 6

Contents:

In this Issue:

If the decade of the sixties was the decade of electronics and the seventies the decade of the computer, what will characterize the decade of the eighties? This issue explores a likely candidate: the very large-scale integrated (VLSI) circuit. The size of the VLSI challenge—the design and production of tiny chips containing hundreds of thousands of transistors—is such that it will tax the creativity and ingenuity of everyone involved. To reduce the design time for such complex circuits to a reasonable amount, the complete design process must be addressed. It is not enough simply to create tools for VLSI designers. The creation of sophisticated design tools will help, but a balanced strategy encompassing tools, methodologies and quick-turnaround fabrication must be employed if the design time is to be significantly reduced.

In this issue, you will have an opportunity to review some of the VLSI design concepts that have worked for Hewlett-Packard Company. Design strategies and design tools are continuously evolving at Hewlett-Packard divisions and in Hewlett-Packard Laboratories. The combination of industrial development and university research is quickly moving IC design capability into the hands of hundreds of very creative engineers who have not had access to this additional degree of freedom before. This design strength will impact new product development throughout the world. With this new strength, we may find that in the future we will be limited more by the capital required to build IC facilities than by process technologies or design resources.

(Cover: See page 25**)**

Merrill W. Brooksby, Guest Editor

Editor, Richard P. Dolan • Associate Editor, Kenneth A. Shaw • Art Director, Photographer, Arvid A. Danielson
Illustrator, Nancy S. Vanderbloom • Administrative Services, Typography, Anne S. LoPresti • European Production Manager, Dick Leeksma

Marco Negrete on Structured VLSI Design

WITH THE CAPABILITY of putting 450,000 devices on a chip, enough to implement a 32-bit microcomputer, designers are well into the VLSI era (see box, page 11). If the trend of 2-to-1 increases in density per year established over the last decade continues, we will soon be talking about ultra-LSI. The rewards are great as higher levels of integration are achieved. The most obvious benefit is the dramatic decrease in the cost per function as the cost of a chip is amortized over a larger number of functions. Equally important are the increases in performance and overall reliability that accrue as devices shrink in size. These advances open up entirely new applications, and there's more to come, since by some estimates, densities can increase by a factor of 100 before the fundamental limits of the technology are reached.

Achieving higher levels of integration depends on continual improvements in device fabrication, but if past experience is any indication, advances in device fabrication will continue to outstrip design capabilities. Substantial improvements in design approach are continually being made as new tools are integrated into the design process, but some fundamental breakthroughs will be required to reverse this trend.

Major Problems

Some of the problems encountered are simply results of the tremendous increase in the number of devices. Others are results of the increases in functional complexity made possible by the scale of integration now achievable.

For example, it is clear from the number of devices alone that a designer or even a team of designers can no longer deal with individual devices. Using the traditional "paper doll" approach of laying out individual polygons on mylar, digitizing and then assembling these into an overall layout would take, by some estimates, 60 years to lay out a chip with 100,000 devices.

The nature of the chip design task has changed as well. Now that it is possible to put a complete microprocessor on a chip, the designer must be concerned with a much broader range of problems. Compared to the days when chip design consisted of a combination of device physics and circuit design, the span of design has expanded greatly to include logic design, machine organization, and even system architecture.

Integrated Design Approach

Finding an answer to these problems is the subject of intense interest within the industry and the university community. There is no simple answer, and numerous approaches are being pursued. A common theme is emerging, however, namely structured design. The goal is to structure the design process by splitting the design into successively smaller pieces to take advantage of the simplification that can be achieved, but to split it in such a way that the individual pieces can later be combined in a coherent fashion.

The design process is divided into several phases such as functional description, logic design, circuit design, simulation, layout, design verification, and so on. In each phase, different levels of abstraction are used to highlight those features of the design important during that particular phase and to minimize the amount of information the designer has to deal with. This allows the designer to focus attention on the critical items, thereby greatly simplifying the total process and increasing the reliability of the design. During each phase of the design, increasingly detailed representations of the design are developed, and the design tasks alternate between design and design verification.

In a typical design, for example, a designer starts by defining the functions to be performed and partitioning the design into a set of logical blocks that together are capable of performing the desired functions. A functional simulation is performed to determine if the functional design is capable of producing the desired results. Once satisfied that the functional description is correct, the designer develops a chip plan, which shows how the various pieces go together on silicon. Then comes the logic and circuit design phase, a process of decomposing and refining the design into successively smaller blocks until all blocks are implemented. Logic and circuit simulations are performed to determine whether the blocks perform properly. The designer then proceeds to lay out the circuit in silicon according to a set of design rules for the process to be used to fabricate the devices. Finally, a design rule verification is performed to see whether or not any design rules have been violated.

At each stage of the design, the design consists of a set of symbolic reprepresentations. In general, each of these representations is relatively independent of the others and is keyed to a particular design focus reflecting the set of design choices appropriate to the corresponding phase of design. They are closely tied, however, in that as the design progresses through its various phases, information generated in previous phases provides the base on which succeeding phases of the design are built. The emphasis is on maintaining continuity from one phase to the next, starting with the algorithm describing the basic set of functions to be performed and evolving into the set of masks used to fabricate the device. In many ways the design process is analogous to the top-down structured design approach used in designing complex software systems.

In addition to the simplification inherent in separating the design into separate design phases, simplification is achieved in the process of partitioning the overall system. This starts with deciding what part of the total system will go on the chip in the first place and defining a suitable interface between the chip and the rest of the system. In a similar fashion, the design task may be divided into several smaller subtasks or modules during each phase of the design. By defining a set of standard modules, the number of different modules that have to be designed can be significantly reduced. Furthermore, by combining existing modules to form even larger modules, the entire chip can be built up out of progressively larger blocks, resulting in a highly leveraged and regular chip design.

Separating the design into phases and partitioning the system into modules is not new. What is new is the need to pull these diverse design activities together into an integrated design approach. The chip is now a complex system with subsystems and subsystem interfaces of its own. In addition, the coupling between the subsystems is generally much tighter than the coupling between subsystems that can be physically isolated. Any change in a subsystem can ripple through the entire chip, causing the design task to grow exponentially. Consequently, structuring of the design to maintain independence of the individual modules and increasing the regularity of the design are key considerations.

Circuit Design and Simulation

Today the design process is far from ideal. It is split into two distinct areas: 1) circuit design and simulation and 2) artwork and design rule checking. Means of bridging the gap between these two are relatively primitive. This is largely because tools have evolved starting at opposite ends of the spectrum, with paper dolls at one extreme and circuit design at the other. Tool development has been driven by the need to improve constantly on existing techniques.

In circuit design and simulation, the designer now has tools for circuit level or gate level design in a hierarchical fashion. Circuit elements are added and interconnected interactively on a CRT-based graphics editor. Once defined, these circuits can be included in other circuits along with other primitive circuit elements. As a result, the designer is able to build up an arbitrarily large circuit out of increasingly complex cells, starting, for example, with a set of primitive elements such as transistors, diodes, and resistors to build a gate, a set of gates to build a register, and so on. These circuits can then be used to form the building blocks that have been previously defined in the process of top-down design.

Since building breadboards is no longer feasible at the present scale of integration, the designer must rely on software simulation to verify the design. There is a choice of circuit, logic, or functional simulation, depending on the size of the circuit and the level of detail desired. The tedious and error-prone task of entering the circuit or logic level description has been greatly simplified. The description can now be generated automatically either from the computer-based schematic that has already been developed in the circuit design phase or from the artwork that has been created in the layout phase.

Artwork and Design Rule Checking

Artwork and design rule checking have received more attention than design and simulation. Design techniques have progressed steadily from the use of paper dolls to hierarchical, symbolic design with interactive graphics input and automated design rule verification. A designer now has the capability of building up a total chip out of progressively larger cells, each of which has been checked against a set of design rules for the process to be used in fabricating the device. By using symbolic design, the designer can focus attention on those aspects of the design that are important at that particular stage in the design. Unnecessary details are hidden and do not consume valuable resources.

By establishing cell libraries and providing a means of incorporating predefined cells in different combinations to satisfy custom designs, substantial improvement in design efficiency can be achieved, limited primarily by the ease with which individual cells can be interconnected. By structuring these cells so that they plug together easily, the remaining interconnect problem can be greatly alleviated.

These techniques have been successfully applied in ASAP, a design methodology developed by HP's Colorado Springs Division. In addition to the use of standard cells and symbolic design, design rules have been carefully chosen so that symbolic elements are constrained to fall on a multiple of the design grid, thus creating a secondary or coarse grid. With a given resolution, this enables the designer to view a proportionately greater area in symbolic mode.

Eliminating the artwork step altogether is even better, if the designer is willing to put up with the constraints imposed and a lower level of integration. For example, using gate arrays, all of the circuit elements are already laid out. The designer specifies which of the gates are to be used and how they are to be interconnected schematically, and the interconnection is done automatically. Another example is HP Laboratories' programmable state machine, PSM, in which the interconnect is completed by cutting holes, or vias, between conducting layers. The via pattern is generated automatically from the Boolean logic equations describing the circuit.

Closing the Gap

The problem of determining whether or not the layout meets the original design goals still remains. However, a partial solution exists, since it is now possible to obtain feedback in the design process by extracting an equivalent circuit directly from the artwork and comparing it with the schematic generated by the designer during the circuit design phase. Since this procedure is cumbersome and consumes a great deal of computer time, it is generally used to verify the design on a cell-by-cell basis. To the extent that the individual cells are relatively independent and no adverse affects are encountered as cells are combined with other cells, it is an effective means of determining whether or not the layout accurately represents the desired circuit. Parasitics that are generated as a result of the interconnections between cells may still cause problems, however, and the means of extracting these parasitic elements and including them in a chip level simulation are relatively primitive.

In the long run, providing a strong forward path would greatly simplify the problem, since it would eliminate the need for complex feedback paths altogether. One such approach is being developed by Lou Scheffer at Stanford. It constrains the design in such a way that cells can be defined procedurally and combined without introducing deleterious side effects. The procedural definitions are broad enough to meet the requirements imposed in the process of the top-down design of the chip.

A more highly automated approach is the silicon compiler being developed by Carver Mead at California Institute of Technology in cooperation with industry as part of the silicon structures project. It enables the designer to describe the chip programatically and automatically generates artwork in much the same way as a compiler is able to generate executable code directly from a program written in a high-level language.

Whatever the approach, it is clear that several key ingredients are required before ultra-VLSI can become a reality: high-level languages to describe the chip in its various phases of development, the ability to generate procedural descriptions of cells which can be used as basic building blocks, means of tailoring these cells to fit the particular application, and a means of assembling individual cells into a composite chip with the guarantee that the result will satisfy all the constraints imposed by the top-down design.

Marco R. Negrete

Marco Negrete is engineering manager of HP's Technical Computer Group. With HP since 1956, he became engineering manager of HP's Loveland, Colorado Division in 1961 and served in that capacity for nine years. He then served as general manager of the Loveland Divisions, general manager of the Microwave Divisions, engineering manager of the Instrument Group, and engineering manager of the Computer Systems Group. Marco holds a BS degree in physics from California Institute of Technology and an MSEE degree from Stanford University. A member of IEEE, he has authored a number of IEEE papers on various subjects. He's married and has five children.

VLSI Design Strategies and Tools

by William J. Haydamack and Daniel J. Griffin

DRAMATIC CHANGES have occurred in integrated circuit technology since 1961. In that year Fairchild introduced the first IC, containing four transistors. Today's ICs contain over 100,000 transistors and perform functions that were not available with the most powerful computers of twenty years ago.

Although there are many interesting aspects to the evolution of integrated circuits, the ones discussed here are the design tools and methodologies. Most historical accounts cite a few major milestones during the development period. Milestones were certainly present, but more interesting is the phenomenon of a continually advancing technology.

Gordon Moore of Intel observed that the number of transistors in the most complex devices was doubling every year. This observation later became known as Moore's Law. Knowing the size of future integrated circuits, as predicted by Moore's Law, leads one to speculate on the new design tools and methodologies that will be required. These requirements are best understood by reviewing where IC design has been and where it is today.

Early design of ICs was done by cutting sections of a red material called rubylith, and placing them on sheets of mylar. When completed, the drawing was photographed and the negative used to prepare masks for photolithographic processing. Given the number of devices, the dimensional tolerances, and available drawing aids, this was an acceptable methodology. Unfortunately, as technology advanced, more components were required on the drawings. Either the size of the drawing had to be increased, or the feature sizes had to be reduced. A dilemma arose when features reached a minimum size. Increasing the drawing size meant the designer couldn't reach all parts of the drawing without the possibility of disturbing features already placed. HP's first major design tool was a suspension platform that allowed the designer, lying prone, to move about above the drawing. Additions and corrections could be made without disturbing other figures on the drawing. It was a clever scheme, but its usefulness was short-lived and newer tools were required.

The computer was soon used to assist in the design of ICs. Pattern generators became available that could produce masks from computer data stored on magnetic tape. Although providing many advantages, these tools added certain constraints on the design methodology. They would accept data only in a rectangular form, and the rectangles had restrictions on their minimum and maximum sizes.

HP's first artwork system, CAA (computer-aided artwork), was developed at the Santa Clara Division and released in 1970. It accepted data in a punched card format and was capable of producing data to drive both plotters and pattern generators. It was later enhanced to accept coordinate inputs from a digitizer, display data on a graphics terminal, and permit some interactive editing. This capability was made available in 1972. This system provided some major improvements in the mechanical aspects of the design, but also had its impact on the methodology. The engineer was able to segment the design of the IC into units which were easier to conceptualize and develop, and later to merge these segments, with the help of the computer, into the final circuit. With simple IC designs this approach had only small benefits, but as complexity increased, it became an extremely important capability.

Although the CAA system enabled the development of more complicated IC designs, these designs in themselves created new problems. Earlier designs were easy to inspect visually to determine if they met process requirements and represented the circuit the engineers wanted. Designs produced by the CAA system were much more complex. Visual checking took weeks and even months, and still errors were missed.

A new set of design tools emerged which included mask data verification and powerful simulation programs. A design-rule-checking program (developed initially by NCA Corporation, Santa Clara, California) was used to test processing requirements such as the width and spacing of metal lines. Errors were identified and plotted. Tools were also developed that would literally extract a schematic from the mask data for comparison with the designer's original concept. The SPICE circuit simulator (originally developed by the University of California at Berkeley) was introduced at Hewlett-Packard. This simulator provided the engineer with familiar voltage and current waveforms similar to those that would appear on an oscilloscope. Accuracy of this simulator was enhanced by the TECAP system from Stanford University which provides model parameters based on device measurements.

A Set of Design Tools

At this time a set of design tools (Fig. 1) was in place that allowed the designer to create relatively large circuits.

The set contains three editors for entry of data. ED is a general-purpose text editor. IGS is a graphics editor used for the design of masks. DRAW is also a graphical editor but is used for schematic data entry rather than entering mask data. Both DRAW and IGS are described in other articles in this issue.

Fig. 1 also shows the various simulators. SUPREM is a process-level simulator that lets the user determine levels of impurity doping concentration as a function of processing variables such as the temperature of the diffusion furnaces. HP-SPICE is a very accurate circuit simulator that can determine voltage and current values with respect to time. Transistors are modeled by analytical expressions for current and charge storage. MOTIS-C is a simplified circuit simulator that uses data in a table to represent the transistors. TECAP is used to generate models for MOTIS-C and HP-SPICE. Logic-level simulation is done by TESTAID-IC. Instead of voltage and current levels, its results are in terms of logic levels.

The three electrical simulators HP-SPICE, MOTIS-C, and

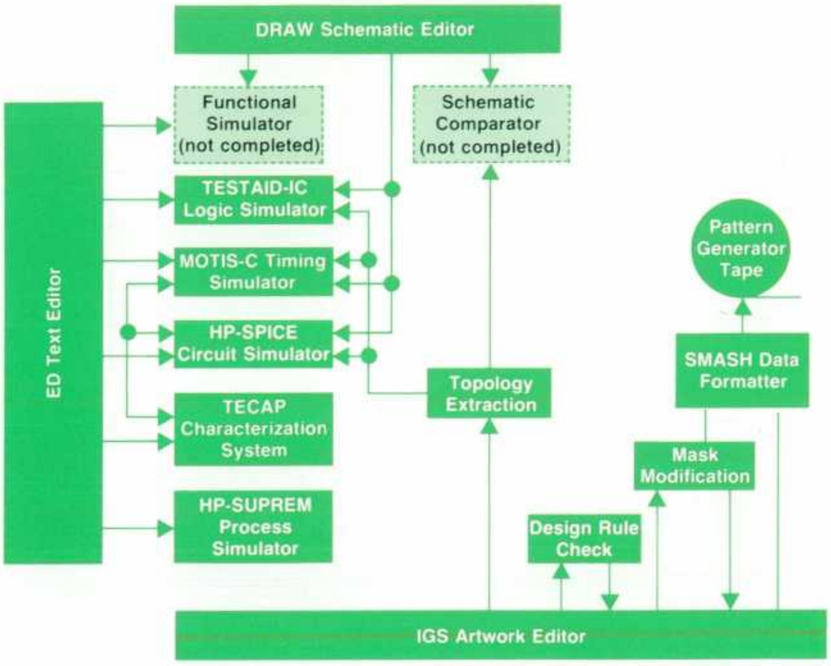

DRAW Schematic Editor
Functional Simulator (not completed)
Schematic Comparator (not completed)
ED Text Editor
TESTAID-IC Logic Simulator
MOTIS-C Timing Simulator
HP-SPICE Circuit Simulator
TECAP Characterization System
HP-SUPREM Process Simulator
Topology Extraction
Pattern Generator Tape
SMASH Data Formatter
Mask Modification
Design Rule Check
IGS Artwork Editor

Fig. 1. *Hewlett-Packard design aids for integrated circuit design.*

TESTAID-IC provide a range of capabilities differing in accuracy of results, size of circuits that can be simulated, and computational requirements. A 10,000-gate circuit can be evaluated in a few minutes with TESTAID-IC on a 16-bit minicomputer, whereas a similar analysis with MOTIS-C requires hours on the Amdahl 470-V8 computer system.

Various methods of artwork verification are also shown in Fig. 1. The design-rule-checking program (DRC) is used to determine the conformance of a design to the IC process' geometrical requirements. These may include widths, spacings, enclosure distances and areas. Another verification tool is EXTRACT which creates an electrical schematic from the mask artwork. The schematic can be input to any of the simulators. Although extracted at a transistor level, a program exists for conversion of this data to the higher level required by the logic simulator. This method of determining whether the artwork meets electrical requirements by simulation has been used at HP. However, a more direct method of verification is desirable. This would involve comparing the extracted schematic with the designer's original intent, as input by DRAW. At present this program, COMPARE, is not complete.

Two other programs are listed in Fig. 1. The SMASH program is used to convert artwork data into a form suitable for use by the photolithography equipment used to create the processing masks. Mask Modification is a program used to make minor geometrical modifications to the artwork data before final processing.

These design aids are currently used at Hewlett-Packard. They provide a solution to many of today's IC design problems and support many different design methodologies. Unfortunately none of the design methodologies is guaranteed to produce a good circuit.

Problems of Current Tools

The need for a methodology that will guarantee a good circuit is paramount. If an IC is fabricated and doesn't work, little information is available other than that it doesn't work. In some cases simulation models have been modified to try to represent the behavior of the failing circuit, but success with this method requires very clever people with very clever ideas. In general, the problem of dealing with an error becomes more difficult the later the error is discovered.

The problems of tools and computers are perhaps best exemplified by the design-rule-checking program. The algorithms used now dictate that the computer time required will increase as the amount of data raised to the 3/2 power.

At present Hewlett-Packard is using one of the largest computers available (Amdahl 470-V8) and it may take up to two days to complete a design-rule test for a large IC. Unfortunately, this problem will only get worse as complexity increases.

There is no single or simple answer to these problems. Although improvements may be made in existing design aids, new tools and methodologies will be needed. Tools and methodologies will become much more closely coupled than they are today. Today's tools are often applied over a wide range of methodologies. Some of tomorrow's tools will be more specifically oriented toward a particular methodology.

Design Methodologies

Hewlett-Packard is currently using six different design methodologies, as illustrated in Fig. 2. These methodologies may be distinguished by the process technology used, the size of circuits being designed, and the tradeoffs between minimizing design time and minimizing the area of the circuit. At the low end of the spectrum (small bipolar circuits) a methodology is being implemented that will be used for programmable state machines (PSMs) of up to 500 terms. In such designs the active devices and interconnecting lines are predetermined. The designer, using either a graphics editor or some higher-level description such as a state-transition table, describes the connections required between various conducting layers to complete the circuit.

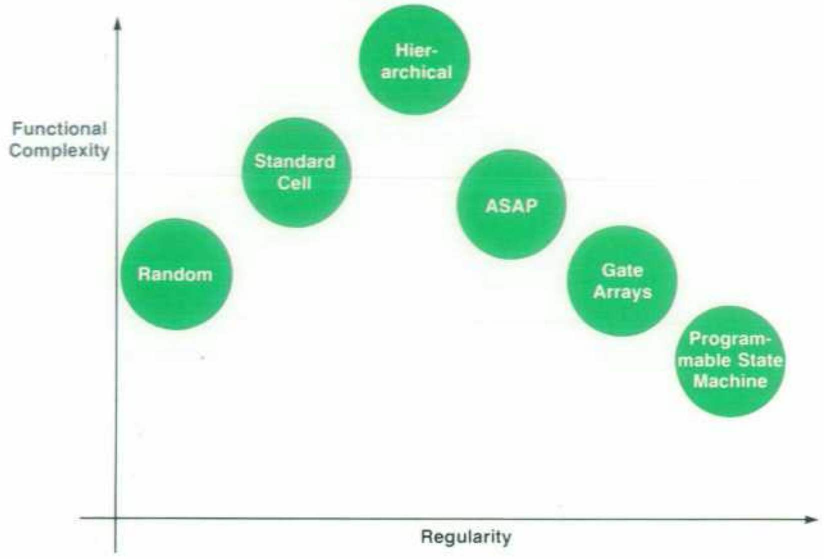


Fig. 2. *Six different IC design methodologies (circled areas) are used at Hewlett-Packard, depending on the functional complexity and regularity of the circuit being designed.*

Not only does this methodology lead to rather fast design but it also leads to rather fast processing turnaround. Once the designer has determined that the state-transition table is correct, the methodology guarantees a good circuit without having to use tools such as design-rule checking and circuit schematic extraction.

For larger bipolar circuits HP often uses the popular gate array techniques. Gate arrays are similar to PSMs, since the active components have predefined locations, and again wafer processing can be started before the design is finished. Gate arrays give the designer more freedom because the active components can be combined into various logic-gate configurations and interconnecting wires can be routed as the requirements of the circuit dictate. Often the design tools include automatic routing and placement programs which, given a schematic of the intended circuit, will interconnect the necessary active components. Although certainly not VLSI design, gate array techniques have been used for developing ICs with a complexity of up to 7000 logic gates.

The top end of HP's bipolar design methods uses the ASAP system described on page 8. The ASAP system allows the designer to place cells at any location and therefore excludes any prior processing of wafers before the design is completed. ASAP uses symbolic techniques which raise the designer from the rectangle level to a more familiar schematic level.

Although very distinct differences appear in bipolar designs, the same is not true for MOS designs. Many terms have been used to describe MOS design including structured, standard cell, top-down, and hierarchical; and in fact, many methodologies have been used. Two styles predominate today within Hewlett-Packard. One may be described as a traditional standard cell approach, the other a structured hierarchical approach. In the standard cell approach several logic gates are defined as cells. The quantity varies, but is usually less than twenty. The design is a two-part process: design of the cells and design of the interconnects. This might be considered a two-level hierarchy; however, in many cases several levels may be employed in the design of the cells.

Structured Hierarchical Design

The structured hierarchical approach is worthy of closer examination for several reasons. It has proved very effective in its reduction of design time, use of circuit area, and ability to support a multiple-person design effort. It is also of interest because of the close coupling of this strategy with that prescribed by Carver Mead and Lynn Conway in their textbook, Introduction to VLSI Systems.[1]

At the topmost level, the hierarchical approach can treat an integrated circuit as a collection of black boxes, each performing a useful function and communicating with other such boxes in a well-defined manner. The data and control pathways can be explicitly indicated in the form of a block diagram (coupled with a flow chart if necessary) without being obscured by the detail of the inner workings of any individual functional box. At this point the IC designer starts to construct the floor plan of the chip. This floor plan is the arrangement of all the functional boxes as they are expected to be physically located in the final mask artwork. The floor plan is optimized for minimum interconnect complexity and data pathway distance. With a suitable computer graphics system, this floor plan and corresponding interconnects can be interactively entered and meaningful names assigned to all the data and control pathways. This data base forms the standard against which all subsequent levels of hierarchy can be computer-verified for agreement.

At the next lower level of a hierarchical design each functional box is represented in additional detail. For example, a register at the topmost level will now be represented as several individual bit cells or a fast adder will now be represented as a combination of sum cells and carry cells. By appropriate choice of symbology, this additional complexity can be verified against the original design. In addition, a graphics system such as IGS (described on page 18) can display or suppress this additional detail at the designer's command. Each functional box can be worked on by different design groups at differing rates and their work checked with a hierarchical verification system. A functional box also should be designed using floor plan and interconnect techniques if it is at all complex.

(continued on page 10)

Advanced Symbolic Artwork Preparation (ASAP)

by Kyle M. Black and P. Kent Hardage

When considered in light of the capability it will provide, VLSI (very large scale integration) is very exciting to think about. When considered in terms of design and implementation, VLSI is frightening. The tremendous number of components present in VLSI circuits will make the design problem one of managing complexity while trying to maintain performance. The management of complexity has usually not been considered even in LSI design. We are more accustomed to the transistor-level design approach now used to achieve high performance.

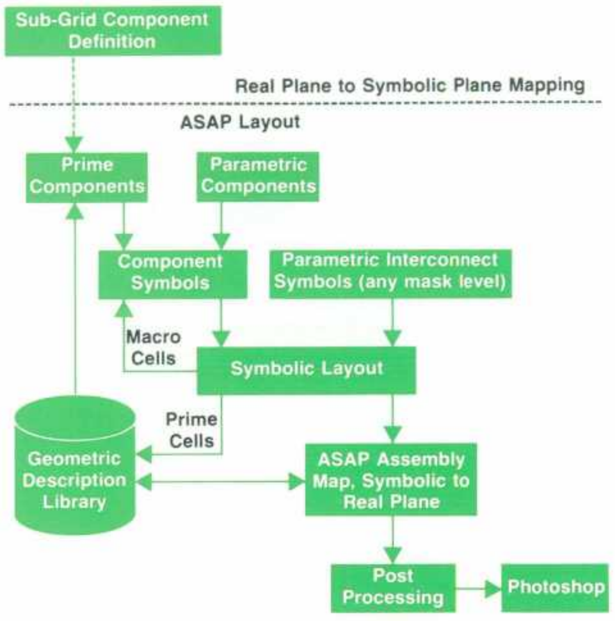

Sub-Grid Component Definition
Real Plane to Symbolic Plane Mapping
ASAP Layout
Prime Components
Parametric Components
Component Symbols
Parametric Interconnect Symbols (any mask level)
Macro Cells
Symbolic Layout
Prime Cells
Geometric Description Library
ASAP Assembly Map, Symbolic to Real Plane
Post Processing
Photoshop

Fig. 1. *Flow diagram of the symbolic layout process.*

Advanced symbolic artwork preparation (ASAP) was the answer that HP's Colorado Springs Division found to manage the complexity of IC design. Before the introduction of ASAP, typical design time for mask artwork alone was as long as 50 weeks. The ASAP design system was formulated to address not only the problem of mask layout but also the entire IC design cycle.

Conventional computer-aided design (CAD) systems use graphic design tools to capture, edit, and output the detailed physical descriptions of IC masks. While artwork generation systems have traditionally been the central focus of CAD efforts, HP's Colorado Springs Division found it necessary to address the complete design cycle to be effective in managing circuits of LSI complexity.

Today's software methodologies manage this degree of complexity by using macrodescriptions, high-level languages, and a structured design approach. ASAP offers a similar set of capabilities to custom IC designers. This is done by extending the software concept of mapping a program into a one-dimensional array of memory to that of mapping a symbolic hardware description into the two-dimensional physical planes of IC masks. ASAP enables the designer and the CAD system to use symbolic abstractions for structural, physical, and behavioral properties. Completed designs can be readily verified for correctness before they are assembled into descriptions for IC mask fabrication.

The ASAP system uses a graphic design description, modifying and extending the data representation in the following ways:

- Superfluous physical information is reduced by using high-level symbolic representations of components and interconnections.
- Using symbolic representations, the designer can see more of

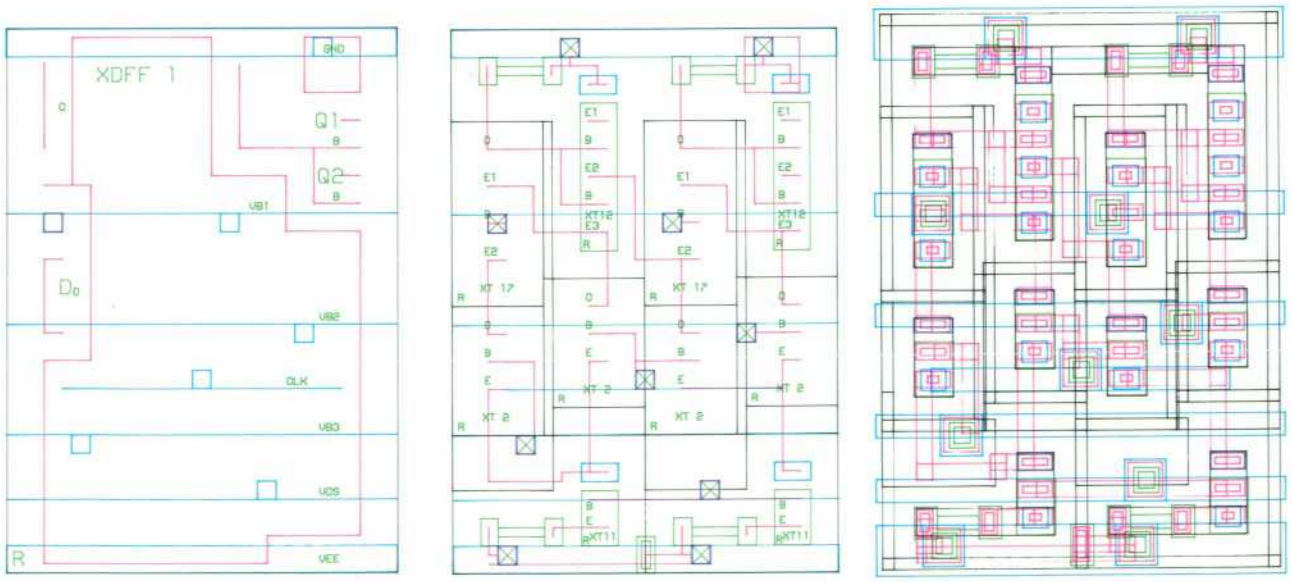


Fig. 2. *The ASAP design starts with the simple symbols on the far left and progresses to a more detailed symbolic representation which is translated into the geometrical layout used to fabricate the circuit shown on the far right.*

the design at a smaller scale, thus gaining a better overall perspective of the circuit.

- Structural and behavioral descriptions are added to aid design verification. The structural description defines the logical interconnections of the system, while the behavioral descriptions provide basic information about circuit interconnections.
- The mapping from the high-level symbolic description to the physical mask description is done by a chip assembler. This step occurs after the completion of the design process.

ASAP IC designs are built from a hierarchy of components. A component may be a primitive element as small as a via* or transistor, or may be assembled from a large collection of such primitives. The structural (and some behavioral) aspects of a component are captured along with the physical description. The external structure and behavior are defined by the component's physical boundries and interconnection points. Each component is named and its interconnection points defined. These definitions appear in the graphic descriptions for visual reference and in the data base for use in design verification. The component boundaries and interconnection points must lie on a grid network that is larger than the detailed design rules (minimum element width or spacing allowed) although any set of design rules can be used inside the primitive components themselves.

In addition to the primitive components, there are high-level component symbols. These symbols are PRIME if the component symbols are available from the system library and have predetermined geometry, or PARAMETRIC if the component symbol often changes size in the layout and an algorithm for mapping of that particular parametric component exists in the assembler. Fig. 1 shows the flow of the symbolic layout process. All parametric components have unique names and are recognized visually by their unique geometry. An algorithm in the assembler contains the components' structural and behavioral descriptions.

To use a component symbol correctly, a boundary must be defined for each mask level. These boundaries are an abstraction of the internal geometry of the component, and indicate legal wiring areas. Symbolic circuit design begins after a set of the most primitive symbolic components has been established for an IC process technology. Fig. 2 shows an example of the mapping that takes place in building an ASAP cell.

The complete ASAP design cycle shown in Fig. 3 addresses more than just the layout structure. The IC device and cell definitions are critically important to the design cycle. A hardware measurement set allows ac and dc device parameters to be determined. The parameters are used to generate models for the circuit simulator (SPICE). A logic library is then derived from the circuit simulator runs of individual cells. A circuit can be simulated with these logic cells on a logic simulator (TESTAID-IC) for state and timing information. A library of structured cells has been designed and descriptions placed in the geometric, circuit and logic libraries. The schematic is entered into the system by means of an interactive computer graphics station. Once entered, it can be simulated (circuit or logic simulation) and the results checked against the desired performance. If the desired performance is not achieved, new cells may be generated. The data base created by the schematic can also be used for comparison with the layout data base, thus checking continuity of the circuit with the schematic. Some behavioral checks are also made during the continuity check. With the aid of ASAP, system designers without detailed knowledge of integrated circuit device physics can now specify designs successfully.

ASAP has been used to complete some twenty custom IC chip designs. These chips support a wide range of products and applications in HP's analog oscilloscopes and CRT displays, analog-to-digital converters, and digital circuits for logic analyzers. Table I compares five ASAP IC designs to five earlier designs. Both sets of circuits use the same logic family and address similar

*Connection between a conductor in one interconnection level and a conductor in another level.

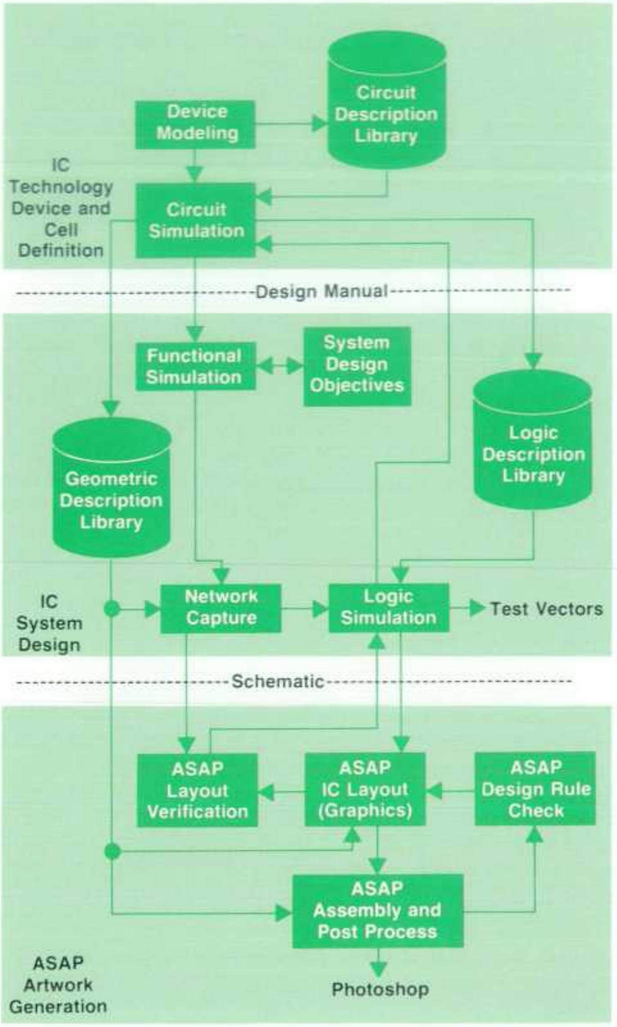


Fig. 3. *The ASAP design cycle has three sections as shown.*

Table I

Comparison of layout time and number of components designed per day for representative chips designed using a physical design approach (R) or a symbolic design approach (S).

Chip No.*	Layout Method	Chip Area (mm²)	Component Density (#/mm²)	Components Designed (#/day)	Layout Time (weeks)
1	R	9.7	165	8	40
2	R	12.3	125	8	40
3	R	10.2	180	5	65
4	R	9.0	155	9	32
5	R	14.8	200	9	65
6	S	9.6	195	80	3.1
7	S	8.7	210	103	2.4
8	S	13.7	200	103	3.6
9	S	13.2	230	138	3.1
10	S	13.5	170	76	4.3

*Odd-numbered chips have a regular architecture and the even-numbered chips have a random architecture.

P. Kent Hardage

Kent Hardage is CAD/tools manager with HP's Logic Systems Operation in Colorado Springs. With HP since 1966, he's been a CRT production engineer, an instrument design engineer, a project manager, and an IC design group leader. He's authored five papers on CAD and IC design and is co-author of a chapter in a forthcoming book on VLSI. He's a member of IEEE. Kent was born in Maud, Texas and attended Texas Tech University, graduating with a BSEE degree in 1966 and an MSEE in 1968. He is married, has three children, enjoys woodworking and camping, and lives in Manitou Springs, Colorado.

Kyle M. Black

Kyle Black received the BS and MS degrees in electrical engineering from the University of California at Berkeley in 1972 and 1973. Before joining HP in 1978, his experience included the development of the first bipolar FET op-amp, LM 156 (he is named inventor on a patent on its output stage). At HP, Kyle has worked on high-speed digital ICs and now is an IC design group leader with HP's Colorado Springs Division. He is a member of the IEEE. Born in Salinas, California, Kyle is married, has two children, and lives in Colorado Springs. He is active in his church and enjoys camping, fishing, woodworking, and carving.

design needs. Each group contains both random and regular architectures. The average design time has dropped from 46 weeks to three weeks, and the average number of components designed per day has increased from eight to over 100.

The structuring of the design process does not reduce component density. As stated above, global perspective and wiring management (the most important aspects of complex designs) have been improved by minimizing symbolic information not related to interconnection. IC design using ASAP emphasizes wiring management. As Table I shows, the component density has actually increased over that of earlier IC designs done by a detailed design approach.

Acknowledgments

The authors thank Jeff Sondeen for his contribution to the overall system software and particularly to the network capture and topological verification. John Morgan contributed to the graphics data capture and assembler software. Mary Howard and Judy Barrett accepted the challenge of being the first to design masks for IC chips in the "symbolic plane." The authors also thank John Strathman and Larry McBride of the technology system staff for their support and aid during the original project proposal.

Continuing to lower levels, each cell is expanded into its component parts, and at the bottom level the designer will be manipulating the actual geometric shapes as they will be processed into an IC mask. To reduce the total design effort it is highly desirable to create a library of the lowest-level cells (such as a single register bit), include a symbolic representation for each cell, verify each cell for electrical and geometrical correctness, and thereafter design a multitude of ICs using symbolic descriptions entirely. This allows a greater functional complexity overall (i.e., a greater number of active devices) to be incorporated on a single chip while not exceeding the capacity of the human mind to comprehend a large collection of "things". The parallel of this endeavor is found in the model of computer programming productivity where it is assumed that a software writer can produce x lines of code per day. If the writer is writing in a higher-level language such as PASCAL or ALGOL, the lines of code can perform many times more powerful operations than can x lines of assembly language code.

At HP's Desktop Computer Division in Fort Collins, Colorado this method of hierarchical design was successfully applied to an NMOS* IC having 10,200 transistors. By using IGS the design team was able to compose each level of the hierarchy in the manner just described and complete the artwork process in six months. This effort included the creation of a cell library because none previously existed. Hierarchical verification programs were not available to this project, so that any time an existing verification program was used, it was forced to operate on the level of greatest complexity. The time-versus-data relationship mentioned earlier was readily noticed. By careful grouping of items and using many features of IGS, the greatest number of errors found at any verification step was less than 10. This ensured that the final (entire chip) verification did not generate an overwhelming number of errors.

Before the six-month artwork period there were eight months of chip functional description, block diagram creation, and functional verification. On one occasion the block diagram was modified because the initial floor plan indicated an undesirable interconnect pattern. This helps point out the fact that, while there may be several acceptable floor plan arrangements for an IC, a wrong choice may make the IC physically unrealizable. The critical first step needed in any successful design is to know completely how the device must respond to external stimulus (the functional description). All the input and output characteristics need to be agreed upon by both the user of the device and the designer of the device. This description should be stable and accurate before proceeding beyond the floor plan stage. While the functional description is being finalized, preliminary floor plans will be useful to ensure realizability, because it can be very, very difficult to stuff another black box into an IC that has progressed to the mask layout phase.

An Attempt at Prophecy

What about the future? Perhaps the safest prediction is that the future will lead to further change. Several factors dictate th[illegible] The design tools used now are simply running out of capability. Design verification is perhaps the most critical area today because verification of a complex circuit can take 30 CPU hours on the largest computers available. Plotting and analysis of results usually takes weeks. Assum-

*N-channel metal-oxide-semiconductor.

ing the programs do detect errors, correction of these errors may require the relative skills of a brain surgeon, so as not to disturb any of the other million or so features on the mask set. We must have a design methodology that will allow small pieces to be designed, tested, and used without fear of possible adverse effects. This is not a new idea, it has been used in structured computer programming for many years. In fact, the best clues as to how to manage future designs may come from the technology developed for programming computers.

Before the acceptance of structured programming, programmers had the freedom to use unlimited and unconstrained control statements. Program flow could move freely to and from various segments of code by means of the GO TO statement. An individual piece of code could be used in a variety of ways because of the unlimited branching capability. With this methodology the ultimate constraint is the fact that a section of code can only be tested in the context of the complete program. A programmer might create a segment of code with one function in mind, but it could be used later by another programmer to perform a different task, often with questionable results. Modifications often had adverse effects on sections of code that were considered to be immune to such changes.

Structured programming placed severe restrictions on programmers because they could only enter and exit a segment of code with a single entry point and a single exit point. Unrestricted branching was eliminated. However, the benefits were great. With the new techniques a segment of code could be written and tested independently of the program to which the segment would be attached. Testing was reduced to simple testing of small segments rather than exhaustive and often inconclusive testing of a complete program. The programs were also more understandable, logical, orderly, and complete. History has shown that this structured methodology has given benefits far exceeding those that were removed by constraining the software designers.

Lou Scheffer, a member of Hewlett-Packard's design aids team and a PhD student at Stanford University, has proposed a constrained methodology for the hardware design of integrated circuits similar to the constraints imposed on software design by structured programming. Fundamental to Lou's approach is the ability to design cells and verify them independently of the complete circuit. This is done by eliminating the overlapping of cells and the placement of active components near the boundaries of cells. Lou does not restrict the boundaries of cells to simple rectangles as do many methodologies today, but permits irregularly shaped boundaries. Many constrained methodologies decrease circuit density, but this one, at least in theory, does not. It remains to be seen whether designers can actually achieve this density in practice.

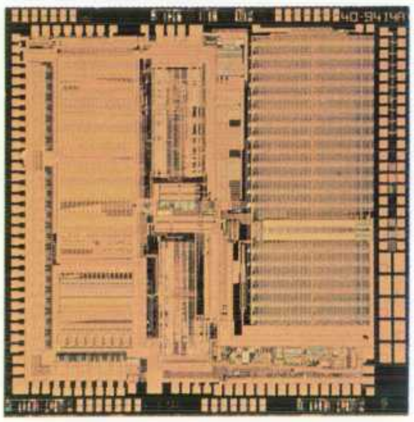
Silicon Compiler

Another parallel to the use of software methods in the design of hardware is the silicon compiler. Although still very much restricted to universities it is becoming of increasing interest to industry. In designing with a silicon compiler, a programming language is used to describe the geometry of cells. These descriptions then become procedures in the language. It is possible to pass parameters to these procedures to generate a variety of different cell structures, depending on the value of the parameter. A simple example might be a procedure for the design of an inverter. The passed parameter might determine the circuit loading for the inverter. The procedure would then use this value to calculate the sizes of the transistors to be used in the inverter. These procedures would then be tied together by other procedures until the complete integrated circuit is de-

VLSI Makes 32-Bit CPU Chip Possible

The design capability and complexity possible with VLSI technology are demonstrated by a 32-bit processor designed and fabricated on a single chip by Hewlett-Packard in Fort Collins, Colorado. The chip (see photo) contains 450,000 transistors interconnected by two layers of metallization with minimum feature size and spacing of 1.5 and 1 μm, respectively. This design and the fabrication process were described in two papers presented at the 1981 International Solid-State Circuits Conference.[1,2] This 6.35-mm-square chip vividly illustrates the need for the design tools and methodologies discussed in this issue. For example, if drawings of each mask layer are prepared with a scale where a 2-mm-wide line represents the narrowest metal line on the actual chip, a room at least ten metres square would be required to lay out the drawings! Add to this the requirement that each mask layer must be checked for proper alignment with the other layers within less than 1 millimetre at any point on these huge drawings, one can see that reviewing the chip design without tools and a methodology would be very difficult, if not impossible.

Only by partitioning the design into smaller segments and using design and artwork verification tools was the Fort Collins Division able to reduce the difficulties associated with fabricating this complex system as a VLSI circuit.



References

1. J. Beyers, et al., "A 32b VLSI Chip," Digest of Technical Papers, 1981 IEEE International Solid-State Circuits Conference, THAM 9.1.
2. J. Mikkelson, et al., "An NMOS VLSI Process for Fabrication of a 32b CPU Chip," Digest of Technical Papers, 1981 IEEE International Solid-State Circuits Conference, THAM 9.2.

scribed. At present this technology is in its infancy. Many problems exist, but it is at least theoretically possible to describe any circuit with a silicon compiler language. The greatest promise of this technique is perhaps in our knowledge of creating large software programs by assembling many small subprograms.

Correct the First Time

The difficulties in discovering errors later in the design process were discussed previously. Stan Mintz of HP's Corvallis Division in Oregon has proposed a "correct-the-first-time" methodology. Stan proposes that the functional description is the control document for both the development of an IC and any tests that will be applied to it. In addition, as the design progresses through logic, circuit, and mask design, Stan proposes that each stage be compared to the functional description. Furthermore, he states that the test program used to evaluate the IC be generated from the functional description. In theory, if each path rigorously follows the verification procedures, when the finished IC and test program meet at the tester there should be no reason for a failure. Stan's proposal has many implications for both methodology and tools, but if enforced, should relieve the disastrous effects of discovering an error late in the design sequence.

Finally, as an extension of Stan's proposal for establishing the functional description as the premier description of an IC, the potential exists for completely synthesizing the design from this description. There is some very encouraging work progressing at Carnegie-Mellon University. Their work involves synthesis of a design from a high-level ISPS language description into implementable components.[2] Their techniques involve high-level decisions as to the selection of a design style and implementation of a design based on that design style.

References

1. C. Mead and L. Conway, "Introduction to VLSI Systems," Addison-Wesley, 1980.
2. C. Bell and A. Newell, "Computer Structures: Readings and Examples," McGraw-Hill, 1971.

William J. Haydamack

Bill Haydamack is a native of the province of Alberta, Canada. He attended the University of Alberta in Edmonton and received BSEE and MSEE degrees in 1963 and 1966. After three years of automatic test system design, he joined HP's Automatic Measurements Division in 1969 and served as project manager and group leader. He's now section manager for design verification with HP's design aids group. He has authored eight papers on computer-aided design and his work has resulted in three U.S. patents. A resident of San Jose, California, Bill is married and has two sons. He is active in YMCA youth activities, having served in various offices and as a baseball and basketball coach. He enjoys various sports and outdoor activities and does his own photographic darkroom work.

Daniel J. Griffin

Dan Griffin received his BSEE degree from Massachusetts Institute of Technology in 1971 and his MSEE degree from the University of California at Berkeley in 1972. With HP's Desktop Computer Division since 1972, he was on the project teams for the 9845 Computer and the NMOS processor chip set used in the 9825 and 9845. He was in IC production engineering for two years and is currently project manager for three NMOS LSI chips. He's a member of IEEE. Dan was born in Denver, Colorado and now lives in Fort Collins. He's married and enjoys photography and hot air ballooning.

Design and Simulation of VLSI Circuits

by Louis K. Scheffer, Richard I. Dowell, and Ravi M. Apte

A VLSI CIRCUIT is a complex structure containing tens of thousands of individual transistors. The design of VLSI proceeds from a high-level architectural description down to the layout of the artwork. The circuit design process is the activity that occurs between the functional description and the start of the physical layout. However, for integrated circuit design, there is usually no clean separation at either end of the process. System architectures are specified with a final implementation in mind, and layout and device designs often dictate changes in the system design. The responsibilities of the circuit designer are to insure that the circuit performs its intended function, verify that the function is performed at the required speed, and guarantee that the circuit can be manufactured and tested for a reasonable cost.

Circuit design may be divided into several stages. In the first stage, the designer considers the functional specifications and produces a logic design. Various portions of the allowable delays and the power consumption are allocated to parts of the total circuit. This step requires a significant amount of experience, since optimizing a VLSI design is far different from optimizing a design in a technology such as TTL. Speed, power, space, and design time can all be traded off against each other, but a wrong decision can lead to a great deal of backtracking, since all portions of a design may be interdependent. At this stage of the design, a logic simulator (such as TESTAID-IC) may be used to verify that the desired function is indeed performed.

This translation step from functional to logical circuit description is of great interest to current researchers. To lessen the required integrated circuit knowledge and reduce the design time and errors, structured methodologies have emerged that permit the computer to assume a greater role in the design.

The next step is to turn the logic design into a transistor level design. In many cases, this is not done explicitly, for it is simpler to design the artwork directly from the logic specification. Where performance is critical, however, detailed investigation of the transistor level designs is required. The circuit simulator SPICE is used to do verification on this level. It allows the designer to look at speeds, powers, noise margins, best and worst cases, and the effects of process variations. By using the results of these detailed simulations, designers can significantly improve the reliability and yield of the final product.

Another necessary step is the definition of the tests for the chip. The designer must guarantee that the chip can be tested. This is partially achieved in the logic design stage, by making sure that the storage elements can be read and controlled, but the actual definition of tests must take place after the logic design is completed. Logic simulators (TESTAID-IC) offer considerable help with this task.

All of the preceding steps deal with logic diagrams and schematics, which must be converted to machine readable form so that simulators may use them. Furthermore, several different representations of a circuit usually exist, and a hierarchical design is almost a necessity to keep all the details straight. DRAW is a program that helps the designer in these areas. With a graphics terminal and a digitizing pad, the designer can enter and edit schematic diagrams interactively. The hierarchy is easily maintained and used, and the program can convert conventional schematics to the textual forms required by the simulators.

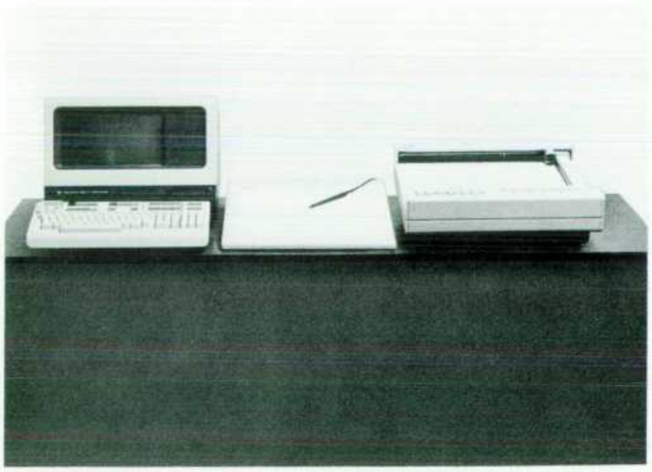


Fig. 1. *DRAW circuit design station consists of an interactive graphics terminal, a digitizing tablet, and a four-color plotter, all tied to a timeshared computer. DRAW is a graphics editor designed for editing schematics.*

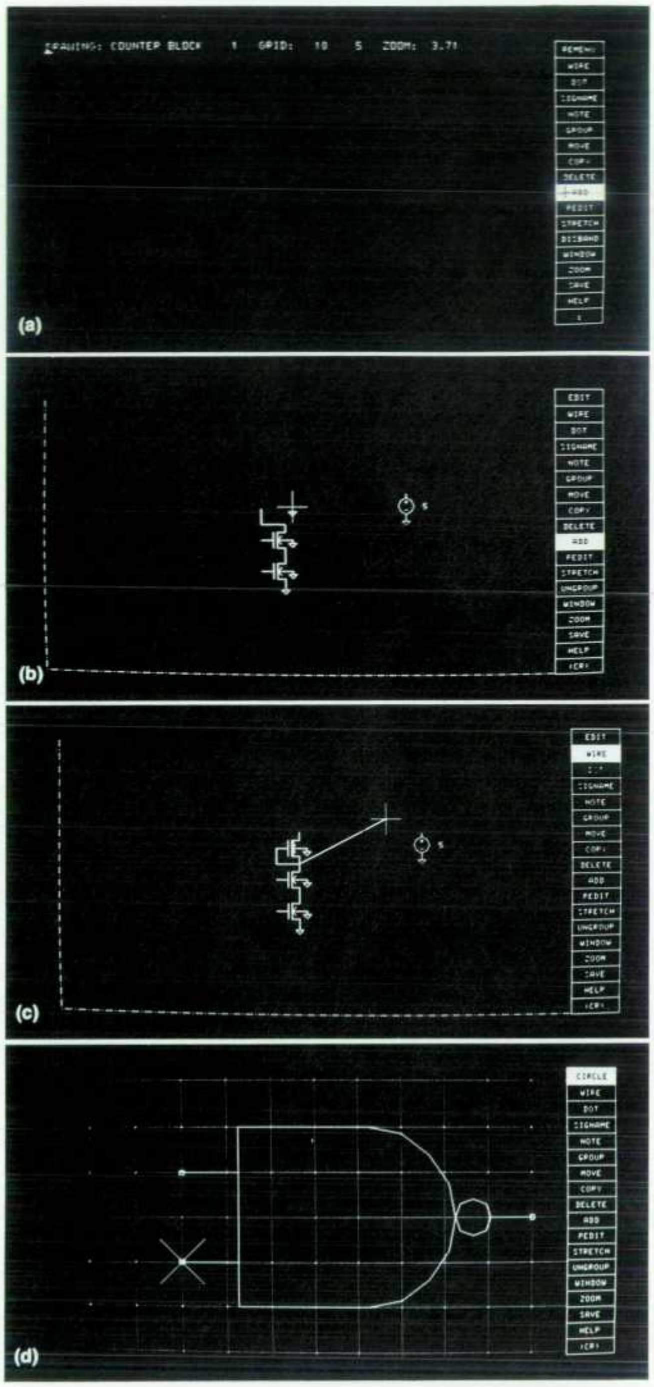


Fig. 2. *Using the DRAW schematic editor. (a) Select the ADD command. (b) Place a transistor. (c) Wire two points together. (d) Define an external VIEW drawing to be used on higher-level schematics.*

Schematic Input Using DRAW

Schematic entry using DRAW involves a combination of tablet and keyboard input (Fig. 1). Schematics are kept in a

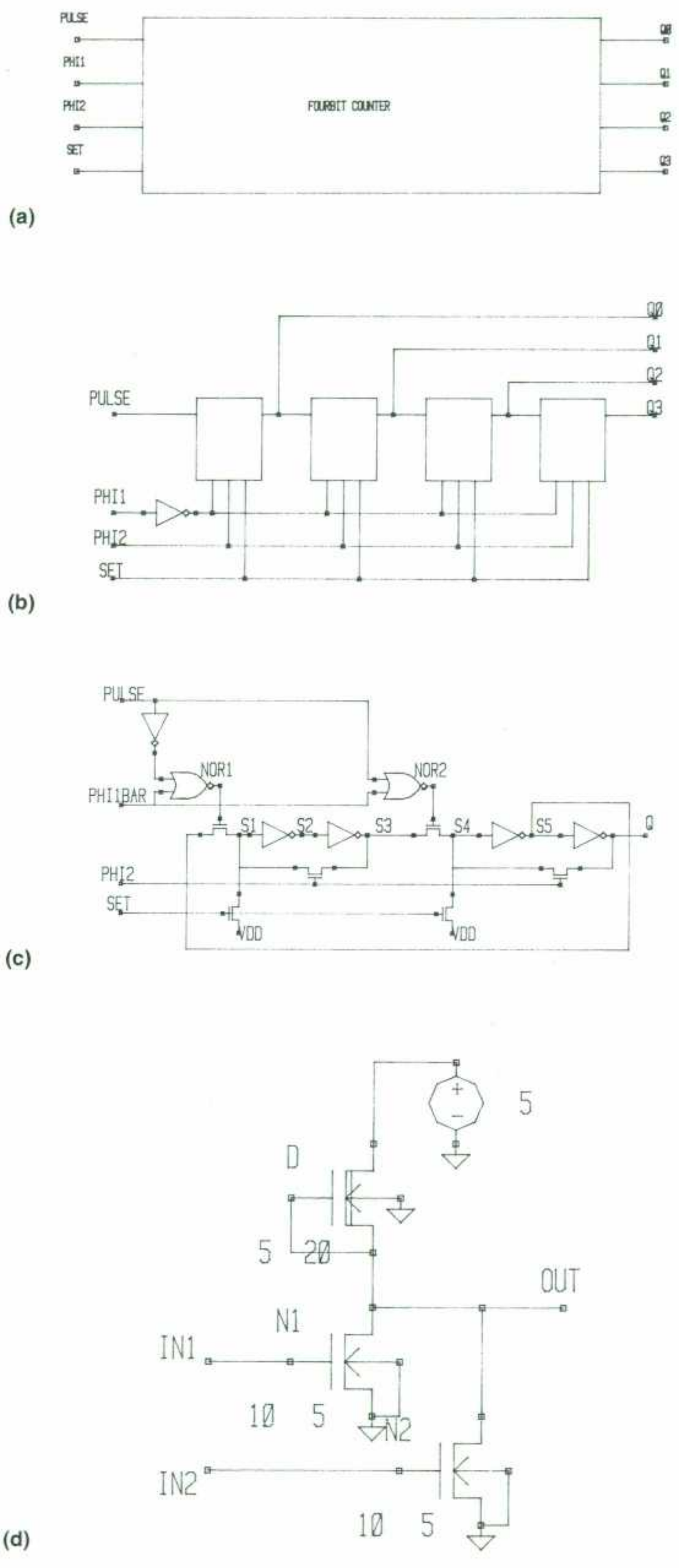


Fig. 3. *Schematics for a four-bit counter, created by DRAW. (a) VIEW-level plot of the counter. (b) Block-level plot. (c) Block-level plot of a one-bit cell. (d) HP-SPICE plot of a NOR gate.*

library which is managed by DRAW on the user's account. A "clean sheet of paper" is created by asking to edit a presently nonexistent drawing. The screen of the HP 2648A Terminal provides a window to the work area, as shown in Fig. 2. Both text and graphics displays are used. In Fig. 2a, the cursor has been placed on top of the ADD command and the stylus depressed. DRAW interprets the position of the cursor to be a request for the ADD command and highlights it as shown in the figure. The type of component to be added is typed at the keyboard together with its parameter values, if any. The placement of an NMOS transistor onto a partially completed schematic is shown in Fig. 2b. The WIRE command is used to interconnect the components. The use of this command is aided by the rubber-band line mode of the terminal. The cursor with a loose wire attached is shown in Fig. 2c. User-defined components that are to be used in higher-level schematics are defined by an external VIEW. The description of an external VIEW drawing is shown in Fig. 2d. The grid is provided to aid in the creation of an outline for the component, which is drawn using the WIRE command.

A four-bit counter can be used to illustrate the hierarchical description of DRAW. The highest level of description is shown in Fig. 3a as an implementation-independent four-bit counter with four inputs and four outputs. Figs. 3b, 3c, and 3d show schematics at different stages of design. Both user-defined components such as the NOR gates and SPICE components such as MOS transistors have been used. These schematics are easily incorporated into larger systems and can be used to drive simulators such as TESTAID-IC and SPICE.

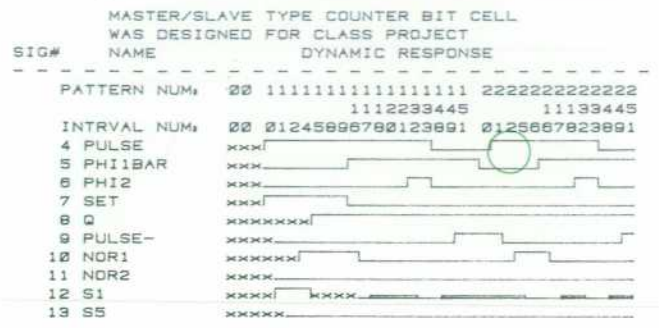
Logic Simulation Using TESTAID-IC

TESTAID-IC is a multilevel digital logic simulator for verifying that a proposed design is logically correct and meets timing requirements. Circuit descriptions permit components to be described at Boolean, logic, and transistor levels. Data not appropriate for schematic display is kept in an associated text file. Such data typically consists of transition delay times, Boolean equations and other nongraphic descriptions. In response to a command, DRAW creates a text file containing a TESTAID description. Fig. 4a shows the output generated by TESTAID-IC after simulating the four-bit counter shown in Fig. 3b. At this level the designer is interested only in checking out the functional behavior of the circuit. An output format using 1s, 0s, and Xs is chosen to display the results. The circuit can be run for all the possible states for checking out complete functionality.

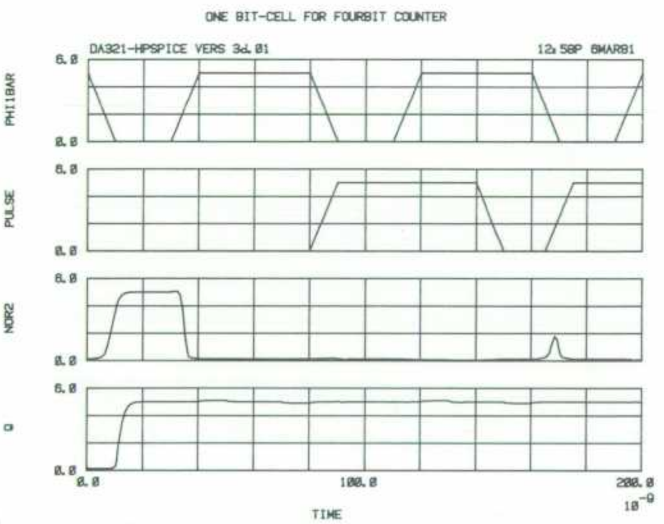
Fig. 4b shows the output waveform generated by TESTAID-IC after simulating the one-bit latch shown in Fig. 3c. The waveform shows the functional response of the circuit to the user-specified stimulus at the logic level. The stimulus applied has a very precarious timing relationship between the SET and PULSE inputs and the two-phase clock. The close timing of the PULSE and PHI1BAR signals is shown circled in Fig. 4b. The waveform in Fig. 4c shows the consequences of the short delay between PULSE and PHI1BAR inputs. The NOR gate, NOR2, does not respond fast enough to the changes on the input side because of its inertia, and consequently it fails to toggle the Q line. The SPICE output shows the NOR2 signal starting to rise but returning to the

```
      MASTER/SLAVE TYPE FOUR-BIT COUNTER

SIG#     NAME                 DYNAMIC RESPONSE
- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
      PATTERN NUM>   00 1111 2222 3333 4444 3333 4444 3333 4444

                         134  134  134  134  134  134  134  134
      INTRVAL NUM>   00 0628 0628 0628 0628 0628 0628 0628 0628

 4       PHI1        .. 1000 1000 1000 1000 1000 1000 1000 1000
 5       PHI2        .. 0010 0010 0010 0010 0010 0010 0010 0010
 7       SET         .. 1111 0000 0000 0000 0000 0000 0000 0000
 6       PULSE       .. 0000 0000 1111 0000 1111 0000 1111 0000
 8       Q3          .. 1111 1111 1111 0000 0000 0000 0000 0000
 9       Q2          .. 1111 1111 1111 0000 0000 0000 0000 0000
10       Q1          .. 1111 1111 1111 0000 0000 0000 0000 1111
11       Q0          .. 1111 1111 1111 0000 0000 1111 1111 0000
```

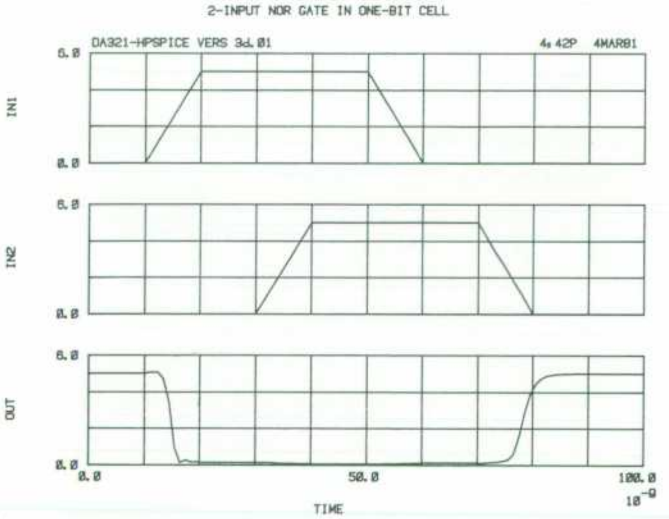
(a)



(b)



(c)



(d)

Fig. 4. *TESTAID-IC and HP-SPICE plots. (a) Functional behavior of a four-bit counter. (b) Logic response of a one-bit counter cell. (c) HP-SPICE plot of a one-bit cell response, showing a toggle failure. (d) HP-SPICE plot of a two-input NOR gate response.*

zero state instead of switching to the one state. Further study using SPICE would probably be done at the single-gate level. Other potential problems such as uninitialized nodes and nodes in the high-impedence state are shown with Xs and double lines, respectively in Fig. 4b.

A means to evaluate the effectiveness of circuit testing schemes is provided by a fault simulation option. Under fault simulation mode, circuit nodes are clamped to logic 0 and logic 1 sequentially. The proposed test vector (set of input values) is applied and the faulty outputs compared with the no-fault output. A difference implies that this fault can be detected. For a given set of test vectors, TESTAID-IC runs the fault analysis and computes the percentage of faults that will be detected. Fault analysis is used as an adjunct to functional testing, since a full functional test is not always economical.

Circuit Simulation Using HP-SPICE

The SPICE circuit simulation program developed at the University of California at Berkeley has become a workhorse for the IC industry. SPICE performs a detailed analysis of circuits using the interconnect data from DRAW. Accurate analytical models are provided for the semiconductor devices.

The DRAW display of a basic NMOS NOR gate is shown in Fig. 3d. The library of circuit primitives provides the transistors, capacitors, resistors and other SPICE elements. Stored in the associated text file are device model parameters, source excitation, and SPICE analysis control information.

A typical output from HP-SPICE is plotted in Fig. 4d. The waveforms show the gate response for worst-case conditions of circuit speed. The signal delay through a typical gate is often studied using these simulated waveforms. Circuit designers typically perform hundreds of similar simulations during the course of a design. The simulations can provide the detailed information necessary for assuring that the final design will be reliable. The program's interactiveness, flexibility, and graphical output have been found to be important contributors to designers' productivity.

While simulators are usually most effective for a certain level in the circuit hierarchy, it is vitally important for simulators to be usable at adjacent levels. On the whole, simulations are usually carried out in the most economical manner, with the assurance that sticky problems can be solved using less efficient but more accurate simulations at the next level of detail. The counter cell shown in Fig. 3c is made up of the gates whose switching performance is shown in Fig. 4d. This simulation could be used to confirm timing calculations done from rough extrapolations from the gate performance.

Conclusions

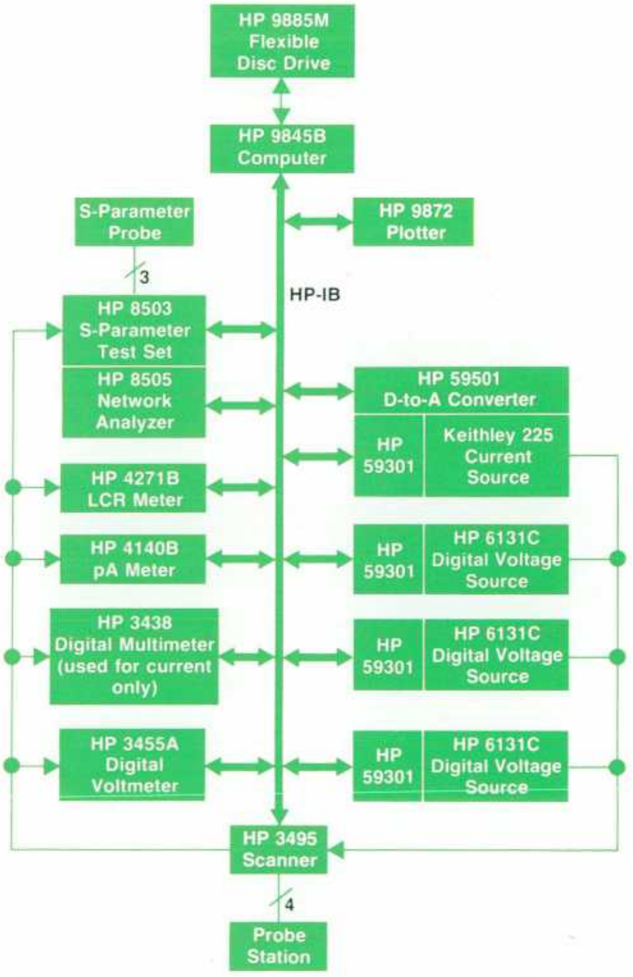
The advent of VLSI has forced the designers of both systems and design tools to come to grips with the necessity for multilevel system descriptions. The schematic-based system, DRAW, has been offered as one appropriate means for describing VLSI systems. The simulators that have been interfaced to DRAW provide a powerful capability, but are not adequate for evaluation of complete VLSI designs. Simulators and description languages focusing on levels higher than logic states will have to be integrated into a VLSI design package. In addition, the need to simulate

Transistor Electrical Characterization and Analysis Program (TECAP)

by Ebrahim Khalily

As simulation becomes a cornerstone of VLSI design and as technology advances toward fabrication of small-geometry devices for VLSI, greater process control and more accurate simulation models become more important. With the increases in the complexity of device models, an automated characterization system is needed to perform quality transistor measurements and provide reliable model parameters interactively and in a reasonable time.

The Transistor Electrical Characterization and Analysis Program (TECAP) is a characterization system based on the HP 9845B Desktop Computer and several HP-IB-compatible measurement instruments. The system is intended for the IC design environment, where understanding of device and circuit limitations is important. TECAP is designed as a general measurement and characterization tool, with interactive measurements on a variety of devices, including resistors, capacitors, bipolar NPN and PNP transistors, enhancement and depletion mode P and N-channel MOSFET transistors, and JFETs. The model parameters for bipolar and MOS transistors are obtained from these measurements. The extracted parameters are then used in a model analysis program to check the validity of the model through direct comparison with measured results.



HP 9885M Flexible Disc Drive
HP 9845B Computer
S-Parameter Probe
HP 9872 Plotter
3
HP-IB
HP 8503 S-Parameter Test Set
HP 8505 Network Analyzer
HP 59501 D-to-A Converter
HP 59301
Keithley 225 Current Source
HP 4271B LCR Meter
HP 4140B pA Meter
HP 59301
HP 6131C Digital Voltage Source
HP 3438 Digital Multimeter (used for current only)
HP 59301
HP 6131C Digital Voltage Source
HP 3455A Digital Voltmeter
HP 59301
HP 6131C Digital Voltage Source
HP 3495 Scanner
4
Probe Station

Fig. 1. *Block diagram of the TECAP (transistor electrical characterization and analysis program) system. TECAP makes measurements on bipolar and MOS transistors and computes the parameters of models of these devices.*

The HP 9845B Desktop Computer is used as the system controller. The enhanced BASIC language provides an easy-to-program, easy-to-use system. A flexible disc drive provides accessible storage for programs and data. The graphics capability of the HP 9845B and the HP 9872 Multicolor X-Y Plotter give fast visual feedback to the user.

The HP-IB* is the communication link between the computer and all of the instruments in the system. The HP-IB commands are easy to program, and the standard connections of the HP-IB provide hardware modularity, allowing any instrument to be replaced by another HP-IB-compatible instrument without any hard wiring and with minimum software modifications.

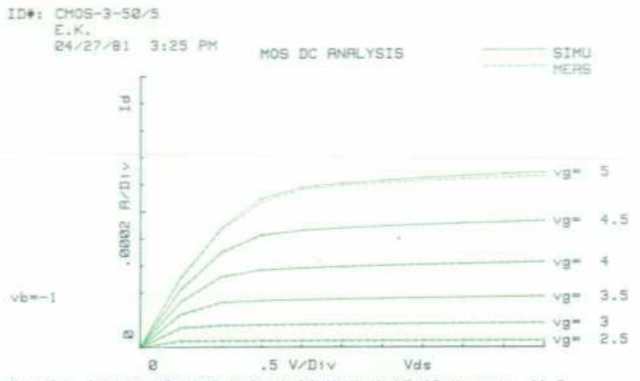
The block diagram of the system is shown in Fig. 1. Three HP 6131C Digital Voltage Sources provide ±100V, 0.5A outputs with a resolution of 1 mV. An HP 59301 ASCII-to-Parallel Converter interfaces each source to the HP-IB. Current sourcing capability is provided by a Keithley 225 Current Source modified for remote programming. An HP 59501 Digital-to-Analog Converter is used to control the current source in the range of ±100 mA with a resolution of 0.1 nA. Voltage measurement is done by an HP 3455A Digital Voltmeter with a resolution of 10 μV. Low current measurements (less than 20 mA) are done by an HP 4140B pA Meter with 1 fA resolution. Currents greater than 20 mA are measured by an HP 3438A Digital Multimeter. The HP 4271B LCR Meter provides capacitance measurement capability from 0.001 pF to 10 nF. The HP 8505 Network Analyzer accompanied by an HP 8503 S-Parameter Test Set enables the system to measure high-frequency (0.5-1300 MHz) parameters. The 3495A Scanner, a matrix switch constructed of 40 programmable two-pole switches, connects the instruments to the four terminals of the system. These four terminals can be either directly connected to the device under test or connected to a probe station for on-wafer measurements.

To maintain system modularity, TECAP uses separate subroutines to control each instrument on the HP-IB. These subroutines enable the user to write or modify the programs using a high-level language. Each section of the program is stored in a separate file on the system flexible disc, and is loaded into the computer and executed only when it is needed. Thirty-two function keys on the desktop computer provide fast access to different tests and measurements.

In general, the application of the system can be divided into two different categories, process control and device characterizations.

A process control engineer is more interested in data on physical parameters of the process. For example, C-V (capacitance versus voltage) measurements can help determine oxide thickness, surface state charge density Q_{ss}, and threshold voltage.

*Hewlett-Packard's implementation of IEEE Standard 488-1978.



```
Beta0=4.37E-04 U0= 530 Vt0=+1.28 Nsub=1.9E+16 Vnorm=  41.6
Desat=2.24E+09 Ecrit=2.73E+04 Etra=1.18E+05 Tox=7.50E-08
Rs=  26 Rd=  26 Xj=1.00E-06
Ld=1.18E-06 Wd=1.35E-06 L=5.00E-06 W=5.00E-05
```

Fig. 2. *A TECAP dc characterization of an MOS transistor. Model parameters have been extracted from measured data and the simulated result is compared with the measurement.*

Also, parameters like sheet resistances and breakdown voltages yield valuable information about the quality of the process. Statistical analysis of the data reveals process variations.

An IC designer, on the other hand, uses the electrical characteristics of the device. Parameters like saturation current I_s and forward current gain β at different current levels for bipolar transistors and mobility μ_o for MOS transistors are some of the essential design parameters. Special phenomena such as saturation, subthreshold, and punchthrough of a transistor are also of great importance in the design of ICs. With an accurate computer model, the only obstacle to successful circuit simulation is the determination of good model parameters. TECAP extracts model parameters through direct measurements on actual transistors. Parameter extraction programs can then be merged with model and parameter analysis programs to compare the simulations to the measurement, thereby checking the validity and accuracy of the model. The MOS and bipolar models used in TECAP are similar to models available on HP-SPICE.

TECAP can also be used to tie the circuit simulators to the device and process simulators. The output of the process modeling program HP-SUPREM, which simulates the impurity profiles in different cross-sections of the process, is used with device modeling programs like SEDAN or GEMINI, which generate device I-V (current versus voltage) characteristics. TECAP can then work with simulated I-V data instead of measurements to extract the model parameters for SPICE. This complete line of process device and circuit simulation allows users to simulate circuits based on process data, and is a powerful tool for studying the effects of process parameters on the performance of the final circuits.

Fig. 2 shows an example of dc characterization on an MOS transistor. The model parameters, including the parameters for short-channel effects, are extracted from measured data and the simulated result is compared with the measurement. The analysis part of the system lets the user simulate and plot drain current and its derivatives and compare them with the measurement.

Acknowledgments

TECAP, SUPREM, SEDAN and GEMINI are programs that have been developed at Stanford University under the direction of Bob Dutton and sponsored by both HP and the Advanced Research Projects Agency (sponsored by the U.S. Army Electronics Technology and Devices Laboratory).

Ebrahim Khalily

Ebrahim Khalily is a development engineer specializing in modeling and characterization with HP's design aids group. Born in Kermanshah, Iran, he received his BS degree in electronics from Tehran Technical University in 1973 and his MS and Engineer degrees in integrated circuits from Stanford University in 1978 and 1979. He joined HP in 1979. He has authored papers on automated characterization and process and device simulation and is a member of IEEE. Ebrahim lives in Cupertino, California and enjoys outdoor activities, sports, music, and bridge.

larger structures in existing simulators has spawned multilevel simulators to deal simultaneously with detailed and logical signal levels. While much work is yet to be done to improve the capabilities for designing with VLSI, one must not lose sight of the designer's fundamental concerns: Will the design work? Will it perform to specifications? Can it be manufactured and tested at reasonable cost?

The computer is able to perform a larger role in providing the information necessary to answer these questions. We must recognize that as tools become more sophisticated, design requirements are keeping pace. If we are to remain successful, we must provide a design environment that has the flexibility to permit plenty of "good old-fashioned engineering" to be added by creative design engineers.

Acknowledgments

The contributions in simulation from Professors D.O.Pederson and A.R. Newton and their students at the University of California at Berkeley are gratefully acknowledged. We appreciate and acknowledge the semiconductor device modeling work done under the direction of Professor R.W. Dutton at Stanford, which plays an essential role in the usefulness of SPICE. The DRAW team of D.B Gorman, K.M. Mann, and J.P. Skudlarek has done a fine job of creating a real design tool, as has the HP-SPICE team of I.D. Klein

Louis K. Scheffer

Lou Scheffer is a member of HP's design aids group and a PhD candidate at Stanford University. He received his BS degree in engineering in 1974 and his MS degree in electrical engineering in 1975, both from California Institute of Technology. He joined HP in 1975. He has designed digital filters for the 3582A Spectrum Analyzer and developed microprocessor assemblers and a high-level language. He has authored three papers on IC design tools. Lou is a native of Rochester, New York. His hobbies are audio electronics and bicycling.

Richard I. Dowell

Dick Dowell studied electrical engineering at the University of California at Berkeley, receiving his BS, MS, and PhD degrees in 1966, 1969, and 1972. His PhD research was on computer-aided simulation and optimization. He came to HP from Bell Telephone Laboratories in 1977 to work on simulation and modeling with HP's design aids group, and is now project manager for development of simulators and post-processors. A member of IEEE, he's authored three papers on computer-aided design and has lectured on computer-aided circuit simulation at Berkeley. Born in Birmingham, Alabama, he lives in Saratoga, California and enjoys blue grass music and flying.

Ravi M. Apte

Ravi Apte holds a B. Tech degree in electrical engineering from the Indian Institute of Technology in Bombay, an MS in computer science from the University of Missouri at Rolla, and a PhD in electrical engineering from Southern Methodist University in Dallas. With HP since 1976, he's served as project leader for studies in testing, logic simulation, test pattern generation, and topology extraction. He's authored three papers and participated in several workshops on computer-aided design, and is a member of IEEE and the IEEE Computer Society. Before joining HP he served as a consultant to HP and other companies. Ravi is married, has two sons, and lives in San Jose, California. He has traveled extensively and enjoys reading and gardening.

and Y.C. Yuan. TESTAID-IC evolved from the HP product TESTAID,[1] and has undergone considerable enhancement by B.V. Infante, C.G. Jue, D.W. Nelson, and D.S. Wu.

Reference

1. K.P. Parker, "Software Simulator Speeds Digital Board Test Generation," Hewlett-Packard Journal, March 1979.

An Interactive Graphics System for Structured Design of Integrated Circuits

by Diane F. Bracken and William J. McCalla

A LARGE PORTION of the design cycle time of LSI and VLSI circuits is devoted to creating a physical layout and verifying that the layout obeys design rules and represents the intended circuit schematic. IGS, an HP interactive graphics system,[1] supports structured, hierarchical design of custom integrated circuits through an advanced, fully integrated set of features. User-definable multilevel symbolic representation and incremental design rule checking facilitate design and layout in an abstract or symbolic format. Input and editing are done interactively at a color graphics workstation. User specification of process information makes the system technology-independent while retaining the features that make it a useful design tool. These features are enhanced through the availability of a flexible macrocommand capability. Access to comprehensive design verification and mask generation capabilities is provided, and both on-line and off-line plotting are supported.

IGS System Overview

IGS is a multiprocessor system. It uses an HP 3000 as the host computer and up to four HP 1000s connected via high-speed links to the host as graphics workstations. The basic workstation (Fig. 1) consists of an HP 1000F, an HP 2645A alphanumeric CRT terminal, an HP 9111A Graphics

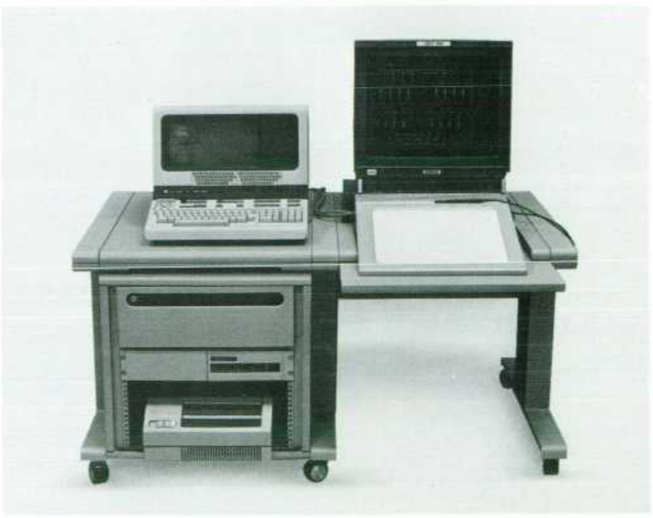


Fig. 1. *IGS workstation consists of computer, CRT terminal, graphics tablet, and high-resolution color display.*

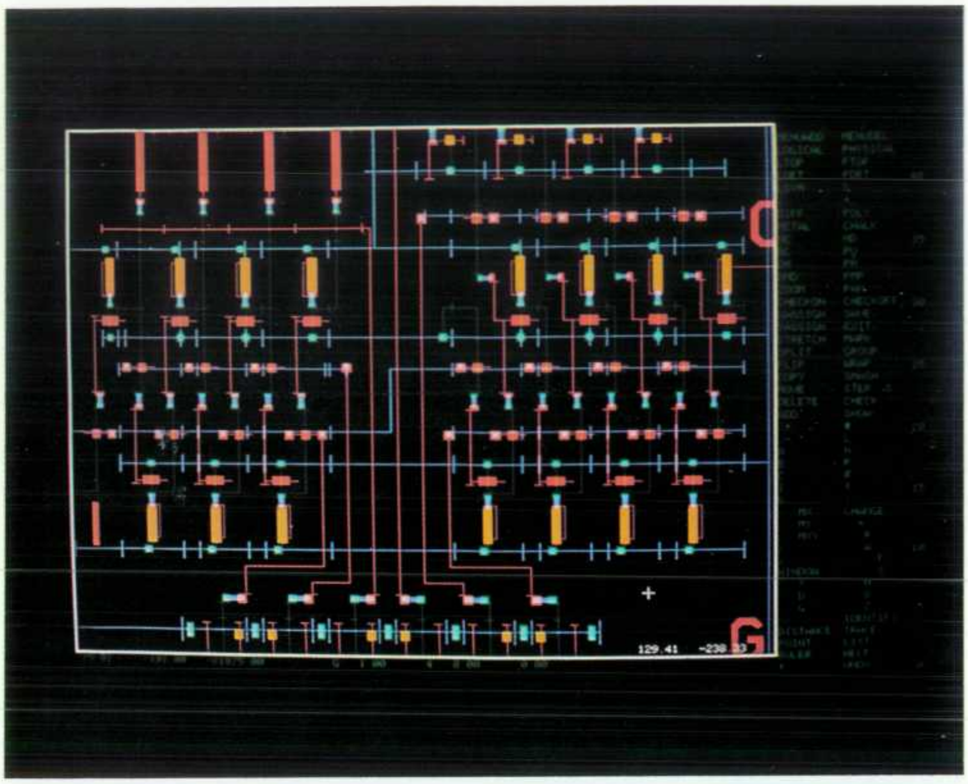


Fig. 2. *A typical IGS graphics display.*

Tablet, and a high-resolution color display. The HP 1000 is used both as an intelligent display processor and as an I/O controller for the tablet, terminal and display. Optionally, any of the following HP plotters can be supported locally at the design station: 2631G, 7221, 9872, or 9876.

Cursor display, scaling, translation, mirroring, rotation, and clipping are some of the functions performed by the HP 1000 display processor. In its role as I/O controller, the HP 1000 is also used to route status information transmitted from the host to the CRT, and to merge tablet and keyboard inputs into a single continuous stream for input to the host. Use of this separate, high-speed processor operating concurrently with the host off-loads complex graphics calculations from the host and frees it for data retrieval and update.

The color display is a high-performance color monitor driven by a 768×512-bit frame buffer with four separate graphics memory planes. Three memory planes (red, blue, and green) are used for data display. A fourth memory plane, always displayed in white, is used for cursor data, for component highlighting, and as a graphic scratchpad. The monitor screen (Fig. 2) is partitioned by software into a command menu area and a graphic data display area.

The IGS system commands are divided into six major categories: 1) executive level commands, 2) process commands, 3) macro commands, 4) graphic edit commands, 5) plot commands, and 6) output commands.

Executive level commands are used to invoke the various subsystems within IGS, to create plots and output files, and to perform chip level maintenance functions such as archiving and retrieving chips, appending devices from other chips, and purging devices. Process commands, which are handled by the process subsystem, are used to establish a working environment for a particular IC process. A typical process file as created within the process subsystem is described in more detail in the next section. Closely related to the process subsystem as a means of tailoring IGS to a particular technology and design methodology is the macro subsystem. Features of this subsystem are also described in more detail below.

Graphic components are created and modified within the edit subsystem using the fourth group of commands. Allowed transformations and modifications on components include rotation, translation, replication, deletion, mirroring and others. The sizes and/or shapes of components may be changed by stretching or shrinking along a vertex or edge. Single components may be split into two or more separate components with a single command. Components may be collected into groups either by selecting individual components or by drawing a boundary around the desired set of components. This makes it easy to shift an entire section of a chip so that, for example, a metal line can be inserted.

The selection of specific components to which the above transformations apply can be restricted in several ways. First, the specific mask levels displayed may be classified as modifiable or nonmodifiable. Second, modification can be restricted to selected component types. Third, edit commands may impose a temporary restriction to a particular component type and/or mask level. In addition, the level of detail, content, and positioning of the display can be manipulated easily through the use of graphic display commands.

The fifth set of commands, available within the plot subsystem, is used to define standard parameter values for off-line plots such as layers to be plotted, pen assignments, and scale factors.

The last set of commands, handled by the output subsystem, is used primarily for creating control files for optical

IC Layout on a Desktop Computer

by Thomas H. Baker

The first installation of the HP Interactive Graphics System (IGS) was within the Integrated Circuit Department of HP's Loveland Instrument Division. IGS soon proved to be a valuable design tool and stimulated interest in providing such design aids to designers within their work areas. Convenient access to a responsive, interactive graphics capability encourages a designer to try out designs as part of a natural thought process. Specific observations about IGS are:

- The designer can work directly with the system. No operator is required to interface to an unfriendly system. The command syntax is relatively easy to teach to new users and a graphics tablet input means that users do not have to be good typists.
- A medium-resolution color display proved adequate for all edit work so that precise multicolor plots on paper are not necessary. The relationships between various mask levels can be distinguished easily. Rapid windowing and zooming make a high-resolution display unnecessary.
- The interaction allows designing directly on the screen, using the system as a sketchpad. When satisfied with a design, the user already has it in a data base. This mode of design has significant advantages compared with a draw-and-digitize design mode.
- The presentation is natural to designers; they can work with shapes and spatial relationships rather than specific coordinates. The designer is not forced to keep a mental picture of the actual format of the data base, but works more with the concepts the data represents at a higher level of comprehension.

The IGS hardware configuration requires that workstations be located within 250 feet of the host HP 3000 Computer. The design engineers at the Loveland facility required a portable, self-contained system compatible with the existing IGS system and low enough in cost so that one could be made available at each designer's desk or work area. If a system is constantly available, it can be used to try out new ideas, to see what a concept looks like.

It is expected that such a local station will be used to design cells and portions of a chip, and to do architecture studies and block diagrams or floorplans. The larger IGS system can be used to integrate entire chips, with pads and scribe masks, and to apply supporting programs such as design rule checks, mask generation, and topology extraction.

The local stand-alone system should have the following features. It should be interactive enough to encourage its use as a sketchpad to eliminate precise paper drawings and digitizing. Symbolic and hierarchical design capability is desired to encourage use of top-down design techniques. It should have enough capacity to design most of the cells of an IC chip, but not necessarily enough to handle an entire chip. The local design station syntax and interaction should duplicate that of IGS as closely as possible to avoid designers having to learn a new language and a new set of reflexes. Its data base should be compatible with IGS so that designs can be transferred back and forth.

The most obvious processor candidate was the HP 9845C Desktop Computer,[1] because it has a high-quality color display and a moderately powerful processor. An internal software development system based on PASCAL was used to write the system. The local design station, known as PIGLET, was implemented with a subset of IGS commands, the ones used for 80-90% of typical work. Much attention was given to maintaining compatibility with IGS. Display speeds are maximized wherever possible; for example, all display calculations are done as integers. Component modifications and medium-size cell displays run at about the same speed in the two systems. However, PIGLET runs somewhat slower in displaying large complex devices.

A PIGLET station (Fig. 1) consists of HP's 9845B/C Desktop Computer, 9885M/S Flexible Disc Drive for mass storage, 9111A Graphics Tablet[2] or 9874A Digitizer[3], and an optional 9872B/C[4] or 7580A Plotter. An HP 7906 Disc Drive is also supported by the system. The operating system, program and all data files reside on a floppy disc. An auxiliary floppy drive may be used to extend the storage area, or to hold a disc with library parts on it.

Components in PIGLET are the same as in IGS: rectangles, polygons, lines with or without width, circles, notes (stick characters), text (block characters), and instances (copies) of another device with rotation, mirror, and scale. Devices, which are a named collection of primitive components, are stored on the floppy disc by name and read into memory as needed. Only one copy of each device is needed in memory, regardless of how many times it is referenced. The actual generation of the device instance is done on the fly at display time. A hierarchy of devices may be nested to any desired depth. Files are easily copied from disc to disc, so libraries of commonly used devices can be maintained. The 9845B/C supports up to 449K bytes of memory, which is a substantial data base for a local station (but not enough for a large chip). For example, PIGLET may have as many as 7000 rectangles in memory at once, divided among as many different devices as desired. This can easily represent 100,000 rectangles on a complete device, by making use of hierarchical design techniques and regular structures. A current restriction with PIGLET is that it does not have the ability to do design rule checks. Ideally design rule checks occur at design time, and do in IGS, but that is beyond the capability of the local station at this time. This means that feedback of design rule violations is separated in time and space from the actual design phase.

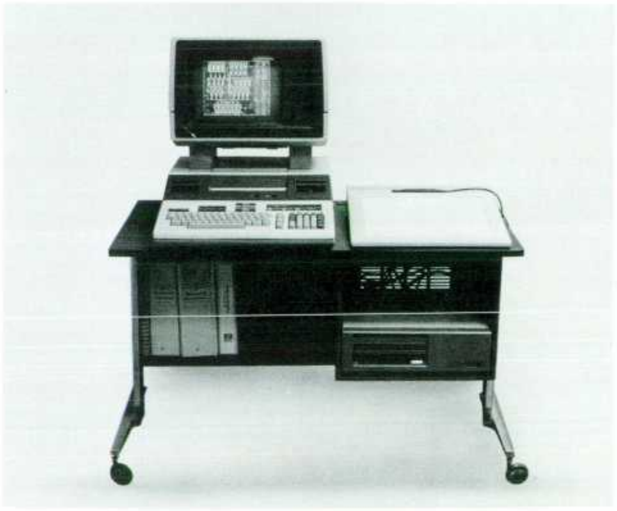


Fig. 1. *The PIGLET station provides a compact interactive graphics design system for IC and printed-circuit layouts, schematic drawings, plant layout, and illustrations. It supports the same symbolic design capability that is available with IGS.*

Commands for PIGLET may be entered from keyboard, taken from a disc file, or selected from a menu that is very similar to that of IGS. Users may change, store and recall menus at will; thus, the displayed commands can be customized for particular needs. Process information, such as display color and plotter pen number for each defined mask level, is kept in a file. PIGLET supports a compatible macrocapability with parameters, so the user can write a personal command set and can reduce commonly used command strings to a single item. PIGLET supports the same symbolic design capability that is available in IGS. Hard copy of the display or color plots is available for device documentation. Data is transferred to the IGS system in an ASCII format, using a utility program that allows the 9845B/C to become an HP 3000 terminal.

The presence of this small but powerful graphics tool has stimulated many people in other areas to consider its potential impact on their jobs. Applications for which interactive graphics was previously considered inappropriate or not considered at all became candidates for some effective computer assistance. The command set defined for IGS is general enough to serve a broader base than just for IC design. The first use of PIGLET for work other than IC layout was in the printed circuit board area. Most printed-circuit-board design people have not had access to interactive graphics, so they were delighted at the ability to lay out a board on a terminal rather than penciling in interconnects between paper doll devices. A library of commonly used parts was quickly developed along with an interface to a photoplotter. Many engineers use the ability to do filled plots to get rapid turnaround on prototype boards by plotting the data on mylar at one or two times real size and using the plot to generate artwork directly, thus avoiding a queue on the photoplotter. Other uses of the graphics capabilities of PIGLET are schematic drawings, manual illustrations, overhead slide preparation, and plant layout.

References:

1. J. Frost and W. Hale, "Color Enhances Computer Graphics System," Hewlett-Packard Journal, December 1980.
2. D. Stavely, "A High-Quality Low-Cost Graphics Tablet," Hewlett-Packard Journal, January 1981.
3. F. Carau, "Easy-to-Use, High-Resolution Digitizer Increases Operator Efficiency," Hewlett-Packard Journal, December 1977.
4. L. Brunetti, "A New Family of Intelligent Multicolor X-Y Plotters," Hewlett-Packard Journal, September 1977.

Thomas H. Baker

Tom Baker received his SB and SM degrees in electrical engineering from Massachusetts Institute of Technology in 1964 and 1966. He joined HP in 1966 and has worked on several instrument designs, set up laser trim production facilities for thin-film resistors, and served as a support and resource person for IC designers. He's currently researching concepts for local design stations and interactive graphics. Tom is a member of IEEE and a registered professional engineer in Colorado. Recently he spent a year on HP's faculty loan program, teaching at North Carolina A&T State University. Born in Knoxville, Tennessee, he is married, has four children, and lives in Loveland, Colorado. He's active in his church and enjoys backpacking, "non-technical" mountain climbing, and bicycling.

and electron beam pattern generator tapes or for access to a full chip design rule checker.

Several of these systems are described in more detail below by means of examples. To provide cohesion for the material presented below, a five-mask NMOS* process (diffusion, implant, polysilicon, contact and metal) as described in Mead and Conway's Introduction to VLSI Systems[2] is used throughout as the basis for examples.

Process Description

An IGS process file created via process subsystem commands is shown in Fig. 3. The information consists of mask layer, type, name, color, and design rule declarations.

Several considerations influenced the decisions as to the number and types of masks to be declared. Different sets of mask layers are required for the detailed definition of a device and for its symbolic definition. Further, a subset of the detailed mask set consisting of those masks used for interconnecting devices or their symbols is distinguished for reasons that will become more apparent later.

Two of the five masks in the process at hand, implant and contact, are used only in the detailed definition of devices and are declared to be of type DETail. Three masks (diffusion, polysilicon, and metal) are used both in the detailed definition of devices and for the interconnection of devices or their symbols and therefore are declared to be of type INTerconnect. The key concept here is that even when device symbols are displayed it is both desirable and necessary that they be interconnected. Thus INTerconnect masks

*N-channel metal-oxide semiconductor.

```
PROCESS FILE:       PROVLSI.VLSIDEMO.CAD
CREATION DATE:      JANUARY 15, 1981
LAST MODIFIED:      JANUARY 31, 1981    1:56 PM

RESOLUTION:         100 UNITS PER UM

MASK INFORMATION

                                      EXPL.  MIN.  MIN.   COMM
MSK  TYPE  MNEMONIC  COLOR            BUMP   AREA  WIDTH  MASK

1    INT   DIFUSION  GREEN            YES     4     4      2
2    DET   IMPLANT   YELLOW DOT       YES    36     6
3    INT   POLYSIL   RED              YES     4     2      4
4    DET   CONTACT   WHITE            YES     4     2     1,3,5
5    INT   METAL     BLUE             YES     9     3      4
11   SYM   DIFFCON   GREEN            NO
12   SYM   IMPLSYM   YELLOW           NO
13   SYM   POLYCON   RED              NO
15   SYM   METALCON  BLUE             NO
20   SYM   OUTLINE   WHITE            NO

IMPLICIT BUMPER INFORMATION

REFERENCE            RELATIVE
MASK       DISTANCE  MASK

1 DIFUSION    3      1 DIFUSION
              1      3 POLYSIL
3 POLYSIL     1      1 DIFUSION
              2      3 POLYSIL
4 CONTACT     2      4 CONTACT
5 METAL       3      5 METAL
```

Fig. 3. *NMOS process file for symbolic design.*

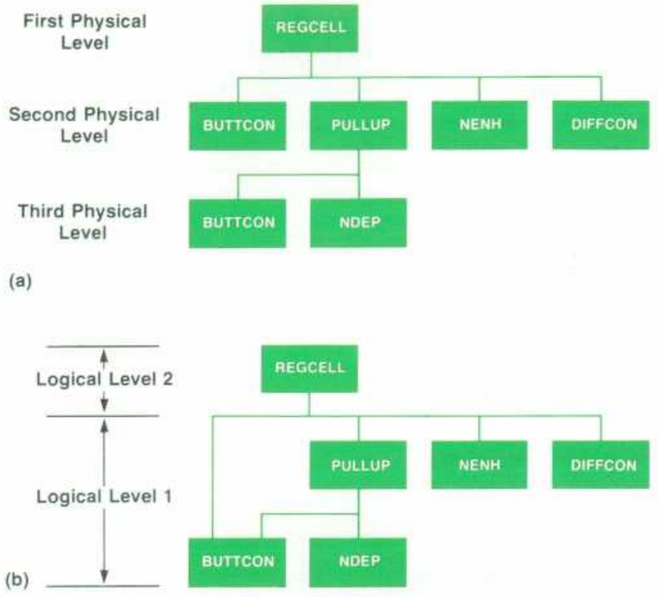


Fig. 4. *(a) Physical nesting as viewed from REGCELL. (b) Logical nesting as viewed from REGCELL.*

are characterized by the property that they are displayable in both a detail display mode and a symbolic display mode.

The second set of masks, layers 11, 12, 13, and 15, are SYMbolic masks corresponding to DETail masks 1, 2, 3 and 5, respectively. These SYMbolic mask layers are used for building stick diagrams, logic diagrams, block diagrams, and others that present more simplified, abstract design views when displayed. Later examples illustrate the use of these techniques more fully. The final mask layer, 20, is also SYMbolic and is used to outline and label functional block symbols where the external boundary and functional name are important.

In this example, the choice of colors is in conformance with Mead and Conway.[2] Explicit bumpers, to be described later, are allowed for masks 1 through 5, while minimum required widths, areas and intramask and intermask spacings are also defined.

Macrocommands and Logical Nesting

IGS provides a flexible user-defined macrocommand capability, which can greatly simplify the implementation of a symbolic design methodology. Before describing the interaction between macrocommands, primitive IGS commands and the process file, it is necessary to describe, as illustrated in Fig. 4, the concept of logical cell nesting as opposed to physical cell nesting. Here REGCELL is the device currently being edited so that physically all display references are levels below REGCELL. The device REGCELL is a shift register cell including instances of PULLUP, NENH, DIFFCON and BUTTCON. When viewed from REGCELL, one instance of BUTTCON is physically nested at the second level down and another is physically nested at the third level down. The problem created by this situation in a symbolic design environment is that if symbols are to retain a consistent level of representation, it is desirable to see all occurrences of a given device or no occurrence. For this example, if the display is controlled only on the basis of physical nesting, then BUTTCON will be seen once in outline form in REGCELL if only the first level is displayed, once in outline and once in detail if the second level down is also displayed, and finally, twice in detail if all three levels are requested.

The basic idea behind logical nesting is to have the designer define a logical hierarchy of device instances that overlays the physical hierarchy. For the example of Fig. 4, the devices BUTTCON, DIFFCON, NENH and NDEP are all low-level devices and hence can be arbitrarily assigned a logical function level of 1, the lowest possible level. The device REGCELL, which provides a true logical function, is therefore assigned a higher functional level of 2. What about PULLUP? The device PULLUP typically serves as a two-terminal active load for inverters and other circuits,

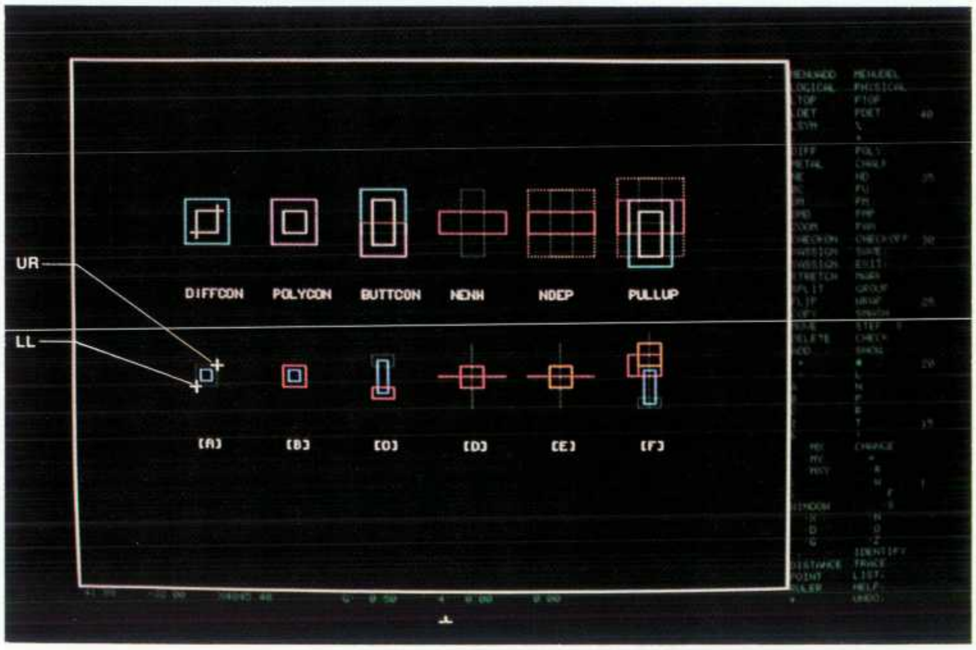


Fig. 5. *IGS primitive devices. Reference crosses are shown on DIFFCON.*

and therefore is also a primitive device. Thus it is assigned a logical function level of 1, the same as its constituent parts.

To a degree, level control is maintained and enforced by IGS. Level 2 devices cannot be added to level 1 devices but can be added to other level 2 or higher-level devices. Thus, there is no conflict in assigning PULLUP the same logical function level as its constituent parts. For the example of Fig. 4, in a logical nesting sense, display is from REGCELL down to logic level 2 devices (i.e., REGCELL itself) or down to logic level 1 devices, the primitive transistors and contacts. As a result, the desired effect of having no instances of BUTTCON or all instances displayed is achieved. As a further point, when the symbols for level 1 devices are displayed, the INTerconnect masks for all devices level 2 and above are automatically displayed to indicate how the symbols are interconnected. This aspect illustrates the true significance of INTerconnect masks and the reason for distinguishing them as a subset of the detail masks.

To summarize, the advantage of working with logical nesting levels is the preservation of a consistent display and multiple symbolic representations. The multiple representations derive from the fact that each time the logical function level is increased, a new symbolic representation may be defined.

With the concept of logical nesting now in mind, the use of the IGS macrocommand facility to provide the basic mechanism for moving back and forth between various display representations in logical display mode can now be described. Two macrocommands, DETAIL and SYMBOL, can be defined as follows:

```
DEFINE DETAIL <LV '1'> ' WINDOW  :D10,<LV> ';
DEFINE SYMBOL <LV '1'> 'WINDOW  :S<LV>,<LV> ';
```

The DETAIL macrocommand uses a WINDOW command to display detail from an arbitrary logical function level of 10 down through a logical function level set by the parameter LV, which has a default value of 1.

The SYMBOL macrocommand is similar to the DETAIL macrocommand. The WINDOW command establishes the logical function level for which symbols are to be displayed. In particular, in a symbolic display mode, it is presumed that new symbols are defined at each new logical function level starting from stick-like symbols for level 1 devices and proceeding to block-like symbols for level 2 devices and above. Most of the time, it is desirable to view only a single level of symbols as well as all interconnections between those symbols. Thus for the SYMBOL macrocommand, the parameter LV is used for both arguments of :S in the WINDOW command. The interconnections between symbols are presumed to be on masks of type INTerconnect and are automatically displayed for all logical device levels from that of the device currently being edited down through the level above that whose symbols are being displayed.

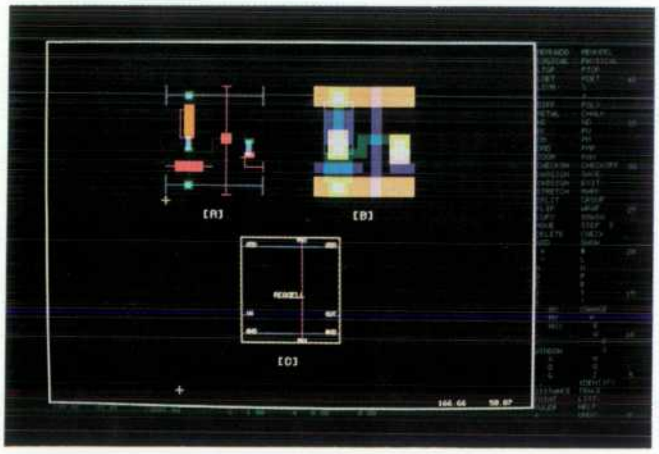


Fig. 6. *Stick, detailed, and block symbolic views of a shift register cell, produced using the SYMBOL and DETAIL macrocommands.*

Incremental Design Rule Checking

Several types of design rules may be specified and checked for in IGS. Design rules are either entered in the IGS process subsystem (implicit bumpers) or directly into device definitions (explicit bumpers). Both types are applied incrementally in the edit subsystem as the user modifies the data base.

Design rule checks in IGS fall into two basic types: design rules that impose constraints on component geometry, and design rules that impose constraints on component topology (i.e., placement of components). Constraints of the former type include minimum area and width specifications for mask layers. Algorithms for checking geometric components against these constraints are well described in IC design literature.

Topological design constraints are specified in IGS through the use of bumpers. Bumpers are pseudo-layers which specify regions that may not be overlapped by actual layers in the chip. Each actual layer m has a corresponding bumper layer −m. When a component, for example a metal line on mask 5, is placed in a cell, it may not overlap components on its bumper layer, mask −5. In IGS, bumpers are used primarily to enforce minimum spacing rules. Bumpers are not intended to take the place of complete off-line design rule checking procedures. Rather, they are intended to shorten the design cycle by detecting a large class of errors during the interactive phases of the design.

Bumpers are created by the designer in two ways. First, they may be created explicitly by adding geometric components on bumper layers. Components created in this fashion are termed explicit bumpers. Second, bumpers may be created as a result of the mask layer minimum spacing rules specified in the IGS process subsystem, as indicated in Fig. 3. Conceptually, implicit bumper components are automaticaly created on negative reference mask layers whenever components are added or modified on positive mask layers. The bumper components are automatically oversized by the minimum distances, effectively implementing the spacing rules. Spacing rules are ignored whenever components touch or overlap so that components may be electrically connected.

The decision to define topological constraints in terms of geometric components has the advantage that bumpers can be treated like any other component in IGS. Not only does this allow manipulation with the standard set of commands, but it also allows parameterization. Therefore, stretchable cells can have stretchable bumpers. In addition, the ability of the user to create bumpers explicitly or have them created based on a set of global rules widens their range of potential application. If any explicit bumper is defined in a cell, then IGS ignores all corresponding implicit bumpers with respect to that cell and any included cells. This characteristic allows the designer to override the globally defined spacing rules with custom rules when necessary. Implicit bumpers are normally defined with a conservative set of design rules. Typically, they find their greatest application when routing interconnects between cells and in intracell designs where cost considerations do not justify optimal designs. Explicit bumpers, on the other hand, may be employed to tailor design rules in cells where more aggressive design approaches have been taken to increase speed, to save area, or for some other reason.

Primitive Devices and Symbols

The following paragraphs describe the creation of detail and symbols for level 1 primitive devices as illustrated in Fig. 5. The first of these devices, DIFFCON, a diffusion-to-metal contact, consists of a 4λ diffusion rectangle on mask 1, a 2λ contact rectangle on mask 4, and a 4λ metal rectangle on mask 5 (λ is the minimum increment for defining dimensions in a design). The symbol defined for this is a 2λ rectangle on mask 11, the symbolic diffusion mask, and a 1λ rectangle on mask 15, the symbolic metal mask. This choice of symbol was based on the following observations. In symbolic display mode, lines of any arbitrary width are displayed as centerlines. The minimum width for a diffusion line is 2λ. Thus in symbolic mode, any diffusion line that touches the symbolic metal rectangle is also a valid, design-rule-correct connection.

As previously mentioned, IGS provides for the definition of stretchable or parameterized cells. For a cell such as DIFFCON, two reference crosses arbitrarily labeled LL for lower-left and UR for upper-right can be declared as shown in Fig. 5a. The X and/or Y coordinates of any vertex of any rectangle, line, polygon, or circle on any mask can be assigned to any declared reference cross. By default, all vertices of all primitives are initially assigned to the device origin. In DIFFCON, the X and Y coordinates of the lower-left vertices of each rectangle on each mask are assigned to the X and Y coordinates of the reference cross LL. The upper-right vertex coordinates are assigned in similar fashion to UR. The result is that whenever an instance of DIFFCON is used in a higher-level device, its size, both detail and symbol, may be parametrically stretched as needed by appropriately locating the reference crosses LL and UR.

The polysilicon-to-metal contact, POLYCON, is similar to DIFFCON, as shown in Fig. 5b. The butting contact of Fig. 5c is described in detail in Mead and Conway.[2]

In practice, an enhancement-mode pull-down transistor can be formed simply by crossing polysilicon over diffusion. However, to be sure that necessary minimum widths and extensions are maintained and that all design rules can be checked, it is useful to define an explicit stretchable cell, NENH. As shown in Fig. 5d, it consists of a 2λ-by-6λ horizontal polysilicon rectangle, which serves as the gate contact, and a similar vertical rectangle on the diffusion mask. Both rectangles are stretchable, with their lower-left and upper-right vertices assigned to two reference crosses, LL and UR, located at the corners of the active region (i.e., the region of intersection of the two rectangles). The symbolic representation is simply a symbolic zero-width line on mask 13 horizontally crossing a vertical symbolic zero-width line on mask 11 and a 2λ-by-2λ rectangle, also on mask 13. The resulting symbol is appropriately assigned to the reference crosses.

A depletion-mode NMOS transistor NDEP is constructed like the enhancement-mode transistor, as shown in Fig. 5e. Note that the size of the symbolic rectangles accurately reflects the size of the active gate area of the transistors. The final level 1 primitive device, PULLUP, is a depletion-mode NMOS transistor combined with a butting contact for use as a pullup or active load in gates and inverters. An actual default-sized instance of the butting contact device BUTTCON is used together with primitive rectangles and lines as in the device NDEP to provide direct assignment of vertices to achieve stretchability. An additional zero-width line is used in the symbolic view to reflect the shorting of gate to source, as shown in Fig. 5f.

Building a Shift-Register Cell and Its Symbol

A simple, single-bit shift register can be constructed from a chain of inverters gated by pass transistors driven by two alternate, nonoverlapping clock signals. Fig. 6a shows a symbolic stick representation and Fig. 6b a detailed representation for a basic shift register cell, REGCELL, assumed to be a level 2 device. The inverter consists of a depletion-mode MOS transistor with its gate and source shorted via a butting contact, the stretchable PULLUP device described previously, and an enhancement-mode MOS transistor, the NENH device, also described previously, used as a pulldown. The diffusion-to-metal contact, DIFFCON, is used to connect the structure to power and ground lines.The output of the inverter is taken from the butting contact in diffusion, and after being gated by a polysilicon clock line, is returned back to polysilicon with a second butting contact. The power, ground, and output signal lines of one cell align with the power, ground, and input signal lines of the next. Thus the cells can simply be concatenated for as many bits as desired with two cells required for each bit.

Fig. 6c shows a new block symbolic view of the same shift register cell consisting of a boundary rectangle on mask 20 together with various labels for signal names including power, ground, clock, input and output. Two zero-width lines on the symbolic metal mask (15) and a single zero-width line on the symbolic polysilicon mask (13) are extended completely through the cell boundary to indicate the direct, continuous path of these signals through the actual device. Short stubs, also on mask 15, are used to indicate the exact location of the input and output signals. This symbolic block view is displayed whenever level 2 cells are requested via the SYMBOL macrocommand. It contains the essential information required to use this device to create still higher-level functional blocks such as a complete shift register array. It conveys more clearly the functional organization of that higher-level cell. Finally, it contains less graphic detail, thereby resulting in a faster display.

IGS in Practice

The same principles used in the design of a symbolic representation of the simple shift register cell can be used at successively higher levels of chip design, the goal being to strike a balance between abstraction for clarity and detail

Diane F. Bracken

Diane Bracken was a member of the original IGS design team and is now IGS project manager. She received her BSEE degree (computer science option) from Massachusetts Institute of Technology in 1975 and her MSEE degree in computer architecture from Stanford University in 1977. She has done image processing using small computers, and is co-author of two papers on IGS and interactive design techniques. Her responsibilities include liaison with MIT in HP's university grants program. Born in North Tonawanda, New York, she now lives in Palo Alto, California and enjoys hiking, ice skating, swimming, and cross-country skiing.

William J. McCalla

Bill McCalla has been with HP since 1976 as a member of the technical staff and project manager for IGS development. He's now section manager for artwork systems and graphics with HP's design aids group. A native of Tustin, California, he received his BS, MS, and PhD degrees in electrical engineering from the University of California at Berkeley in 1967, 1968, and 1972. He came to HP with ten years of experience in circuit simulation and numerical analysis. He has authored more than a dozen papers on circuit simulation and IC layout and has conducted graduate seminars in circuit simulation at Berkeley and Stanford. He's a senior member of IEEE and a member of ACM, and serves on the IEEE CANDE Committee. Bill is married, has a growing family, and enjoys photography, running, and guitar. He lives in Los Altos, California.

for accuracy. To illustrate the concept of structured, hierarchial, symbolic design, two views of one of the early chips designed with IGS are shown on the cover of this issue. In the background, a plot as might be seen from a more conventional IC layout system is shown. This plot includes so much detail that any perception of function or organization is lost. By contrast, the symbolic view from which a major part of the design was done is shown in the foreground. Here, function and organization are shown much more clearly.

Acknowledgments

It is a pleasure to acknowledge the many contributors to the success of IGS. These include Beatriz Infante, Ellis Cohen, Sam Yamakoshi, Kamran Elahian, Josie Diehl, Lind Gee, Dave Hoffman, Larry Dwyer, Don Van Deventer, Randy Smith, David Henke and Bob Rogers. In the area of training, documentation and customer support, Jim Lipman, Larry Smith, David Sternlicht and Sam Boles have each contributed significantly. Finally, the contribution of Merrill Brooksby in providing the support and resources to get the job done as well as the freedom to exercise and stretch the creative talents of the project team members is greatly appreciated.

References

1. B. Infante, D. Bracken, W. McCalla, S. Yamakoshi, and E. Cohen, "An Interactive Graphics System for Design of Integrated Circuits," 15th Design Automation Conference Proceedings, June 1978, pp. 182-187.
2. C. Mead and L. Conway, "Introduction to VLSI Systems," Addison-Wesley, 1980.

VLSI Design and Artwork Verification

by Michael G. Tucker and William J. Haydamack

THE COMPLETION OF ARTWORK DESIGN does not signal the end of the design activity for an integrated circuit. Geometric and circuit-level checking must still be done to verify proper operation of the circuit. At HP, the circuit level check is done using HP's EXTRACT system, and the geometrical check is done using the HP Design Rule Checking system.

EXTRACT

Artwork design for today's VLSI circuits is a complicated and therefore error-prone process. With the TTL breadboards of years past, the designer could sit down with an oscilloscope and probe the circuit to find bugs. Not so with

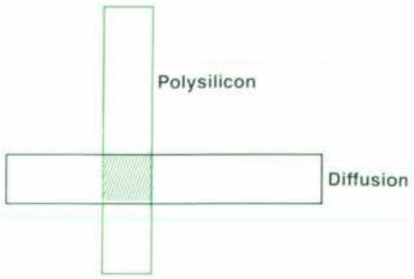


Fig. 1. *Isolation of an NMOS transistor gate.*

VLSI circuits. Instead, the EXTRACT system allows the designer to extract from the artwork a circuit schematic which can then be verified functionally using a circuit simulator.

EXTRACT is usable with any MOS process. The user must provide instructions to specify exact operations to be performed for each process. Circuit extraction for all processes, however, consists of four steps:

- Electrical feature isolation. For example, Fig. 1 shows a transistor formed using a popular NMOS process. The transistor itself is not drawn on any layer, but is implied by the crossing of diffusion with polysilicon. A gate layer is created by logically ANDing the area of the two input layers (see "Design Rule Checking" below).
- Continuity determination. All electrical artwork is converted to a polygon representation. Evaluation of contact layers is used to determine interlayer connections.
- Transistor and capacitance extraction. This step can create output plots of node identification, of transistor sizes, and of any electrically nonsensical structures, which signify design errors.
- Simulator file creation. This circuit description is textual and is not as easily readable as the original graphical description. It can be directly simulated, however, using SPICE (see article, page 12).

Fig. 2 shows the process of extracting a full-bit adder in an NMOS process. The electrically significant layers are shown in Fig. 2a. Fig. 2b shows all the transistors in the circuit, along with the length and width dimensions of each. Fig. 2c is the textual circuit description output created by EXTRACT.

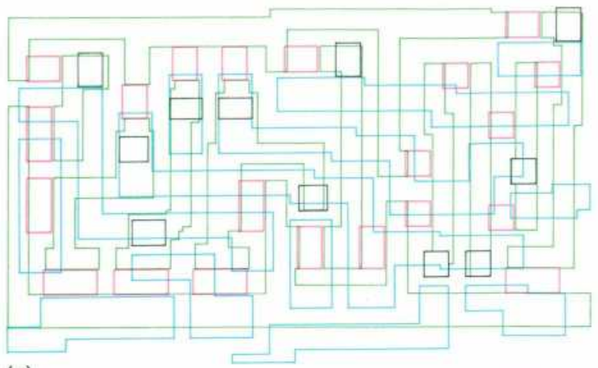
Because small cells like those shown in Fig. 2 can be extracted in a few minutes, the designer is able to verify the functionality of each piece of a circuit at the time of design. While EXTRACT can create schematics from full-chip artwork, this is much more time-consuming.

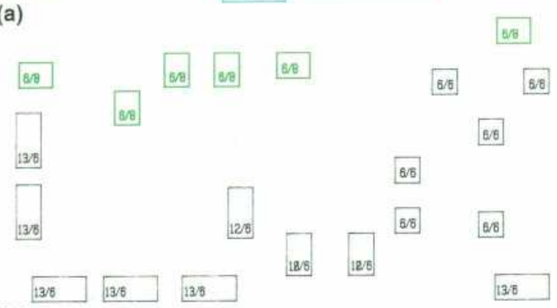
An important design goal of EXTRACT was that extraction of large circuits should not require excessive amounts of main memory. This goal has been achieved by geometrically partitioning the artwork data before continuity extraction. The partitioning is done along a two-dimensional grid, so that the circuit is divided into a checkerboard-like set of artwork squares. Because the grid size is user-adjustable, arbitrarily dense circuits can be extracted within a fixed amount of main memory by decreasing the square size. For less dense circuits, use of a larger square size reduces computer time, since fewer interpartition figures must be processed.

Design Rule Checking

Design rule verification is another area where computers have been found very useful. Each IC fabrication facility publishes a list of minimum spacings and widths an IC process can reliably fabricate. These design rules are observed by the artwork designer wherever possible, but here too, mistakes are made because of the complexity of the task. The HP design rule checking (DRC) system checks artwork for compliance with these process design rules.

The HP DRC system is flexible enough to allow checking of all IC processes used within HP, plus numerous thin-film and other microcircuit fabrication processes. The key to this flexibility is a set of primitive operations and tests which can be specified using a DRC command language. The


(a)



(b)

```
MEn1     7  2  1 0 Enh W=   6.00U L=   6.00U
MEn2     9  2  8 0 Enh W=   6.00U L=   6.00U
MEn3     8  4  7 0 Enh W=   6.00U L=   6.00U
MEn4    13 10  3 0 Enh W=  13.00U L=   6.00U
MEn5    12  4  9 0 Enh W=   6.00U L=   6.00U
MEn6    13  6  9 0 Enh W=   6.00U L=   6.00U
MEn7     8  6  7 0 Enh W=   6.00U L=   6.00U
MEn8     4  3 13 0 Enh W=  12.00U L=   6.00U
MEn9    13 10  6 0 Enh W=  13.00U L=   6.00U
MEn10    2  6 13 0 Enh W=  10.00U L=   6.00U
MEn11   13  4  2 0 Enh W=  10.00U L=   6.00U
MEn12    1  8 13 0 Enh W=  13.00U L=   6.00U
MEn13    4  5 13 0 Enh W=  13.00U L=   6.00U
MEn14    5 11 13 0 Enh W=  13.00U L=   6.00U
MEn15    6 11 13 0 Enh W=  13.00U L=   6.00U
MDe1    12  1  1 0 Depl     W=   6.00U L=   8.00U
MDe2    12  2  2 0 Depl     W=   6.00U L=   8.00U
MDe3    12  4  4 0 Depl     W=   6.00U L=   8.00U
MDe4    12  5  5 0 Depl     W=   6.00U L=   8.00U
MDe5    12  3  3 0 Depl     W=   6.00U L=   8.00U
MDe6    12  6  6 0 Depl     W=   6.00U L=   8.00U
C1       1 0      .0367P
C2       2 0      .0447P
C3       3 0      .0382P
C4       4 0      .0755P
C5       5 0      .0423P
C6       6 0      .0621P
C7       7 0      .0203P
C8       8 0      .0272P
C9       9 0      .0336P
C10     10 0      .0112P
C11     11 0      .0135P
C12     12 0      .0812P
C13     13 0      .1291P
```
(c)

Fig. 2. *Extraction of a NMOS full-bit adder. (a) Input artwork. (b) Transistors shown with width and length dimensions. (c) Circuit schematic generated for input to SPICE simulator.*

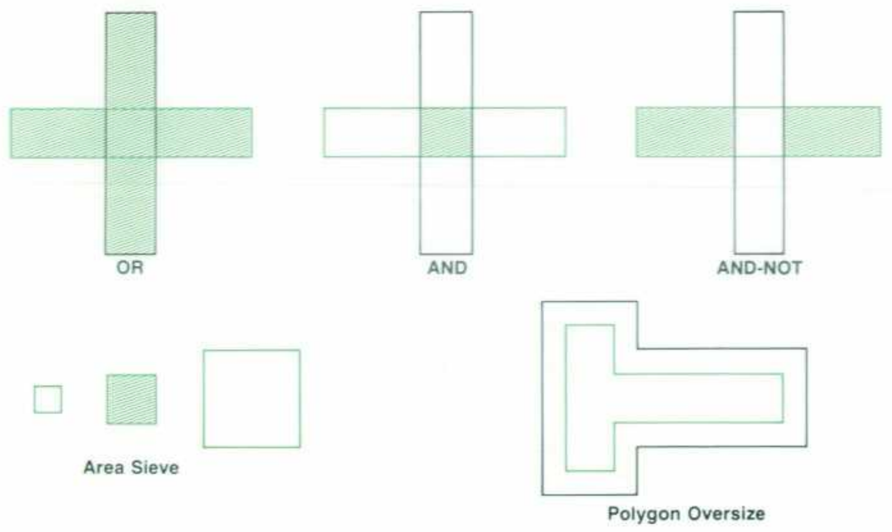


Fig. 3. *Design rule checking (DRC) operations.*

primitive operations available are shown in Fig. 3. Each operation reads one or two input layers and produces a single output layer. Operations can therefore be combined in sequence to isolate the features of the artwork that are to be tested. Feature isolation is necessary to allow tests to be made with one of the three test operations shown in Fig. 4.

The translation of process design rules into a sequence of operations and tests is often a challenging task. Fig. 5 shows the sequence of operations and tests that can be used to test for compliance with one design rule in a polysilicon gate NMOS process. An IC process may have over 25 of these rules, and command files usually contain 30-150 commands.

The test commands indicate errors by highlighting the line segments that cause the error. Errors can be plotted or can be viewed on an IGS work station. For IGS viewing each DRC error is automatically matched with the appropriate IGS device, and is superimposed on the displayed device artwork.

The DRC system allows checking whole circuits or small pieces of circuits. With both DRC and EXTRACT, the advantage of checking small pieces is that mistakes can be found while they can be easily corrected. A final check of the entire circuit is also desirable, however, to find errors made while fitting the pieces together. This final check can involve more than 10^6 separate geometric figures for a VLSI circuit, and efficient algorithms are required if computer run times are to be kept reasonable. The algorithms used in most of the DRC operations and tests are based on an efficient line-segment representation. Line segments representing the polygon outline of the artwork are sorted in both X and Y dimensions. For efficiency, no horizontal line segments are shown, since their existence can be implied from the remaining line segments. Fig. 6 shows how a polygon outline is represented using the line-segment representation.

The operations built around this line-segment representation process the data in bands. A band includes all line segments that intersect a horizontal line drawn across the circuit. This horizontal line slowly scans the circuit from bottom to top. To allow fast processing, all line segments in the band being processed must be contained in main memory.

The memory requirements of this algorithm vary as the square root of the circuit complexity. If circuit complexity doubles each year as predicted by Gordon Moore of Intel,

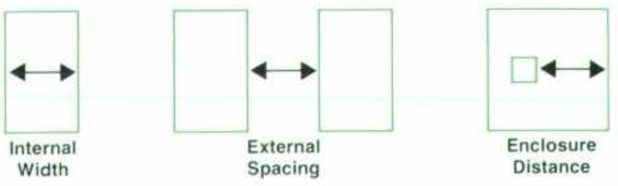


Fig. 4. *DRC tests.*

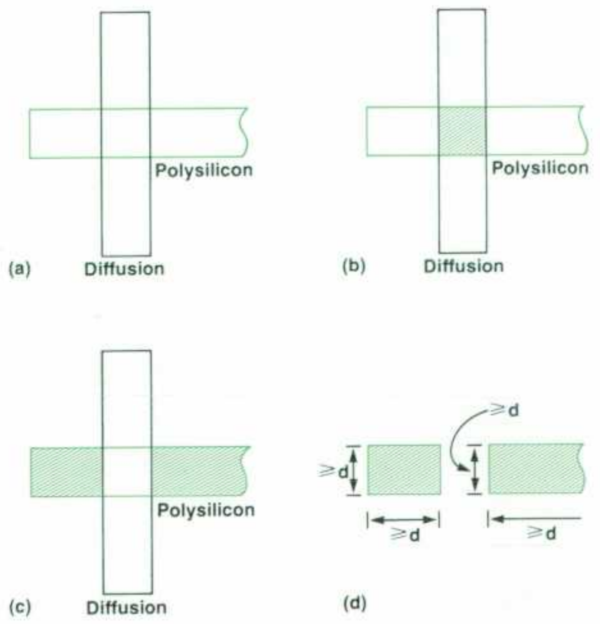


Fig. 5. *An example of rule isolation and test. One design rule for an NMOS transistor gate is that polysilicon must overlap diffusion by a distance d. (a) Original artwork. (b) Polysilicon is ANDed with diffusion to create the active area. (b) Polysilicon is AND-NOTed with the active area to create the nonactive polysilicon. (d) The internal width of the nonactive polysilicon is tested to be sure that it exceeds the minimum d.*

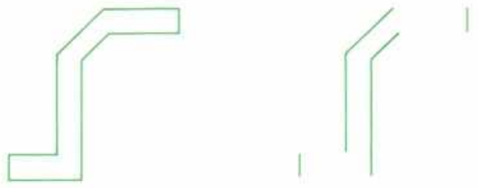


Fig. 6. *DRC data representation. Polygon outlines are represented using an efficient line-segment method.*

DRC memory requirements will double every two years. Today's commonly used computer architectures have addressing limitations from 2^{16} to 2^{24} bytes. The DRC programs require 2^{19} bytes to check the largest circuits seen today. Clearly, the unbounded memory requirements of the current system will eventually force development of new algorithms that allow some type of circuit partitioning.

Mask Modification

The operations used by the DRC system can also be used to create modified artwork layers for use in chip fabrication. Uses of this capability include: 1) performance enhancement, where minimum process geometries have shrunk, 2) layout simplification, in which artwork for one or more layers is algorithmically generated from other layers, and 3) process adjustments, in which feature sizes are biased to compensate for process parameters.

Mask modification is operationally part of the DRC system, and uses the same command language. The additional function required is the ability to convert line-segmented data into rectangles that can be sent to a reticle pattern generator. This is accomplished by 1) converting line-segmented data into closed polygons, 2) splitting polygons with more than 100 edges, and 3) generating rectangles to fill each polygon, called "boxing." Since boxing of polygons is computationally complex, the addition of step 2 speeds up the conversion considerably.

Fig. 7 shows an application of performance enhancement in an NMOS process. The metal is to be made thinner to decrease capacitance, but the contact enclosure distance must not be decreased. The sequence of steps used in this operation is:

INPUT LAYER 5	Contacts
INPUT LAYER 6	Metal
OVERSIZE LAYER 6 BY −1.25 MICRON TO LAYER 10	Shrunk metal
OVERSIZE LAYER 5 BY 1 MICRON TO LAYER 11	Preserve metal around contact
OR LAYER 10 WITH LAYER 11 TO LAYER I2	Combine to produce new metal

The most common use of mask modification has proved to be for process adjustments. The ability to do artwork modification has allowed artwork design to be done for fabrication processes that are not yet completely defined.

Summary

Computer aids have proved very useful both in verifying IC artwork and in modifying it. Since the first of these tools

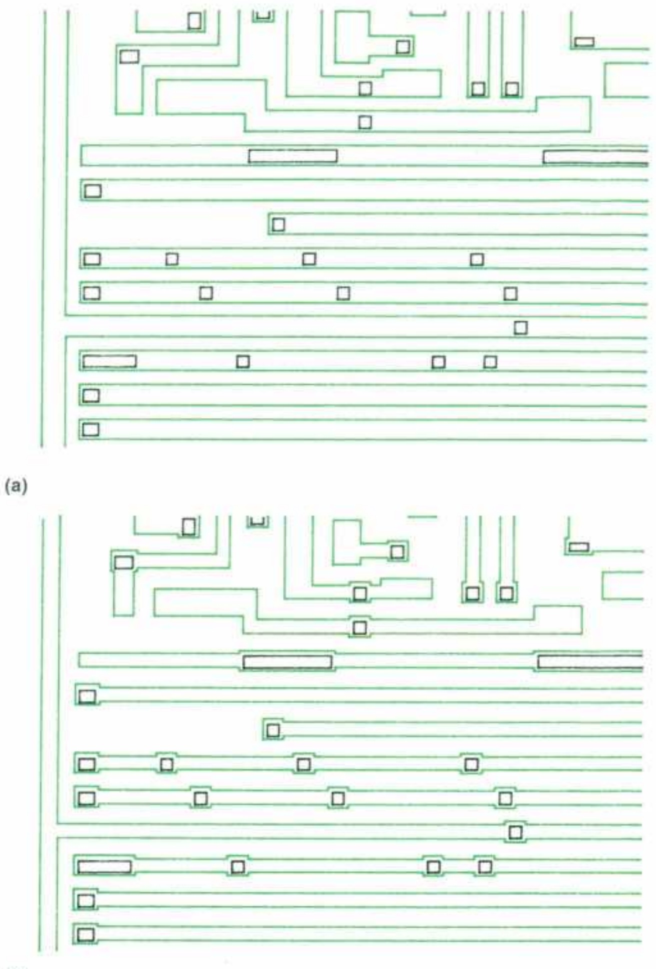


Fig. 7. *Mask modification using the DRC system. (a) Original circuit (metal shown in color, contacts in black). (b) Circuit with undersized metal except at contacts.*

Michael G. Tucker

Mike Tucker is a project manager for design verification with HP's design aids group. He's been a member of that group since 1976. He's also a member of the IEEE Computer Society and author of a paper on mask modification. A native of Davenport, Iowa, he received his BS degree in electrical engineering from Iowa State University in 1974 and his MSEE degree from Stanford University in 1975. He joined HP in 1975 as a calculator production engineer. Mike is married and lives in Los Altos, California. He enjoys photography, sings with the Schola Cantorum, a 150-voice symphonic choir, and has been known to design and build stereo equipment.

became available, IC complexity has increased to the point that the use of these tools is now mandatory. At the same time, the largest existing circuits strain the available computer resources. Clearly, continuing tool development will be needed as circuit complexity continues upward.

Acknowledgments

The authors would like to acknowledge the contributions made by Lou Scheffer, who developed EXTRACT, and Dave Alexander of NCA Corporation, who wrote the original version of DRC.

University and Industrial Cooperation for VLSI

by Merrill W. Brooksby and Patricia L. Castro

IN 1975 HEWLETT-PACKARD established an industrial grant with Stanford University. The major objective was to fund university efforts that could become effective resources for HP integrated circuit facilities. To accomplish this, research and development was divided into two major functions: product R&D and facility R&D. Product R&D is directly related to company products sold in the marketplace. Facility R&D enhances engineering efficiency or extends engineering capability in such areas as:

- Characterization methods
- Analysis methods
- Process control R&D
- Design aids standards
- Modeling.

Facility R&D can be non-proprietary and can provide an excellent opportunity for university assistance. Universities have always done an excellent job in providing well-trained engineers and improved engineering methods, so this is very consistent with their basic objectives.

With the above concepts in mind, HP extended the program with Stanford to three other universities; the University of California at Berkeley (UC Berkeley), the University of Illinois, and the California Institute of Technology (Caltech). Long-term programs were organized with the success of each project measured by how well the results were transferred to various HP divisions. Lengthy written reports were not required and the universities were treated very much as if they were a part of HP. The programs were based on a simple one-page written Project Data Sheet and had regular project reviews by interested engineers from Hewlett-Packard's Corporate Engineering, HP Laboratories and operating divisions. This form of peer review is exceptionally valuable because it is immediate and the reviewers are interested in seeing the project produce useful results. The difference in technical perspective is every bit as invigorating for students as the exchange between the divisions. In virtually every project to date, the progress in student development was accelerated by these interactions. On an informal basis, the student also spends time at Hewlett-Packard during the periods of project transfer.

These intensely focused activities increase awareness of how industry functions and what relevance means. Some of the major benefits to universities are:

- Industrial involvement improves project definition and research objectives.
- Direct contact with industrial colleagues is demanding and stimulating.
- Better equipment opens new doors to student thinking and attracts better students.
- Students develop incentive and feel a major personal reward by seeing their research applied.

There are however, special considerations related to university activities. Universities are tax exempt organizations so all donations must be unrestricted grants. Because students and professors are often measured by their publications, proprietary programs are not recommended. Finally, to establish a good reputation and attract the very best people, a program should be continuous over an extended period of time (four or more years).

In the early seventies, Carver Mead at Caltech and Lynn Conway's VLSI systems area at Xerox's Palo Alto Research Center (PARC) were exploring the development of new VLSI design methodologies. The early success of this work in 1979 led to the publication of their textbook, Introduction to VLSI Systems, in 1980.[1] In the beginning, these concepts had not been proven effective in industry and so it was felt by many that they were only theory.

Using a preliminary draft of the textbook, a course of study was developed in 1977 that taught the basics of integrated systems design necessary to undertake an IC design. It was followed by a project in which students did a complete IC design from architecture through to layout, using a symbolic approach. In the fall of 1978, Lynn Conway introduced this course at the Massachusetts Institute of Technology (MIT). It was felt that if these concepts could be taught to students with no prior IC design experience and they then could do an IC design from initial concept through to finished parts in one school term, the value of this methodology would be verified.

At the time of the final plans for implementation of the first VLSI class at MIT, Jonathan Allen of MIT approached Hewlett-Packard concerning the fabrication of the IC for its

(continued on page 32)

A Process Control Network

by Christopher R. Clare

The successful processing of MIT's first Multi Project Chip (MPC) was aided by a proprietary computer-aided manufacturing system which was developed at HP's Integrated Circuits Processing Laboratory (ICPL). A similar system is installed at HP's Corvallis Division.[1] This system consists of a distributed network of computers and direct digital controlled (DDC) processing equipment. The system is partitioned to minimize communication between the nodes in the network.

The process of making integrated circuits consists of a series of steps called operations. Each operation is performed using different tools which provide the environment for the desired reaction on the wafer. Many tools in ICPL are under direct digital control to provide rapid operation changes and consistent results.

The term DDC implies several features in the equipment. First, the method for achieving the desired parameter value, called the setpoint, is through digital computation. The IC process parameters such as temperature or pressure are sensed by transducers and converted to digital form by analog-to-digital converters. The computer, operating as a controller, samples the parameter values on a periodic basis and computes a new control value based on a control algorithm. The control value is converted to signals that vary power or flow to achieve the desired setpoint. The control algorithm can be refined by changing control constants in the software to achieve tighter control than would normally be achieved by non-computer methods.

A second major feature of DDC is that the setpoints for control parameters can be varied as a function of time according to an operation description which can be loaded from a remote computer in less than 15 seconds. Once loaded, the operation will run to completion regardless of breaks in communication between computers.

Another feature of DDC is that processing parameters are constantly monitored for deviations outside of normal control limits. Any such deviation causes an alarm which alerts the operator to investigate the equipment for a possible malfunction. This feature tends to improve confidence in the DDC equipment.

The diffusion furnace and plasma etcher are examples of DDC tools in ICPL. The DDC diffusion furnace, shown in Fig. 1, provides a clean gaseous environment at temperatures up to 1100°C for such operations as growing oxides, diffusing dopants or annealing implants. There are no manual settings on this machine.

The proprietary plasma system, shown in Fig. 2, was designed and built in ICPL. This system uses an RF plasma to cause chemical reactions to take place on the wafer surface in a dry gas environment. Otherwise, such reactions require the use of dangerous liquid chemicals. All functions are computer controlled including a system to detect the completion of etching for the particular film being etched.

A process like NMOS done for MPC consists of 40 or more operations that must be done in the proper sequence and under tight control. At ICPL, as at other plants at HP, more than one process may be conducted at the same time in one facility. The wafers, grouped in lots of approximately 20, follow different paths through the facility. At each operation, the lots may compete for machine time with other lots. Lots waiting to be worked on are queued at the operation just like a line of people waiting at the bank. Each lot may require a different operation. To smooth the flow of wafers, HP relies upon a process control network called the Process Control System (PCS). There are more than 30 computers in the network used at ICPL.

PCS is organized in a network having one central HP 1000 system as shown in Fig. 3. Computers in PCS use a standard protocol format[2] for all communication with other computers. The network is based on a simple, modular structure. Some controllers may control more than one piece of equipment such as low-pressure chemical-vapor-deposition (LPCVD) or plasma equipment. One or more controllers communicate with a bank computer. The bank computer communicates directly with the central computer. The bank acts as an interface between lab personnel and the controllers.

PCS communicates with personnel through display terminals which have essentially the same functions and commands regardless of their location in the network. As wafers flow through the process, the PCS records their movement in individual lot histories which can be recalled and viewed on any display terminal during processing.

The process engineer wishing to fabricate wafers through ICPL's facility creates a detailed list of the sequence of operations for the process. While a lot is in process, part of the operation description sets up the automated equipment and the rest gives directions to the lab personnel to insure faithful implementation of that operation. Included in each operation are the target values

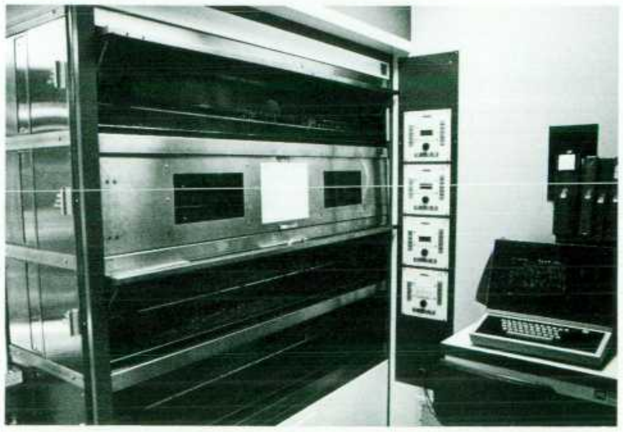


Fig. 1. *A diffusion furnace under computer control is part of a distributed network in the ICPL process facility.*

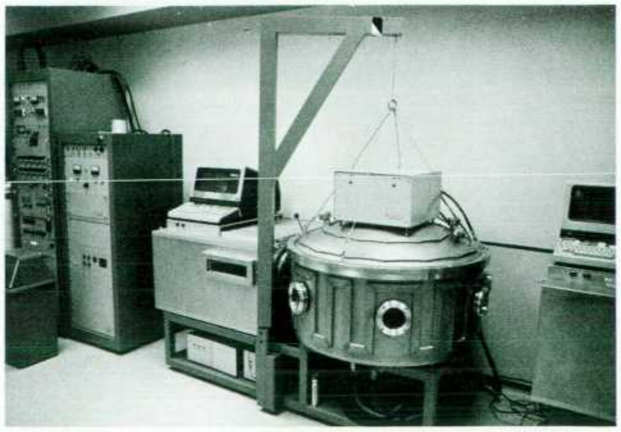


Fig. 2. *ICPL's computer-controlled plasma systems were designed in-house to meet the requirements of VLSI processing.*

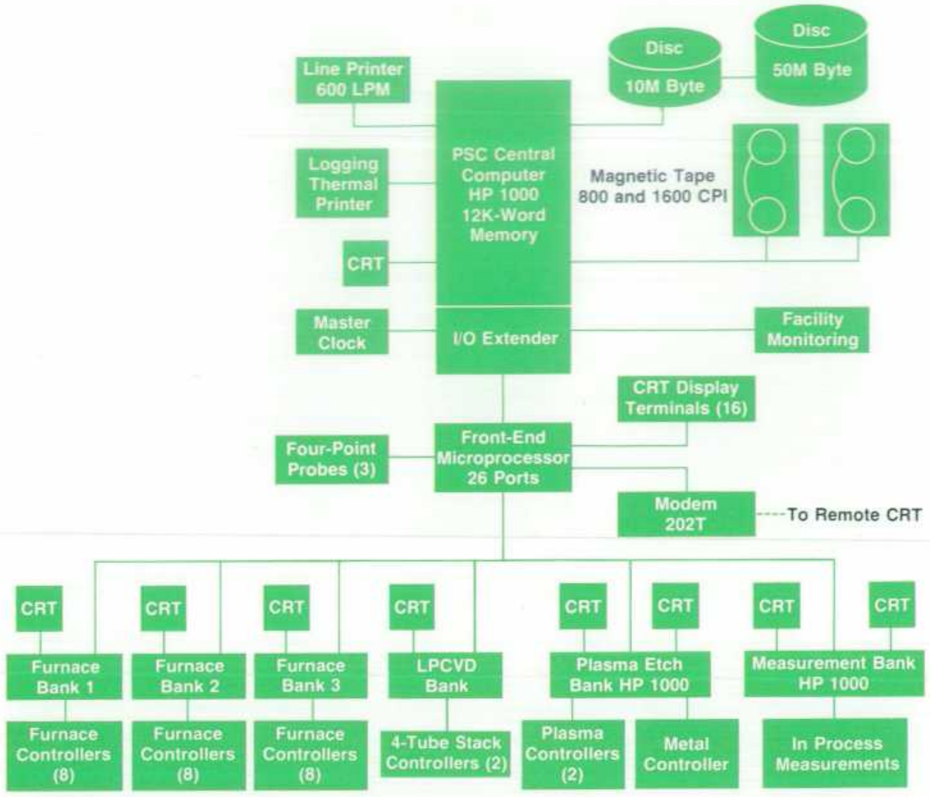


Fig. 3. *The PCS distributed computer network enhances the communication between operators, engineers and equipment to maintain a smooth wafer flow and a quick turnaround time.*

and ranges that the operation monitor should observe to allow the wafers to proceed to the next operation. Using PCS, the lot is automatically queued at the next operation in the process sequence after verification of the successful completion of the previous operation. Delays involved in engineering interactions are reduced by status displays that the engineer can access without entering the clean room and by standard operations that the engineers can intermix with other more experimental steps to reduce operational mistakes. Because the process is documented in a computer-readable form, it is easy to transport a copy of the process to another similarly equipped facility as was done in copying the Corvallis NMOS process in ICPL. At other times, parts of a process can be duplicated at two locations for backup of critical equipment.

Acknowledgments

PCS and DDC equipment are the results of efforts at ICPL in both hardware and software involving many people along the way. Among those deserving credit for their contributions are Shane Dickey, John Youden, Gary Modrell, Subhash Tandon, Mary Dryden, Bill Goodman, Ulrich Kaempf, Eric Slutz and Gerry Alonzo. Of course credit also goes to Bob Grimm and Pat Castro who gave support and encouragement while providing the environment for the PCS development at ICPL.

References

1. D. Clemans and G. Castleman, "Distributed Computer Network Takes Charge in IC Facility," Electronics, June 5, 1980.
2. The computer process is based on a serial communication between RS-232-type ports according to SEMI Equipment Communications Standard (SECS) available from Semiconductor Equipment and Materials Institute, Inc., 625 Ellis Street, Mountain View, California 94043.

Christopher R. Clare

Chris Clare holds a BS degree in electronic engineering from California Polytechnic University (1966) and an MS in electronic engineering from Stanford University (1968). He has been with HP Laboratories (HPL) since 1966, when he was on the design team for HP's first desktop computer. Following that, he contributed to a variety of logic design and electronic projects and wrote some papers on kinesthetic feedback in keyboards. In 1973 he authored a book, "Designing Logic Systems Using State Machines" (McGraw-Hill). He has lectured at Stanford and the University of California at Berkeley on logic design and taught classes on design within HP. From 1975 to 1980, Chris directed the process automation effort at the Integrated Circuits Processing Laboratory at HPL. Based on this experience, he authored the SEMI Equipment Communications Standard for interfacing between IC processing equipment and a host computer. His present work focuses on the future system requirements for VLSI manufacturing control.

initial class project. Design rules for a silicon-gate NMOS process as well as the device parameters predicted by the device simulation models were discussed with representatives of HP's Integrated Circuit Processing Laboratory (ICPL). Although this laboratory did not have at that time the NMOS process that matched the design rules being used, HP was confident that a manufacturing process used by its Corvallis Division in Oregon was a close match and could be transferred to ICPL. Jonathan Allen and Lynn Conway requested rapid wafer-fabrication processing in order to return the final designed and processed chips to the student before the term ended. This meant that, if the masks were available for processing on December 15th, processed wafers should be available to MIT in early January.

Because Corvallis Division's facility was very similar to ICPL's facility, there was much confidence that the NMOS process at Corvallis could be transported by entering the process operation descriptions into ICPL's Process Control System (PCS) and then duplicating the process. Agreement was made with MIT to provide wafer fabrication for the 1978 class. Details for mask specifications, such as the proper fields per layer, critical dimension compensation and mask labels were incorporated with the final electron-beam mask format. This format was sent to Micro Mask for mask generation. The mask layers were generated and delivered in the sequence of their use to allow the wafers to be processed as rapidly as possible. Two wafer lots were processed in seven working days and the packaged chips were available to the students at MIT in the desired time. The NMOS process transferred from Corvallis Division and used for the first time at ICPL for these lots resulted in all parameters meeting the desired specifications.

This experience provided a unique opportunity to prove the value of university and industrial cooperation. For the university it provided verification of their theories and IC design methodologies. For industry, it provided an opportunity to demonstrate the value of quick-turnaround processing in verifying IC designs (see article on page 33 for discussion of other benefits), direct digital process control for transferring and controlling processes (see box on page 30) and for the first time, it created a "can-do" attitude for computer scientists as well as electrical engineers toward rapid integrated circuit design.

After the success of the 1978 MIT project, ICPL collaborated with Xerox and Micro Mask for the MPC79 project in which eleven universities submitted 82 designs from 124 participating designers. Two different mask sets were required to make these projects in sufficient quantity to insure each circuit would have sufficient packaged parts to test. A later project in May 1980 (MPC580) required five different mask sets to handle 171 design projects from fifteen participating universities.

Xerox PARC had the capability of merging all project design information into the mask-specification data to provide the electron-beam-formatted tape for mask making. Micro Mask agreed to cooperate with rapid response time for generating electron beam masks.

However, fabrication of such small quantities of wafers was not economically feasible for a semiconductor manufacturer. HP's ICPL facility, with its flexible processing capability and ability to respond for quick turnaround on small quantities of wafers, was ideally suited to handle the fabrication. This allowed the universities to have actual working circuits quickly and prove the concepts of the course. With the leveraged cost of merging mask making and processing spread over the many projects on each mask set, the actual cost per student project can be in the affordable range of a few hundred dollars each.

HP's design aids group has developed its own VLSI systems design course which just completed its second round of classes and student projects. This class is a ten-week in-house version of the Mead/Conway VLSI Design Course. Twenty students took part in the first session, and color video tapes were made that are now used for all subsequent courses. Ten IC projects were completed and

Merrill W. Brooksby

Merrill Brooksby set up HP's design aids group and is now responsible for a corporate design aids development program. He joined HP as a development engineer in 1959, then became section manager for high-performance counters. He was one of the early users of ICs at HP, helped outfit HP's first IC facility, and helped develop a 2-GHz bipolar IC process. He managed the Santa Clara Division's IC facility for two years and spent a year and one-half planning HP Laboratories' IC facility. In 1974 he joined the newly formed corporate engineering department responsible for IC facilities and strategies throughout HP, and in 1977 started the design aids group. Merrill earned his BSEE degree at Brigham Young University over a period that included interruptions for military service and church missionary service. He received his BSEE in 1959 and his MSEE from Stanford University in 1962. He is a member of IEEE, has held a private pilot's license for 28 years, and is interested in hydroponic gardening and youth groups. He's married and has five children.

Patricia L. Castro

Pat Castro is director of the IC Processing Laboratory (ICPL) at HP Laboratories (HPL). She joined HPL's Solid State Laboratory in 1973 and was a member of the task force responsible for defining HPL's IC facility. She was named operations manager of the new facility and became director of ICPL when it was formed in 1979. Pat received her BSEE degree in 1959 from Northwestern University and her MSEE degree in 1963 from Stanford University. Before joining HP she was involved with materials research, IC process development, and IC production for seven years. She has held many offices in the Electrochemical Society and has authored many papers on materials and IC processing. She serves on the board of directors of the Santa Clara Valley Science and Engineering Fair Association and on advisory boards at local colleges. A native of Tupelo, Mississippi, she is married and has a son.

fabricated with ICPL's quick-turnaround capability.

This course was offered next to ninety students at four locations in October 1980. The course material was presented by video tapes from the first class and additional live lectures. At each class site (Palo Alto, Cupertino and Sunnyvale in California and Fort Collins in Colorado) a tutor was available to answer questions and assist the students in the project phase of the class. The course was run at a university level; homework was required, graded and returned to the students. A mid-term exam was given and graded, and during the last five weeks of the course, an IC project was required. The projects ranged from simple circuits with one hundred or so transistors to fairly large designs with thousands of active elements. During this session, thirty projects were completed and already the benefits of this approach are paying off in terms of new circuit designs. Approximately 75% of the circuits designed in the latest class are for new products now on the drawing boards. Wafer processing of two lots for this session was completed in thirteen working days after allowing for the delay in mask delivery. This cycle time includes three days of delay caused by scheduled equipment maintenance. Without this delay, the turnaround time is expected to be ten working days which represents twice the end-to-end absolute minimum processing time required for this NMOS process on a 24-hour working day basis.

A CMOS version of this course began in April of this year at HP's Corvallis Division. Quick-turnaround processing of the projects for this class will be provided by the Corvallis CMOS facility.

Acknowledgments

The authors thank Jim Lipman, Ken Picek, Rob Mathews, John Newkirk, Doug Fairbairn (Publisher of Lambda Magazine), Ernest Erni, Mike Beaver, Bob Dutton of Stanford University, Don Pederson of UC Berkeley, Carver Mead of Caltech, Jonathan Allen of MIT, and members of HP Laboratories (HPL) and the IC Processing Laboratory (ICPL) for their contributions and suggestions.

Reference:

1. C. Mead and L. Conway, "Introduction to VLSI Systems," Addison-Wesley, 1980.

Benefits of Quick-Turnaround Integrated Circuit Processing

by Merrill W. Brooksby, Patricia L. Castro, and Fred L. Hanson

THE VALUE OF SHORT IC PROCESS cycle times for research and development is usually taken for granted, but often, since production can be scheduled to agree with any reasonable cycle time, short manufacturing cycles times have not been considered to be very important. However, the demand within Hewlett-Packard has placed significant emphasis on fast turnaround for both production and research and development. To better understand this demand, consider the advantages of being able to fabricate ICs in the minimum possible time.

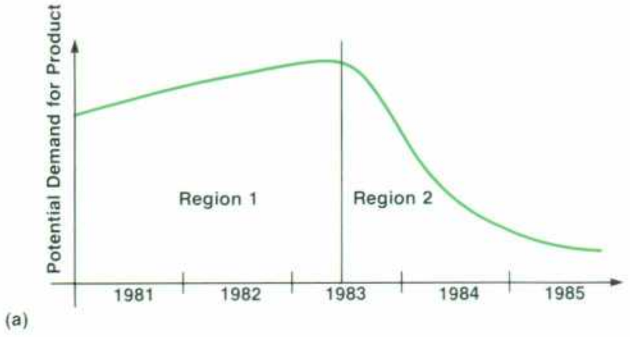
R&D Benefits

Why is quick turnaround so important in research and development? Why don't engineers just schedule their work so that the process cycle times can be arranged in parallel with other activities? Resources would then be efficiently and effectively used and life might be a lot less hectic. There are, however, important reasons for R&D experiments to be done in a hurry.

To help understand why, let's look at a hypothetical product life cycle curve for a product we'll call SRF1. One conceptual point is fundamental to the understanding of this argument. That is, once an idea is formed into a particular product embodiment (still as an idea) its life cycle curve is fixed. This curve is fixed not only by what the product is, but also by whatever competitive, social and economic factors influence it. This product life cycle curve, or potential demand for the product, exists whether or not the product ever even comes into existence.

Fig. 1a shows the life cycle curve for our hypothetical product, the SRF1. For the sake of simplicity in explaining this concept, the curve is made up of just two regions. In Region 1, the potential demand is usually fairly stable, although it could be growing or shrinking slightly due to a lot of factors. It is the onset of Region 2 that is of concern to us. In this particular example, conditions for Region 2 are such that technology will have advanced to the point that competitors will be introducing superior models of the SRF1 to the marketplace as well as the fact that the next generation product, the SRF2, will be displacing SRF1s in many applications. It is difficult to do much about Region 2 unless we change what our SRF1 is and/or what it costs.

Let's assume for the moment that we embark on the development of the SRF1, and it is ready for start of manufacturing at the end of 1982. Fig. 1b shows what the actual sales volume would be for the SRF1, assuming that it takes six months in 1983 for manufacturing to build up sufficient volume to meet the demand. A glance at the shaded area in Fig. 1b shows the impact on sales of the SRF1 if the development can be speeded up by one year. Notice that the



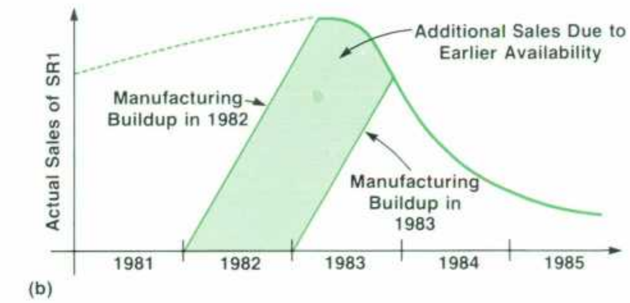


Fig. 1. *(a) Product life cycle curve for a hypothetical product, the SRF1. This curve is independent of the product's existence. (b) Actual sales achieved for SRF1 if the product development is completed at the end of 1982 or one year earlier at the end of 1981.*

delay or speedup of shipments affects the mature-volume shipments, not the low-volume periods. Depending on the product, earlier market entry may very well double the total sales of the product during its life. The belief in a product life cycle is a primary driving force for the R&D organization's insistence on shorter experiment times which can result in shorter product development schedules.

Another insidious impact that longer cycle times tend to have on IC development is that engineers tend to be less innovative if it is months from the time the design is completed until the photomasks are available and the ICs are processed. The always-present schedule pressure is there and very often the feeling is, "maybe we had better not run the risk of trying that neat new idea that could eliminate some expensive components. If the idea doesn't work, months will be added to the schedule." Thus the overall impact can be quite detrimental.

Production Benefits

Within Hewlett-Packard there has always been emphasis placed on having short manufacturing process times. However, in an IC facility, this becomes significantly more important because of yields and high inventory costs.

One of the most significant advantages of quick-turnaround IC processing is its impact on yield. The rate of yield improvement is an inverse function of the cycle time required to complete an engineering experiment. Such experiments provide the feedback required by process engineers for implementing process improvements. It is estimated that an experiment designed to improve a process will most likely result in a yield improvement from −2.5% to 12.5%. It can be further estimated that some small, but positive yield improvement (typically 5%) for each completed engineering experiment is the most probable result. To increase the rate of yield improvement the feedback loops (turnaround time) should be as short as possible. The net effect of cycle time on yield improvement is illustrated in Fig. 2 where the advantage of a short cycle time results in an earlier significant yield increase.

Quick-Turnaround Operation

IC manufacturing management is often reluctant to take all of the steps necessary to minimize cycle time. The arguments generally center on efficiency, or as it is sometimes expressed, "operator efficiency." They argue that if there is not a reasonable amount of inventory between stations or steps in the process, each operator is paced by the previous station. Any slowdown at one station causes a similar slowdown at all downstream stations. This means operators are often standing around waiting for material. There is some truth to this argument, but it can be turned into an advantage if both the operations and engineering personnel respond in the proper ways. The first requirement is that manufacturing and engineering management must have information available on a near continuous basis about the situation at each step in the process. If there is a problem at any station, it must receive immediate attention. If engineering help is needed, it must be called in immediately. Since there is no inventory between stations, the visibility of a slowdown is much greater and the likelihood of a problem going undetected for any length of time is small.

Once a solution to a problem is proven out, another requirement of the strategy comes into play. Operators must be trained to handle some of the operations upstream from their respective stations so that a continuous flow of product can be restored in the shortest possible time. It is also necessary to provide an equipment capacity somewhat greater than that required for normal product flow to allow quick recovery from stoppages.

From a manufacturing viewpoint, the long or short process cycle time tends to be almost a binary decision. If an organization chooses to allow a slightly longer cycle time in order to "gain a little more efficiency," the small stack of

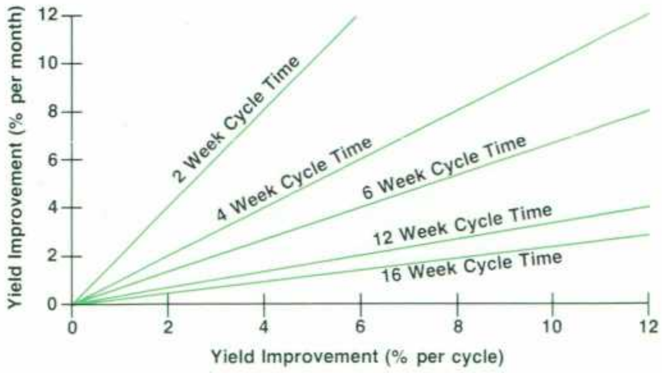


Fig. 2. *Plots of probable process yield improvement versus time for different process cycle times. As can be seen, short cycle times can result in more rapid yield improvement.*

wafers awaiting processing at each station is comforting. On the other hand, if any of those stacks of wafers are bad.....this thought often causes management to institute a much more elaborate and expensive in-process testing capability, one or more tests following each step in the process. This also tends to require a small stack of wafers at each station to ensure that it can run efficiently and smoothly. Thus, the adding of a few wafers before each step causes the addition of more steps and more stacks of wafers. With these stacks of wafers everywhere, it becomes much more difficult to know about a slowdown or a technical problem anywhere in the process area. This in-process inventory also significantly increases costs.

The opposite approach, a short production cycle time, is characterized by a quite different type of operation. Inventory between stations is minimized. Process monitoring is used but not allowed to increase process time. In many cases, only a sample of each lot of wafers is tested. Operators must be more versatile. If a particular step in the process has a problem getting sufficient quantity out, the next station knows about it almost immediately, and so on down the line. This brings prompt supervisory or engineering attention. As the problem comes under control, the operators on down the line are shifted upstream to help get wafers moving again.

With the reduced emphasis on total in-process testing of all wafers, a compensating capability is instituted. The capability for diagnosing processing problems must be first class. It is debatable whether or not this approach increases or decreases the likelihood of a significant problem. It is obvious, however, that if you have a two-week cycle time and any process step goes down, the output will go to zero in two weeks or less. On the other hand, once the problem is solved, the output can be quickly restored. In the worst case, only two weeks worth of in-process inventory will have to be scrapped. It could have been three months worth, and this has happened!

Once a decision is made to run a shop with a short cycle time, it is not always easy to make it happen. An easily missed point is that control of the number of wafers input at the start of the process is the primary control on cycle time for an IC process. If for any reason the flow of wafers out of the process is slowed down or stopped, a corresponding change must be made at the input end. One lot of wafers out allows one lot of wafers to go in. If more wafers are allowed to go in than come out, the in-process inventory and cycle time grows. If you want more wafers out, change the process, the equipment set, the number of people, or another factor that really changes capacity. Then and only then should the input be changed. If cycle time grows beyond the desired time, cut the inputs back until it balances.

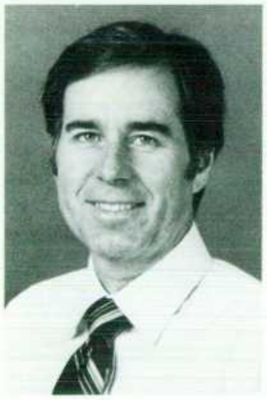
One of the more difficult tests that this mode of operation encounters occurs when the supply of good circuits was just adequate and suddenly the yield drops. The instinctive response is likely to be "we are not getting enough out, let's increase the starts." This often is the first response. The system then responds with no more wafers out, the cycle time grows longer, but at least the yield responds; it drops even lower! This may not have been the anticipated result, but the fact is, it does happen. This is because the available resources are used to start new wafers, not fix the problem. In effect the emphasis has been quickly changed from quality to quantity. This is not good for any manufacturing operation, but for an IC manufacturing process it is deadly.

Short IC process cycle times are possible if the operating philosophy is in agreement with the constraints discussed above. For instance, engineering lot cycle time for the Corvallis Division's CMOS LSI process is less than two weeks, while other facilities are typically running with process times from one to three months.

Acknowledgments

The authors acknowledge the contributions of Jim Colwell, Terry Pierce, Ed Shideler, and Mike Beaver, whose implementations of these ideas have confirmed their validity.

Fred L. Hanson



Fred Hanson is engineering manager at HP's Corvallis, Oregon Division. A native of Ogden, Utah, he received his BSEE and MSEE degrees from Utah State University in 1962 and 1963. He then joined HP's Loveland, Colorado Division, helped develop the 741A Differential Voltmeter/DC Standard, and served as project manager for the 745A AC Calibrator and the 746A High-Voltage Amplifier. Several papers and one patent resulted from that work. Then came a year in production engineering, three years as an R&D section manager, three years as manager of the Loveland Instrument Division integrated circuits department, and a year and one-half managing the start-up of the Corvallis Division IC operations. Away from the job, Fred enjoys running, fishing, hunting, and backpacking, and is a novice scuba diver. He's also held leadership positions in Boy Scout programs. He's married, has four children, and lives in Corvallis.

David Packard on University and Industry Cooperation

THE HEWLETT-PACKARD COMPANY has benefited in many ways from cooperative research programs with universities. Our closest and most intensive involvement in this regard has been with Stanford University. HP's first product, the audio oscillator, was the outcome of research on the negative feedback principle done by Bill Hewlett while at Stanford, and many other HP instruments have come from company-supported research there over the years.

For example, in 1948-49 we supported research on high-speed circuitry for nuclear counters, which was being done by two graduate students, Al Bagley and Quentin McKenna. This work made the Model 524 10-megahertz frequency counter possible, and it was one of our most successful instruments of that era.

Cooperative research programs with universities can generate new ideas for new products, and also can provide the opportunity to get acquainted with outstanding students who are prospective employees. The research project mentioned above not only brought us an outstanding new product idea, but an outstanding employee named Al Bagley, now engineering manager of our Instrument Groups.

In the early years of the company it was largely the close personal relationship Bill Hewlett and I had with Fred Terman, dean of the Stanford School of Engineering, that made possible these very effective cooperative research programs. Fred Terman's vision of university-industry cooperation brought about the development of electronics research and education at Stanford to rank with the best in the country, if not in the world.

Of course research has been a significant factor in the advancement of technological knowledge at many educational institutions other than Stanford, and in many fields other than electronics. University research is one of the most important sources of new technology, and it should be nurtured and supported to the fullest possible extent by those industries and individual companies, such as Hewlett-Packard, that are highly dependent on technological advancements for their progress and growth.

In the early years of the 1960s, there was concern at the universities about the status of the humanities. The so-called hard sciences such as physics, chemistry, and engineering were receiving good levels of support from the United States government, and from industrial companies such as HP. There was, accordingly, a drive by university people for more unrestricted financial support from business and industry, and this drive met with some considerable success.

A few years ago we decided to expand HP's university research support program. Relying on our earlier experience, we sought out professors at various universities who were doing research in areas in which we were interested. We provided these people with a level of financial support that would be significant, and over a long enough period of time for them to be productive. We endeavored to establish a personal, cooperative relationship between HP people and university research people that would contribute to the effectiveness of the research, and enhance the benefits for all those involved.

I am very pleased about the way these programs are going, and I hope that as time goes along we will be able to initiate similar programs in new areas of interest to our company. I also hope that the experience and success we at HP have had in university-industry cooperative research will encourage other companies throughout industry to develop, or expand, similar programs in the future.

David Packard
Dave Packard is co-founder and chairman of the board of HP. He attended Stanford University, receiving a BA degree in 1934 and an MSEE degree in 1939. He served as a partner from HP's founding in 1939 until it incorporated in 1947, when he became president. In 1964, he was elected chairman of the board and chief executive officer. He left HP in January 1969, to become U.S. Deputy Secretary of Defense. After resigning in December 1971, he returned to California and was re-elected HP's chairman. He is a Fellow of the Institute of Electrical and Electronics Engineers, a member of the National Academy of Engineering, and an honorary lifetime member of the Instrument Society of America. He has served on the Stanford University Board of Trustees and was president of the board from 1958 to 1960.

Hewlett-Packard Company, 1501 Page Mill Road, Palo Alto, California 94304

Bulk Rate
U.S. Postage
Paid
Hewlett-Packard
Company

HEWLETT-PACKARD JOURNAL

JUNE 1981 Volume 32 • Number 6

Technical Information from the Laboratories of Hewlett-Packard Company

Hewlett-Packard Company, 1501 Page Mill Road
Palo Alto, California 94304 U.S.A.
Hewlett-Packard Central Mailing Department
Van Heuven Goedhartlaan 121
1181 KK Amstelveen, The Netherlands
Yokogawa-Hewlett-Packard Ltd., Suginami-Ku
Tokyo 168 Japan

CHANGE OF ADDRESS: To change your address or delete your name from our mailing list please send us your old address label. Send changes to Hewlett-Packard Journal, 1501 Page Mill Road, Palo Alto, California 94304 U.S.A. Allow 60 days.