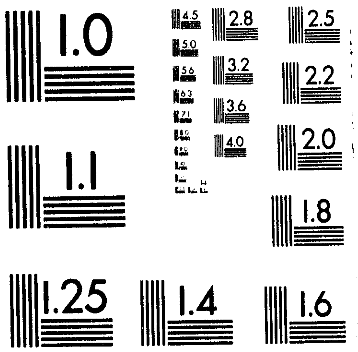
1.0
1.1
1.25
1.4
1.6
1.8
2.0
2.2
2.5
2.8
3.2
3.6
4.0
4.5
5.0
5.6
6.3
7.1

1 of 1

Conf-940453--2
SAND94-0402C

The Advent of Failure Analysis Software Technology

Christopher L. Henderson and Richard D. Barnard*
Sandia National Laboratories P.O. Box 5800, Albuquerque NM 87185
*Schlumberger Technologies, 1601 Technology Drive, San Jose CA 95110

Abstract: The increasing complexity of integrated circuits demands that software tools, in addition to hardware tools, be used for successful diagnosis of failure. A series of customizable software tools have been developed that organize failure analysis information and provide expert level help to failure analysts to increase their productivity and success.

I. MOTIVATION

As integrated circuit (IC) technology continues to increase in complexity, the way in which we perform failure analysis must change. Intel indicates that the success rate for analyzing complex microprocessors will approach zero by the year 2000 [1]. IBM estimates for its current production technologies that the lower levels of interconnect are between 92 and 95% obscured by the upper levels of interconnect [2]. Needless to say, the analyst's ability to obtain data from the IC through physical methods is becoming increasingly limited. In the future, failure analysis must turn to the computer to aid in the failure analysis of these complex ICs.

Historically, failure analysts have relied on various types of hardware to obtain information from a failing IC. For early IC technologies, failure analysts used optical microscopes and microprobe stations. As IC manufacturers added more functionality to circuits and improved the fabrication process, the feature sizes on the ICs were made smaller. The scanning electron microscope became the workhorse. Today's ICs are so complex that days or weeks of ATE testing, electron beam probing, focused ion beam studies and transmission electron microscope analysis may be required to isolate and determine the cause of failure. This hardware is very expensive to purchase and operate. A typical failure analysis laboratory attached to a large semiconductor manufacturer may have an investment of 3 or 4 million dollars in equipment for performing failure analysis [3]. This large equipment investment places complete failure analysis capabilities beyond all but the largest semiconductor manufacturers.

By contrast, software for failure analysis is largely an undeveloped field. During the late 1980's some computer-controlled hardware became available, but there is still very little software that helps the analyst determine the cause of IC failures [4,5,6]. As hardware reaches fundamental limitations in analytical capability, we need a change in thinking. We need to begin putting research dollars into software as well as hardware. A small investment in software troubleshooting research has the potential for a large payoff by generating affordable tools that can circumvent the inability to control and observe internal nodes on today's ICs.

This paper describes the development of a series of computer products that will aid in the task of failure analysis. We have developed three failure analysis tools (computer software programs) that move failure analysis from exclusively a hardware task to a software task. The four products FASTData, FASTReport, FASTHelp, and FASTAdvice use the common acronym FAST (Failure Analysis Software Technology). FASTData organizes all the failure analysis data into an object-oriented database. FASTReport is a subset of FASTData; the routines allow the efficient generation of failure analysis reports. FASTHelp centralizes all failure analysis help information into a hypertext file system. And FASTAdvice provides expert level assistance and training. These programs can form the basis for more powerful computer diagnostic tools (discussed briefly at the end of this paper). The FAST software products run on a SUN Workstation using X-Windows as a user interface (see Fig. 1). The SUN workstation was chosen because of its capabilities for handling IC layouts and schematics and its compatibility with existing software products.

II. FASTDATA

Failure analysis information is not always well-structured data. Most failure analysis laboratories use several filing systems to organize and track failure analysis data. A typical laboratory might store the basic failure analysis data (customer, date submitted, analyst, etc.) in a simple database on a personal computer. Some larger corporations may store this information in a relational database on a mainframe computer. As each new analysis is started, a new entry in the database is generated. The entry is updated after the analysis is finished. Information generated during the analysis (curve tracer plots, waveforms, photographs, textual data, and the failure analysis report) are typically stored as hardcopy in file cabinets. The layout and schematic plots are stored as hardcopy in tubes or

DISCLAIMER

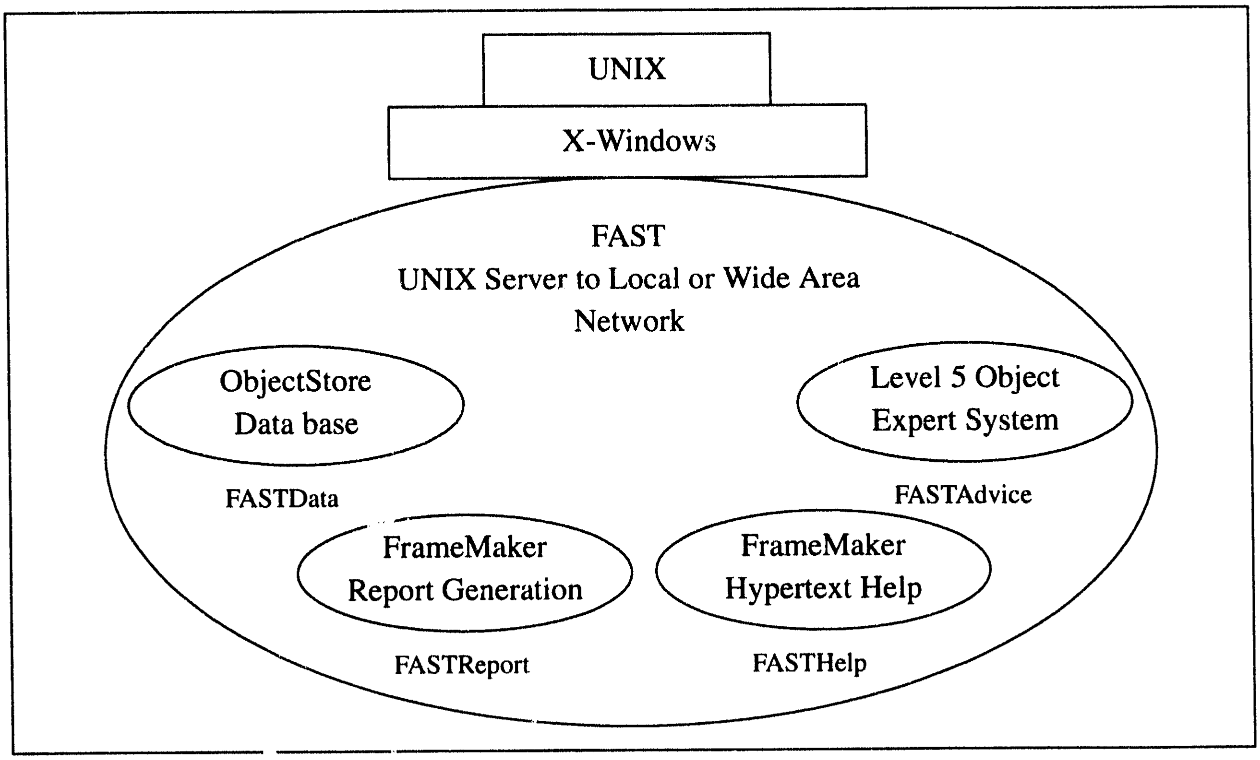

Figure 1. The Architecture of the FAST Project

map cabinets, or not at all. These data can all be stored efficiently in an object-oriented database (see Fig. 2).

Database Foundation

An object-oriented database allows the creation of classes of data. Within each class the developer can describe different types of data ranging from boolean data to text data, audio and video data, to layout and schematic data. Classes can inherit characteristics (or certain variables) from other classes. Data are entered into the variables within the classes by creating instances (or copies) of the class. An object-oriented database was chosen to allow more interactive storage and retrieval of different types of information. Relational databases are as much as two to three times slower than an object-oriented model for the heterogeneous types of data involved. If the class structure is designed correctly, the object-oriented database can also store information in a more space efficient manner than a relational database (the developer only creates instances for the data already available). Because the data are treated as objects, they can be manipulated and edited by routines that work on single types of objects. For example, an image manipulation and display program can be used to edit and manipulate any image in the database.

The Framework

FASTdata is a framework for organizing failure analysis information. The purpose of the FASTdata tool is twofold. For the analyst, it provides easy storage and retrieval of information gathered during an analysis. For the manager, it facilitates lab management, resource allocation, and period reporting. The framework manages text, images, waveforms, audio, and other lab information using industry accepted common formats, provides intuitive information retrieval, and easily generates professional reports. The foundation of the framework is a distributed object oriented database. A distributed database allows multiple users to share information. Lab information is easily supported by this object-based model (see Fig. 2). Lab objects include activity types, engineers, equipment, and analyses, all of which can be scheduled. An analysis is a collection of lab activities with assigned equipment and engineers. Similarly, an activity consists of a collection of objects such as text files, images, and analog and digital waveforms. Advanced databases also handle large binary objects such as audio and video information.

FASTdata has three defined work areas: Scratch Pad, Database, and Report. Data, as represented by the icons shown in Fig. 3, is moved among the three areas in a drag-and-drop fashion. The Scratch Pad is the framework's interface to the external file system. Objects in the Scratch Pad are created from imported files. Likewise, files can be created by exporting an object to the file system. The Database work area allows the user to organize information relating to a lab activity, engineer, and equipment. It also stores start-up information, such as customer and device type, associated with the analysis. Queries about objects or object attributes can also

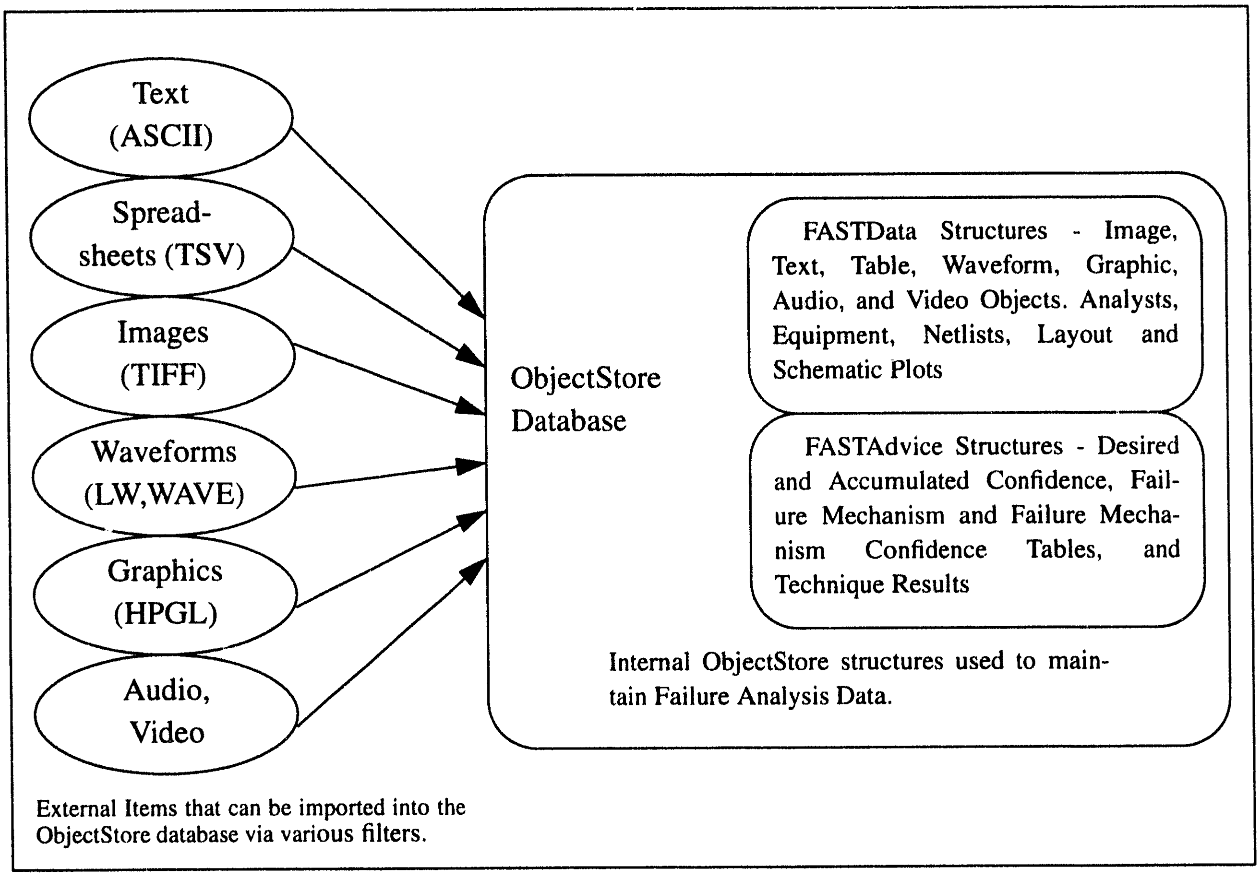

Figure 2. Diagram illustrating the information stored in FASTData.

generate information in either the Scratch Pad or Database work areas. Lastly, the Report work area allows the user to organize a subset of information into a final summary report.

FASTData is configured with basic information, which includes a list of activity types performed in the failure analysis lab, such as failure analysis tasks, equipment used for each activity, and engineers qualified to perform a task. Other framework items such as report distribution lists and tag index names, used in database searches, are also initially set up. Entering this configuration information permits easy "point and click" entry selection when accessed by the user. It also safeguards against erroneous data entry into the database.

Collecting Information

Information is categorized in the database on a per analysis basis. Start-up information such as customer, device type, point of failure, and package type is included in the database. From this information the analyst begins the examination of the failure. Since this information is usually kept either in a database or a file, the framework provides a means to import the information and parse it into the correct database fields. A panel for this data entry is also provided.

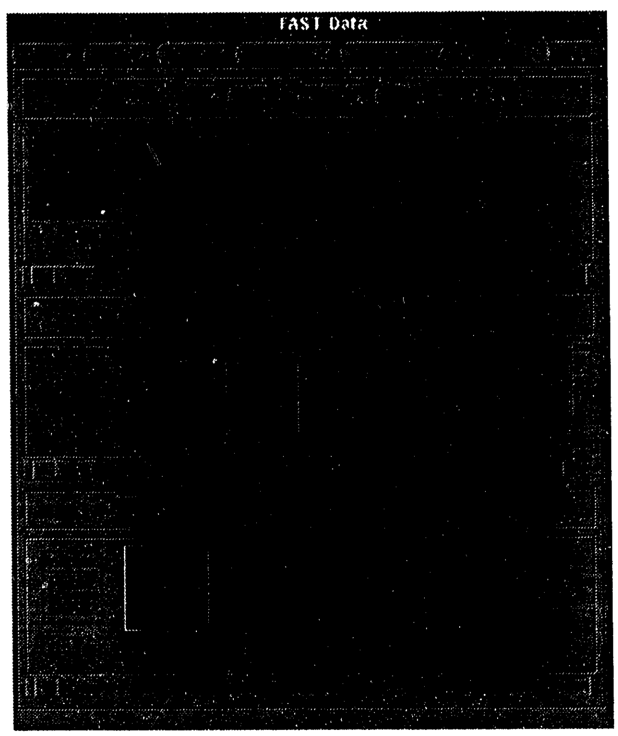

Figure 3. The FASTData User Interface.

FASTData imports images, waveforms, text, and audio through the UNIX file system. Standard formats are used, such as ASCII for simple text, TIFF or IDS image formats for images, VERILOG, IDS LW, or AWAVE formats for waveforms. Word processor formats such as FrameMaker MIF (Maker Interchange Format) and Microsoft Word RTF (Rich Text Format) are also used. These standard formats facilitate data collection during the analysis. As information is imported into the framework, objects representing the data are created. Images are compressed and waveforms sampled to produce icons representing the object information.

Database

The database organizes information collected during an analysis according to the associated lab activity. This is called a collection class in the database. Time stamps (e.g. start and complete time), equipment used, and task engineers associated with each activity task are also included within the database. The database collection class for an activity is represented as a folder in the database work area.

Information such as images, waveforms, text, or audio objects can simply be dragged and placed over the folder associating the information with the lab activity. Another special feature of the database work area is the ability to add an index tag to an object. For example, one may wish to add the tag "oxide rupture" for images showing capacitor oxide ruptures. This would allow engineers to easily locate images of this type in subsequent analyses. Lastly, data can be easily modified. Selecting an object automatically loads its respective editor. For text, the user can use any ASCII editor. For images, the public domain program, xv, is invoked with the image object. From xv the user can enhance, crop or change the size of the image. Saving the information, and then exiting the respective editor, signals the framework to automatically update the object stored in the database with the new information.

Creating Reports

The analysis report is the end result of the device analysis procedure. Information (data objects) can be dropped into this work area. If the user chooses to review the information in the report area, each object renders itself in the MIF format and the resulting document is displayed using the popular FrameMaker publishing system. A filter from MIF to RTF can also be invoked from the framework. Another key feature is the automatic insertion of report distribution lists and analysis start up information. Report templates can be applied automatically to the report to add such things as the company logo, standard margins, and page numbering.

Reviewing Past Analyses

The analyst can review all the information stored in the database for an analysis. The review function in the database work area shows the entire history of the analysis currently viewed. It sequentially opens each activity folder in the database, rendering each object and related information about engineers and equipment as a document. The editor is then automatically loaded with this document. This history function is useful to the analyst reviewing the current status of an ongoing analysis and to the analyst performing subsequent analyses requiring more information than the final summary report. Another means of reviewing information is by creating a query. Objects from previous analyses can be copied into a current analysis when the object's attributes match the query. Queries are formulated by selecting an object's attributes and entering data to be matched. As previously mentioned, customized tags can also be associated with each object. These tags are very useful in relating data from subsequent analyses. Lastly, summary reports can be indexed from the framework and easily loaded into the integrated editor.

Scheduling

Date and time information stored in the database can also be used for scheduling resources such as equipment and engineering staff. As activities and new analyses are created in the database, date and time information can be entered. This information can then be used for scheduling purposes. To enter date and time values a calendar panel is provided. Also a monthly view of schedules for engineers, equipment, activities, and analyses is provided.

Measuring Lab Productivity

FASTData can also assist the lab manager. Information from different objects, such as date and time, customer name, and failure mode, can be retrieved from the database and a period report can be generated. Period reports include cumulative information on activities, equipment, engineers and customers. Each period report collects information according to the specified date range. This data is formatted to accommodate data export to a spreadsheet or database.

In summary, FASTData uses ObjectStore, an object-oriented database, to store the failure analysis information. Furthermore, FASTData provides the mechanisms necessary to import many types of failure analysis data into the database. FASTData also provides the means to examine the data in the database through the use of the structured query language. FASTData can also be used to maintain calendars for equipment usage and analyst availability. Once the data has been imported into FASTData or generated within FASTData, a report for the analysis can be generated semi-automatically as the user drags the necessary objects from the scratch pad area to the report area.

III. FASTHELP

Failure analysis help information can be difficult to locate. Most failure analysis help information is either in book for-

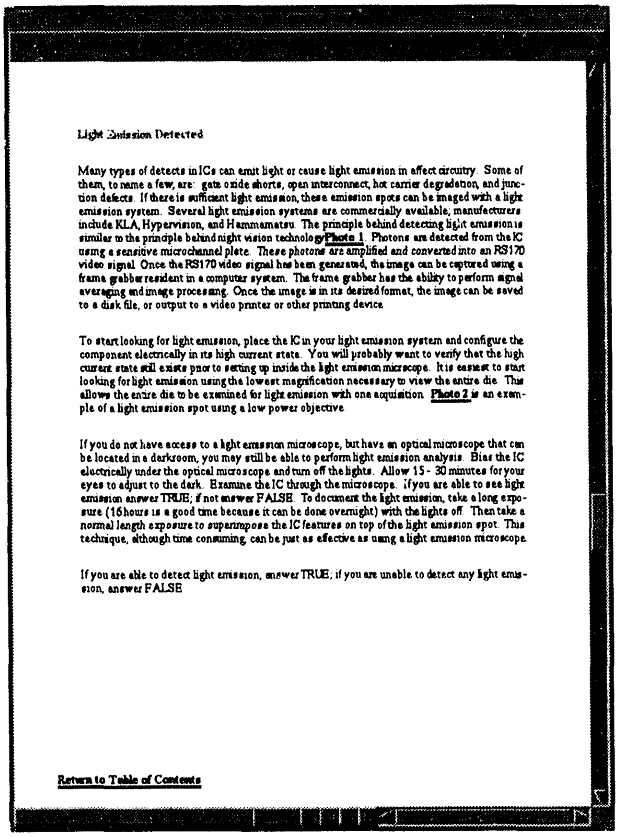

Figure 4. The FASTHelp Hypertext System.

mat, technical papers, (both non-electronic) or in the analyst's brain. However, there are only a few failure analysis books or manuals available to the failure analyst [7,8,9]. Even if these books are available, they may be difficult to locate (unless the analyst personally owns a copy) or may cover failure analysis techniques for technologies that are very old. Help for analyzing the most current technologies and information on the most current techniques usually exists in a few expert analysts' minds. These experts may not always be available (because of other job demands, or worse yet, because the company cannot afford to hire them or keep them).

We have constructed a hypertext help system to address the issue of providing timely help to the failure analyst wherever and whenever he or she needs it. An example screen of the hypertext help system is shown in Fig. 4. The help system is a series of documents that are electronically linked by a concept known as hypertext. A hypertext link allows the analyst to click on a highlighted word or topic and go right to the subject of interest. The help system is comprised of a master index and a help file that corresponds to each of the modules in FASTAdvice. Both general subject matter and specific failure analysis techniques are addressed. The hypertext help system contains textual material, graphs, and photographs and can also support audio and video clips.

The help system is implemented in FrameReader, a view-only version of FrameMaker. FASTHelp can be run as a stand-alone program or can be called from within FASTData or FASTAdvice. The photographic, audio, and video information is stored outside of the help file to allow FASTHelp to run faster. Photographs are stored as TIFF (tagged information format files). The audio/visual data in FASTHelp requires considerable storage space. Photographs consume approximately 250 kilobytes per photograph. CD-ROM quality audio requires 44 kilobytes per second while video requires approximately 7.5 Megabytes per second. Because of the large disk requirements for video and the inability to serve video over a local area network, video help data is probably still several years off.

IV. FASTADVICE

Historically, expert advice on failure analysis has been provided by analysts who have had years of experience analyzing failed ICs. These analysts act as mentors to the newcomers, tutoring them in the skills and techniques necessary for failure analysis. These experienced analysts can be hard to come by. There is currently no way of storing the knowledge that is contained in an experienced analyst's mind. If that individual leaves or is promoted, the experience and knowledge leaves with the individual. When an experienced analyst leaves, the operation of a failure analysis laboratory can be devastated.

We have developed a software program to provide expert level assistance and training in a failure analysis laboratory. Based on ICFAX [10] (Integrated Circuit Failure Analysis Expert System), this program can provide expert level advice on one failure analysis technique or on the entire failure analysis process. Although the existing ruleset is geared toward CMOS ICs, the software contains a graphic rule editing routine that allows easy manipulation of the ruleset. The analyst in the failure analysis lab can modify existing rules, delete existing rules, or add new rules to the existing set. The analyst will also have the ability to perform multiple tasks in parallel (when analyzing multiple failures or when performing yield analysis) and to perform analyses with differing equipment sets.

Failure analysis reasoning is a two-stage reasoning process. First, you deduce what you know and what to do next (using the information already on hand). Second, you gather data and proceed along a line of reasoning as you perform a technique. FASTAdvice uses this model as the basis for its reasoning. The overall process is summarized in Fig. 5.

FASTAdvice is a modular program with 82 separate modules. The same reasoning system is applied within each module. Each module contains a similar object structure, rule structure, demon structure, and a single goal. FASTAdvice

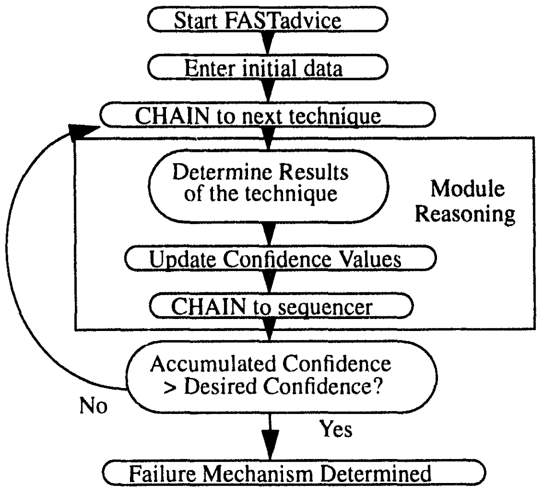

Figure 5. Basic FASTAdvice flow diagram.

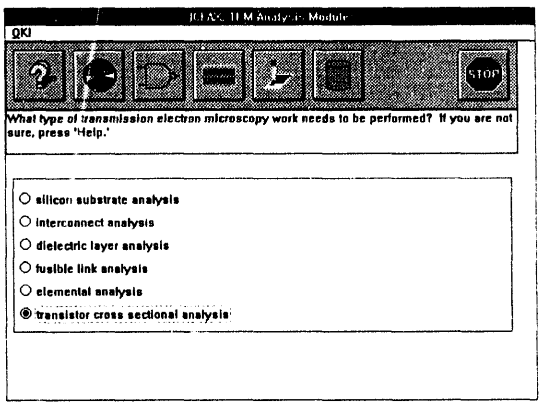

Figure 6. The FASTAdvice Graphical User Interface.

uses the Level 5 Object expert system as its inference engine. The graphical user interface is depicted in Fig. 6.

The Module Reasoning System

The Module Objects: Level 5 Object uses objects as its paradigm for representing knowledge and data. Level 5 Object lets the developer define classes, instances of the classes, and attributes (objects). Within the FASTAdvice modules there are six classes: the Domain class, the Features class, the Help class, the Pie Chart class, the Pie Chart Data class, and the Technique class. The Domain class contains information used across each of the modules. This information includes the results of each of the techniques, the confidence values of the failure mechanisms, the desired confidence level, and some basic device information. This allows each module to have access to the results of previous tests and device information in order to reason about what to do. The Features class contains the mechanics necessary to launch the applications displayed in the tool bar at the top of FASTAdvice. Each button is tied to an attribute in the Features class. When the analyst presses one of the buttons, Level 5 Object executes a method to launch the routine. The Pie Chart Class contains the structure of the pie chart used for displaying confidence values. The class defines the type of chart, the style of the chart, the label attachments, and the series attachments. The Pie Chart Data class contains an attribute to store the Failure Mechanism Codes that will be displayed in the pie chart and an attribute to store the confidence values to be displayed in the pie chart. The Help class contains the structure necessary to launch FrameMaker (the help system). The keyword, the document, and the help file activation status are stored in this class. The Technique class contains the failure analysis knowledge for the particular technique. The attributes correspond to the questions and results the failure analyst would ask and determine for the technique or procedure. The class may also contain additional attributes that handle multiple levels of reasoning (backchaining) if the procedure is complex. The class also contains an attribute that serves as a goal for the module.

The Module Goal: The goal can be thought of simply as the results of a particular technique. For example, if the analyst is performing a gross leak test on an IC, the goal would be the results of gross leak testing.

The Module Rules: The rules in FASTAdvice serve a different function than rules in a standard artificial intelligence application. Normally rules are "heuristics" or rules of thumb concerning a particular subject. In FASTAdvice the rules control the order in which questions are asked. For example:

```
RULE for xray foreign object in package material
IF xray radiographs OF Technique IS foreign object in package
material
AND NOT object in package material bridging OF Technique
AND comments OF Technique <> ""
THEN Xray Results OF Technique
AND Xray Results := "foreign object in package material"
```

This rule can be understood in the following manner. When Level 5 Object begins execution of this module, an introduction screen is displayed. The x-ray module has no introduction screen, so this step is by-passed. The introduction screen is used mainly to alert the analyst that the following technique or step is destructive. Level 5 Object then loads the goal (in this case Xray Results OF Technique). Level 5 Object then queues all of the rules that have as an element of their conclusion Xray Results OF Technique. Level 5 Object then tries each of these rules. The rules are constructed so that the order is not important. The first attribute of the first rule is tried. In FASTAdvice each attribute used in the ruleset has a text facet attached to it. This text is displayed to the analyst when the attribute is tried. Each attribute also has a facet that describes its type. FASTAdvice makes use of the simple (boolean), compound, multicompound, time, and numeric attributes. After the analyst makes a choice from the answers, the value of the attribute is set. If the value matches the value in the rule that is currently being considered, the condition succeeds and the next condition in the rule is tried. If the value does not match, the condition fails and the rule fails. If the rule fails, the inference engine tries the next rule in the ruleset. Because of the construction of FASTAdvice, one rule will always succeed. Once a rule succeeds, it is "fired". The actions list in the rule are tried. In the example above, Xray Results OF Technique is set to TRUE, and Xray Results (shared domain attribute) is set to "foreign object in package material".

The Module Demons: The demons in FASTAdvice control the failure mechanisms and their confidences. An example of a demon in FASTAdvice is shown below.

```
DEMON for conf foreign object in cavity
IF Xray Results = "foreign object in cavity"
THEN Failure Mechanism Code Probability Matrix[68]:=25
AND Failure Mechanism Code Probability Matrix[51]:=25
AND Failure Mechanism Code Probability Matrix[52]:=8
AND Failure Mechanism Code Probability Matrix[106]:=7
AND Failure Mechanism Code Probability Matrix[61]:=8
AND Failure Mechanism Code Probability Matrix[15]:=27
AND PURSUE adjust mechanism OF technique
AND CHAIN "icfax"
```

Level 5 Object will check to see if it can process demons anytime that the value of one or more of the conditions changes. In the case of the demon above, when Xray Results receives the value "foreign object in cavity" the demon will be fired. The demon will then update the confidence in six different failure mechanisms. Mechanism 68 receives a confidence of 25, mechanism 51 receives a confidence of 25, and so on. The PURSUE action causes the sum of the confidence values to be renormalized to 100. This eliminates the possibility of a failure mechanism having a confidence greater than 100. Finally, the CHAIN action causes Level 5 Object to leave this particular technique and return to the sequencing module. These confidences are based on the expert knowledge upon which FASTAdvice is constructed.

The Sequencer Reasoning System

The Sequencer Reasoning System has been constructed to handle four confidence regimes, differing equipment sets, and both wafer and packaged part analysis. The knowledge base was developed with three overarching considerations: (1) tests will be performed from least destructive to more destructive; (2) test will be performed from least costly to more costly; and (3) tests will be performed from the least time consuming to the most time consuming.

The Top Level Goal: The Sequencing Module contains the top level goal for FASTAdvice. The top level goal is Failure Mechanism. It is worthwhile to note that the top level goal is not Root Cause. For the purposes of this paper the failure mechanism is the observable cause of failure on the IC. The root cause is the event, action, or design decision that initiated or caused the failure. For example, the failure mechanism ascribed to a failing IC might be electrical overstress. The root cause might be that a technician used an improperly grounded oscilloscope. Identification of the failure mechanism is a more bounded task and can be accomplished (in most cases) without in-depth knowledge of the wafer fabrication, package assembly, or troubleshooting operations. Identification of the root cause may require in-depth knowledge of the wafer fabrication process, the package assembly process, or the user's application and environment. While these activities are integral to the failure analysis activity, they are difficult to bound and quantify. The developers will investigate the possibility of capturing this knowledge for future releases. It is important to realize that the software, as it exists, will not provide assistance to determine the underlying root cause.

The Sequencer Objects: The Sequencing Module contains eleven classes. The Technique, Features, Help, Chart, Pie Chart Data, Series and Domain classes are structured identically to the module classes. Four additional classes exist in the sequencing module. These are: the Activity Conditions class, the Activity Equipment Available class, the Equipment Available class, and the Analyses class. The Activity Conditions class contains 82 simple (TRUE/FALSE) attributes that match one to one for each module. These attributes are set to TRUE by rules that determine when it is time to perform the activity or technique. The Activity Equipment Available class also contains 82 simple attributes that correspond one for one with each module. These attributes are set to TRUE when the equipment for performing the activity of technique is available. The Equipment Available class contains an attribute for every piece of equipment that is normally used for failure analysis. When the equipment is available, the attribute is marked as TRUE (Note: more than one piece of equipment may be required to perform an activity or technique). Finally, the Analyses class provides a link into the FASTData database to store or retrieve analysis information. Its attributes match those of the FASTData database.

The Sequencer Rules: There are two groups of sequencing rules: (1) the rule group for determining if the equipment is available to perform the technique, and (2) the rule group for determining if the time is right for performing that activity. The rule group for equipment availability consists of 82 rules (one per module). An example rule is shown below.

```
RULE for edxpack equipment available
IF scanning electron microscope OF Equipment Available
AND energy dispersive xray detector OF Equipment Available
THEN EDX Package OF Activity Equipment Available
```

Simply, this states that if a scanning electron microscope and an energy dispersive x-ray detection system are available, the analyst has the equipment necessary to perform an energy dispersive x-ray (EDX) analysis of the device package. This is one of the prerequisites required to perform EDX analysis on the device package. The rule group for determining the correct time also consists of 82 rules. An example of this type of rule is shown below.

```
RULE for grosleak fineleak pass
IF Fine Leak Results = "passed"
THEN Gross Leak OF Activity Conditions
```

This rule states that a gross leak hermetic seal test can be performed after a fineleak hermetic seal test. This condition is a prerequisite to performing a gross leak hermetic seal test.

The Sequencer Demons: The overall execution of FASTAdvice is data-driven. The core of the data-driven approach is captured in the sequencer demons. The sequencing demon set consists of approximately 500 demons. These demons have the following structure.

```
DEMON for acoustic conf exhaustive package single
IF Accumulated Confidence < Desired Confidence
AND Acoustic Imaging Results = "not known"
AND Analysis Comments = "exhaustive packaged single"
AND PURSUE Acoustic Imaging OF Activity Equipment Available = TRUE
AND PURSUE Acoustic Imaging OF Activity Conditions = TRUE
THEN CHAIN "acoustic"
```

The first line of the demon is an identification descriptor. The second line is a check to make sure that the accumulated confidence is still less than the desired confidence. If the accumulated confidence exceeds the desired confidence, then all of the sequencing demons will fail and the failure mechanism will have been determined. The third line of the demon checks to make sure that the technique or procedure has not already been performed. This prevents FASTAdvice from entering an infinite loop. The fourth line of the demon determines the type of regime under which FASTAdvice will operate. There are twelve operating regimes based on low, medium, high, and exhaustive levels of confidence, wafer level failure analysis, single package part analysis, and multiple packaged part analysis. The fifth line of the demon checks to see if the equipment is available to perform the failure analysis (additional discussion located in The Sequencer Rules section). Finally, the sixth line of the demon checks to see if the time is right to perform the test (i.e., have all of the steps leading up to this activity been performed).

Failure Mechanism Confidence

One important feature designed into FASTAdvice is a confidence factor for the outcome of the analysis. The desired confidence attribute allows the analyst to enter a confidence value that will determine how "sure" he or she would like to be in the failure mechanism. A desired confidence value of 90 roughly means that the analyst would like to be 90 percent confident in the particular failure mechanism for the failure mechanism determined for the failing IC. A higher confidence requires more analysis to achieve the confidence level. For example, if the analyst received a packaged IC for failure analysis, performed some electrical testing, and determined that pin 7 is electrically open, he or she would be more than 50 percent confident that the failure is electrical overstress (in the absence of any further information). Now of course there are other mechanisms that can cause pin 7 to be open, but the analyst is reasonably confident (at least 50 percent) that the failure is caused by electrical overstress. The analyst arrives at this conclusion based on experience performing failure analysis for some number of years. If the analyst needs to be 90 percent confident in the failure mechanism, he or she will need to perform additional analysis (at a minimum, delid the IC and examine the die and bond wires using an optical microscope). This brief example describes the way FASTAdvice uses confidence.

The amount of work the analyst performs is directly related to the desired confidence level. Furthermore, there are four confidence regimes that further define the amount of work to do during the analysis. If the analyst requires only a low confidence, then FASTAdvice avoids expensive and time-consuming activities and techniques. If the analyst requires a very high level of confidence, then FASTAdvice utilizes all of the techniques available and uses rechecks at various important junctures.

FASTAdvice does not use any form of probability statistics, such as Bayesian or Certainty theory, to perform its confidence calculations. The confidence values are based solely on Sandia failure analysis expertise. Because Sandia experience may not be valid for all products, the FASTAdvice confidence data set can be modified. FASTAdvice is also designed to disallow confidence values below 0 and above 100. Each of the modules contains a method to normalize the sum of the confidence values to 100.

One important reason confidence is employed in FASTAdvice is to make the decision upfront as to the importance of the analysis. The desired confidence value should be set at the beginning of the analysis and not adjusted during the middle of the analysis. Important failure analysis work should be given a high confidence value so that the analyst has the greatest chance at a successful outcome. Failures ranging from one-of-a-kind field returns to other sensitive customer returns require that the job be done right the first time [11]. Failure to determine the true mechanism may result in loss of credibility or contracts, or have life threatening consequences

V. CONCLUSION

More complex devices require new approaches to performing failure analysis. We have introduced a framework of failure analysis software technology that can provide the basis for more advanced computer techniques to aid the analyst in the diagnosis of failing ICs. Within the next several years, ICs may very well become too complex to analyze using standard IC failure analysis tools. We need to begin developing advanced diagnostic tools that can circumvent the need for extensive physical troubleshooting. Software tools that can utilize the layout, schematic, and netlist information in conjunction with testing or simulation data must be developed if we are to provide value to the IC manufacturer and user in the future. There has already been preliminary work in this area[12-15]. The software tools described in this paper provide structured data models and reasoning methodologies for the task of failure analysis. More fundamental research needs to be done is in the area of model and structural based reasoning[16]. There are computational limitations associated with reasoning from first principles. Techniques for minimizing the computational overhead through partitioning and hierarchy need to be investigated. These techniques promise to bring failure analysis into the 90's.

ACKNOWLEDGMENTS

The authors would like to thank Bill Lee, Jerry Soden, and Dan Barton for their review and suggestions. This work was performed by Sandia National Laboratories and Schlumberger Technologies. The work at Sandia National Laboratories was supported by the U.S. Department of Energy under contract DE-AC04-94AL85000.

REFERENCES

1. Vimod Dham, (to be published in the *Proceedings of the 20th International Symposium for Testing and Failure Analysis*).

2. Dick Ross, Personal communication, Nov. 16, 1993.

3. Howard Dicken, "Semiconductor Reliability News," Vol. V, No. 8, Aug. 1993, p. 1.

4. N. Khurana and C. L. Chiang, "Analysis of Product Hot Electron Problems by Gated Emission Microscopy," *Proceedings of the International Reliability Physics Symposium*, Apr. 1986, pp. 189-194.

5. S. Concina and N. Richardson, "Workstation Driven E-beam Prober," *Proceedings of the International Test Conference*, Sept. 1987, pp. 554-560.

6. L. M. Bellay, P. B. Ghate and L. C. Wagner, "Computers in Failure Analysis," *Proceedings of the 16th International Symposium for Testing and Failure Analysis*, Nov. 1990, pp. 89-95.

7. *The Microelectronics Desk Reference*, eds. T. W. Lee and S. V. Pabbisetty, ASM International, 1993.

8. J. R. Devaney, G. L. Hill, R. G. Seippel, *Failure Analysis Mechanisms, Techniques, and Photo Atlas*, Failure Recognition and Training Services, 1986.

9. *Failure Analysis Techniques - A Procedural Guide*, eds. E. Doyle Jr. and B. Morris, IIT Research Institute, 1980.

10. C. L. Henderson and J. M. Soden, "ICFAX, An Integrated Circuit Failure Analysis Expert System," *Proceedings of the 29th Annual Reliability Physics Symposium*, Apr. 1991, pp. 142-150.

11. J. M. Soden and R. E. Anderson, "IC Failure Analysis: Techniques and Tools for Quality and Reliability Improvement," *Proceedings of the IEEE*, Vol. 81, No. 5, May 1993.

12. A. Noble, "IDA: A Tool for Computer-Aided Failure Analysis," *Proceedings of the International Test Conference*, Sept. 1992, pp. 848-853.

13. T. Viacroze, G. Fourquet, and M. Lequex, "An Expert System for Help to Fault Diagnosis on VLSI Memories," *Proceedings of the International Symposium for Testing and Failure Analysis*, Nov. 1988, pp. 153-160.

14. M. Marzouki and B. Courtois, "Debugging Integrated Circuits: A.I. Can Help," *Proceedings of the 1st European Test Conference*, Apr. 1989, pp. 184-191.

15. P. Mauri, "Computer-Aided Analysis of Integrated Circuit Reliability," *Proceedings of the NATO Advanced Research Workshop-Semiconductor Device Reliability*, Jun. 1989, pp. 127-136.

16. R. Davis, "Diagnostic Reasoning Based on Structure and Behavior," *Artificial Intelligence* Vol. 24, 1984, pp. 347-410.

DATE
FILMED
4 / 6 / 94
END