The
Computer
Museum

300 Congress Street
Boston, MA 02210
(617) 426-2800

March 5, 1984

Stephen Swerling
Vice President-Engineering
Mentor Graphics Corporation
8500 SW Creekside Place
Beaverton, OR 97005-7191

Dear Steve,

Delighted to have met you at the AEA meeting in Portland. It was especially nice for me that you were enthusiastic about The Computer Museum. A copy of the most recent issue of our quarterly, The Computer Museum Report, is enclosed.

You are personally invited to become one of our Founders — a unique opportunity that lasts until June — to help an international institution get off the ground.

I hope that you will urge Mentor Graphics Corporation to become a Corporate Founder ($2500).

Software is one of the most difficult things to exhibit at the Museum — and the issue that everyone talks about. If you have any ideas about how your products might be used to explain CAD, I'd be delighted. We are planning a major gallery on "The Computer and the Image" developed by Dr. Oliver Strimpel. His proposal for this gallery is enclosed and if you see any way for Menotor to be involved, we'd love to have you. We do have an excellent relation with all the folks at Apollo and will have a work station or two, as needed.

To give you some background, the Museum was started by Digital in 1979 based on Ken Olsen's and Gordon's collection. Then, in 1982, it was established as a separate entity with its own Board of Directors. This year a home was found for it on Boston's Museum Wharf, where it will open on October 1. It is a non-profit, public, charitable foundation dedicated to preserving and exhibiting an industry-wide, broad-based collection of the history of information processing. Computer history is interpreted through exhibits, publications, videotapes, lectures, educational programs and other activities. The Museum archives both artifacts and documentation and makes the materials available for scholarly use.

When you are in Boston let us know and we will give you a behind-the-scenes look at the Museum.

Cordially,

Gwen Bell
Director

GB/jb
encl.

cc: Oliver Strimpel

Original Apollo : 'Green Node' :
Now green node;

Sales off Lexington:
Rt 4 & 225 — 8635776
Bedford St
E Regional Sales Mgr Denis Watts Ed
3 cents
N of Rt 128
Rt 2
crossing

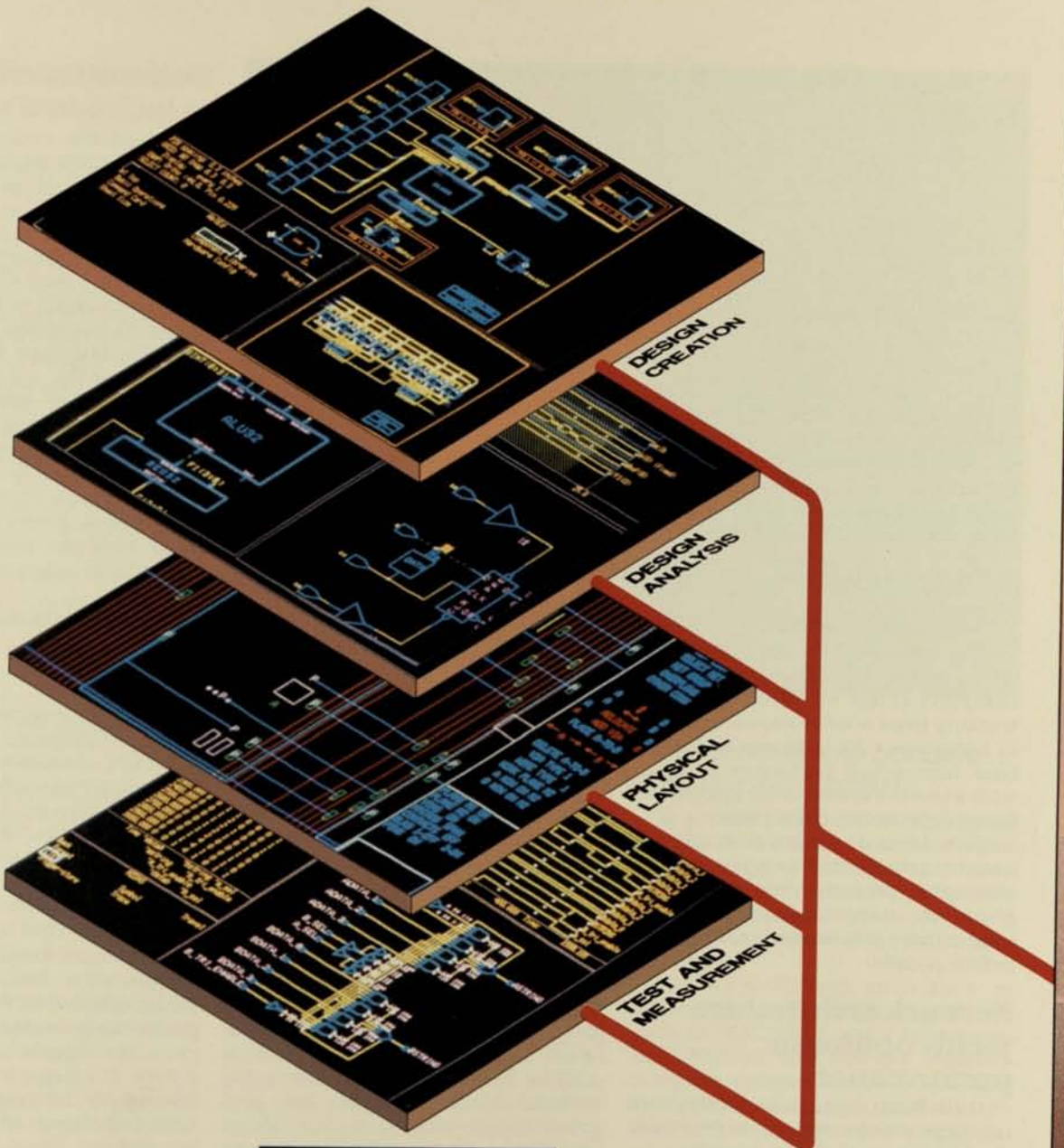
503-626-7000

SWERLING, Stephen

Mentor

# IDEA 1000

**A computer-integrated engineering system**




# The first fully integrated and computerized design path from concept to prototype test.

Rising gate counts. Tighter real estate requirements. Increased functionality. Compressed product development cycles.

They're all facts of life for today's hardware design engineer. And they won't go away any time soon.

The only real solution is to take full advantage of state-of-the-art VLSI computing power and make it an integral part of your hardware design team. From initial concept all the way to verification of the prototype.

Moreover, this computer power must be applied in such a way that each individual computerized engineering tool is fully integrated into the design system as a whole. And that the system itself gives you maximum flexibility, performance and growth potential.

Only one computer-aided engineering (CAE) system satisfies all these requirements.

The Mentor Graphics IDEA 1000.

Why? Because Mentor Graphics is both a pioneer and leader in the field of computer-aided engineering. We're working from a solid knowledge base to bring you CAE solutions that combine unmatched performance depth with a level of system-wide integration found only in the IDEA 1000.

With Mentor Graphics, you're assured of a single CAE design path from concept to prototype test. A path that gives you more room for increased productivity and innovation than ever before possible.

## Network architecture yields optimum performance.

At the heart of the IDEA 1000 system is a networked computer architecture that brings you all the benefits of a timeshared host computer, but none of the disadvantages.

The IDEA system uses any number of engineering workstations connected together in a high-speed (12 Mbits/sec) network. Each workstation has its own Apollo DOMAIN 32-bit computer with resident hard disk storage and high-resolution graphics hardware. Every workstation has complete, transparent access to data stored anywhere within the system. You get the centralized file organization of a host system, but without performance degradation of timesharing on a central CPU.

Further, the Mentor network can include gateways to mainframes for access to existing CAD tools through protocols such as Ethernet, X.25, HASP or 3270 emulation. Also, peripheral and computational servers within the network can provide all workstations with common resources such as printers, plotters and array processors.

## DBMS system forms a solid yet flexible foundation.

The core of any engineering design system is the data used to describe and test the design. Because the database plays such a critical role, Mentor has designed its own Full Relational Database Management System targeted specifically at the needs of electronic engineering organizations.

You get a complete set of data management tools which control and safeguard your data movement as it evolves during the engineering process. Files are automatically monitored for conditions such as version updates, data interdependencies and access rights.

Mentor's DBMS also allows a great deal of flexibility without any risk to the integrity of stored data. Different programs can access the same data file in any format a particular application calls for. New data formats can evolve without modifying applications programs. Equally important, new design engineering tools can be introduced without modifying existing ones or the data they use.

## Uniform interface molds to individual styles.

One of the central challenges in computer-aided engineering is to present the user with an interface that remains consistent as different design tools are used. This way, the engineer's learning curve is kept to a minimum and applies across the entire system.

Mentor's IDEA 1000 interface solves the problem through a high resolution, graphics interface that remains the same across the entire Mentor tool set. This interactive interface includes a rich set of features such as on-line help, tutorials, command macros, soft keys and programmable display windows. The result is an interface that is rapidly learned and grows in sophistication along with the user, who can "customize" it to individual working styles.

## Select from a high-performance hardware lineup.

A Mentor engineering workstation can be configured from any Apollo DOMAIN computer, giving you broad latitude in selecting the optimum hardware mix for your particular engineering environment. All workstations feature powerful 32-bit processing, local hard disk storage and high resolution graphics hardware.

The IDEA 1001 and 1002 represent the ultimate workstation processing performance. Each has a CPU with 32-bit external and internal data paths, providing throughput comparable with even the largest of supermini computers. At a single workstation, you can run up to 24 concurrent processes, with 256 Mbytes of virtual memory per process.

The IDEA 1200/1300 puts mainframe processing power at the desktop level and becomes a cost-effective addition to any Mentor network.


The IDEA 1000 Series brings powerful 32-bit processing and high-resolution color graphics to the entire range of Mentor design tools.

# Capture your front-end creativity as well as your schematics.

## Hierarchical design entry promotes innovation.

Mentor goes a step beyond graphic entry of "flat" schematic sheets and also lets you approach your design in a hierarchical fashion. You express your design in its broadest functional terms at the "top" of the hierarchy as a series of interconnected function blocks.

As your project progresses, lower layers in the design hierarchy express these functional elements in ever finer detail. At the bottom you have a fully expanded design expressed in its most primitive elements, such as interconnected gates or transistors. When building your hierarchy, you can start at any level you choose and then work either up or down.

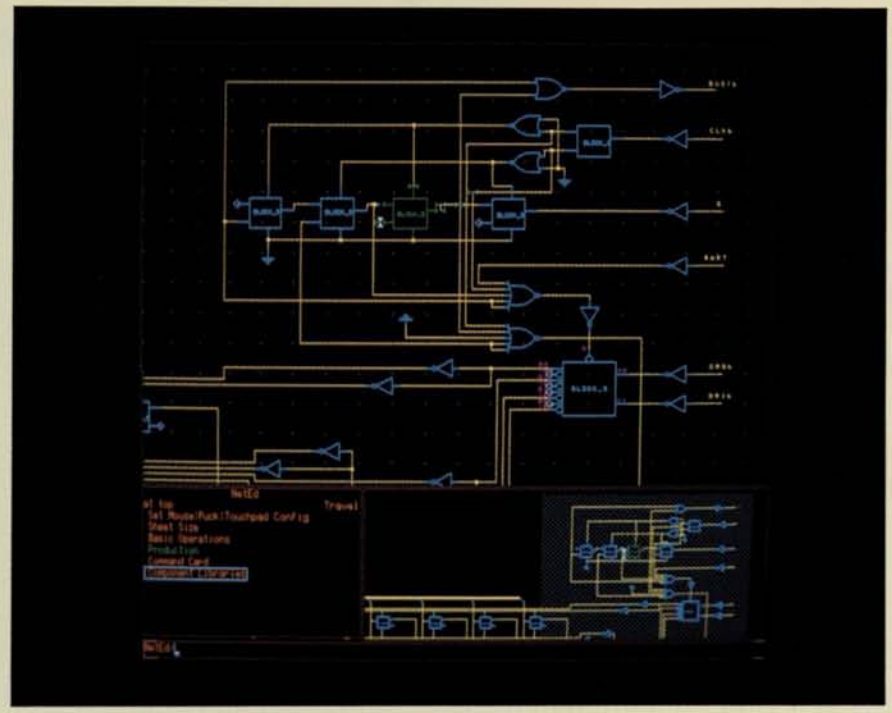
The advantage of this approach is that you can view and perceive your design from many different perspectives, depending on your position in the hierarchy. This gives you vastly expanded opportunities to make innovative modifications. And to make them quickly and easily.

## Use standard symbol parts or create your own.

While some schematics require only "standard" electrical parts symbols, such as TTL, many others will require custom parts. In either case, Mentor gives you a solid solution.

For standard parts, Mentor furnishes a series of parts libraries that meet many different types of applications requirements, including gate array design.

For custom parts, The IDEA 1000 Symbol Editor gives you a powerful graphic tool set to create symbols for even the most complex parts.



## Graphic tools speed schematic entry.

The graphic placement and interconnection of parts symbols often consumes a very large portion of a designer's time. Accordingly, Mentor gives you a high-performance Network Editor that puts a powerful array of graphics-oriented tools at your disposal. Schematic entry is faster, easier, and more accurate.

You can invoke powerful graphics commands that simultaneously move, pivot or rotate whole groups of symbols. Or use a single line to define an entire data bus. Or call upon "frames" to express large volumes of repetitive circuitry in just a few symbolic entries.

## Netlisters allow remote simulation/layout.

To protect your current engineering software investment, Mentor provides a broad range of netlisters that transform IDEA 1000 schematic data into formats acceptable by simulator and layout programs resident on a host mainframe.

This includes simulators such as Tegas and Logcap, and layout programs such as SCI-CARDS and Computervision. Further, you can use Mentor's complete programming environment for creating your own netlisters to interface proprietary engineering tools.

The IDEA 1300 is a cost-effective workstation for schematic entry tasks and includes a 17" high-resolution CRT.

# Simulate your way around costly prototype revisions.

## Logic analysis in software.

One of the principle advantages of the Mentor IDEA 1000 system is that your entire design is captured and preserved in the system's database. This means that many of the debug operations that once required a hardware prototype can now be performed entirely in software.

As a result, you can save yourself many costly prototype iterations by performing the vast majority of your debug operations before the design ever reaches the physical prototype stage.

## Start debugging even before your design's complete.

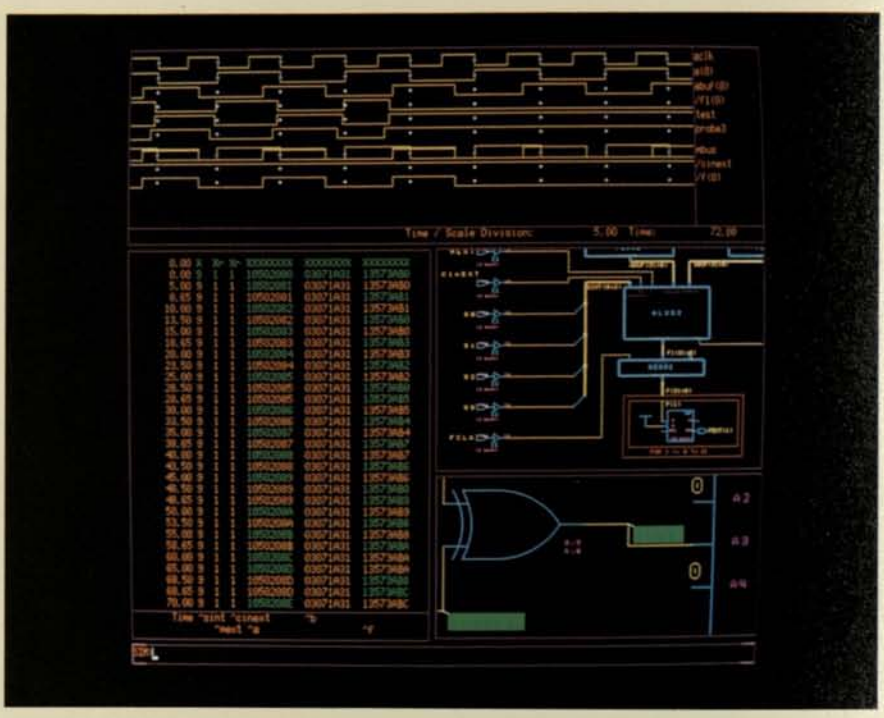
One powerful advantage of simulation using the IDEA 1000 system is that you don't have to wait for the completed design data to begin debugging. You save time, cut costs and considerably reduce the complexity of debug operations.

It's done through a technique called Functional Modeling, which emulates the logical behavior of components that have not yet been designed down to the gate level. Each component is described through a Pascal-based behavior language which describes the circuit's logical outputs in response to input stimulus.

During simulation, these functional models can be freely intermixed with fully designed circuitry to give you a highly accurate preview of how your completed design will perform.

## Interactive logic and timing simulation reveal potential errors.

Mentor's interactive interface puts logic flow displays and circuit diagrams on the same screen at the same time when you're performing simulation tasks. You get a direct visual relationship between the circuitry under test and its logical behavior. You can even graphically "probe" individual circuit nodes to monitor or force their states.



You're not limited by technology, either. Mentor's simulators can accommodate components from gates to PLA's. It also covers MOS as well as TTL and ECL, and supports nine distinct logic states.

## MSPICE models the analog side of your design.

If you are designing down to the analog circuit level, you can model your design's analog behavior through MSPICE, Mentor's version of the popular circuit simulator program.

MSPICE represents a significant advance in analog simulation because it automates the cumbersome input and output operations associated with conventional SPICE programs.

MSPICE works directly with the Mentor design database to extract the information required to perform DC, AC, DC operating point or transient analysis. During a simulation session, you can call up the schematic to be simulated and graphically modify circuit parameters. You can even specify points of interest in the schematic and watch their parameters change during the actual simulation run.

Simulation results are displayed graphically on the high resolution user interface. You can also call in powerful post-simulation processing programs to calculate and graphically display any specific information that your application requires.


The IDEA 1002 provides the 32-bit processing performance that many digital and analog simulations require.

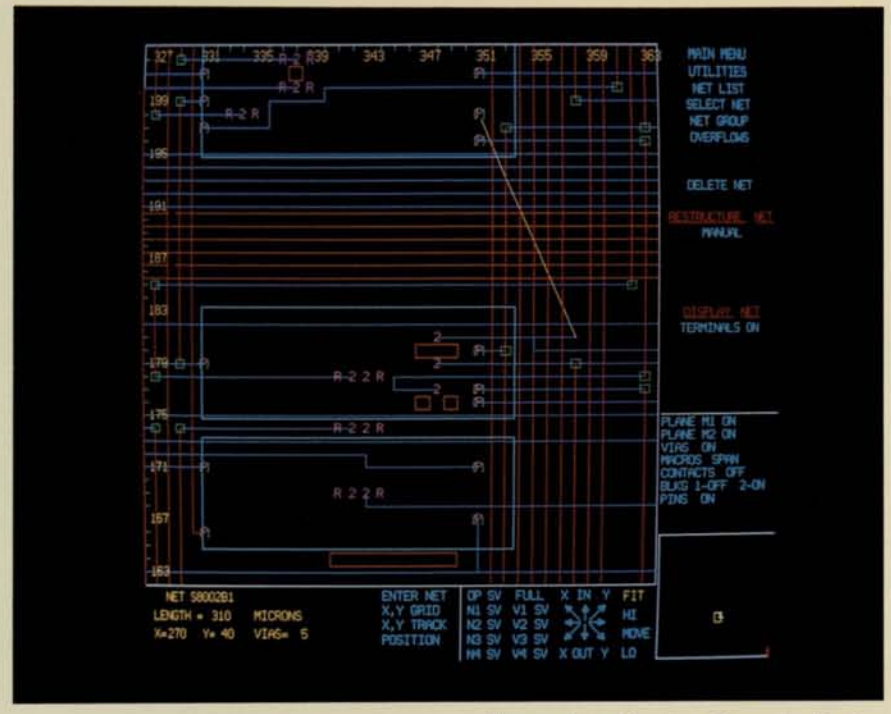
# Automate a wide range of physical layout tasks.

## Fully automatic gate array layout.

Mentor's CADISYS gate array physical design package represents a dramatic leap in the automation of gate array layout. Designs that once took months to move from the schematic stage to the physical layout stage can now be done in a matter of hours.

Besides enormous time savings, you benefit from a state-of-the-art performance level. When converting your schematics from symbols to completed silicon placement, the automated CADISYS typically offers over 90% gate utilization, and connection rates that are typically 98% to 100%.

These outstanding performance levels mean your design will often fit on a smaller, less costly gate array. Also, you can implement more functions in an array of any given size. You save real estate, you save time, you save money.




## A streamlined path from block placement to final layout.

Mentor's CADISYS gets you through the gate array physical layout process in three easy steps that not only get your design done faster, but also more efficiently.

First, Mentor CADIPLACE takes your design and puts it onto the gate array image so that performance and routability are optimized.

Second, Mentor CADIROUTE works with the placed functional blocks and automatically performs the interconnections.

Third, Mentor CADIGRAPH provides an interactive tool for use either before or after automatic routing. You can quickly predefine critical routes or rapidly finish up any overflow connections.

## Design data feedback permits a faster finished product.

When a logic schematic version of a design is transformed into a physical layout for a gate array, delay times can be introduced which may be inconsistent with the original schematic specifications.

At this point, Mentor's system-wide integration of automated design tools becomes invaluable. First, CADISYS automatically produces a file containing net capacitance delay data for every interconnection in the layout. This data is then used by a tool called the Design File Interface to automatically insert this new timing information into the original schematic data.

The updated logic schematic design is then rerun through the simulator tools to verify its adherence to specifications. If the simulation indicates performance problems, the design is modified accordingly and then sent back for another physical layout.

This entire data feedback cycle takes place entirely within the Mentor system. With an extremely high degree of automation and a significant reduction in labor cost.

The IDEA 1002 provides powerful color graphics hardware to present a friendly interface for physical layout tasks.

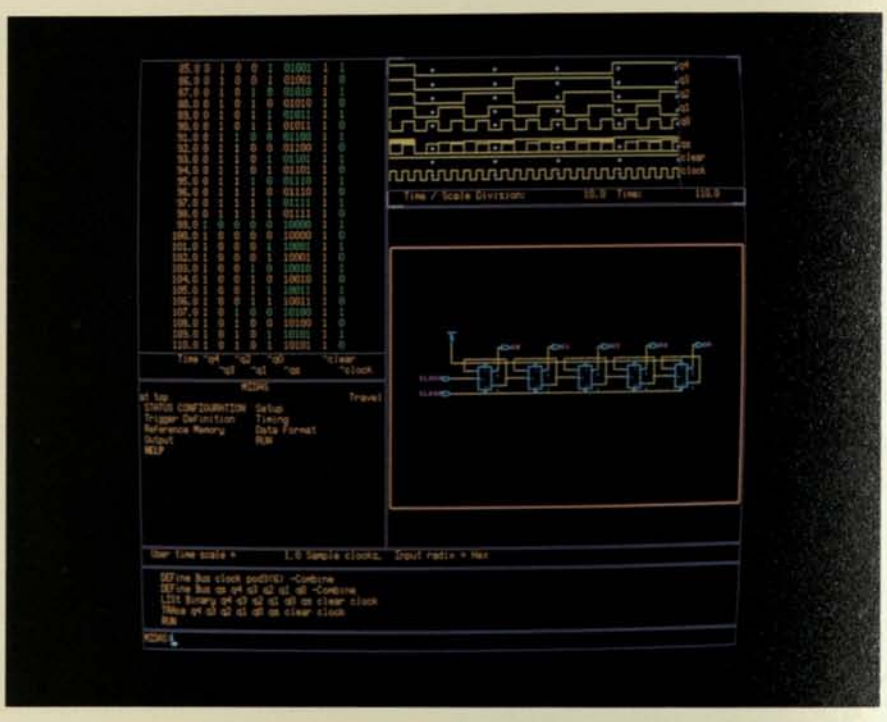
# Verify prototype functions with fully integrated logic analysis.

## MIDAS 7000 brings you high-performance logic analysis.

The hardware design cycle doesn't end with physical realization, so neither does Mentor. The MIDAS 7000 is a fully integrated logic analysis system that lets you perform all your prototype test and verification functions from any IDEA 1000 workstation.

MIDAS 7000 gives you a full array of logic state and timing instrumentation and is controlled through the same powerful graphics interface as all other Mentor design tools. You can even use it in conjunction with other design tools to create powerful synergies only possible with Mentor.

MIDAS 7000 has a modular architecture that accepts a wide range of instrumentation cards. You get maximum latitude in selecting the right performance combination plus the capability to easily update your instrumentation lineup.



## Interactive state analysis with up to 15 trigger states.

MIDAS 7000 lets you tackle even the most convoluted software acquisition problems through sophisticated clocking and triggering, and data widths up to 80 channels.

Sample clocks can be synthesized from up to 7 clock lines, giving you the power to decode the buses of advanced 16-bit and 32-bit microcomputer systems.

Triggering uses a RAM-based state machine and can include up to 15 distinct states that allow for both data acquisition and qualification.

## Timing analysis up to 100 MHz with unprecedented data depth.

MIDAS 7000's timing analyzer not only lets you sample 16 channels up to 100 MHz, it also gives you wider acquisition windows than ever before possible. It's done through a data compression technique called transition triggering that lets you acquire data windows up to 130,000 samples wide.

What's more, you can cross-link the MIDAS 7000 timing and state analyzers for time-synchronized acquisition of both hardware and software data.

## Powerful post-acquisition analysis.

Once again, the system-wide integration of IDEA 1000 tools makes the difference. All data acquired by the MIDAS 7000 can be automatically transferred to a workstation's main memory or disc storage for unlimited post-acquisition processing.

This includes automated comparisons with data accumulated by other Mentor tools, such as the Interactive Logic Simulator. Compare physical data from a prototype with the simulator model to measure accuracy. It also includes programs such as performance analysis histograms and mnemonic disassembly packages.

The MIDAS 7000 system gives all the power of a CAE workstation coupled with a high-performance logic analyzer.

# Manage and document the entire design process.

## DOC: the premier technical documentation tool.

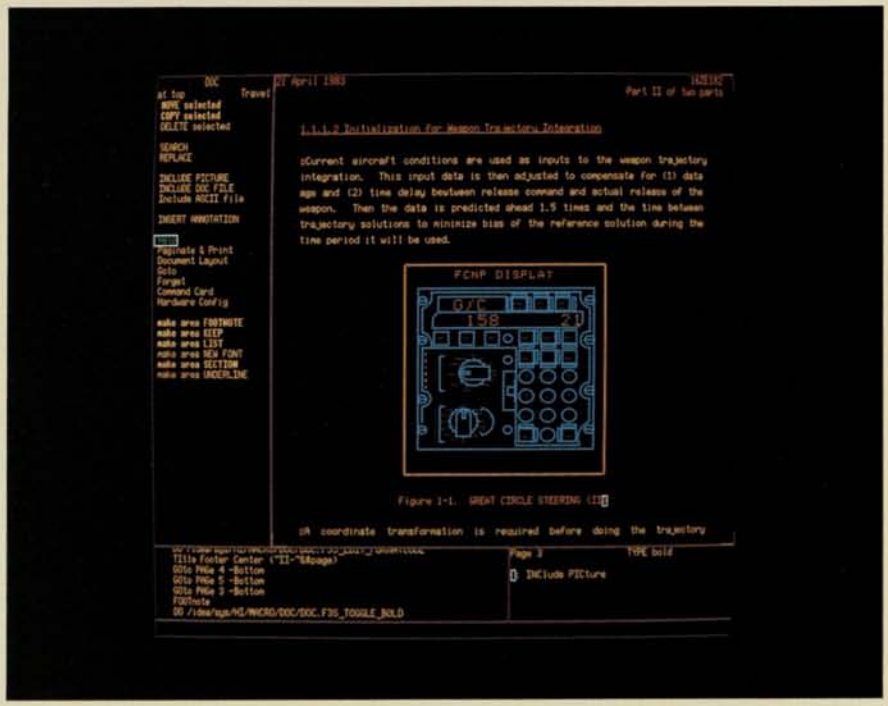
Research shows that over half the typical engineer's day is spent on non-design tasks, such as documentation and conference time. Accordingly, Mentor has integrated these functions into its system in a way that keeps the time and cost of your design "overhead" to an absolute minimum.

DOC is a professional documentation system several generations beyond conventional word processing. It includes the usual "cut and paste" editing features, but then expands to give you a series of features that completely cover the needs of the professional engineering environment.

For instance, you can reach into the Mentor database and pull diagrams into your text from sources such as schematics, symbol libraries, simulator and logic analysis displays.

Certain numbering tasks, such as figure and section numbers are eliminated altogether through automatic incrementing, even when nested. Footnotes are automatically placed and numbered. "Bookmarks" allow rapid travel to anywhere in the text. You can even annotate a document without affecting its existing format. And use multiple fonts within the same text, such as Bold, Greek, Math and many others.

Also, you can divide a single document into "sub-documents" to allow several writers to work on a single project.




## Electronic mail keeps you in constant touch.

Another obstacle to design productivity is the amount of conference time needed to keep a project moving forward. Here you're faced with countless schedule conflicts and meeting space arrangements.

Mentor solves this problem through electronic mail, which conquers both space and time when it comes to engineering team communications. Each user has an electronic memo writer to send messages and an electronic mailbox to receive them.

With electronic mail, your team communicates better, works faster, and achieves superior end results.

The IDEA 1400 workstation provides a cost-effective vehicle to handle project management and documentation tasks.

# Mentor


**Mentor Graphics Corporation**
8500 S.W. Creekside Place
Beaverton, Oregon 97005-7191
503-626-7000
Telex 4742102 MENTOR

**Mentor Graphics Corporation**
2620 Augustine Drive
Suite 280
Santa Clara, CA 95051
(408) 727-9667


## U.S. Sales Offices

Lexington, MA
617-863-5776

Orlando, FL
305-422-1841

Fort Lee, NJ
201-585-9395

Dallas, TX
214-392-2992

Albuquerque, NM
505-296-3767

Minneapolis, MN
612-835-7007

San Jose, CA
408-294-9933

Beaverton, OR
503-626-7000

Irvine, CA
714-752-0200

## International Sales Offices

Mentor Graphics Japan
Akasaka Twin Tower Bldg.
17-22 Akasaka 7-Chome
Minato-K, Tokyo, Japan

Mentor Graphics (U.K.) Ltd.
Reading, Berks., England, RG7 1SS
0734-884888
Telex: 849975 MENTOR G

Mentor Graphics (Deutschland) Gmbh
Eching, West Germany
089/319-1003
Telex: 5214474 MENT D

Orion Technologies Ltd.
1 Korazin Street
Givatayim, 53583 Israel
03-772031
Telex: 32188 CLLSY

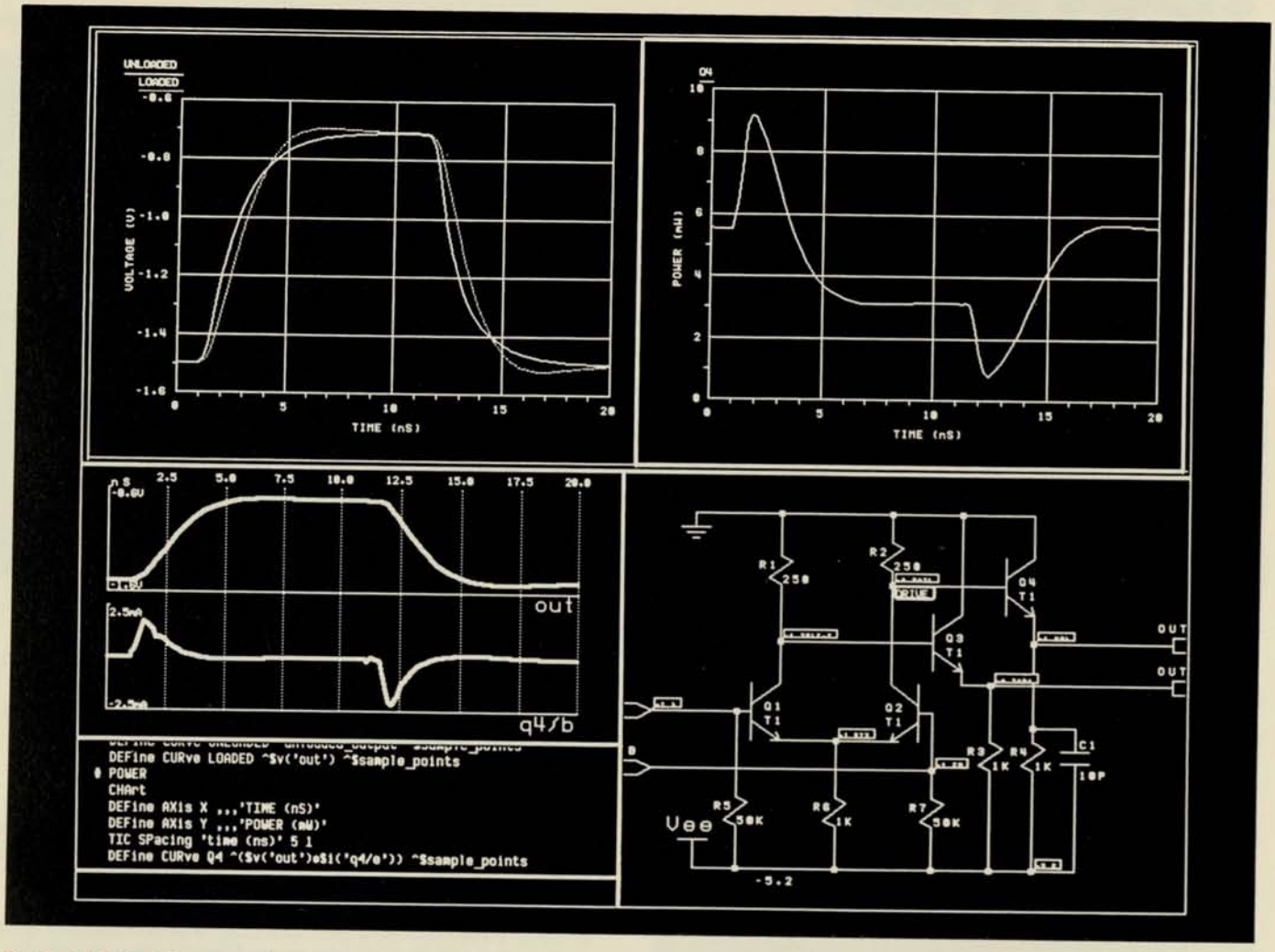
Mentor Graphics (France) SARL
Centre Commercial et Industriel Silic
59-61 Rue des Solets
552 Rungis
France



Printed in USA 1008A 02/84

# MSPICE

**Mentor**




## MSPICE Overview

Mentor Graphics offers MSPICE (Mentor Spice), a program for interactive analog circuit analysis. MSPICE is offered for use in conjunction with Mentor IDEA 1000 computer-aided engineering (CAE) systems, which are designed with the depth and flexibility to support all aspects of complex electronic circuit design and analysis.

Mentor SPICE consists of an interactive graphics pre- and post-processor, coupled with the analytical strength of the most up to date version of Berkeley SPICE. When used in conjunction with the IDEA 1000 workstation, you have an integrated, comprehensive computerized circuit design system versus a stand alone design tool, such as SPICE. Rather than entering circuit descriptions in a time-consuming and error-prone textual manner, MSPICE runs directly off of the design database you create with the IDEA 1000 Design Creation Tools. This same database is also used by the other IDEA 1000 design verification and physical design tools.

During simulation, MSPICE provides you with highly-interactive analysis control including the ability to monitor and stop analysis, perform user-specifiable custom analysis, and parameter modification on-line. After the simulation is complete, MSPICE's general purpose graphing capability is used to present output in either MSPICE format or any other customized format you specify. Modifications to your existing versions of SPICE are not lost since the MSPICE pre- and post-processor can interface with almost any analog circuit simulator. Regardless of whether that simulator runs on your IDEA 1000 CAE system, another IDEA 1000 in the network, an IDEA high-performance computational system, a mainframe, or special purpose array-processor, MSPICE supports your analysis requirements.

## Creating the design for use with MSPICE

The design to be simulated with MSPICE is created with the Structured Logic/Circuit Design module of the IDEA 1000 system. This integrated set of software tools permits the creation of circuit designs through the user interface in a graphics-oriented format. Circuit components are represented by a set of analog symbols available from the Mentor-supplied parts library, or through symbols that you create with the IDEA 1000

# MSPICE

Symbol Editor. Using the Network Editor, you connect the symbols through nets; lines which represent various electrical paths, to create the circuit schematic. Values for resistors, capacitors, and other passive components are assigned via properties associated with the symbol, and are visible on the schematic. If the symbol represents a diode, transistor, or other active device, its values or "model" are contained in a separate file. A property associated with the symbol identifies the model within this file. If you have more than one active device with the same model, you simply assign them the same property.

You may also design hierarchically with MSPICE. The IDEA 1000 Design Creation Tools allow the schematic to contain higher-level components than the basic passive and active components. For example, a symbol representing an op amp could be created with the Symbol Editor and used in a schematic which contained various passive and active components. The op amp symbol would be fully described by a lower-level schematic containing a set of interconnected transistors. Furthermore, using the hierarchical design capabilities of the IDEA 1000 Design Creation Tools, you may create circuit schematics which represent basic logic functions such as and/or gates, flip-flops, etc., and use the symbols which represent these logic functions in higher-level schematics. Having done this, you now have the design verification capabilities of the IDEA 1000 Interactive Logic Simulator, Interactive Timing Verifier, Design File Analyzer, and MSPICE available.

Once you have entered the circuit schematic on the IDEA 1000, the only step required before entering MSPICE is to expand the design down to the interconnection of primitive components acceptable for analog simulation. This is accomplished through the use of the IDEA 1000 Design Expander. While using the Design Expander, you may wish to utilize the Design File Analyzer which provides the capability to query the entire design at once, perform design rule checks, and perform specialized computations utilizing any IDEA 1000 or user-defined properties associated with symbols. Once a design is expanded, you may perform an unlimited number of MSPICE simulations without re-expanding. Changes made to component values are handled within MSPICE, and therefore, do not require re-expansion. Certain topological changes such as deleting components also do not require re-expansion. For other types of topological changes, only those schematics which have changed need to be re-expanded, rather than the whole design. This greatly increases the efficiency of the expansion process.

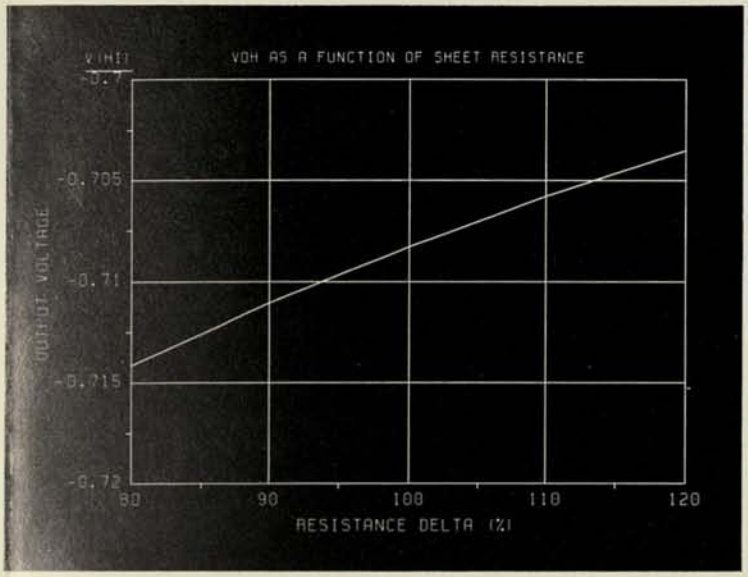
## Performing the simulation

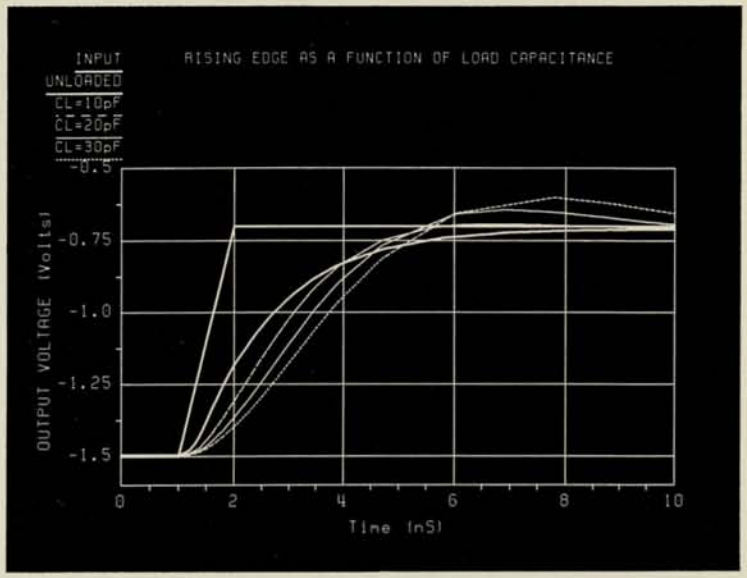
To start the circuit simulation, you simply invoke the MSPICE command specifying the appropriate design name. The pre-expanded design file is transparently translated into syntax required by the analysis routines within MSPICE. Also, the "models" for active devices are

```
*
* THIS IS A MACRO THAT SWEEPS THE RESISTANCE OF EACH OF THE RESISTORS IN THE
* CIRCUIT FROM 80% TO 120% OF NOMINAL
*
DIMENSION LO_DCOPS 5    * ARRAY TO CONTAIN THE Vol'S FOR THE CIRCUIT
DIMENSION HI_DCOPS 5    * ARRAY TO CONTAIN THE Voh'S FOR THE CIRCUIT
DIMENSION SHTRS 5       * ARRAY TO CONTAIN THE SHEET RESISTANCES USED
LOOP I 0 12
    ASSIGN SHTR ^(I* .1)
    ASSIGN SHTRS(I-7) ^SHTR
    *
    * FIRST RESET ALL THE SPICEPARS IN THE CIRCUIT TO THEIR ORIGINAL VALUE
    *
    CHANGE SIMULATION PROPERTY SPICEPAR -ALL
    *
    * NOW LOOP THROUGH ALL THE RESISTANCES AND MODIFY THEM BY 'SHTR'
    *
    ASSIGN INST ^#FIRST_INSTANCE('R')    #GET THE 'FIRST' INSTANCE OF A RESISTOR
    LOOP
        IF ^(INST = '') THEN ; EXIT LOOP ; END IF ;  # END LOOP WHEN NO MORE RESISTORS
        # CHANGE THE VALUE OF THE RESISTOR TO ITS CURRENT VALUE TIMES THE SHEET R PERCENTAGE
        CHANGE SIMULATION PROPERTY SPICEPAR ^INST ^(SHTR * $INSTANCE_PROPERTY('SPICEPAR',INST))
        ASSIGN INST ^#NEXT_INSTANCE('R')
    END LOOP
    *
    * RUN THE ANALYSIS AND SAVE THE DATA
    *
    RUN
    ASSIGN LO_DCOPS(I-7) ^$DCOP_V('OUT')
    ASSIGN HI_DCOPS(I-7) ^$DCOP_V('OUTB')
    *
END LOOP
*
* AND THE HIGH OUTPUT VOLTAGE AS A FUNCTION OF SHEET RESISTANCE
*
CHART
DEFINE AXIS X
DEFINE AXIS Y
DEFINE CURVE V(HI) ^HI_DCOPS ^(SHTRS * 100)
AXIS TITLE Y 'OUTPUT VOLTAGE'
AXIS TITLE X 'RESISTANCE DELTA (%)'
TIC SPACING X 10 5
AXis RAnge y - 72 - 7
TIC SPacing y .005
CHArt Title 'VOH AS A FUNCTION OF SHEET RESISTANCE'
```

appended to the input data being prepared for simulation. You then interactively enter any set-up commands required to control the simulation to be performed. Set-up commands include setting the temperature parameter or any other parameter used to define component values, setting any simulation control values, specifying the type of analysis to be performed (DC operating point, AC, DC, transient, etc.), simulation monitor commands, and setting force parameters for circuit power supplies and inputs. Since power supplies are handled as set-up commands and currents are handled directly by MSPICE output, you no longer need to put voltage sources in your circuit. As you interactively enter set-up commands, they are translated into SPICE analysis commands and appended to the previous input data. Only when you type "RUN" is the input data set to the analysis portion of MSPICE. Therefore, changes to set-up commands can be made prior to the actual execution of the simulation.

Unlike batch versions of SPICE, you retain complete control of the simulation with MSPICE. Instead of the graphics display going blank during the execution of the simulation, you are presented with constantly updated status information on your simulation in an easy to understand graphics format. Also during simulation, a copy of the schematic is displayed on the screen complete with status windows next to the nodes specified in the monitor commands. As the simulation is progressing, signal values are displayed in these windows. Another portion of the screen contains waveforms for selected points in the circuit that you have specified. During simulation, these waveforms are incrementally generated and displayed on the screen. Status information is also given in textual form and provides you with information on whether the simulation is still active, plus the current value of dependent variables in the circuit. With all this information available, you can



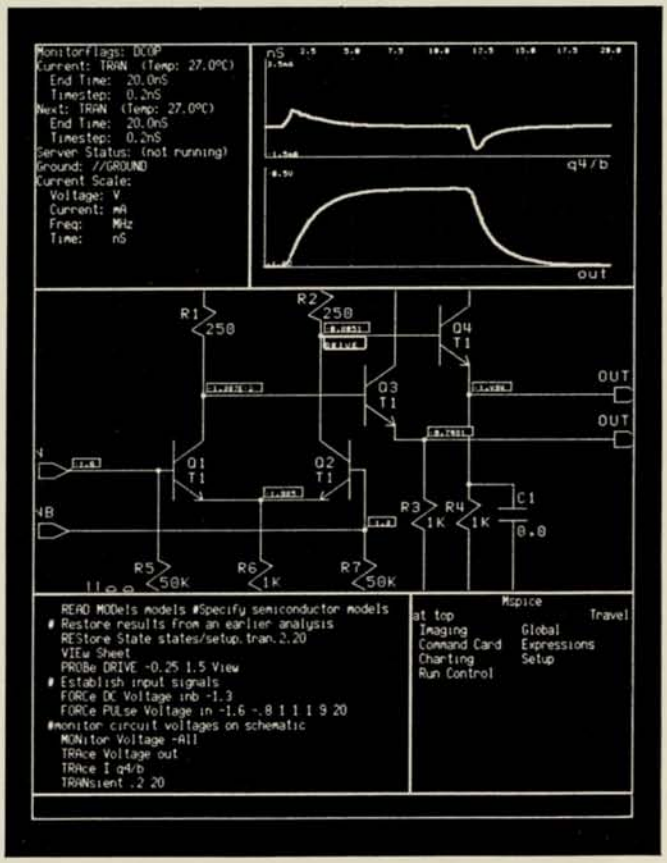




easily determine whether the simulation is progressing satisfactorily, or whether modifications to the circuit or parameters need to be made. If the latter is the case, you can simply stop the analysis at any time with results computed to that point available for study. No longer will you have to wait with crossed fingers for the results of your simulation. Instead, you will make optimum use of your IDEA 1000's computing resources, and more importantly, your own valuable time.

In many cases, after you have decided upon the changes that your circuit requires, you need only enter new set-up commands and run the simulation again. Changes such as defining new power supply values, changing component values, and updating simulation control commands can be made without leaving MSPICE. In some cases, adding and deleting components can be performed without changes to the circuit schematic. Through the use of the IDEA 1000 macro command language, complete with conditionals and looping, you can quickly develop and execute with one command a sequence of multiple set-up and RUN commands. This provides you with the capability to perform analyses which require multiple simulations; a capability not available with batch circuit simulators.

## Post simulation processing

After you have completed the simulation, the MSPICE post-processing capability is available to view and analyze the results. Unlike batch versions of SPICE, you need not specify which points in the circuit you want to view prior to simulation. Upon inspection of the signal values of waveforms presented while monitoring the simulation run, you may request information on any other points in the circuit without re-running the simulation. Unlike many




versions of SPICE, zero-valued voltage sources are not required to display currents within MSPICE. As a result, both voltage and current are directly available to you as output. Output is presented as waveforms in true graphical format as opposed to printer plots available in other simulators. Graphs are easily obtained via a single

Mentor Graphics Corporation
8500 S.W. Creekside Place
Beaverton, OR 97005-7191 (503) 626-7000

command with the detail of axis labeling and scaling handled by MSPICE. Also, you have the option of specifying your own labeling and scaling dependent upon your application. On color IDEA 1000 systems, each waveform in a graph is represented in its own color, while on monochrome IDEA 1000 systems, each waveform is differentiated by using narrow, bold, dotted, or dashed lines. In either case, waveforms are easily discernable, and there is no limit to the number of waveforms on a single graph.

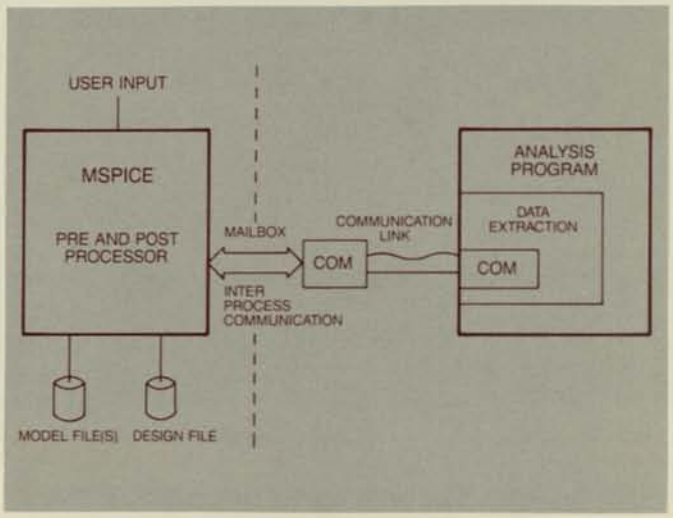
Through the use of expressions allowed in the MSPICE post-processor, waveforms may be generated utilizing any parameter in the circuit and any voltage or current computed by the simulation. Therefore, you are not limited to waveforms representing voltage and current. This allows you to generate waveforms representing such values as instantaneous power. Through the use of arrays within the macro command language, results computed from each simulation run may be stored away for processing at a later time. Coupling arrays with looping and conditionals in macros along with component value modification gives you the ability to perform specialized analysis which require multiple simulation runs. These waveforms from different simulation runs may be combined on the same graph, and through expressions a waveform can be constructed which contains the output from multiple simulations.

## MSPICE Architecture

As delivered with the IDEA 1000 CAE system, MSPICE provides a completely integrated turnkey circuit analysis system. However, the architecture of the MSPICE product has been designed in such a manner that you are not required to choose between MSPICE and your own proprietary version of SPICE which may have been tailored to your specific design and environment.

MSPICE consists of two main software components; a graphics-oriented pre- and post-processor and the simulation server. The pre-processor is used to convert design data created by the IDEA 1000 Design Creation Tools into a format acceptable by the simulation server, while the post-processor creates graphs from results computed by the simulation server. These MSPICE processors always reside on an IDEA 1000 CAE system.

The simulation server consists of three components; the analysis program which computes simulation results, a communications link used to pass data between the pre- and post-processor and the simulation server, and subroutines used to extract analysis data at each analysis




point. As required, you can use the Berkeley SPICE analysis program provided within the MSPICE server or substitute your own. Circuit analysis programs containing your own modifications may be included in the MSPICE server by incorporating the data extract subroutines into your analysis program. The simulation server may run on any IDEA 1000 CAE system, an IDEA high performance computational system, mainframe, or special purpose array processor. Transporting your simulation server, running on a mainframe or other system, to an IDEA 1000 CAE system is easily done since the IDEA 1000 system supports programs written in ANSI standard FORTRAN 77.

When the simulation server is run on an IDEA 1000 CAE system or IDEA high-performance computational system, the communication link consists of the Apollo DOMAIN interprocess communication facility, and if necessary, the DOMAIN local area network. If your simulation server is not running on an IDEA 1000 CAE system, the communications link also includes one of the many Apollo DOMAIN communication gateways.

## U.S. Sales Offices

Fort Lee, NJ
201-585-9395

San Jose, CA
408-294-9933

Lexington, MA
617-863-5776

Orlando, FL
305-422-1841

Irvine, CA
714-752-0200

Dallas, TX
214-392-2992

Beaverton, OR
503-626-7000

Albuquerque, NM
505-296-3767

Minneapolis, MN
612-835-7007

CALIFORNIA AUTOMATED DESIGN, INC.
Santa Clara, CA 408-727-9667

## International Sales Offices

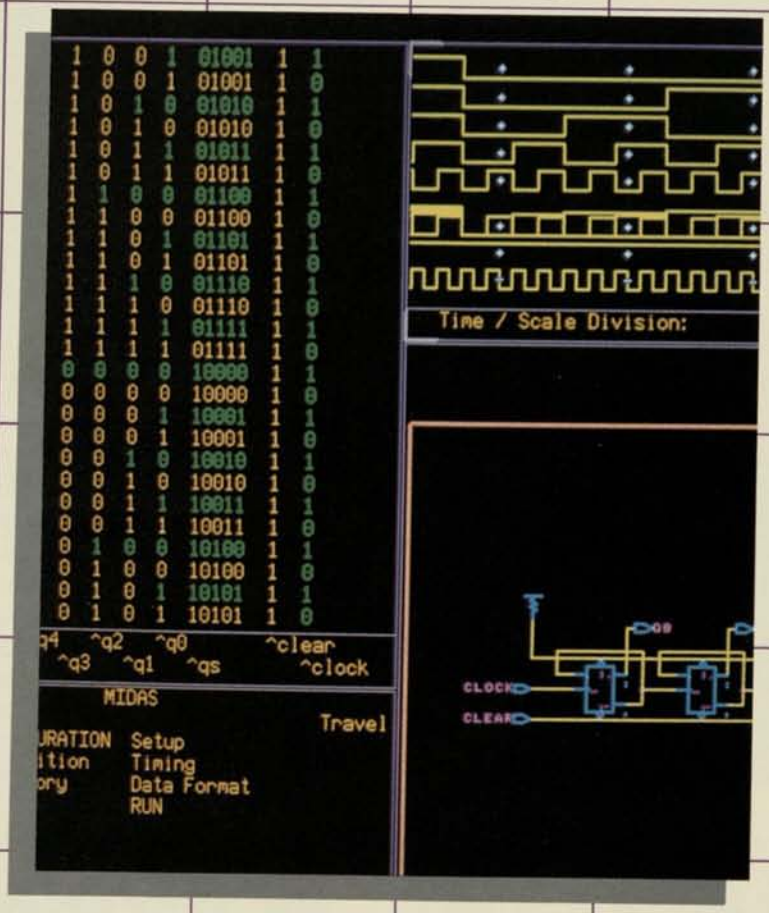
Mentor Graphics Japan Co., Ltd.
Toshima-ku, Tokyo 170, Japan
(03) 989-7950

Mentor Graphics (U.K.) Ltd.
Reading, Berks., England, RG7 1SS
0734-884888
Telex: 849975 MENTOR G

Mentor Graphics (Deutschland) Gmbh
Eching, West Germany
089/319-1003
Telex: 5214474 MENT D

Mentor

# TEST AND MEASUREMENT TOOLS




INTERACTIVE STATE ANALYSIS
INTERACTIVE TIMING ANALYSIS

# TEST AND MEASUREMENT TOOLS

## An Overview

The IDEA 1000 is a self-contained, computer-aided engineering (CAE) workstation dedicated to the design and analysis of complex electronic circuitry. The system is integrated with Apollo DOMAIN computer systems linked together in a distributed network architecture. This way, each operator receives the benefit of independent 32-bit local processing and high resolution graphics, and still has full access to a common full relational data base shared by the Mentor system as a whole. In this networked environment, the user does not experience the performance degradation typical of a multi-user central host architecture.

Within the IDEA 1000 system are a number of software modules used to implement the various phases of the logic or circuit design process. These include Structured Logic/Circuit Design, Interactive Logic Simulation, Timing Verification, Physical Layout, Circuit Simulation, Project Management, Document Preparation, Formattable Netlist/Plot and CAE/CAD Programming. Each of these modules is tied into the high-performance full relational data base management system for fast, efficient access to any required information across the network (Figure 1).

In addition to these software modules, there is a hardware/software Test and Measurement module which is fully integrated with both the IDEA 1000 system software and the Apollo DOMAIN computer system. This module is called the MIDAS 7000, and extends Mentor's CAE tool set to cover the final phase of the hardware design cycle, verification and test. With MIDAS 7000 installed, the IDEA 1000 CAE system covers every phase of circuit design, from product definition to real-time verification of the completed hardware.

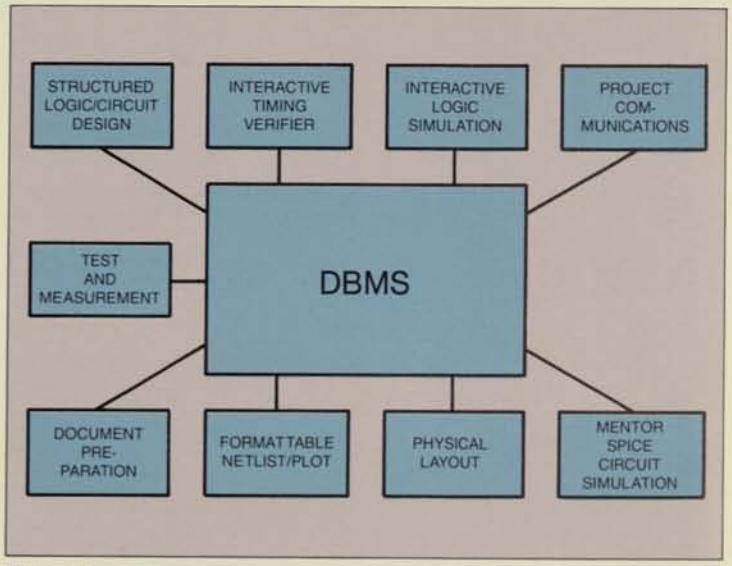



Figure 1

## MIDAS 7000. The Mentor Interactive Design Analysis System.

MIDAS 7000 is a powerful logic analysis system that provides full trace capabilities for both state and timing analysis, with state acquisitions up to 80 channels and timing measurements up to 100 MHz. Since it is fully integrated into the IDEA 1000 system, the user benefits from many capabilities not found in a conventional, stand-alone logic analyzer:

**Standardized User Interface.** All aspects of the MIDAS 7000's operation are controlled from the standard IDEA 1000 user interface, which takes advantage of the workstation's high-resolution color graphics and programmable keyboard. During logic analysis, the user is presented with the same graphic interface format and procedures used with all other IDEA 1000 design tools. This unified interface promotes faster, more accurate work, with no loss of continuity when moving from one tool to another.

**Automatic Comparison of Real-Time and Simulated Data.** Any real-time data acquired by MIDAS 7000 can be retained for powerful post-acquisition processing by the IDEA 1000 workstation's 32-bit CPU. One such application is the comparison of data generated by Mentor's Interactive Logic Simulator and real-time data later acquired by MIDAS 7000 from the actual hardware. This capability gives you a fast, accurate means of checking and verifying hardware performance against the schematic model without ever leaving the IDEA 1000 design environment.

**Performance Analysis.** During hardware/software integration, it is often important to monitor your code's real-time execution and analyze its performance in terms of design specifications. With MIDAS 7000, you can acquire real-time software data, and run it through post-acquisition processing that presents its performance in a graphic format, such as a frequency histogram.

**Mnemonic Disassembly.** The design and debug of microprocessor-based systems often calls for the acquisition and display of software flow as executed on the bus of the target hardware. After a MIDAS 7000 has acquired the software data, the IDEA 1000 can call upon stored mnemonic disassembly tables to convert your code into readable form. This includes both standard microprocessor or special user-defined instruction sets.

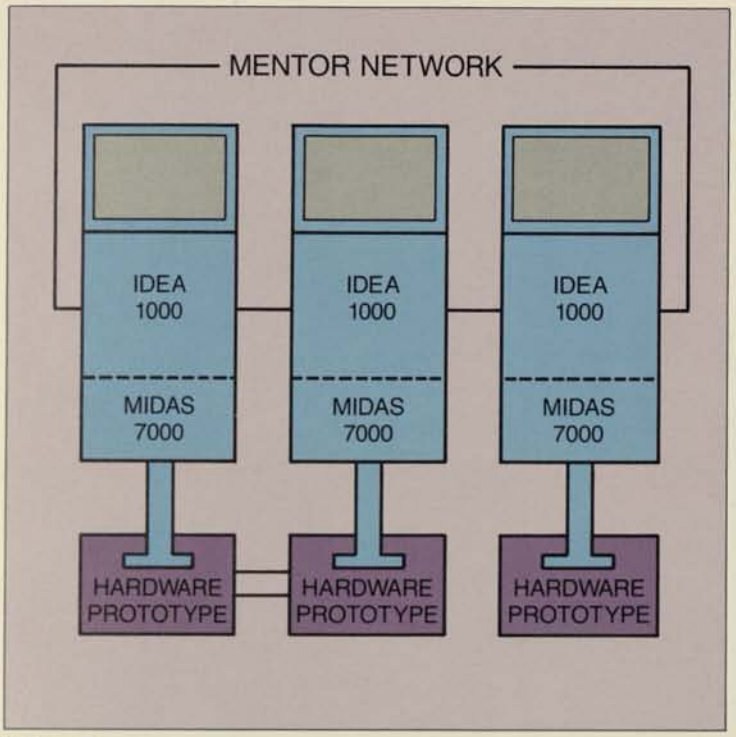



Figure 2

## An Expandable Test and Measurement Architecture

The MIDAS 7000's data acquisition hardware has an expandable modular architecture which accepts individual test and measurement instrumentation cards. For logic analysis, you can install exactly the configuration of state and timing that your application requires. In addition, you will be able to install other types of instrumentation cards as they become available. This way, you have a test and measurement tool with the same room for growth, depth and flexibility found throughout the IDEA 1000 tool set.

## A fully integrated logic analysis tool.

MIDAS 7000 makes logic analysis an integral part of the IDEA 1000 system through its tightly coupled position in the system architecture (Figure 2). Data acquisition probes from MIDAS attach to the hardware under test, and acquired data is stored first in a local data acquisition memory and then uploaded to the local workstation memory, where it can be either stored on disk, processed, displayed or output to peripheral devices.

Because the IDEA 1000 interface allows fully transparent access to both hardware and software

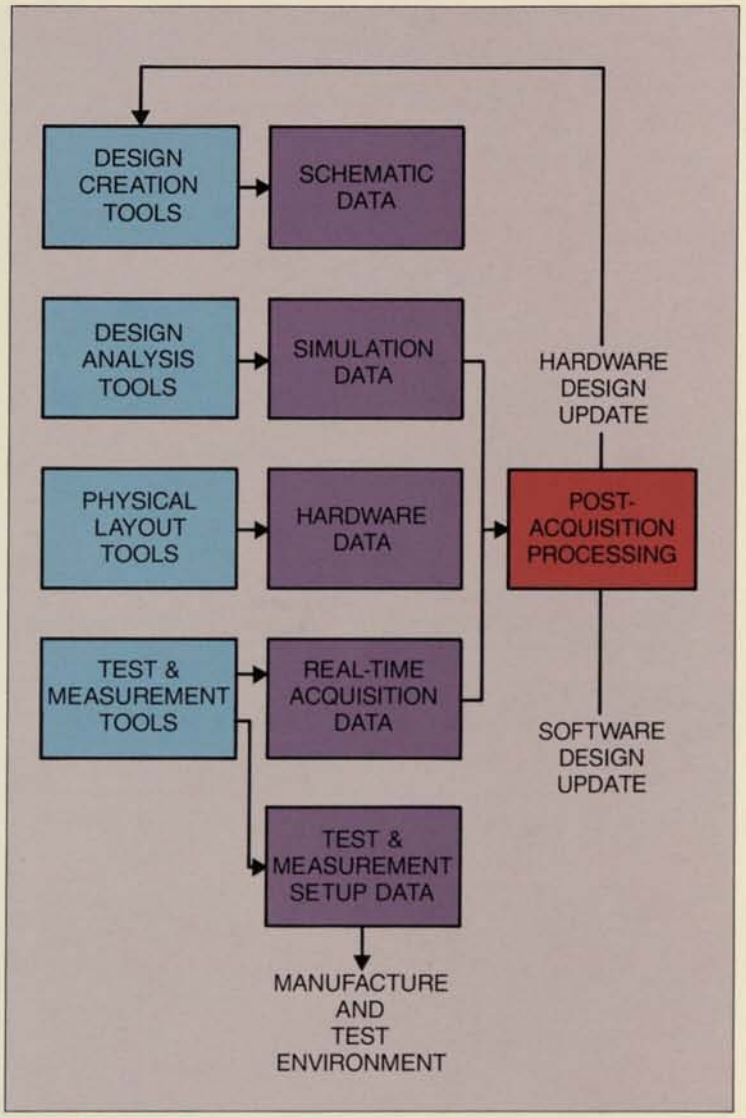
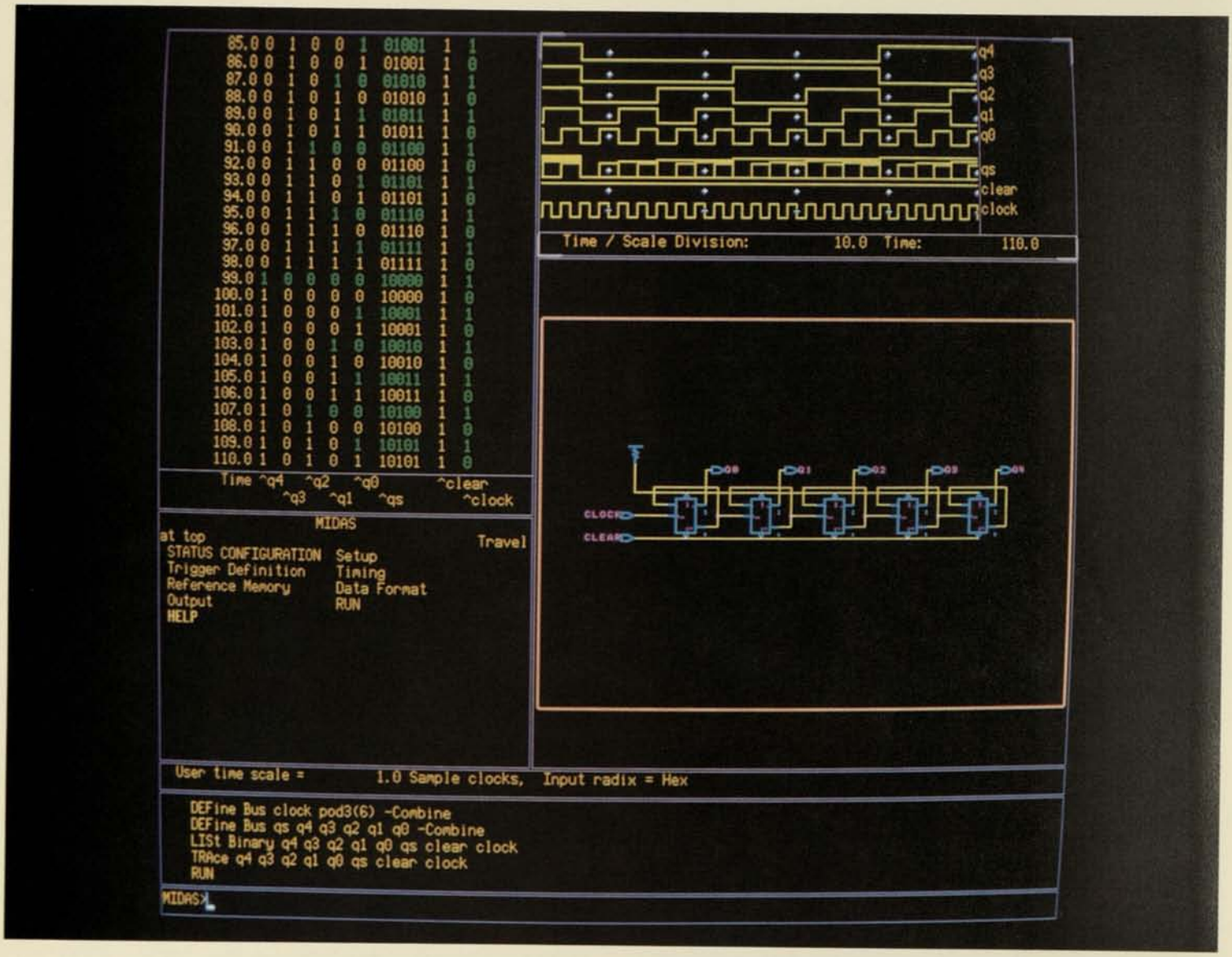



Figure 3

tools anywhere on the network, the MIDAS 7000 analysis functions can be controlled from any workstation on the network. Also, acquired data, reference data, and post-acquisition processing programs can be copied or viewed from any network node.

Figure 3 shows how the MIDAS 7000 test and measurement tools are an integral part of Mentor's support for the design cycle through computer-aided engineering. At the beginning of the cycle, Design Creation Tools allow schematic information to be graphically edited for entry into the design's database. Next, Design Analysis Tools allow the software version of the design to be simulated in terms of logic and timing. Physical Layout Tools then allow the design to be expressed as a physical entity, such as a gate array. To complete the cycle, MIDAS 7000's Test and Measurement Tools allow the

# TEST AND MEASUREMENT TOOLS




physical version of the design to be tested and verified in terms of the original functional specifications. At each and every step in the design cycle, you work through the same Mentor graphics user interface.

Hardware test and verification procedures are ideal candidates for the powerful post-acqusition processing capabilities of the IDEA 1000 System. As mentioned, this includes automatic comparison of simulation and data acquisition files. The information obtained from post-acquisition processing can then be used to update the original hardware design files or to modify the product's software.

An additional benefit of MIDAS 7000 is that test and measurement procedures developed during the design stage of product development can be transported to the Production Engineering test department. Selected instrument setups, reference data and post-acquisition processing can all be retained on disk and then transferred to similar instruments and their controllers located in the manufacturing environment.

## Interactive State Analysis

The MIDAS 7000 Logic State Analyzer allows up to 80 parallel channels of synchronous state acquisition at sample rates up to 10 MHz. The acquisition memory for each channel is 4096 bits deep, which provides data acquisition windows wide enough to trace complex software transactions.

State data is sampled into memory through probes

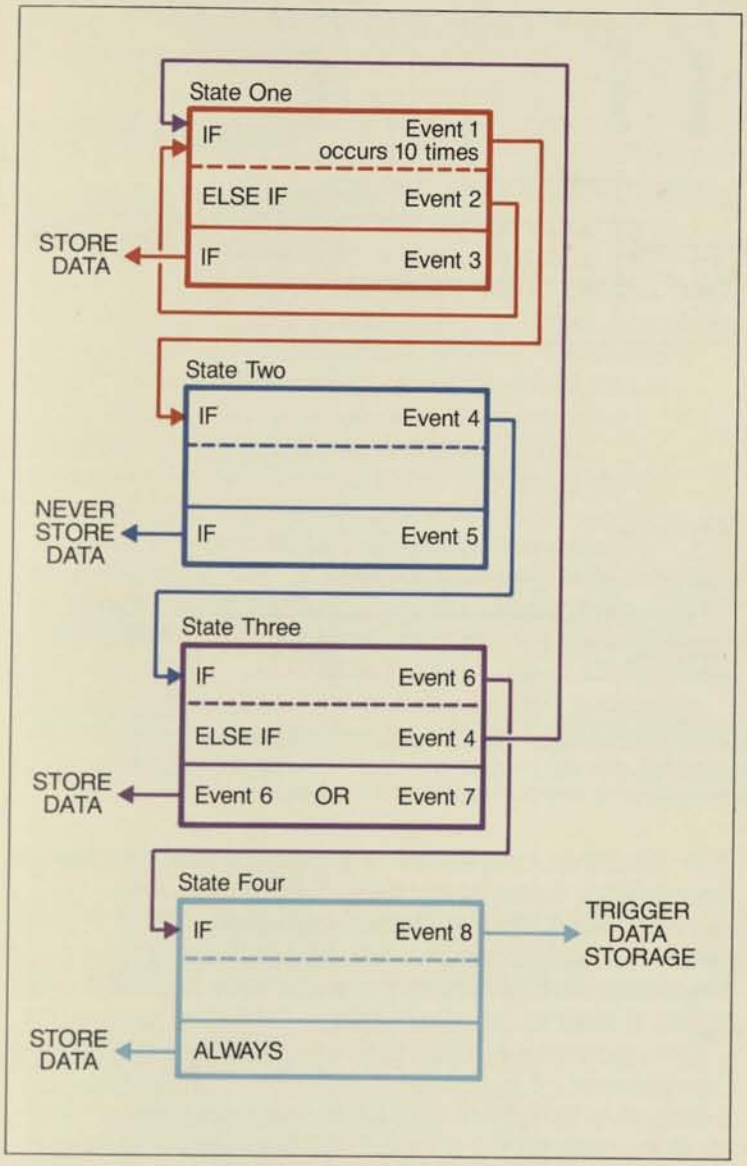



Figure 4 - The Interactive State Analyzer provides up to 15 distinct states for use in trigger definition and data storage qualification. This diagram shows how four such states could use jumps and branches to track complex software flow as it occurred in the system under test.

attached to hardware under test. The sample clock can be synthesized from up to five ORed clock inputs to define the master sample clock. Either the rising or falling edge of each clock signal may be defined as the sampling edge. For acquisition of multiplexed buses, there are also two HOLD clocks which eliminate the need for cumbersome double probing. (A HOLD clock latches the first phase of data sampled off the bus and holds it until the second phase data is present. Both phases are then simultaneously sampled into acquisition memory). Together, these clocking features provide enough flexibility to trace the bus transactions of popular but complex 16-bit and 32-bit microcomputer systems.

MIDAS 7000 data acquisition triggering for state information is implemented through a RAM-based state machine which allows up to 15 distinct states to be defined. During data acquisition, a given state can result in a number of actions, including jumping to another state, triggering and data storage qualification (Figure 4). This arrangement allows you to accurately acquire data even when convoluted software execution paths are followed.

Each state can have its own user-defined mnemonic label to simplify trigger definitions. Within each state, you can call upon up to four word recognizers which define the event or events comprising the state. When an event comes true, its assigned word recognizer initiates a preprogrammed action such as qualifying data for memory storage, halting data acquisition (triggering), starting a counter or transferring data acquisition control to another state.

To simplify the trigger definition process, a symbol definition table lets you specify various trigger events in convenient mnemonic form, which means you can name events using the same labels found in the source code, or any other meaningful terminology.

## Interactive Timing Analysis

The MIDAS 7000 Logic Timing Analyzer acquires parallel channels of data at sample rates up to 100 MHz. The nominal memory depth per channel is 510 bits, however the timing analyzer's architecture includes several features that significantly extend this memory depth during data acquisition.

One such feature is transitional timing, which only samples data into memory when a logic transition actually occurs on one of the data acquisition channels. When a logic transition is detected and sampled, a counter is activated which counts the number of samples until the next transition occurs. When data acquisition is complete, the timing analyzer uses the stored transitions and the counter delta time information to reconstruct a timing diagram representation of the logic activity as it occurred in real-time (Figure 5). This technique enables a relatively small amount of acquisition memory to capture data over an extremely wide time window. In effect, you can acquire real-time data windows up to 130,000 samples wide.

Another data compression feature is the "multi-triggers" mode, which allows you to make 16 "mini-

# TEST AND MEASUREMENT TOOLS

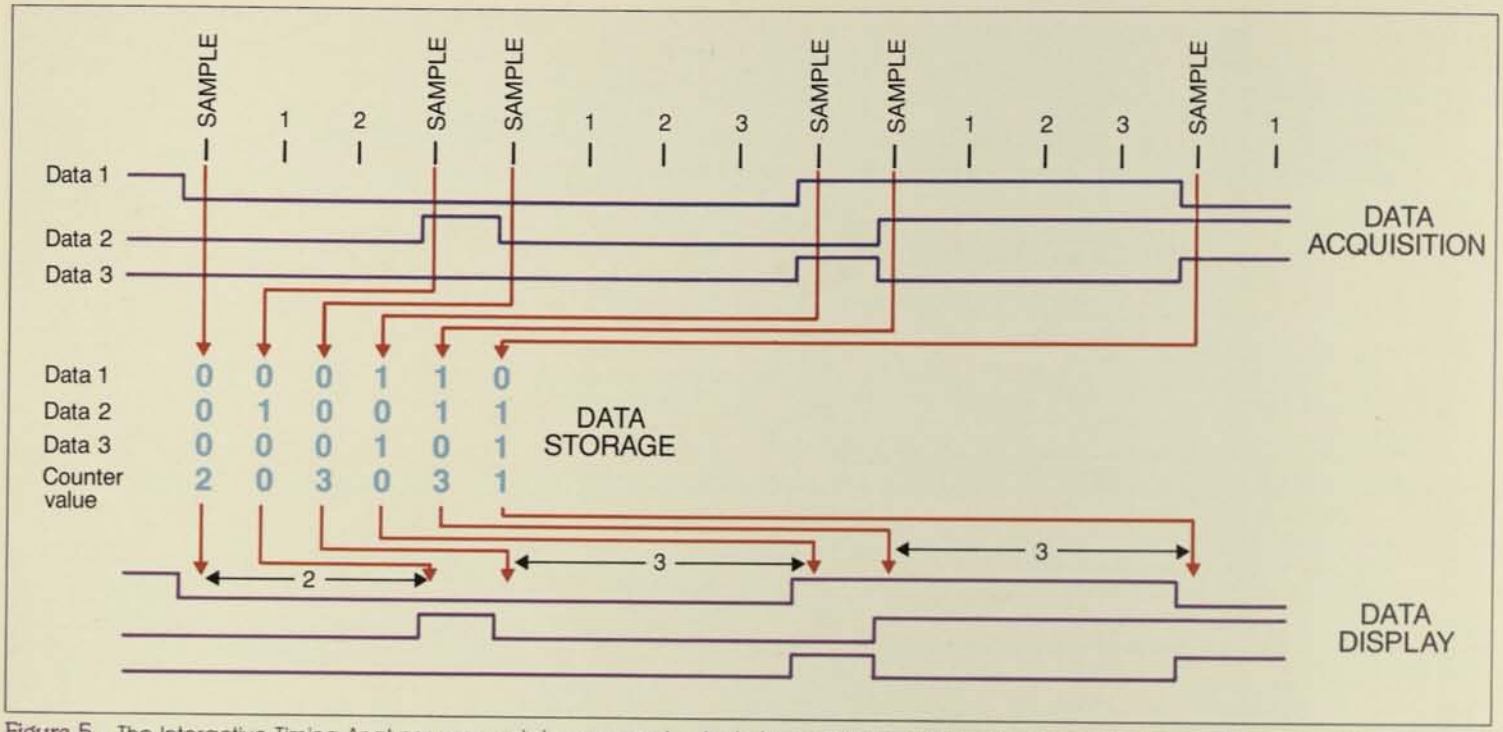



Figure 5 - The Interactive Timing Analyzer uses a data compression technique called transitional timing to expand the data acquisition window. Data sampling only occurs when a transition occurs on one of the channels, thereby minimizing storage requirements. A counter tracks the sample count between transitions, and this information is used to reconstruct a real-time representation of timing activity.

acquisitions" of at least 32 samples each in the course of a single data acquisition. Each of the 16 subgroups has the same trigger event located at its center. During data acquisition, the timing analyzer will acquire at least 32 samples when the trigger event occurs, and then stop storing data until the trigger event occurs again. This cycle is repeated 16 times until the acquisition memory is full. You now have 16 records of events surrounding successive occurrences of the trigger event, with all other data filtered out.

The MIDAS Logic Timing Analyzer can also be programmed into a cross-triggering relationship with the Logic State Analyzer. When a trigger event occurs on the timing analyzer it can be used to either arm or trigger the state analyzer. Conversely, a state trigger can either arm or trigger the timing analyzer. In either case, the data is time-synchronized to let you see the true relationship between software and hardware events as they occurred in the system under test.

## Post-acquisition processing

Any data acquired by either the MIDAS state or timing analyzers is automatically transferred to the main memory of the IDEA 1000 workstation, where it can be either processed or stored on disk for later processing. Since every IDEA 1000 workstation participates fully in a Mentor network and full relational database, the MIDAS 7000 Test and Measurement Tools and the data they produce are tightly linked to all other Mentor CAE tools and data.

One powerful form of post-acquisition processing is comparison of logic data produced by the Mentor Interactive Simulator with data acquired later in the design cycle by the MIDAS 7000 Logic State Analyzer. This comparison is carried out by a Mentor file compare command, and allows all discrepancies between the two data files to input to a third file as well as output to the display or a printer. This comparison technique gives you a powerful means of verifying that the physical hardware's performance conforms to the original design model specifications.

Mentor also will support microcomputer software/hardware integration by offering a series of disassemblers for popular 8-bit, 16-bit and 32-bit processors with bus cycle rates up to 10 MHz. Real-time software flow acquired by MIDAS 7000 off the bus of the hardware under test is post-processed by the disassembler software to provide data similar to an assembly source code listing.

Another Mentor post-acquisition is performance analysis, which statistically measures various aspects

of your code's execution as acquired off the bus of the system under test.

## I/O Functions

Since it is fully integrated into the IDEA 1000 system, MIDAS 7000 can take full advantage of all the system's input/output capabilities. All instrument setups and acquired data can be filed on disk and then accessed by any workstation located anywhere in the system. At an even higher level, these disk-based files can become part of powerful macro command files that automate entire test and measurement procedures, from setup and processing of acquired data to reporting and formatting the final results.

In addition, any MIDAS display of setups or acquired data can be output to a plotter to obtain hardcopy of test procedures and results. Also, logic state lists can be output to a printer. For fast, efficient communication across the network, any MIDAS data can be sent via Mentor's electronic mail system to selected individuals or groups of individuals for review or comments.

## Specifications

### MIDAS 7000 Interactive State Analyzer

#### HARDWARE SPECIFICATIONS

**INPUTS**

CLOCK
CLOCK CHANNELS: 5 OR'ed Sample clocks and 2 Hold clocks
INPUT LOADING OF CLOCK PROBE: 1 TTL FAST load (lin-low = 0.6 ma.)
MAXIMUM INPUT VOLTAGE: 0.5 V to 7.0 V.
DYNAMIC RANGE: 0 V to 5.0 V.
THRESHOLD: 1.6V, TTL
MAXIMUM SYCHRONOUS SAMPLE RATE: 10 MHz
MINIMUM CLOCK PULSE WIDTH: 25 ns.
CLOCK MODES: Normal; Demultiplex

DATA:
DATA CHANNELS: 16 to 80 input channels. 16 channels per P-2902 Data Probe
INPUT LOADING OF DATA PROBE: 1 TTL ALS load (lin-low = 0.1 ma.)
MAXIMUM INPUT VOLTAGE: 0.5 V to 7.0 V.
DYNAMIC RANGE: 0 V to 5.0 V.
THRESHOLD: 1.6 V, TTL
DATA SETUP TIME: 25 ns.
DATA HOLD TIME: 0 ns.
EXTERNAL BNC INPUT: 1 TTL load (lin-low = 1.6 ma.)

**OUTPUTS**

CLOCK PROBE:
REAL-TIME PROGRAMMABLE OUTPUT (FROM STATE MACHINE): 2 lines; open collector; internally pulled up by 4.7 Kohm
RUN/STOP LINE: open collector; internally pulled up by 4.7 Kohm
CROSS-LINK BNC:
CROSS-LINK LINE: programmable output; 50 ohm line driver

**ANALYSIS**

ACQUISITION MEMORY:
Configurable from 16 to 80 channels with 4096 memory depth per channel.
REFERENCE MEMORY:
80 channels × 4096
STATE MACHINE:
NUMBER OF INDEPENDENT STATES: 15
WORD RECOGNIZERS PER STATE: 4 total; 2 used for branching, triggering and output control; 2 used for storage qualification (AND'ed or OR'ed); all words—value, NOT, Don't Care.
OUTPUTS PER STATE: 2 real time output lines and 1 cross link line.
EVENT COUNTERS PER STATE: 1, programmable from 1 to 4096
EVENT COUNTER MODES: after N times, before N times or N times (N = 1 to 4096)
EVENT COUNTER INPUTS: sample clocks, store clocks or word recognizer occurrences

#### SOFTWARE SPECIFICATIONS

FORMAT: channel group assignment; display order; group labels; radix and polarity selection; sample and hold clock and mux/non-mux setup; Run/Stop polarity
TRIGGER: state machine programming; trigger position (begin, center, end and delay to 4096); storage qualification programming; symbolic state definition; Multiple Preview Acquisition selection; programmable start state; store state transitions
SYMBOL: word recognizer symbol name and value definition; find word and mask words
DISPLAY: HEX, OCTAL, BINARY and ASCII display formats; acquisition or reference; single line, page or block scrolling; reference edit; find word (scroll by value).
INPUT/OUTPUT: save and load setup and data to and from disk; or display; print data, symbols or trigger information

GO MODES:
Go once; Go forever; Go until acquisition = reference; Go until acquisition > < reference; Halt

#### MICROPROCESSOR SUPPORT

The MIDAS 7000 can be effectively used with all of the following microprocessors, plus more, using the unique universal clocking scheme:

| | | | |
|---|---|---|---|
| Z80 | 6800/6802 | 8080 | 16032 |
| Z8001 | 6801/6803 | 8085 | 9900 |
| Z8002 | 6809 | 8086 | 1802 |
| | 68000 | 8088 | |
| | 6805 | 80186/286 | |

### MIDAS 7000 Interactive Timing Analyzer

#### HARDWARE SPECIFICATIONS

**CLOCK**

MAXIMUM SAMPLE RATE:
ASYNCHRONOUS: 100 MHz. (10 ns. resolution) - internal clock
SYNCHRONOUS: 100 MHz. using external clock
CLOCK SPECIFICATIONS:
RANGES: 10 ns. to 10 ms. in 1,2,3....9 sequence
ACCURACY: Crystal controlled

**DATA:**

INPUT MODES:
16 CHANNEL MODE: Sample and store input data value with every clock transition; 508 bits memory per channel
10 CHANNEL TRANSITION MODE: Sample and store only on data level changes; 8-bit counter for delta time between transitions; up to 128K effective memory in special case
GLITCH CAPTURE:
5 ns. minimum pulse detection; glitch latch
PROBES:
INPUT IMPEDANCE: 1 megohm, 5 picoFarads
THRESHOLD RANGE (VARIABLE): +/− 5.5 volts
MAXIMUM NON-DESTRUCTIVE INPUT VOLTAGE: +/− 25 volts
SPECIAL HARDWARE CHARACTERISTICS:
MULTI-TRIGGERS MODE: capture up to 15 regions of 32 samples; all center triggered

**ANALYSIS:**

TRIGGER MODES:
1) Pattern occurrence with OR'ed edges (AND'ed levels AND'ed with OR'ed edges)
2) Pattern duration greater than a specified time
3) Pattern duration less than a specified time
4) Start of pattern (transition into pattern)
5) End of pattern (transition out of pattern)
6) Set-up time violation
7) Hold-time violation
8) All modes can be combined with Interactive State Analyzer cross-triggering and arming

#### SOFTWARE SPECIFICATIONS

USER MENUS:
FORMAT: 16 channel or transition mode selection; single or multi-triggers mode selection; sample or glitch mode; channel group assignment; threshold adjustment; trigger setup; trigger position; group labels; channel activity indicator
DISPLAY: timing waveform display; channel label assignment; vertical trigger setup; find word; delta time readout; window and cursor scrolling; magnification selection
LIST: state display of acquired data; absolute or relative time readout between acquisitions; data scrolling
INPUT/OUTPUT: save and load setup and data to and from disk; or display; print data or menu information

### State/Timing Cross Linkage

TRIGGERING:
State triggers timing; timing triggers state

ARMING:
State arms timing to begin looking for an independent trigger event; timing arms state to begin looking for an independent trigger event

**Mentor Graphics Corporation**
10200 S.W. Nimbus Avenue, G-7
Portland, OR 97223 (503) 620-9817
Telex: 4742102 MENTOR

Mentor Graphics Corporation is dedicated to providing the finest computer-aided engineering tools possible, and to ensuring that these tools follow an optimized long-term growth path that produces continuous gains in both productivity and design innovation.

## U.S. Sales Offices

Fort Lee, NJ
201-585-9395

San Jose, CA
408-294-9933

Lexington, MA
617-863-5776

Edina, MN
612-835-7007

Orlando, FL
305-422-1841

Irvine, CA
714-752-0200

Dallas, TX
214-392-2992

Albuquerque, NM
505-821-2775

Portland, OR
503-620-9817

## International Sales Offices

Mentor Graphics (U.K.) Ltd.
Reading, Berks., England RG7 1SS
0734-884888
Telex: 849975 MENTOR G

Mentor Graphics (Deutschland) GmbH
Eching, West Germany
089/319-1003
Telex: 5214474 MENT D

Mentor Graphics Japan Co., Ltd.
Toshima-ku, Tokyo 170, Japan
989-7950



Printed in USA 1007A 11/83

Mentor

# PROJECT MANAGEMENT TOOLS

## 1. The IDEA 1000

The IDEA 1000 Computer Aided Engineering Workstation provides a set of tools for a wide range of engineering activities, including:

- logic design
- logic simulation
- document preparation

You will view all of the tools through a common, global user interface. All tools are supported by a central data-base management system.

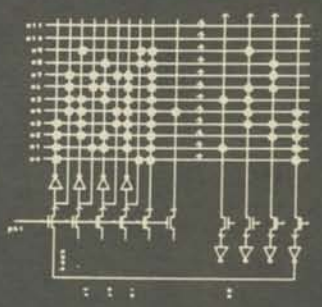


*Figure 1: PLA*

### 1.1 Document Preparation

The document preparation software (DOC) helps you throughout the design cycle. DOC is a page-oriented text editor tailored to the preparation of long documents (such as manuals).

DOC commands allow you to "leaf through" a document in a natural manner. Other commands simplify inserting, rearranging, modifying and deleting text or graphics.

DOC, which is tied to the data manager, can include schematics created in the logic design process (SLD).

DOCUMENT PREPARATION
PROJECT COMMUNICATIONS

# PROJECT MANAGEMENT TOOLS

## An Overview

IDEA 1000 is a self-contained, computer-aided engineering (CAE) system dedicated to the design and analysis of complex electronic circuitry. The system is integrated with Apollo DOMAIN computer workstations, linked together in a distributed network architecture. This way, each operator receives the benefit of independent 32-bit local processing and high resolution graphics, but still has full access to a common data base shared by the system as a whole, without the performance degradation of a central host architecture.

Within the IDEA 1000 system are a number of software modules used to implement the various phases of the logic or circuit design process. These include Structured Logic/Circuit Design, Interactive Logic Simulation, Timing Verification, Project Communications, Document Preparation, Formattable Netlist/Plot and CAE/CAD Programming. Each of these components is tied into a high-performance full relational data base management system for fast, efficient access to any required information (Figure 1).

Two of these basic modules, Project Communications and Document Preparation, provide a complete tool set to handle the management of electronic engineering projects. Almost any project requires extensive intercommunication between team members and documentation of the design effort. For this reason, Mentor has made communication and documentation tools an integral part of the IDEA 1000 computer-aided engineering system. Both of these tools have complete access to the systems's design files, allowing project management to proceed with a high degree of speed and accuracy.

The Document Preparation module takes advantage of the Apollo workstation's high resolution graphic display to provide fast and comprehensive text processing. A user can rapidly enter edit, format and file documentation which includes both text and pictures. The Project Communications module provides a fast, orderly flow of "electronic mail" between team members, allowing intersystem communications that overcome both time and location problems.

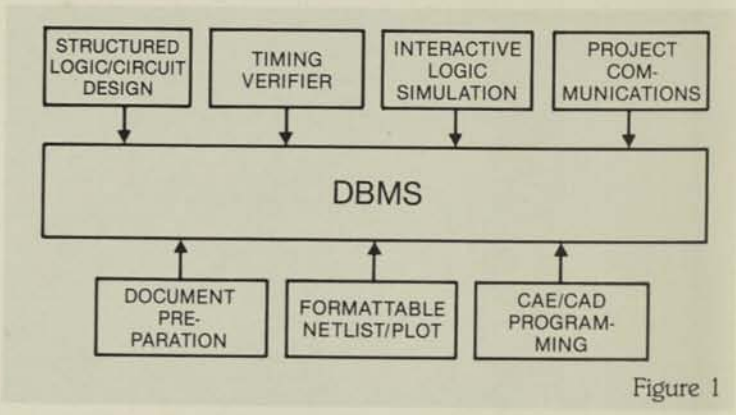



Figure 1

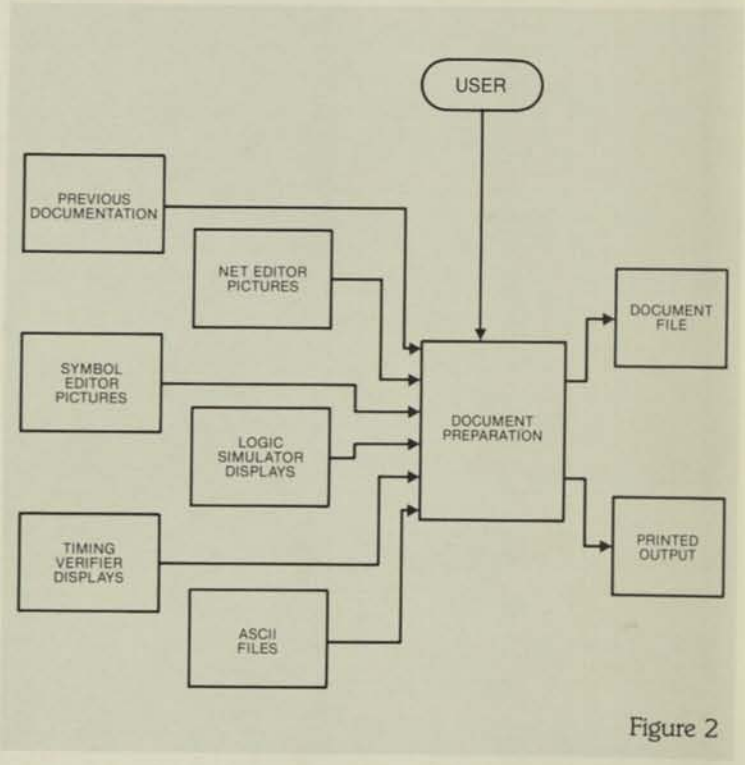



Figure 2

## Document Preparation

Documentation is an essential part of the logic/circuit design process, and the IDEA 1000 Document Preparation Module lets the user generate high quality documentation which can immediately draw upon design information maintained by the system's data base management system. When generating new text, the user can bring in previous documentation or ASCII files, plus graphics produced by the Symbol Editor, Network Editor, Logic Simulator or Timing Verifier (Figure 2). In this manner, users have all the resources to create timely and accurate documentation at their immediate disposal.

Document Preparation gives the user a powerful set of commands to handle text generation while maintaining the comfort level that a conventional "pencil and paper" approach provides. For instance, a command is provided to "turn" through pages of a filed report. Most important, Document Preparation displays generated text on a "what you see is what you get" basis, so the user immediately sees the effect of any command upon the substance of the document, rather than having to wait for later processing to see the final results.

The Document Preparation command set is generally divided into two subsets. One is the editing commands associated with the entry and modification of the text itself. The other is formatting commands which form the text into whatever visual arrangement is desired by the user. In addition there are commands which let the user travel quickly to any location within the document for viewing, editing or formatting purposes. These commands are possible because the editor is context-sensitive and understands concepts such as a "section", "page" or "list".

Figure 3 illustrates a previously completed document that has been called up for editing. Notice that a graphic

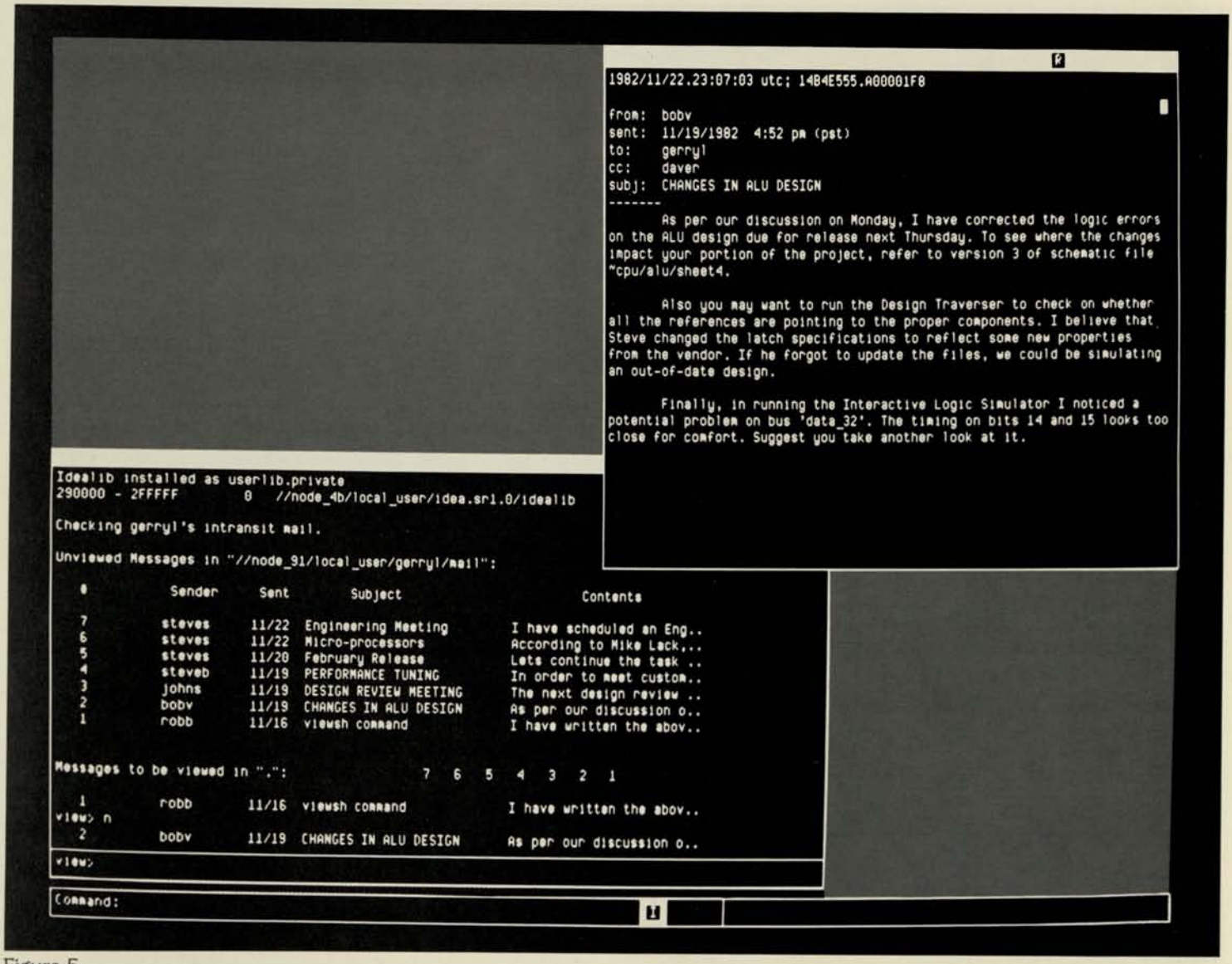



Figure 5

## Project Communications

Effective project management calls for intensive communication among team members creating the logic/circuit design. Communications must be channeled into an orderly flow that minimizes confusion and provides each team member with prompt, easy access to all other team members. The IDEA 1000 system meets these requirements through its Project Communications module, which provides fast efficient "electronic mail" service between users. Project Communications boosts productivity by substantially reducing the need to schedule and conduct time-consuming conferences.

With Project Communications, each system user can conveniently send, receive and file messages from or to any other system user. A single command sets up the display to enter a message and prompts the user to identify those who will receive the message directly and those who will receive "carbon copies". There is also a "subj:" prompt to identify the nature of the message. When the user completes text entry and sends the message, an additional field is automatically added that logs the date and time the message was sent.

Figure 5 shows how the display might appear when the user checks accumulated messages. The window on the left acts as the "mailbox" and draws upon the user's message directory to list all messages that have not yet been viewed. Each message is identified by a unique number and described in terms of the sender, the date sent, the subject and the first few characters of the text. The user can now view any of these messages in the window on the right.

When the messages are viewed, they can be "marked" for special attention, such as an immediate response. They can also be sent on to other users, filed, or responded to. The user's message directory will automatically be updated to show which of these actions was taken when the message was viewed. This way, users always have a current record of the status of their mail.

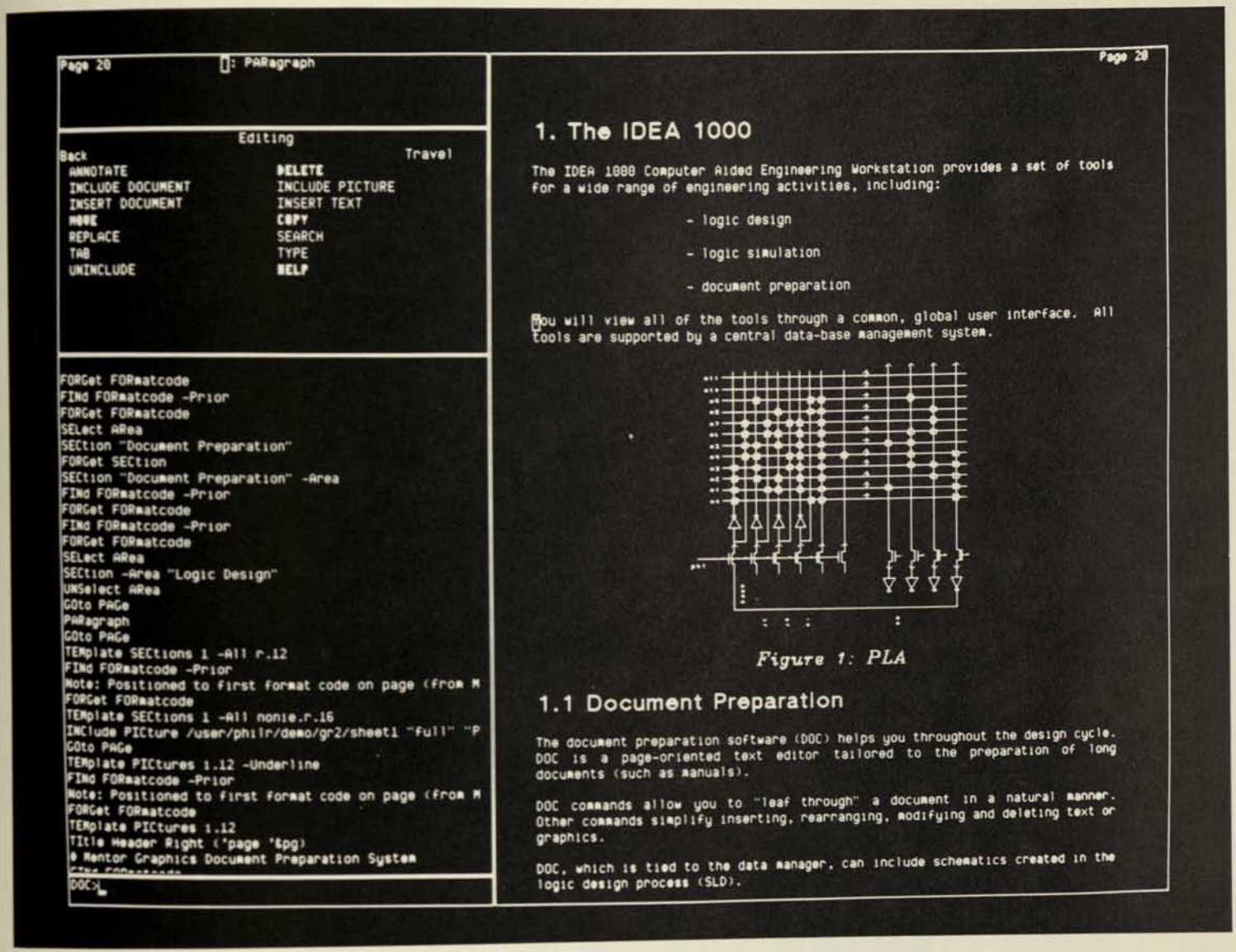



Figure 3

element has been included as an integral part of the text. This feature is unique to IDEA 1000 and illustrates how the user may easily draw upon the system data base. The PLA picture was included in the text through a single command which simply identified the picture according to its file name. Additional parameters can title the picture, scale it and indicate the font to be used for the picture title. Because the picture is on-line to the data base, it is automatically updated if someone edits it after its initial entry into the text.

Document Preparation also contains a full set of "cut and paste" commands which operate under cursor control for fast interaction. By selecting specific text areas with the cursor, the user can move text to any location within the document. It is also possible to include (in a procedure call) other documents residing in the data base. There are also sophisticated search and replace functions to seek out specified text and replace it with the new text if desired.

The Document Preparation formatting commands permit the text to be arranged into a visual configuration which suits the needs of the user. To streamline formatting, special "templates" can be used which control the effects of certain format codes on the text. One of the most basic templates is that which formats the document's page layout in terms of width, height, margins, page numbering information, and several other parameters. Another template sets the format for paragraphs by specifying the degree of indentation, the number of lines to skip at paragraph's end and the minimal number of paragraph lines that must appear at the bottom. This last parameter prevents unsightly "page breaks" where only the first line or two of a paragraph is printed on one page, with the rest carried over to the next page.

Other templates can specify parameters such as the font of picture titles, section titles, newly typed characters; and also the spacing between tab stops and the positioning of page titles.

Since the editor is context-sensitive, certain mundane tasks are completely eliminated. One example is section numbering. Every time the user indicates the beginning of a

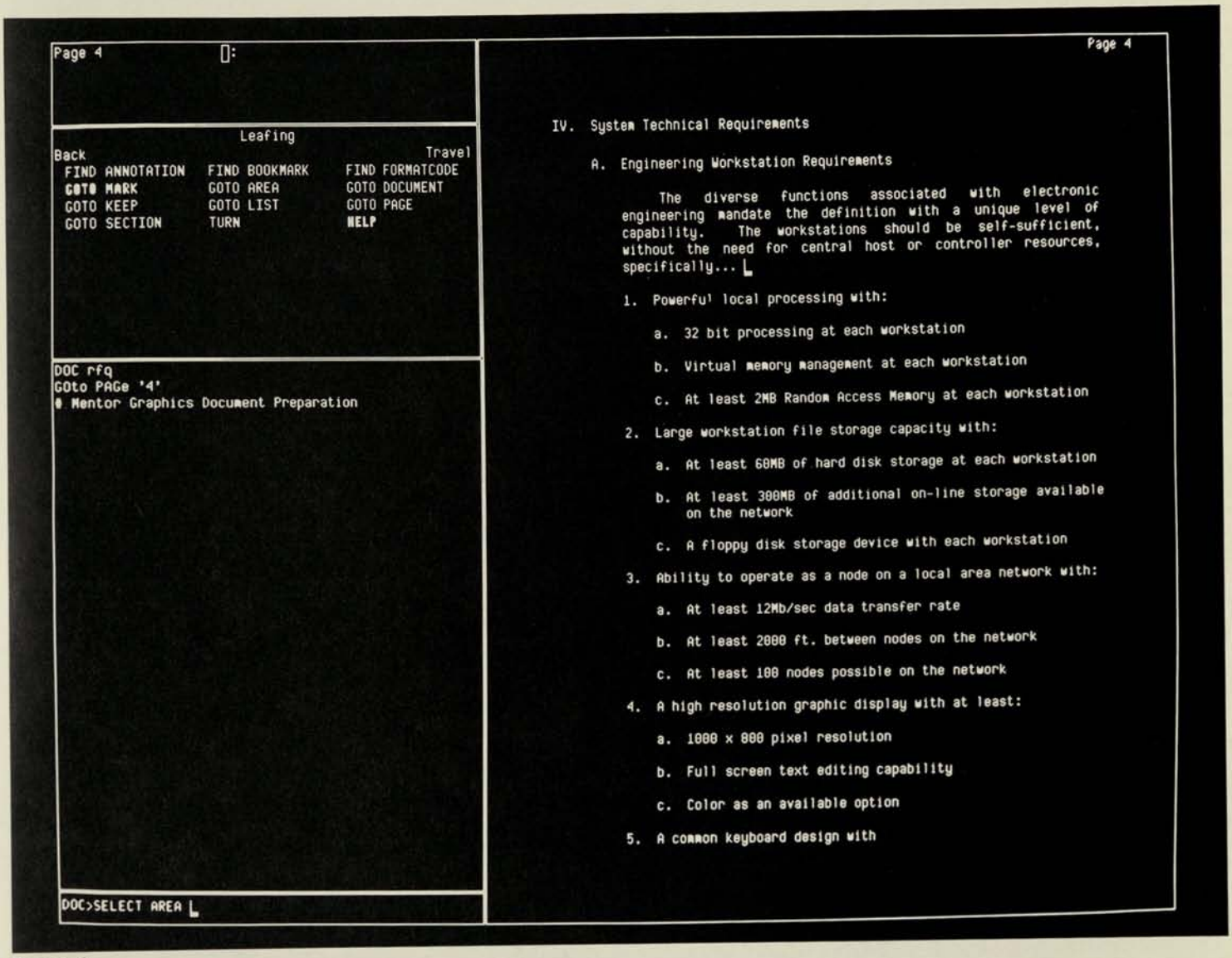



Figure 4

new section of text, the section number will be automatically determined and all other section numbers in the document will be adjusted accordingly. Figure 3 illustrates this feature. The user defined the text above the picture as "1", and the second section below the picture was automatically designated as "1.1". The next section would be 1.2, and so on.

Another example involves listing and is illustrated in Figure 4. The first item in the list is automatically labeled with the roman numeral "IV.". The second item is labeled number "A." and the subitems under it are automatically labeled "1.", "2.", etc. Automatically nested within these subitems are the list elements "a.", "b." and "c."

Formatting of page numbers and headings is another useful feature. For example, a page would be automatically numbered "6-28" if it was the 28th page of text under section 6 of the document. Another format code allows the inclusion of page headers and subheaders, and can center them, align them with the left margin or place them at the bottom of the page. A template is also available to specify a particular font for these items.

Another useful formatting command allows easy inclusion of footnotes. The footnote enumeration is placed in the text by simply positioning the cursor where the enumeration should occur. When the document is paginated, all footnotes will be grouped on the same page as their corresponding enumerations.

Document Preparation includes a set of commands that allow rapid and convenient travel through a document. Movement can be made between specified pages, sections, lists or pictures. In addition, specific text can be defined by a "bookmark" for rapid access at any time.

For fast interaction with the command set, the user can call upon a set of "soft" keys which reduce a command or series of commands to a single keystroke. Once editing and formatting are completed, the display shows the document in exactly the same way it will appear on the printed page. Output to line printers or printer/plotters is at the touch of a button.

Mentor Graphics Corporation
10200 S.W. Nimbus Avenue, G-7
Portland, OR 97223 (503) 620-9817

## Officers

Thomas H. Bruggere
President,
Chief Executive Officer

Gerard H. Langeler
Vice President, Marketing

David C. Moffenbeier
Vice President, Finance

Stephen Swerling
Vice President, Engineering

Thomas J. Fretz
Vice President, Sales

## Directors

Thomas H. Bruggere
President,
Chief Executive Officer

David R. Hathaway
General Partner
Venrock Associates
New York, NY

Robert E. Schroeder
Former President,
Qume Corporation
Palo Alto, CA

## Principal Investors

Greylock Management
Corporation
Boston, MA

Hambrecht & Quist
San Francisco, CA

Lamoreaux, Glynn & Associates
San Francisco, CA

L.F. Rothschild, Unterberg,
,Towbin
New York, NY

Sutter Hill Ventures
Palo Alto, CA

Venrock Associates
New York, NY

## Auditors

Peat, Marwick, Mitchell & Co.
Portland, OR

## Bank

First Interstate Bank of Oregon
Portland, OR

## Legal Counsel

Stoel, Rives, Boley, Fraser & Wyse
Portland, OR

## U.S. Sales Offices

2001 Gateway Place
Suite 195
San Jose, CA 95110
(408) 294-9933

4000 MacArthur Blvd.
Suite 3000
Newport Beach, CA 92660
(714) 752-0200

420 Bedford Street
Suite 160
Lexington, MA 02173
(617) 863-5776

4825 L.B.J. Freeway
Dallas, TX 75234

## International Sales Offices

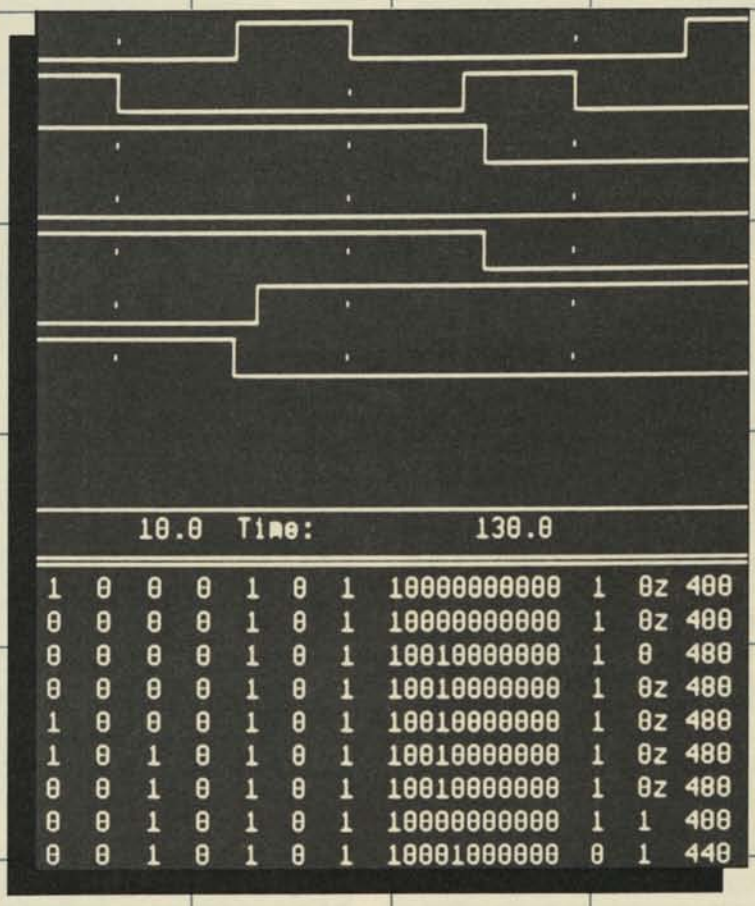
Mentor Graphics Japan Co., Ltd.
Sunshine 60 · 25F
1-1 Higashi-Ikebukuro 3-Chome
Toshima-Ku, Tokyo 170, Japan



Printed in USA 1003A SWW 12/82

Mentor

# DESIGN ANALYSIS TOOLS




INTERACTIVE LOGIC SIMULATOR
–Functional Models

TIMING VERIFIER PROGRAMMABLE NETLISTER

# DESIGN ANALYSIS TOOLS

## An Overview

IDEA 1000 is a self-contained computer-aided engineering (CAE) system dedicated to the design and analysis of complex electronic circuitry. The system is integrated with Apollo DOMAIN computer workstations, linked together in a distributed network architecture. This way, each operator receives the benefit of independent 32-bit-local processing and high resolution graphics, but still has full access to a common data base shared by the system as a whole, without the performance degradation of a central host architecture.

Within the IDEA 1000 system are a number of software modules used to implement the various phases of the logic or circuit design process. These include Structured Logic/Circuit Design, Interactive Logic Simulation, Timing Verification, Project Communications, Document Preparation, Formattable Netlist/Plot and CAD/CAE Programming. Each of these components is tied into a high-performance full relational data base management system for fast, efficient access to any required information (Figure 1).

Two of these basic modules, Interactive Logic Simulation and Timing Verification, function as design analysis tools. Both use the design data generated by the Structured Logic/Circuit Design module, which produces all components and the nets (logical signal paths) that comprise the completed circuit design. Since the design analysis tools allow full simulation of logic and timing flow within a circuit design, they reduce the need to repeatedly produce breadboarded prototypes, which is both costly and time-consuming. In effect, these tools provide the equivalent of a logic analyzer and pattern generator, with complete test capability for both state and timing functions.

Figure 2 presents a block diagram of the IDEA 1000's design analysis environment. The Design Expander output file represents the finalized output of the Structured Logic/Circuit Design module, and serves as the input for both logic simulation and timing verification. For logic simulation, this data is first processed by the Simulation Compiler, which formats it. The Simulation Compiler will also accept input from Functional Models, allowing the inclusion of components that have not been described down to their most primitive logic devices. In this manner, a user can analyze circuitry representing components from any number of levels within a structured design hierarchy (see DESIGN CREATION TOOLS sheet).

To drive the inputs of the design under test, the user creates a stimulus file, which is a set of test patterns optimized to test the logic of the circuit being simulated. The user can call upon several powerful commands that control and modify the simulation process. For instance, breakpoints can be defined and selected circuit nodes forced to particular logic states.

Signal behavior information may be extracted from the circuit design's logic outputs or any internal circuit node the user wishes to examine on a cycle-by-cycle basis. The user interacts with a powerful graphic display which presents signal data in both list and waveform formats, and which also includes a complete graphic representation of the circuit being simulated.

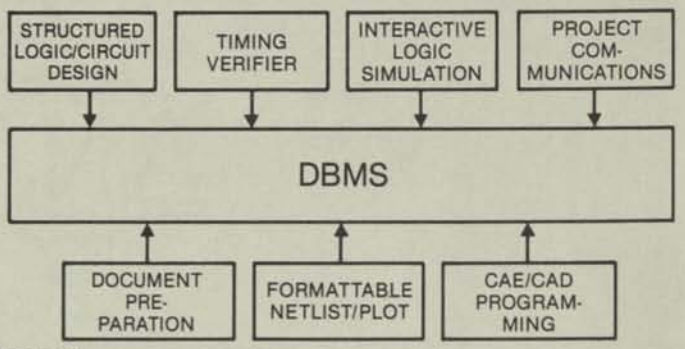


Figure 1

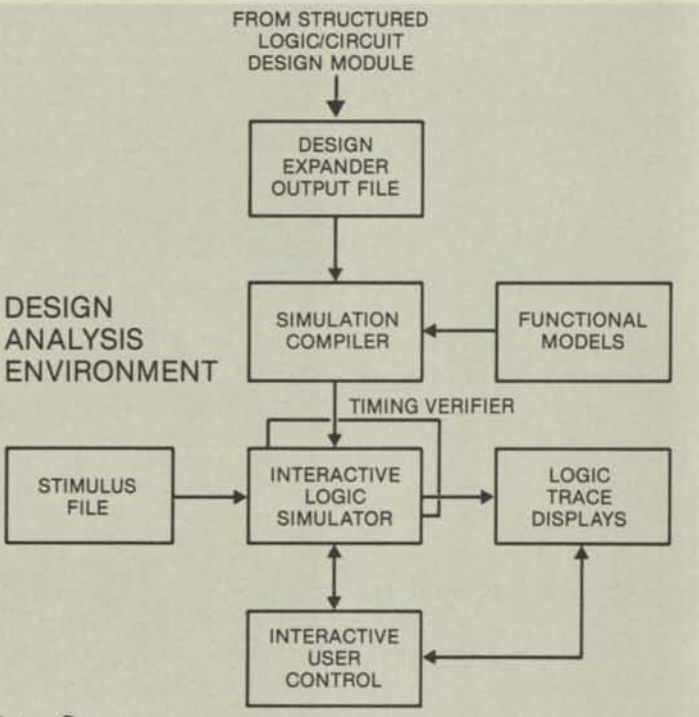


Figure 2

The Timing Verifier also works on the data supplied by the Design Expander output file. It allows the circuit under test to be run under user-specified clock and min/max circuit delay conditions to check for timing errors, which are subsequently identified and displayed.

## Design Expander and Functional Models

Output from the Design Expander, along with Functional Models, produce the input required by the Simulation Compiler. The Design Expander works its way through a hierarchical design until the design's structure is elaborated at its most primitive level as a collection of basic logic components and the signal paths connecting them. The resulting design data is then passed to the Simulation Compiler.

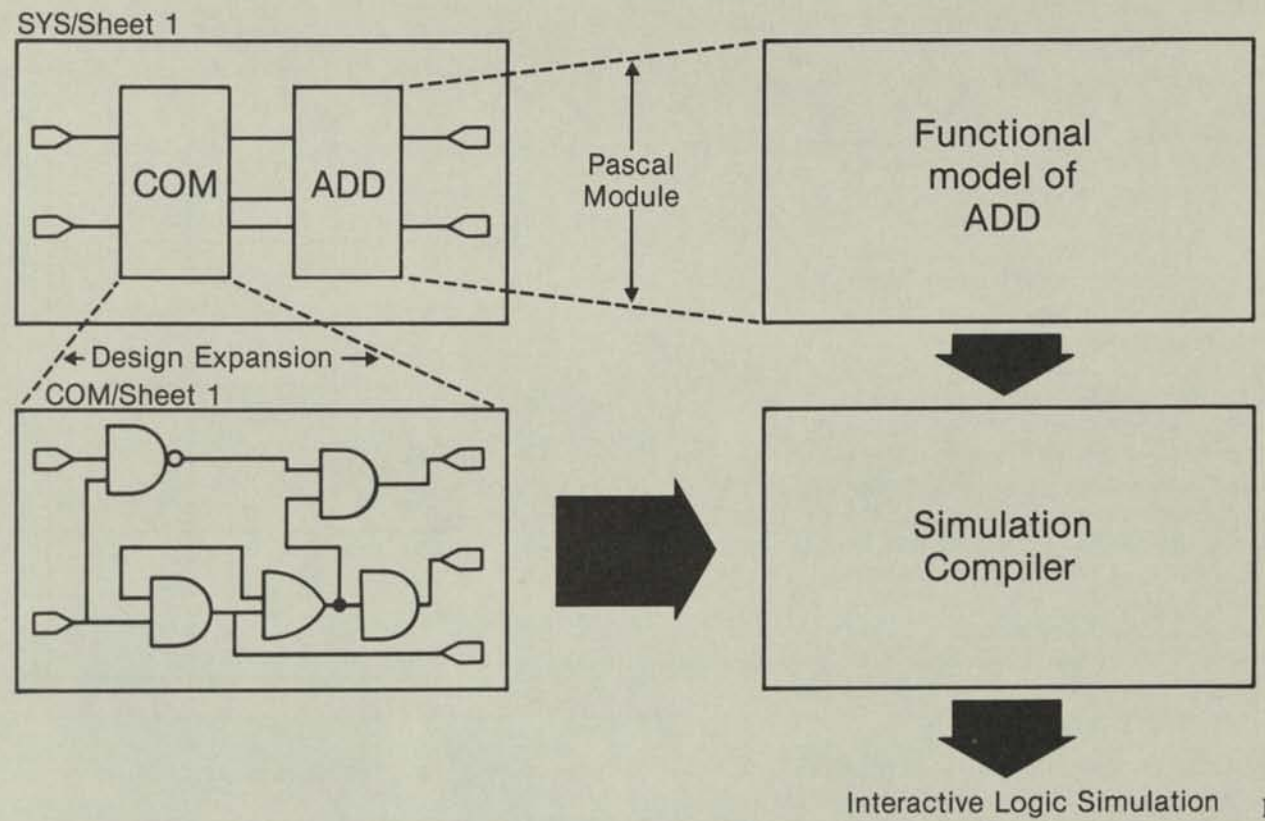



Figure 3

Functional models, on the other hand, allow components at higher levels in a hierarchical design to be described as Pascal procedural descriptions of functional inputs and outputs instead of as a collection of primitive components (e.g., gates) and nets. Once developed, these procedures are accepted by the Simulation Compiler and will interact with primitive-level components or other functional models during actual simulation. Functional models can contain boolean expressions, tabular data or any valid Pascal statement.

Figure 3 illustrates the relationship between primitive-level components produced by the Design Expander and functional models. The top sheet, Sys/sheet1, is located near the top of a typical design hierarchy and contains two interconnected components, COM and ADD. The internal components and nets making up COM have already been drawn on a worksheet called COM/Sheet1, including its output to the component ADD, which has yet to be defined at the "bottom" level of the hierarchy. To compensate for the lack of a fully defined internal logic structure for ADD, a Pascal functional model is substituted to supply the logic function which will later be supplied by ADD's internal components.

When the Design Expander works through the design's hierarchy of worksheets, it will produce a fully "flattened" version of COM, which is passed to the Simulation Compiler. In addition, the functional model for ADD is also passed to the Simulation Compiler. During simulation runs, ADD will now act as a fully functional component which accepts inputs from COM and logically processes them into the correct outputs.

## Simulation Compiler and Stimulus File

Using input from the Design Expander and Functional Models, the Simulation Compiler builds a software module acceptable by the IDEA 1000 Interactive Logic Simulator. The compiler describes each component and net in the design in terms of logic and timing properties that will be used by the simulator. These properties are drawn from information entered earlier by the user during symbol and net editing (see 'DESIGN CREATION TOOLS' sheet).

The simulator can use test patterns defined interactively while simulating, or it can draw upon a user-programmed stimulus file which supplies a series of test patterns to the inputs of the circuit being simulated. Figure 4 presents a simple stimulus file which lists sequentially the binary input to a device with input ports d, c, b and a. Each line contains a forced binary input state for each port. The "run 31" indicates that the forced input states on that line will remain active for the next 31 nanoseconds of simulation time.

The simulator models the functions of all standard logic gates plus RAMS, ROMS and PLAs. It will also accept rise and fall delay times for each component, which are specified as arbitrary time units, typically nanoseconds. Also, decay and

```
PERIOD List 31
LIST Binary d c b a 0 1 2 3 4 5 6 7 8 9

force d 0;force c 0;force b 0;force a 0;run 31
force d 0;force c 0;force b 0;force a 1;run 31
force d 0;force c 0;force b 1;force a 0;run 31
force d 0;force c 0;force b 1;force a 1;run 31
force d 0;force c 1;force b 0;force a 0;run 31
force d 0;force c 1;force b 0;force a 1;run 31
force d 0;force c 1;force b 1;force a 0;run 31
force d 0;force c 1;force b 1;force a 1;run 31
force d 1;force c 0;force b 0;force a 0;run 31
force d 1;force c 0;force b 0;force a 1;run 31
force d 1;force c 0;force b 1;force a 0;run 31
force d 1;force c 0;force b 1;force a 1;run 31
force d 1;force c 1;force b 0;force a 0;run 31
force d 1;force c 1;force b 0;force a 1;run 31
force d 1;force c 1;force b 1;force a 0;run 31
force d 1;force c 1;force b 1;force a 1;run 31

WRite List truths/ls.42.out
bye
```

Figure 4

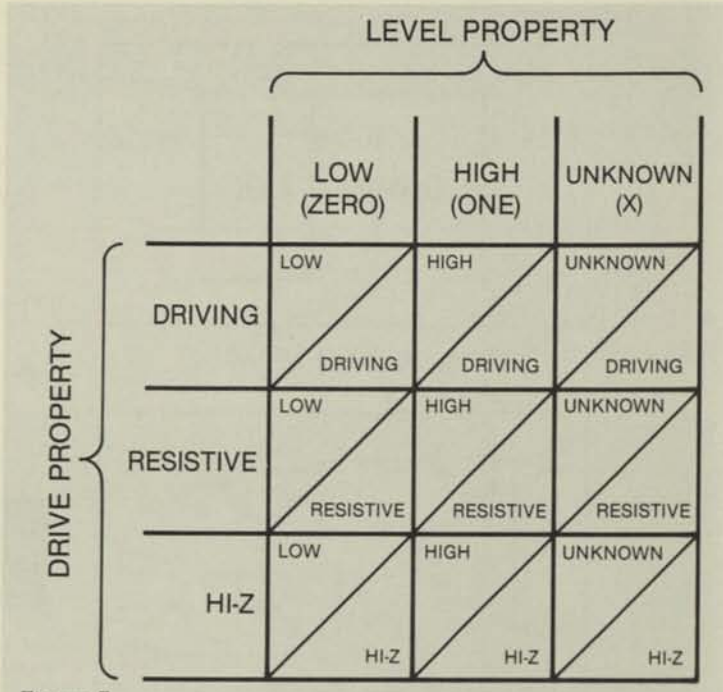



Figure 5

delay times can be specified for nets. In addition, each component can have initialized information attached which describes its logic state at the outset of simulation. For instance a counter could be set to all zeros, or an AND gate set to its true output.

Each component's output can be described as one of nine separate logic states recognized by the IDEA 1000 simulator. (Figure 5) This feature is extremely useful in modeling the various logic state conditions produced by MOS technology, such as wired-or, pull-ups, tri-states and transfer gates. Each state includes both a logic "level" and a "strength" property. Levels are low (zero), high (one) and unknown (X). Strengths are driving (charge transition at an unlimited rate), resistive (charge transition at a limited rate) and hi-Z (no charge transition).

Also, two special conditions can be indicated for each level. One is a spike, where a gate's input toggles faster than its output can respond. The other is a decay, where a charge trapped at a circuit node leaks away, changing the logic level.

The IDEA 1000 simulator differs from unit delay simulators through its ability to process events in terms of real time units. Rather than process events only once for each complete clock cycle, IDEA 1000 models the actual signal propagation from gate to gate in whatever time units the user has specified. This feature aids greatly in modeling asynchronous events, such as interrupts, and MOS conditions such as charge decay times. It also gives the user a better feel for the true operating speed of the circuit under design.

Since all component and net properties are specified during the design phase, no special formatting is required to load the Simulation Compiler, which works directly with the data base created by the Structured Logic/Circuit Design module.

## Simulator Characteristics and Operation

The IDEA 1000 Interactive Logic Simulator is designed specifically for MOS as well as TTL and ECL logic. It is event-driven, meaning that no processing is performed during periods when no logic events are occurring within the circuit. For this reason, there is no processing penalty for increasing the time resolution used in specifying delays. The simulator is also selective trace, meaning that it only examines components when their input states are changing. This feature cuts processing time. Performance has also been maximized by making exhaustive worst case timing verification a separate operation. This way the user can rapidly debug a circuit from a logic and nominal timing perspective and then proceed to worst case timing when ready; or start with timing and then go to logic, if desired.

The simulator has many powerful features under user control. Breakpoints can be set to halt execution on any specified event or set of events. Any logic node within the circuit can be forced to a specified value at a specified time, and then held there, either permanently or temporarily. Any unnamed logic node can also be named and "probed", by pointing to it on the logic diagram, which provides a complete trace history of the node in question.

## Simulator Interactive Graphic Display

The user interacts with the simulator through a powerful graphics display (Figure 6). At the top is a window indicating

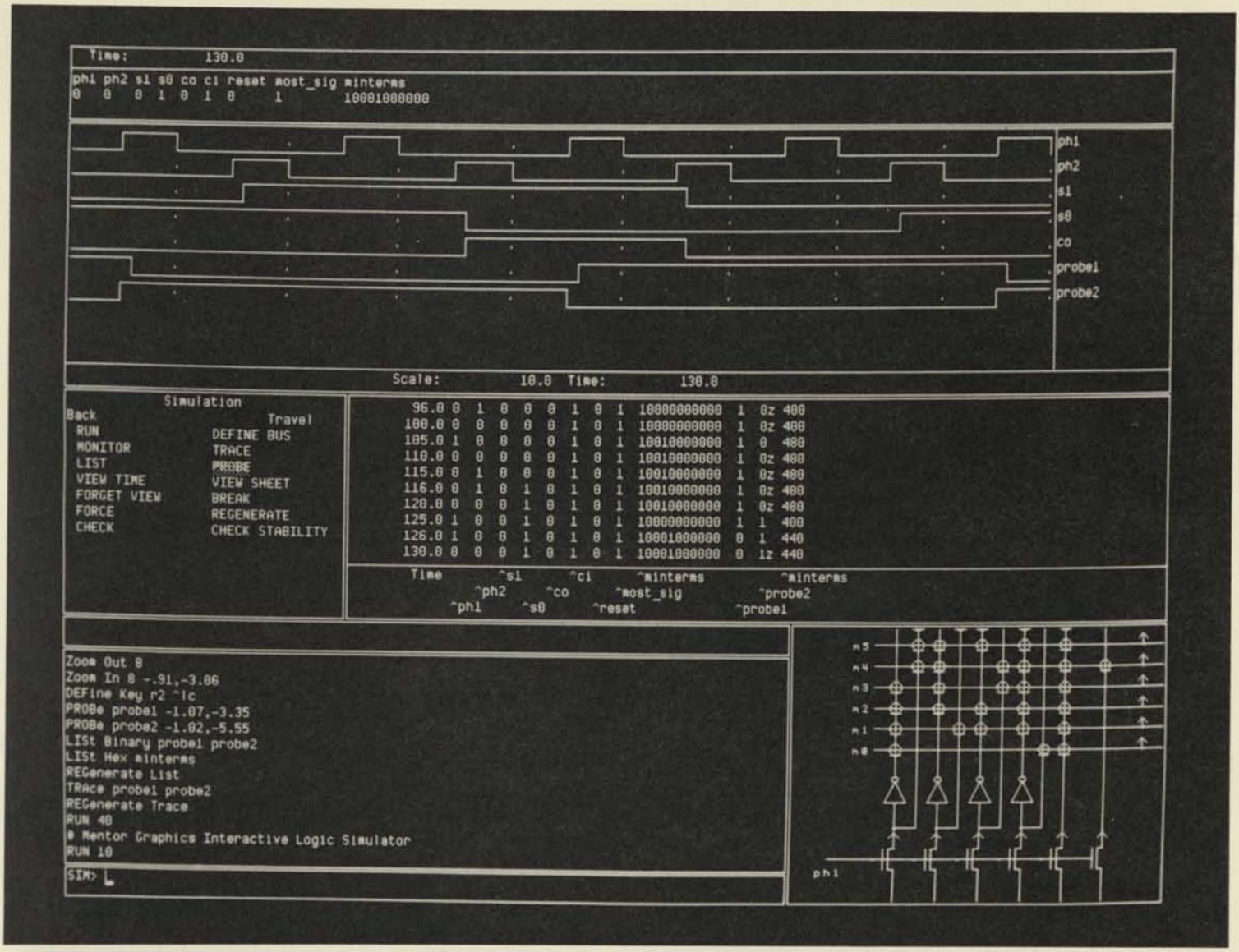



Figure 6

the current clock time and a monitor window indicating the current values of selected circuit nodes. These nodes are usually the outputs of the circuit, and can be given abbreviated names for use during simulation. Below these windows is an optional trace window which shows signal traces in convenient waveform format. Below the trace window is the list window, which shows a simulator trace history at a user-selected clock interval (the "z" subscript attached to some of the logic states indicates a tri-state condition). Below the list window is a view window which shows the circuit under simulation. The user has full graphic control over this window, including panning and zooming. Using a cursor, the user can work within this window to graphically select circuit nodes for probing, which may then be added to the monitor list or trace windows. To the left of the list window is a list of cursor-selectable menu items. To further optimize the interface, the user can call upon a series of "soft" keys for single-button command invocation, and also upon macro files to execute entire command sequences.

## Functional Models

As already described, functional modeling can be a valuable tool when simulating and debugging a circuit with components originating from several levels of a hierarchical design. This feature allows the designer to work from either the "bottom up" or the "top down" since simulation is not confined to hierarchical levels with fully defined circuitry.

In many cases, it is substantially easier to algorithmically model the functions of a particular component than to define all its internal logic in terms of gates and nets. For example, if the design involved interrupt support circuitry for a uP-based system, it would require thousands of gates to describe the processor's internal logic, but only a small program to describe its response to an interrupt input.

Functional modeling is based on the fact that most components remain stable until they receive an input which has the potential to cause some type of state change. A common example is a component which changes state only upon receiving an active clock edge at a specified input.

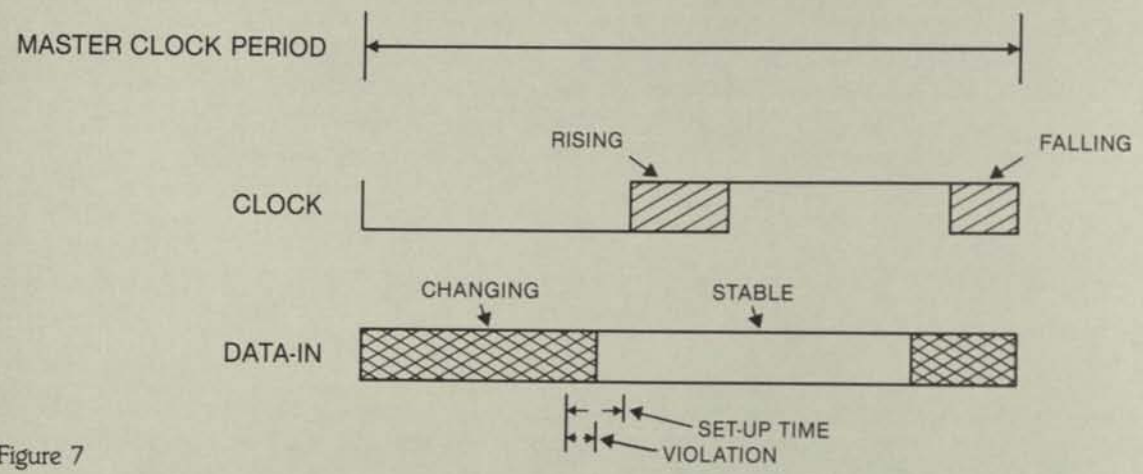



Figure 7

When the clock pulse is received the component will produce an output based on the current state of its data inputs and/or its internal logic state. Usually the number of inputs capable of causing a component state change is small, often only a single clock line. For this reason, it is convenient to write an event handler for this input in the form of a Pascal procedure. It is written so that upon receiving a state change input, the procedure will create a new output for the modeled component based on a functional algorithm which processes its current internal state and data inputs.

Procedures supplied for functional modeling include an "allocate" procedure, which initializes the model and allows it to acquire working storage, if needed. There is also an initialization procedure, which allows the modeled component to set its outputs into a logic configuration that would realistically occur during a powerup state.

Functional modeling provides wide flexibility of operation. For instance, a model can open and close a file, write messages to the display or have internal data storage. Any legal Pascal program operation can be included.

## Timing Verification

Once a logic circuit design has been completed, it must be tested in both the data and time domains. Rather than compromise performance by checking both domains with a single simulator, IDEA 1000 has made the Timing Verifier a separate entity from the Interactive Logic Simulator. The Timing Verifier has a powerful set of features that can exhaustively analyze all the circuit's signal paths. This approach eliminates the error potential present in other systems which limit testing only to user-selected signal paths, or to paths which are activated by a particular set of stimulator test patterns.

The Timing Verifier allows the designer to ascertain if a circuit will function correctly when run at specified clock speeds. The timing verifier provides a complete list of errors which identify both the pin where the error occurred and the actual nature of the error. Also, waveforms of interest can be displayed graphically to enhance comprehension of their timing relationships. In addition, a schematic of the circuit under test can be called up, with pins containing errors automatically highlighted for easy identification.

The Timing Verifier takes all the circuit's signal paths through one complete machine cycle. This simulation of the time domain breaks the full cycle down into very small increments which allow as high a resolution as needed. During execution, the Timing Verifier works with timing characteristics for each signal. These characteristics are expressed in one of two basic formats. The most common is a format which describes the signal as "changing" (in transition from one logic state to another) or "stable" (Logic state transition completed). The other is a format that describes the signal in terms of actual logic value (1 or 0), and also in terms of rising and falling edges. Figure 7 shows the relationship between these two formats and the master clock period.

The timing characteristics for each signal originate from one of two sources. The most common one is a set of properties entered by the user during symbol editing or network editing. The other source is a set of "assertions" supplied by the user, which are usually used to describe the timing behavior of the signal relative to the master clock or other signals within the circuit under test.

The description of the clock signal itself is a special case. It is derived from the master clock period and described in terms of logic values as well as phases and rising and falling edges. The inclusion of logic values is necessary to maintain the correct logic value in cases where the clock passes through gated circuitry, or where events occur only on the rising or falling edge. Also, the clock may be defined as having skew, which describes the area of uncertainty around the occurrences of the clock edge. Skew compensates for the propagation delay that may accumulate as the clock signal passes through the circuitry.

During execution, the Timing Verifier completes two basic tasks. First it produces a timing "profile" for each signal which records its timing characteristics over the master clock period, as shown in Figure 7. Second, it evaluates all timing

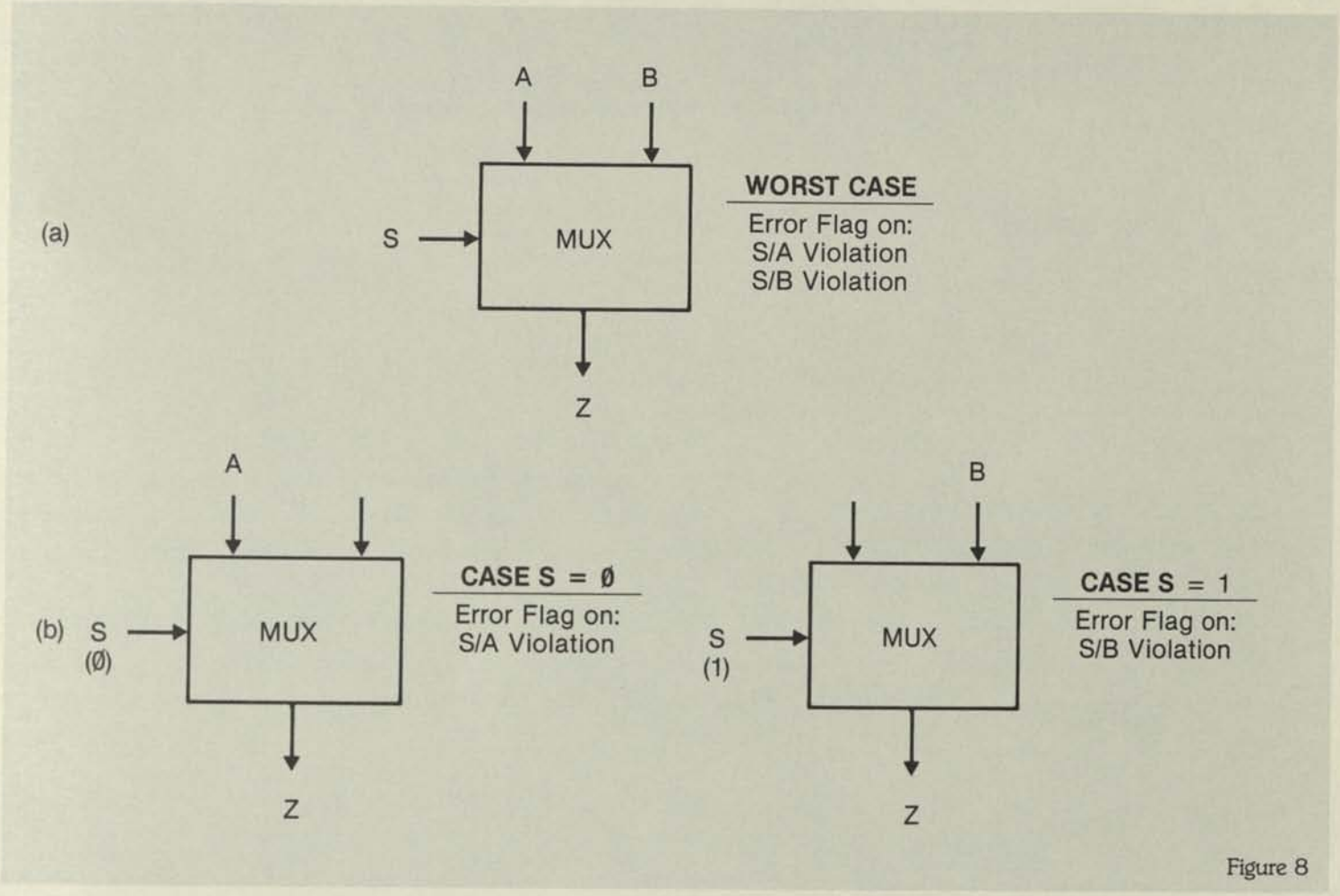



Figure 8

relationships between signals, as defined by their respective properties. If any defined relationship between a signal and the clock (or another signal) is violated, then a timing error is reported.

The Timing Verifier can be run on a "worst case" basis, where all timing relationships are examined on the basis of the changing/stable characteristic alone. Although quite thorough, this approach may call out many timing errors that are not possible given the circuit's intended function. For example, Figure 8a shows a simple multiplex component with data inputs A and B, a selector input S and an output Z. Let's assume that the A and B inputs have a specified setup time in relation to the S input. In a worst case test, the timing verifier would look at only changing/stable characteristics for the A, B and S signals and report any instance where either the A or B signal became stable too late in relation to the S signal. In reality, the only valid relationship is between S and the currently selected input, either A or B. To test within this limited case, a boolean value must be specified for S. The state of this value determines which input has been selected and therefore should be tested for the requisite hold time. Figure 8b illustrates the two possibilities. By introducing individual cases of this type, the user can narrow the range of reported errors to within desired limits.

## Programmable Netlister

The Mentor Netlister is a universal tool for conveying design files to an external simulator, CAD system or data base. It formats the output of the Design Expander according to definitions in a brief Pascal program entered by the user.

The Netlister is provided with a set of Pascal procedures which greatly simplify the task of searching the Design Expander output. This allows the Netlister to function as a data base query tool as well as a netlister to external systems. Calls are available for sequencing through nets, components, pins and properties. Additional procedures allow any of these to be formatted into ASCII strings for storage or printing.

Through Pascal coding, the user has great flexibility in determining the output of the Netlister. For instance, arbitrary text can be included along with connectivity information, component names updated, and specific properties located and processed. In this manner, the IDEA 1000 Netlister avoids the problems associated with fixed format Netlisters. Mentor provides many of the more common program interfaces. For additional formats, only a brief Pascal program is necessary to adapt the Netlister to your current output requirements.

Mentor

Mentor Graphics Corporation
10200 S.W. Nimbus Avenue, G-7
Portland, OR 97223 (503) 620-9817

## Officers

Thomas H. Bruggere
President,
Chief Executive Officer

Gerard H. Langeler
Vice President, Marketing

David C. Moffenbeier
Vice President, Finance

Stephen Swerling
Vice President, Engineering

Thomas J. Fretz
Vice President, Sales

## Directors

Thomas H. Bruggere
President,
Chief Executive Officer

David R. Hathaway
General Partner
Venrock Associates
New York, NY

Robert E. Schroeder
Former President,
Qume Corporation
Palo Alto, CA

## Principal Investors

Greylock Management
Corporation
Boston, MA

Hambrecht & Quist
San Francisco, CA

Lamoreaux, Glynn & Associates
San Francisco, CA

L.F. Rothschild, Unterberg,
Towbin
New York, NY

Sutter Hill Ventures
Palo Alto, CA

Venrock Associates
New York, NY

## Auditors

Peat, Marwick, Mitchell & Co.
Portland, OR

## Bank

First Interstate Bank of Oregon
Portland, OR

## Legal Counsel

Stoel, Rives, Boley, Fraser & Wyse
Portland, OR

## Sales Offices

2001 Gateway Place
Suite 195
San Jose, CA 95110
(408) 294-9933

4000 MacArthur Blvd.
Suite 3000
Newport Beach, CA 92660
(714) 851-6431

420 Bedford Street
Suite 160
Lexington, MA 02173
(617) 863-5776

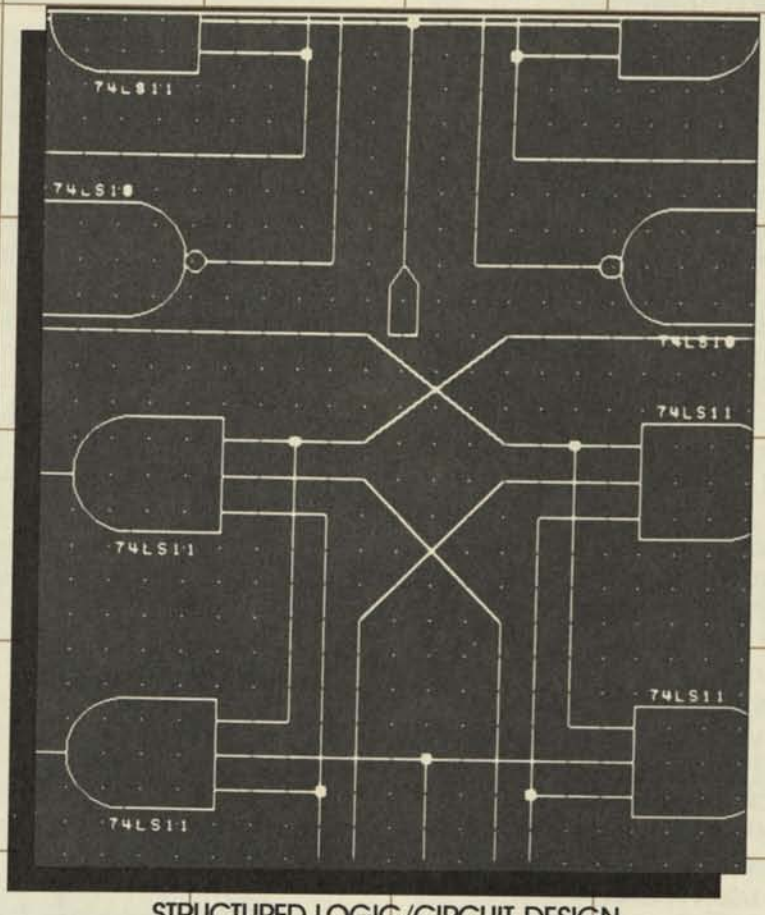
4825 L.B.J. Freeway
Dallas, TX 75234



Printed in USA 1002A SWW 11/82

Mentor

# DESIGN CREATION TOOLS




STRUCTURED LOGIC/CIRCUIT DESIGN

- Symbol Editor and Checker
- Network Editor and Checker
- Designer Traverser

DESIGN EXPANDER

PROGRAMMABLE NETLISTER

# DESIGN CREATION TOOLS

## An Overview

IDEA 1000 is a self-contained, computer-aided engineering (CAE) system dedicated to the design and analysis of complex electronic circuitry. The system is integrated with Apollo DOMAIN computer workstations linked together in a distributed network architecture. This way, each operator receives the benefit of independent 32-bit local processing and high resolution graphics, but still has full access to a common data base shared by the system as a whole, without the performance degradation of a central host architecture.

Within the IDEA 1000 system are a number of software modules used to implement the various phases of the logic or circuit design process. These include Structured Logic/Circuit Design, Interactive Logic Simulation, Timing Verification, Project Communications, Document Preparation, Formattable Netlist/Plot and CAD/CAE Programming. Each of these components is tied into a high-performance full relational data base management system for fast, efficient access to any required information (Figure 1).

The Structured Logic/Circuit Design (SLD) module of IDEA 1000 is an integrated set of software tools permitting the creation of logic and/or circuit designs through a graphics-oriented user interface (Figure 2). These tools let the engineer construct a design using computer graphic techniques which eliminate the drudgery of conventional drafting. First, a set of logic symbols can be created using the Symbol editor. These symbols, along with those from the Mentor-supplied Parts Library, are called upon through the Network Editor to produce the schematic design "worksheet"–a graphic representation of the circuit generated on the IDEA 1000 workstation's CRT. The user interacts with the Network Editor to logically connect the symbols through "nets", lines which represent various signal paths. All user-created symbols and worksheets are subject to checking before use in network editing. This is done to ensure their compatibility with other IDEA 1000 software components. If an inconsistency is found, error messages point to the source.

In the course of a single design, any number of Network Editor worksheets may be produced and connected to describe the completed logic circuitry. These sheets can be symbolically connected in a horizontal manner to represent a "flat" design, and also organized vertically to represent a hierarchical design format. Figure 3 depicts a simple "flat" design comprised of three worksheets. The sheets are automatically linked through the connection points they have in common.

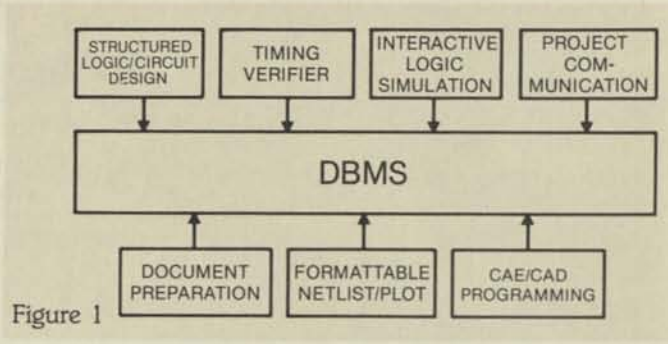


Figure 1

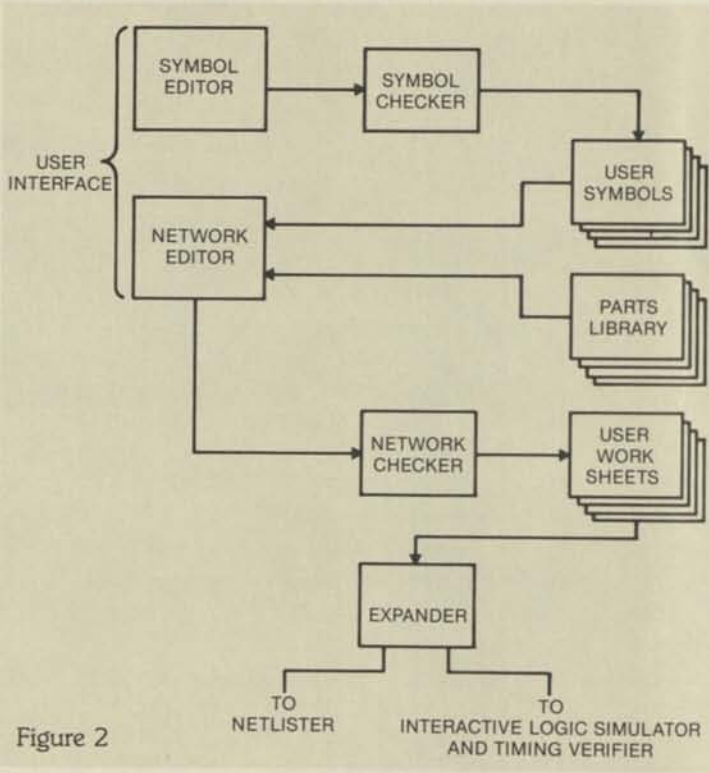


Figure 2

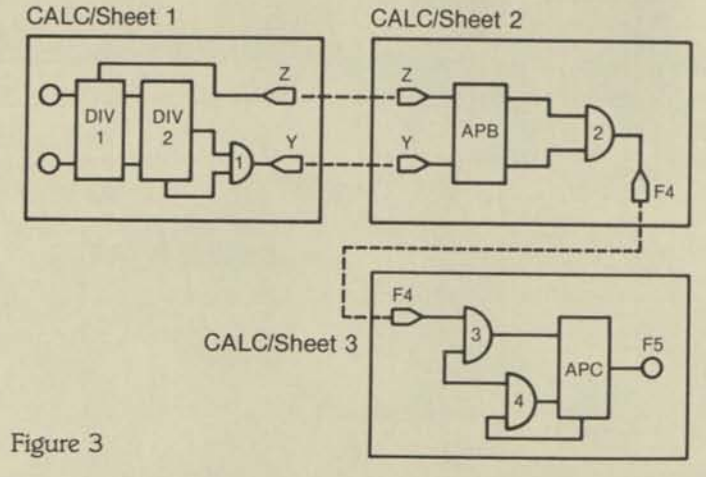


Figure 3

Hierarchical organization of worksheets provides a powerful and sophisticated means of breaking a complex design project down into manageable parts. Figure 4 shows a hypothetical design project organized into three hierarchical levels. At the bottom level, a series of three worksheets is produced using user-created symbols and the Mentor parts library. These sheets are inter-connected in the "flat" manner to make a component called SUBCOM. Using the symbol editor, this component is condensed from three worksheets into a single symbol which will represent all of SUBCOM at the next level up in the hierarchy.

Moving one step up the hierarchy, a device called COMM is represented by a single worksheet. Notice that the SUBCOM symbol from the preceding level is now incorporated in this higher-level worksheet. The other component symbol, ADD2, would have evolved in a similar manner. The AND gate, C3, would have come directly from the parts library. Once again, the Symbol Editor is used to condense the entire COMM worksheet into a single symbol with the same name.

At the top of the hierarchy is the master worksheet, SYSTEM. The symbol COMM now becomes one of the four major function blocks making up the system as a whole. The symbols for the other function blocks would have undergone similar hierarchical processing before being included in this final worksheet, which embodies all the information accumulated from lower-level worksheets.

An important aspect of the hierarchical approach is that the design can be initiated at any level and then proceed either up or down the hierarchical tree. In the example just given, the middle-level component, COMM, could have been created and defined first, using just the symbols for SUBCOM and ADD2. Their related worksheets could have been produced later. There are no practical limits to the number of levels in the hierarchy.

When a worksheet or series of worksheets is complete, it is ready for processing by the Design Expander, which creates a "flattened" design file describing the circuit in its most primitive terms. This file can be processed for simulation, timing verification, or fed to the Netlister. Also, the output of the Design Expander can be merged with files that describe the actual physical properties of the design; for example, propagation delays.

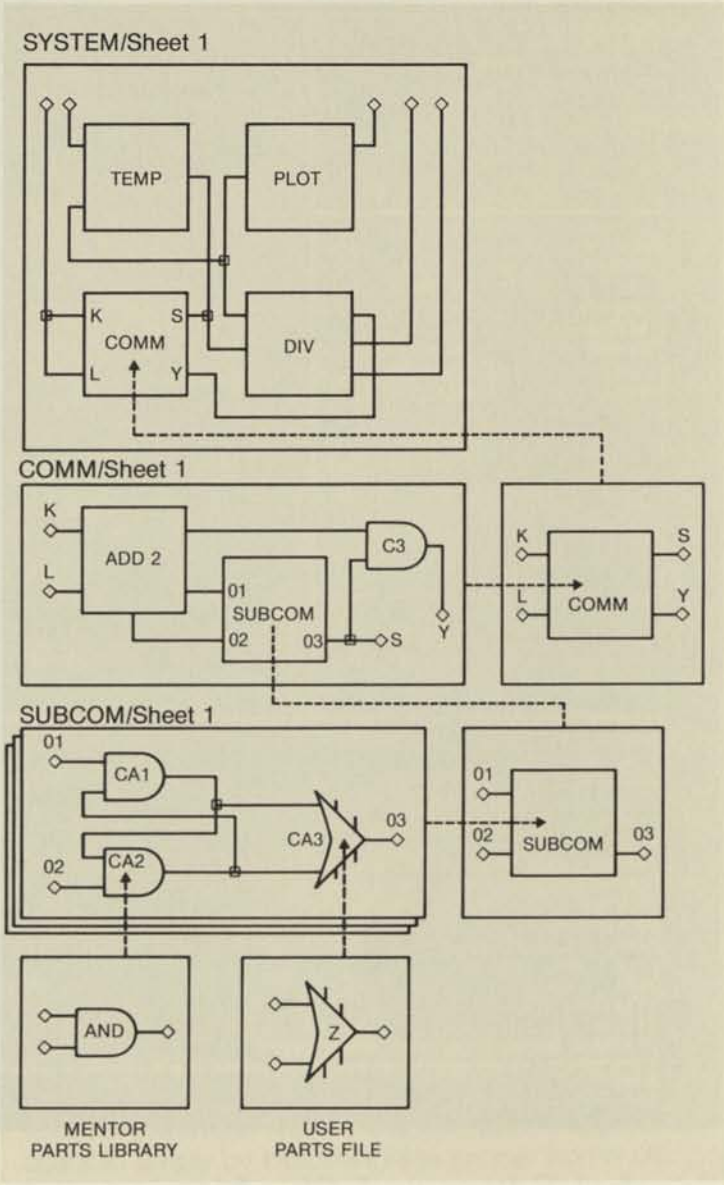



Figure 4

# DESIGN CREATION TOOLS

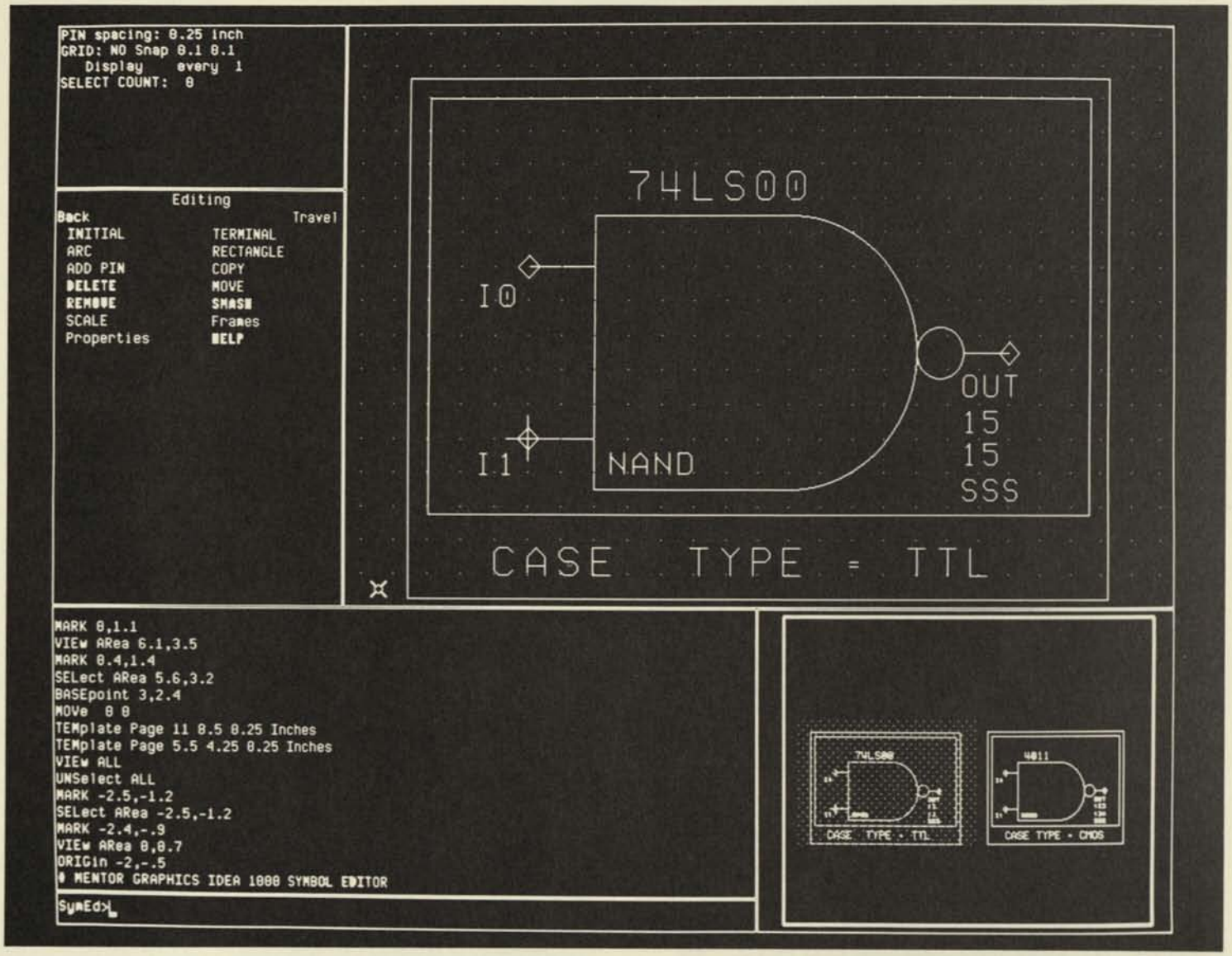



Figure 5

## Symbol Editor and Checker

The Symbol Editor (SYMED) is an interactive graphics program which allows symbols to be created and specific properties attached to them. These symbols are then filed for subsequent use in the network editor. Symbols may be created for "primitives", such as gates; or for arbitrarily complex subsystems.

The typical Symbol Editor display includes two graphics windows and a number of text windows (Figure 5). The top graphics window acts as the worksheet where the user graphically constructs the symbol. The lower graphics window acts as a context view, with the shaded area locating the current position of the main window in relation to the overall worksheet space. This gives the user a complete overview of larger drawings and allows easy movement of the main window to any location within the drawing.

The images in both windows can be expanded, contracted, and moved. Also, the graphic environment within the worksheet can be programmed to meet specific user requirements, such as grid spacing and visibility. The other windows provide a cursor-activated menu, a command transcript list and the current command entered for execution.

The user constructs the actual symbol by using a touchpad- or tablet-controlled cursor in combination

with various commands. There are several ways to enter these commands, including a menu and "soft" keys that permit rapid typing and interaction with the command set. Once drawn, all graphic elements can be quickly moved to any desired location. In building an image for the symbol, the user can call upon certain standardized graphic elements such as lines, arcs and pins, which define logical connection points. In addition, the user can derive symbols from worksheets produced by the Network editor.

The Symbol Editor also allows "properties" to be attached to each symbol. Some of these properties have reserved names and their data is used by other software modules in the IDEA 1000 system. Representative properties reserved by IDEA 1000 include the following types:

**PIN**–labels pins for logical connections with pins attached to other symbols.
**MODEL**–contains generic logic name (e.g. NAND).
**RISE**–indicates rise time in numeric units.
**FALL**–indicates fall time in numeric units.
**DRIVE**–indicates a drive strength for each logic level.

Other properties include any user-defined attributes such as reference designations, pin numbers, etc. There is no limit to the number of user-supplied properties that may be attached to a symbol.

The size and visibility orientation of text representing property data is adjustable to fit constraints presented by other graphic elements. When a symbol is rotated, any associated text is automatically rotated to maintain its readability, eliminating the need for multiple symbols representing all possible symbol/text orientations. Property data can be fixed permanently, or made alterable during network editing.

Another powerful tool within the Symbol Editor is the use of case frames, which can convert generic logic functions into specific logic symbols. The use of case frames is illustrated in Figure 6, where Figure 6a depicts an AND gate as entered into the Symbol Editor's worksheet frame. This gate has two inputs. Obviously, many designs will require variations on this standard version, and this is accomplished by drawing each basic variation and enclosing it in a rectangular frame. Each frame is then labeled by a case statement placed beneath frame. Figure 6b shows how a single Symbol Editor Worksheet and case/frames would be applied to create two versions of an AND gate, one with two inputs, another with three.

Going one step further, case/frames can be nested

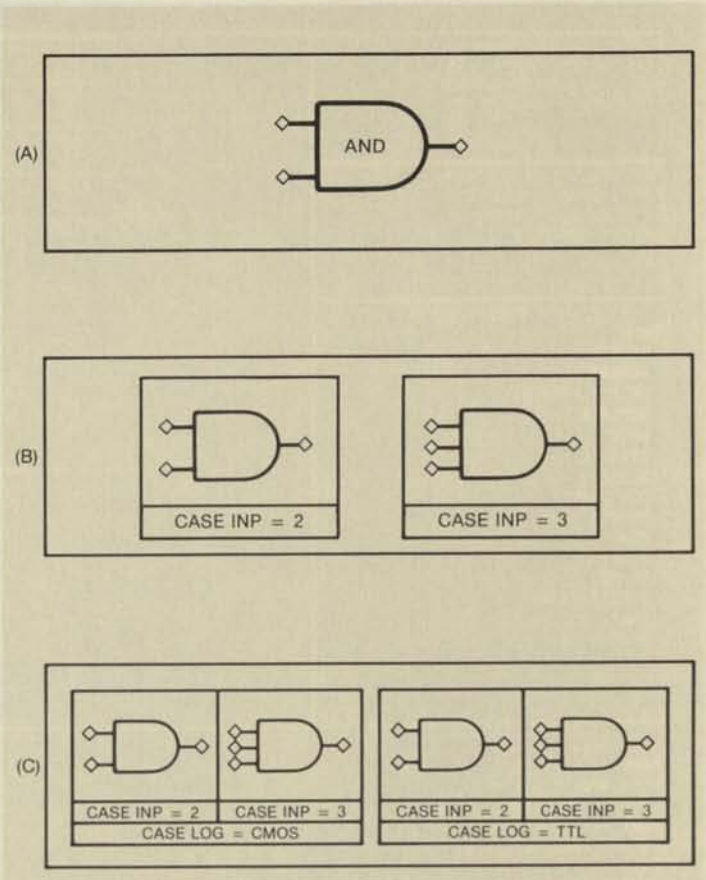



Figure 6

to create an even wider variety of specific symbols from a generic one. In Figure 6C, the frames representing two and three inputs have been nested inside an additional set of frames to handle both CMOS and TTL logic. When using such symbol variations with the Network Editor, each is separately callable simply by indicating the proper frame via the frame parameter.

When a symbol has been completely described using the Symbol Editor, the results are passed through the Symbol Checker (SYMCHECK). This program inspects the symbol to insure that it conforms to the format accepted by the IDEA 1000 system. This includes checking case frames and their nesting, confirming proper use of property text types, and verifying the placement of pins and other graphic elements. In the case of an inconsistency, the Symbol Checker will produce an error message describing the details of the problem.

Once the Symbol Checker has verified a symbol, it can be permanently filed for use during network editing.

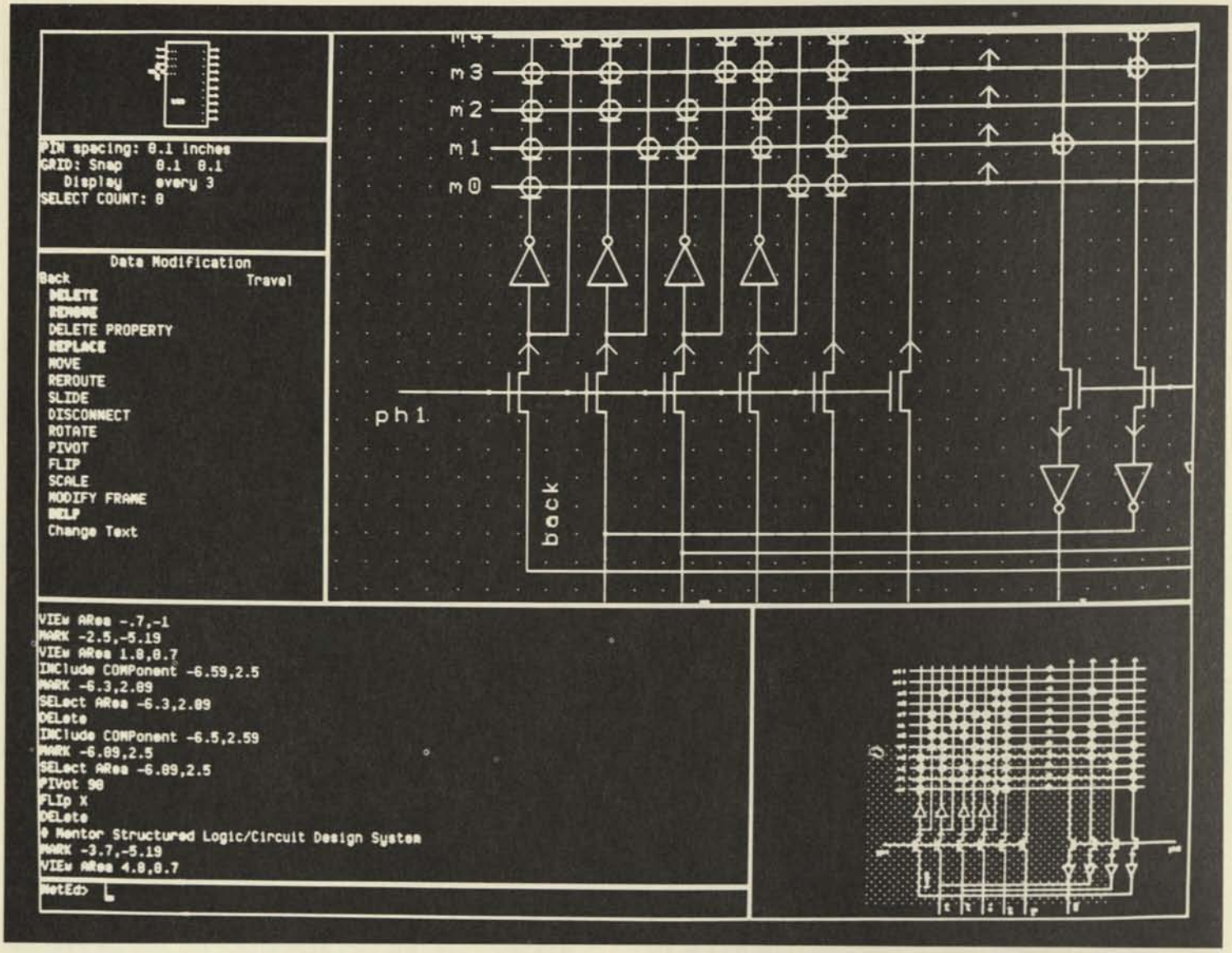



Figure 7

## Network Editor

The Network Editor (NETED) is an interactive graphic system which connects symbols into a circuit design. The symbols used may be user-defined through the Symbol Editor or originate from the Mentor Parts Library. Also, each symbol may represent any level of complexity, from primitive circuit functions to complex components previously defined by the user. The output of the Network Editor is a "worksheet" describing an entire design or a portion of a design that continues onto other worksheets. As presented in the overview, the connections between worksheets may be organized into a "flat" design and also connected vertically in a hierarchy.

Like the Symbol Editor, the Network Editor presents the user with two main graphics windows, one for entering components and making netroute connections, the other acting as a reference map of the overall worksheet (Figure 7). The text windows provide a cursor-directed command menu, a command transcript list, a current command entry and status information on the current worksheet. User interaction with the Network Editor takes place through an optimized set of cursor and command operations.

The Network Editor permits easy entry of symbols into the worksheet space presented by either graphic window. If the symbol files include case frames, the

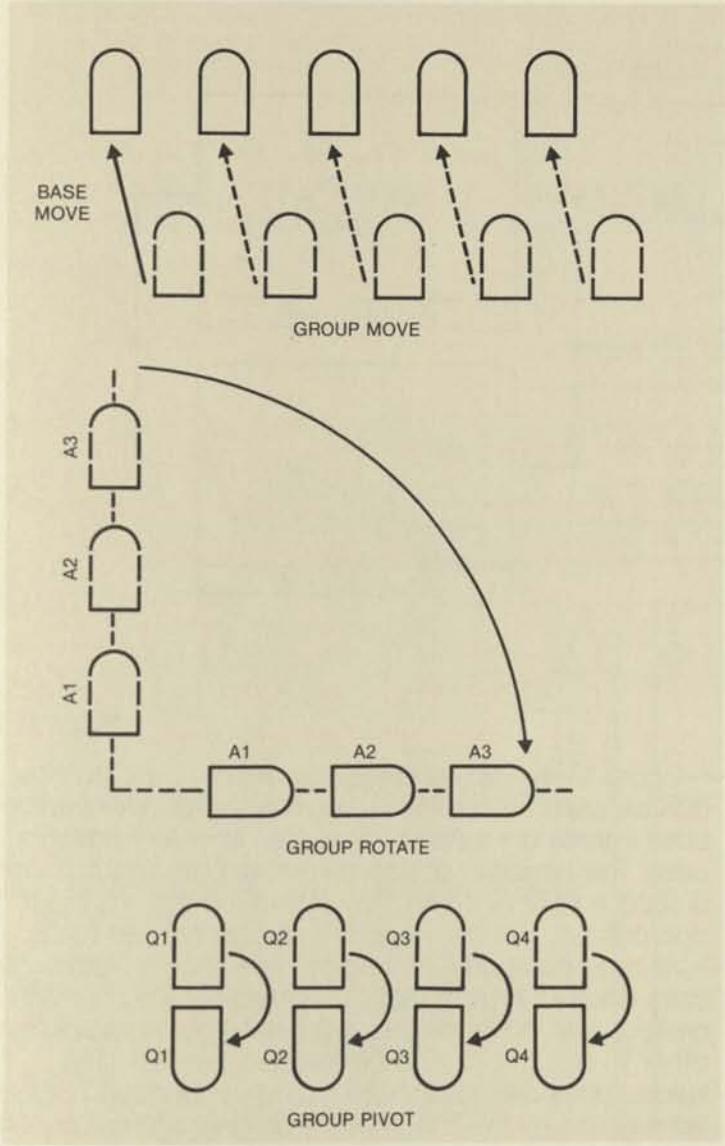



Figure 8

particular case desired is specified at the time the symbol is called. In addition, a "preview" window can be invoked to permit visual inspection of the symbol before it goes to the worksheet. Both the size of the sheet and the size of symbols can be user-defined.

By moving the cursor, copies of the called symbols can be easily entered in either a single or successive locations. Symbols can also be rapidly moved, deleted, rotated and flipped, with text automatically adjusted for a proper fit. Also, the user can zoom, pan and scroll through the worksheet area for rapid access to any given area.

IDEA 1000 also offers a powerful set of commands for manipulating groups of symbols, as shown in Figure 8. A group move command allows a group of cursor-

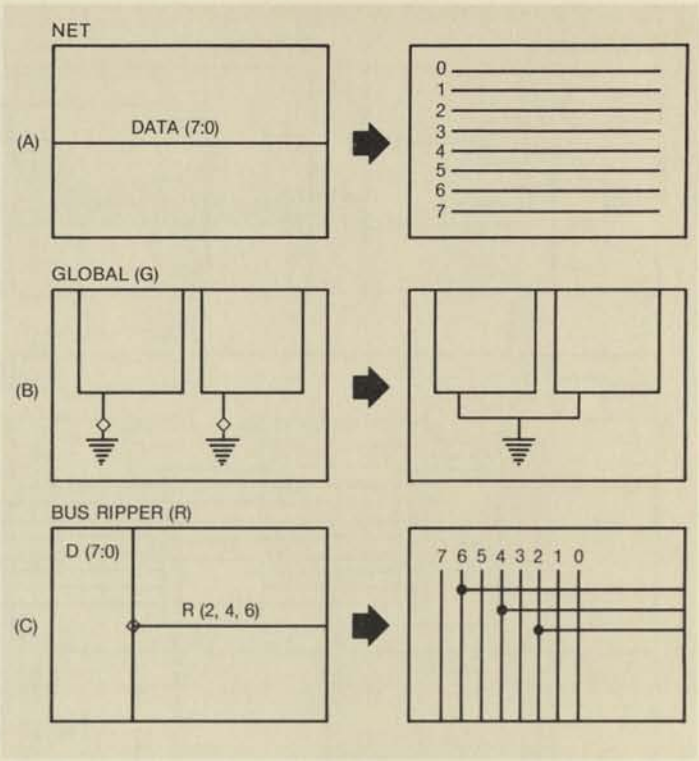



Figure 9

selected symbols to be moved relative to a single "base point". A group rotate command allows an in-line group of symbols to be rotated in 90° increments about a common axis. A group pivot command allows a number of preselected symbols to be rotated around their individual pivot points.

Logical connections between symbols on the worksheet are called "nets", and may represent either a single bit path or bus, which can be entered as a bold line to graphically separate it from individual lines. Nets are entered graphically as line segments that join with symbols or other nets. Each net can have property text (e.g. a signal name) attached to it with the associated property data supplying information used by other programs in the IDEA 1000 system. Whenever symbols are moved, any net connections between them are automatically stretched or "rubberbanded" to maintain logical connections previously entered by the user.

Figure 9 describes the function of several properties associated with the description of nets. Figure 9a shows how the NET property includes text to label a bus DATA and define its width as eight bits. Figure 9b shows how the GLOBAL property is used to define a symbol that represents a universal point, such as ground. Fig. 9c shows how the BUS RIPPER property permits specific lines to be tapped off a main bus at a connector point.

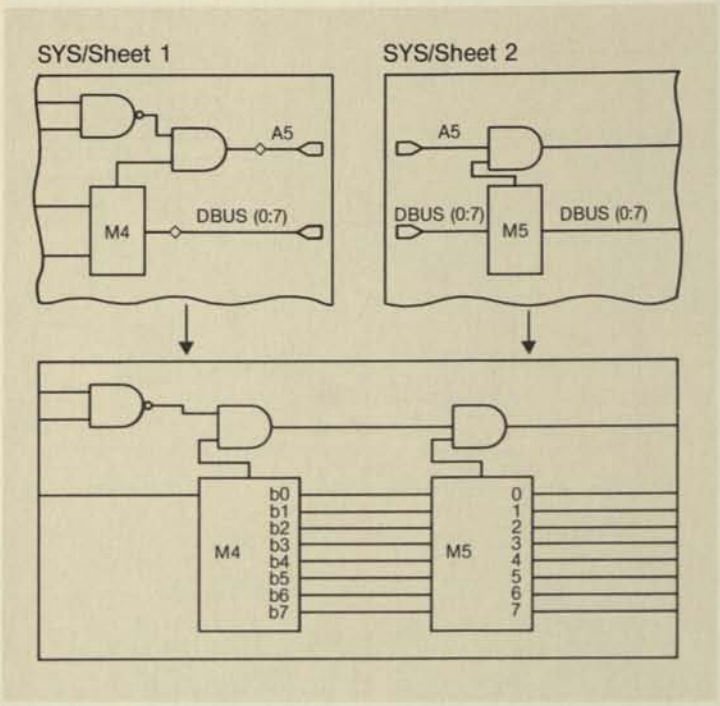



Figure 10

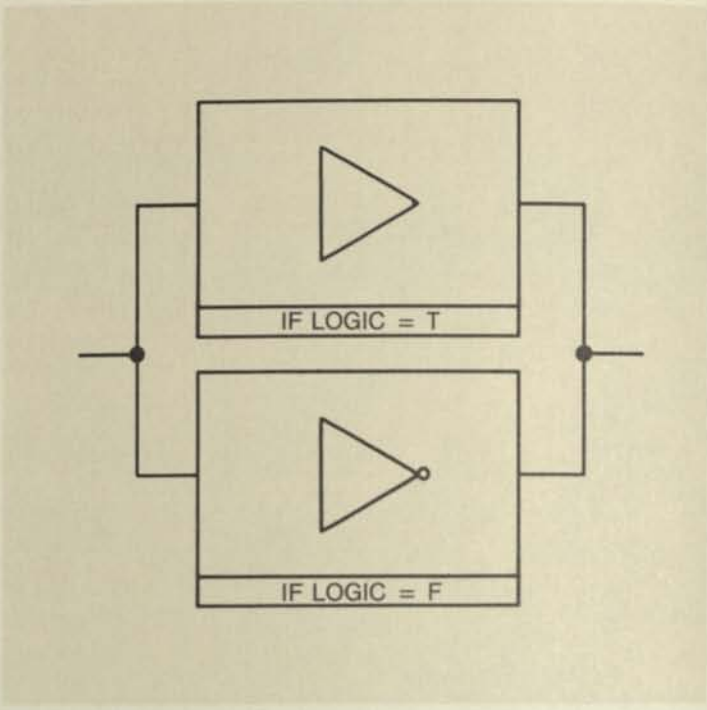



Figure 11

Figure 10 shows the function of a property called CONNECTOR, which graphically defines the connection point between the circuitry on one worksheet and the circuitry on an accompanying sheet. On the sheet marked "SYS/sheet 1", a user-defined connector symbol has been attached to the A5 output pin of the AND gate, and also to the bus labeled DBUS (O:7). Similar connector symbols have been attached to the corresponding nets on "SYS/sheet 2". The bottom illustration shows how the two sheets will later be merged during logic implementation by other parts of the IDEA 1000 system.

Another powerful and important tool for network editing is the use of case frames. Cases are defined by drawing a rectangular frame around the symbol in question and then inserting a case statement underneath the frame. Case frames have two uses in network editing. One is to define alternative conditions, as illustrated in Figure 11, where either an inverted or non-inverted output will be specified depending on the logic polarity chosen in the final design. Case frames of this type are later processed by the Design Expander, which accesses user-supplied data to resolve each set of conditional cases.

Figure 12 gives a detailed look at how case frames can be used for symbol replication. In this instance, the case frames are nested to create a two-dimensional array. The symbolic device has an address input A, and a data output D. When the array is formed, this input and output must be systematically connected to an 8-bit data bus and 4-bit address bus, shown outside the case frames. To accomplish this, two case frames are nested, one to handle the eight data bus hookups, the other to accomplish the address connections. The inside frame creates the array's "rows" and the outside frame creates the "columns". Thus, only a small amount of graphic space and user time have been required to create a fairly complex array. In this example, the case frame parameters were specified as absolutes (e.g., a = O to 7); however, they also could have been specified as variables (e.g., i = O to N) subject to later resolution by the user.

Once a worksheet is completed, it is sent to the Net Checker to insure that the worksheet file corresponds to the conventions of the IDEA 1000 system. Any inconsistencies will result in error messages that detail the particulars of the problem.

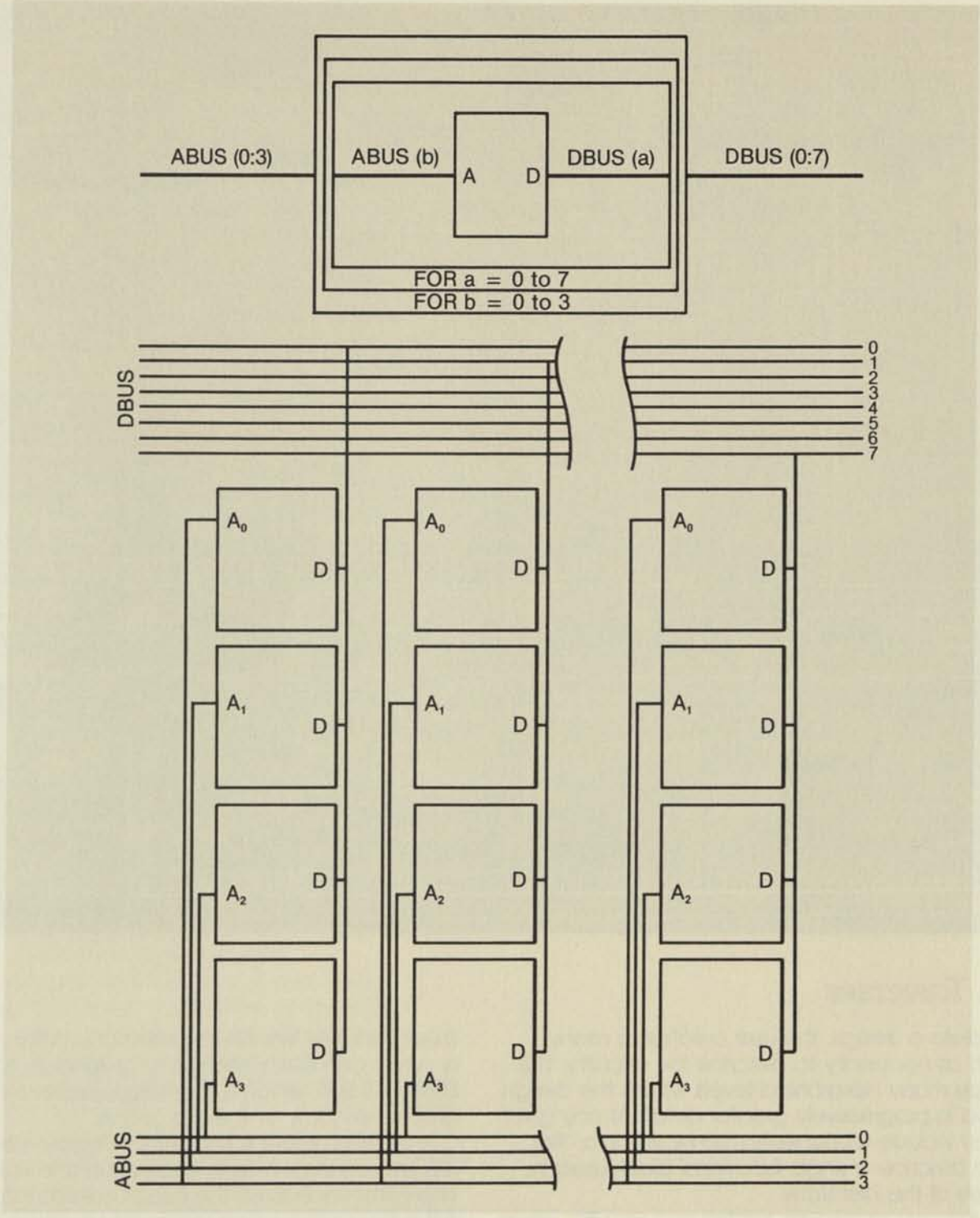

ABUS (0:3)
ABUS (b)
A
D
DBUS (a)
DBUS (0:7)
FOR a = 0 to 7
FOR b = 0 to 3
DBUS
0
1
2
3
4
5
6
7
A0
A1
A2
A3
D
ABUS
0
1
2
3

Figure 12

# DESIGN CREATION TOOLS

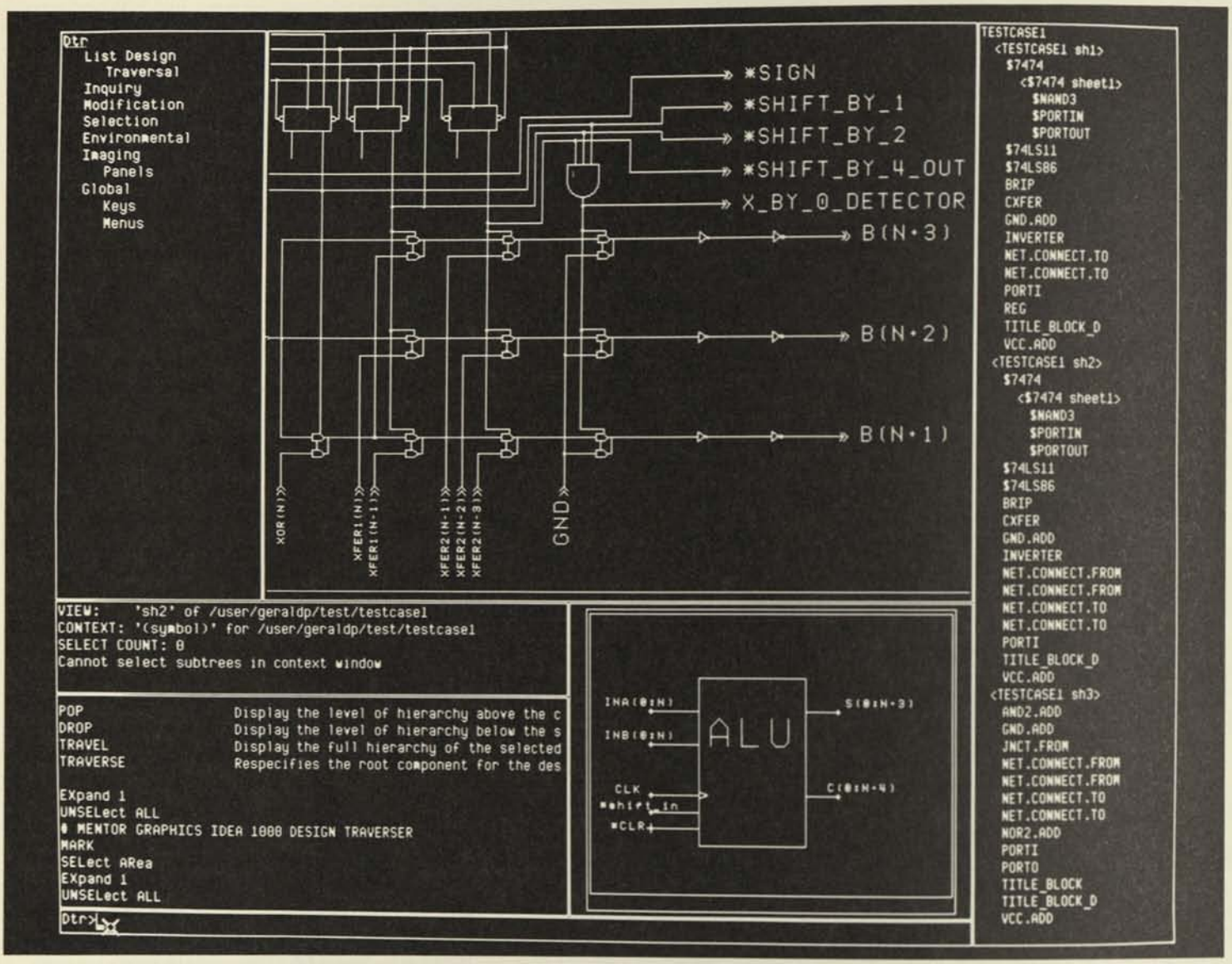



Figure 13

## Design Traverser

To complete a design, the user creates as many worksheets as necessary to describe the circuitry. This may include many hierarchical levels where the design is described in progressively greater detail. At any given level, it may include worksheets connected in a "flat" manner to describe a single functional block used in higher levels of the hierarchy.

To give the user maximum control and mobility when working within a hierarchical design, a Design Traverser package has been included, with a typical design shown in Figure 13. The right-hand column uses a vertical listing scheme to organize the design's various worksheets, and horizontal identation to indicate each sheet's relative level in the hierarchy. Within this list, a cursor can be positioned to graphically select any desired sheet, which is then displayed in the main graphic window at the top center.

In addition there is a context window which performs the same function as its counterparts in the Symbol and Network Editors. The cursor can be positioned on a functional block and the system requested to display how that block is expanded at the next level down in the design hierarchy. Next to the context window is a status window which shows worksheet selection in terms of the IDEA 1000's filing notation. The Design Traverser's combination of indented list, graphics

windows and file status gives an optimum interface for interacting with complex hierarchical designs.

## Design Expander

The task of the Design Expander is to process the completed worksheets and produce a finalized description of the design in absolute detail. The output of the Design Expander is a file usable by other software tools, both internal and external to IDEA 1000.

When processing a design's worksheets, the Design Expander works down through the hierarchy, adding more detail at each level. For each sheet, it performs all operations necessary to expand case frames into absolute design elements. Where replication of components has been called for, multiple copies are produced and each is given an identical number. Where conditional cases have been indicated, the correct choice is selected based on user-supplied parameters.

When the Design Expander has reached the lowest level of the hierarchy specified by the user, the design has been translated into a list of its most primitive components and the nets connecting those components. The Design Expander's processing can be "halted" by the user at any particular hierarchical level. This feature allows the substitution of functional models (see DESIGN ANALYSIS TOOLS sheet) to represent components not yet designed down to their most primitive elements. It also allows functional models to replace completed components in a form that simplifies their processing by other IDEA 1000 tools, such as the Interactive Logic Simulator.

If the design was non-hierarchical and simply involved the "horizontal" connection of series of worksheets, then it can be merged with a file of property values which describe the design's physicalization, including details such as board position and gate packaging, etc. In this case, the result would be an actual hardware description of the design.

In any case, the Design Expander can pass its output to the Netlister, which creates design data file usable by external physical CAD systems or analysis tools. The Design Expander output can also be used by the IDEA 1000's Interactive Logic Simulator or Timing Verifier, powerful debug tools which provide the software equivalent of a state/timing logic analyzer and a pattern generator.

## Programmable Netlister

The Mentor Netlister is a universal tool for conveying design files to an external simulator, CAD system or data base. It formats the output of the Design Expander according to definitions in a brief Pascal program entered by the user.

The Netlister is provided with a set of Pascal procedures which greatly simplify the task of searching the Design Expander output. This allows the Netlister to function as a data base query tool as well as a netlister to external systems. Calls are available for sequencing through nets, components, pins and properties. Additional procedures allow any of these to be formatted into ASCII strings for storage or printing.

Through Pascal coding, the user has great flexibility in determining the output of the Netlister. For instance, arbitrary text can be included along with connectivity information, component names updated, and specific properties located and processed. In this manner, the IDEA 1000 Netlister avoids the problems associated with fixed format Netlisters. Mentor provides many of the more common program interfaces. For additional formats, only a brief Pascal program is necessary to adapt the Netlister to your current output requirements.

Mentor

Mentor Graphics Corporation
10200 S.W. Nimbus Avenue, G-7
Portland, OR 97223 (503) 620-9817

## Officers

Thomas H. Bruggere
President,
Chief Executive Officer

Gerard H. Langeler
Vice President, Marketing

David C. Moffenbeier
Vice President, Finance

Stephen Swerling
Vice President, Engineering

Thomas J. Fretz
Vice President, Sales

## Directors

Thomas H. Bruggere
President,
Chief Executive Officer

David R. Hathaway
General Partner
Venrock Associates
New York, NY

Robert E. Schroeder
Former President,
Qume Corporation
Palo Alto, CA

## Principal Investors

Greylock Management
Corporation
Boston, MA

Hambrecht & Quist
San Francisco, CA

Lamoreaux, Glynn & Associates
San Francisco, CA

L.F. Rothschild, Unterberg,
Towbin
New York, NY

Sutter Hill Ventures
Palo Alto, CA

Venrock Associates
New York, NY

## Auditors

Peat, Marwick, Mitchell & Co.
Portland, OR

## Bank

First Interstate Bank of Oregon
Portland, OR

## Legal Counsel

Stoel, Rives, Boley, Fraser & Wyse
Portland, OR

## Sales Offices

2001 Gateway Place
Suite 195
San Jose, CA 95110
(408) 294-9933

4000 MacArthur Blvd.
Suite 3000
Newport Beach, CA 92660
(714) 851-6431

420 Bedford Street
Suite 160
Lexington, MA 02173
(617) 863-5776

4825 L.B.J. Freeway
Dallas, TX 75234



Printed in USA 1001A SWW 11/82

Mentor : No timeout ?

Please reset the demo when you are done : what ? to do

Remove UNDO

Press centre key over one of the following :

The ~~p~~o

You can draw a line by first selecting ~~that~~ "Draw a Line" ~~from~~ with the centre button.
~~The mouse will then~~
Then press the left button to draw the line.

Leave out Learn to Copy a Symbol.

Pg 1 Electronic Design Station

~~This program enables you to~~

Push the center button on the "mouse" to start. The "tail" of the mouse should point away from you.

pg2

The "mouse" is the small black box in front of you.
The wire should point away from you.

The mouse is the small black box in front of you.
It's "tail"
With the "tail" pointing away from you, like
It's its "tail" should point away from you.
Left

How
Let's learn to use the mouse:

- Left key:
  Action happens
- Center key:
  Selects menu item
- Right key:
  Press to ~~restart~~ start again restart program

Press center button to continue

pg 3

The flashing star, or pointer, is moved by sliding the mouse around on the table.

~~To program the key~~

~~Use the ce~~

To 'program' the action of the left key, move ~~move the flashing pointer star~~ star ~~over the~~ by sliding the mouse around until pointer over ~~it is near~~ the desired menu item in the box below and press the centre button.

Try selecting TWO

wrong - message OK.

pg 4

pg 4 How to draw a line.

Find ~~program the mouse~~ select
"Draw a Line" with the center button.
Do that now.
Then draw a line ~~over~~ to in the ~~the~~ work
area on the right using the left button.
Draw as many lines as you like. ~~The computer will correct them when~~
~~To continue select:~~
~~Next~~ To continue, select:
Learn to copy a symbol with the
center ~~key~~ button

Pg 7 Now ~~you~~ you can work on a ~~real electronic drawing of a timer~~ an electronic timer.

~~start~~

real electronic drawing * of a .....

Find select your skill level below.

Please remember to press the right button when you are finished.

draw a
pg 5 How to ~~copy~~ a symbol ?
~~(such as the logic gates below):~~

~~First select~~ "Draw a Symbol
First choose "~~Select and Copy~~" with
the center key.
Then pick up a symbol (bottom)
by moving the pointer to it and pressing
the left button. & Keep the button
pressed until you are ready to "drop" it.
Now ~~draw~~ draw a symbol.
Try drawing a symbol.

pg 6 Good. Try that again.

Captions under symbols
saying what they are.

Pg 8: Watch the computer
draw a copy of the top drawing

Beginning Earth:
The computer ~~will~~ will now draw
a circuit ~~of~~ diagram of a timer.

(Draw it.)

~~Your task is to copy the upper portion~~
A portion of the timer is now drawn above right.
~~Copy~~
~~Complete~~
~~Try~~ and complete the ~~circuit~~
~~Try~~ Try and copy the circuit by completing the
drawing below right.

Omit UNDO
everywhere.

Acknowledgement when finished successfully, and auto
restart. Timeout?