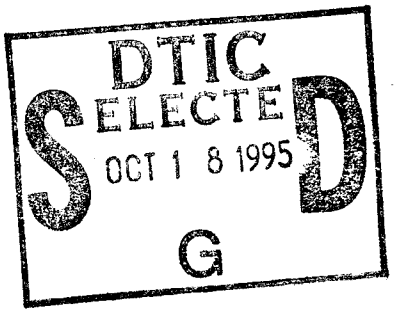
RL-TR-95-162
Final Technical Report
August 1995

INTEGRATED FINITE-ELEMENT GENERATION FOR INTELLIGENT MULTICHIP MODULE RELIABILITY ANALYSIS

Blackboard Technology Group, Inc.

D. Corkill



APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED.

Rome Laboratory
Air Force Materiel Command
Griffiss Air Force Base, New York

This report has been reviewed by the Rome Laboratory Public Affairs Office (PA) and is releasable to the National Technical Information Service (NTIS). At NTIS it will be releasable to the general public, including foreign nations.

RL-TR-95-162 has been reviewed and is approved for publication.

APPROVED: Dale W. Richards

DALE W. RICHARDS
Project Engineer

FOR THE COMMANDER: John J. Bart

JOHN J. BART
Chief Scientist, Reliability Sciences
Electromagnetics & Reliability Directorate

If your address has changed or if you wish to be removed from the Rome Laboratory mailing list, or if the addressee is no longer employed by your organization, please notify RL (ERSR) Griffiss AFB NY 13441. This will assist us in maintaining a current mailing list.

Do not return copies of this report unless contractual obligations or notices on a specific document require that it be returned.

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. AGENCY USE ONLY (Leave Blank)	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	August 1995	Final Jan 94 - Jan 95

4. TITLE AND SUBTITLE
INTEGRATED FINITE-ELEMENT GENERATION FOR INTELLIGENT MULTICHIP MODULE RELIABILITY ANALYSIS

5. FUNDING NUMBERS
C - F30602-94-C-0034
PE - 62702F
PR - 2338
TA - 02
WU - PL

6. AUTHOR(S)
D. Corkill

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
Blackboard Technology, Inc.
401 Main Street
Amherst MA 01002

8. PERFORMING ORGANIZATION REPORT NUMBER
N/A

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)
Rome Laboratory (ERSR)
525 Brooks Rd
Griffiss AFB NY 13441-4505

10. SPONSORING/MONITORING AGENCY REPORT NUMBER
RL-TR-95-162

11. SUPPLEMENTARY NOTES
Rome Laboratory Project Engineer: Dale W. Richards/ERSR/(315) 330-3476

12a. DISTRIBUTION/AVAILABILITY STATEMENT
Approved for public release; distribution unlimited.

12b. DISTRIBUTION CODE

13. ABSTRACT (Maximum 200 words)

This report describes efforts to develop an integrated finite-element mesh generation capability for the Rome Laboratory Intelligent Multichip Module Analyzer (IMCMA). IMCMA is a blackboard-based software tool that automatically applies finite-element and knowledge-based analysis to rapidly assess the thermal reliability of microelectronic multichip module (MCM) designs. IMCMA is a cooperative effort between Rome Laboratory, the Mechanical Engineering and Computer Science Departments at the University of Massachusetts, and Blackboard Technology Group, Inc. In IMCMA, the amount of modeling effort and expertise required of the designer has been reduced through: (1) use of a high-level representation of devices as the interface between the designer and the analysis tools; and (2) incorporating the modeling and analysis expertise of experienced design analysts into the software, making it available to less experienced designers. This was accomplished primarily by replacing modeling and mesh generation activities previously performed in IMCMA by software from Sandia National Laboratory with new, redesigned, and tightly-integrated IMCMA software modules. A 3D-viewer for displaying analysis results (complementing the existing 2D display facility) was also designed and implemented, and the existing graphical user interface was extended and enhanced. This has resulted in an IMCMA that is smaller, faster, more flexible, more portable, more available, and more self-contained than its predecessor.

14. SUBJECT TERMS
Multichip, Module, MCM, Blackboard, Artificial intelligence, Reliability, Thermal analysis, Computer-aided (see reverse)

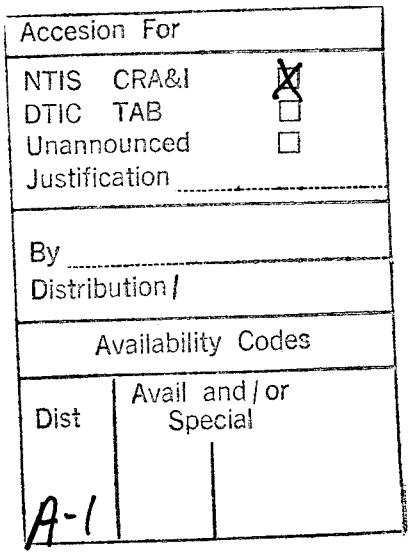
15. NUMBER OF PAGES

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
UNCLASSIFIED	UNCLASSIFIED	UNCLASSIFIED	UL

14. (Cont'd)

design, Finite element analysis, FEA, Modeling



Executive Summary

This report describes the results of our one-year effort in developing integrated finite-element mesh generation capabilities for the *Intelligent Multichip Module Analyzer* (IMCMA). IMCMA is a prototype architecture for an intelligent, automated analysis system that can be used to provide rapid reliability assessments of multichip microelectronic module designs. This effort is part of a cooperative effort between the Design Analysis Branch at Rome Laboratory, the Mechanical Engineering and Computer Science Departments at the University of Massachusetts, and Blackboard Technology Group, Inc. The IMCMA system is a blackboard-based software tool that automatically applies finite-element and knowledge-based analysis to rapidly assess the reliability of microelectronic multichip module (MCM) designs [1,2]. The software bases its evaluation on environmentally and operationally induced failure mechanisms. It is useful in assessing the reliability of advanced microelectronic devices that have no historical reliability data.

Early detection of design-related reliability problems can significantly decrease the development, manufacture, and testing costs of MCMs. This potential savings is offset by the labor-intensive nature of modeling as a means of detecting design-related reliability problems. Modeling an MCM requires significant expertise to build and, if necessary, remodel critical regions of an initial finite-element model. Typically, it may take a senior design analyst several weeks to construct and analyze an initial computer-based model of a microelectronic device. An expert design analyst must decide how to model the individual components, how to represent them so they can be used by analysis tools (such as an automated mesh generator), and how to interpret the results. The numerical results from the initial model will often indicate critical regions of the device that must be remodeled in order to obtain an accurate analysis of the mechanical stresses of these regions.

Prior to IMCMA, analysis tools lacked high-level models of microelectronic devices and loading conditions. Based on low-level geometric representations, these tools required a skilled expert to translate the high-level model of the device into

representations that the analysis tools could utilize. Conversion between different representations used in each tool was also left to the design engineer.

In the IMCMA system, modeling effort and expert-operator requirements have been reduced through two main techniques:

- Use of a high-level representation of devices as the interface between the designer and the analysis tools
- Capturing the expertise of experienced design analysts in an intelligent assistant to be made available to less experienced designers

Increasing the productivity of design experts significantly reduces the development effort, lead time, and cost associated with new MCMs.

The original IMCMA prototype architecture (Version 1.0) used FORTRAN-based mesh-generation codes developed by Sandia National Laboratory. Although these codes were successfully integrated into the IMCMA prototype, a number of problems stem from their use, including:

- **The Sandia codes are not well-integrated into IMCMA.** One of the requirements of the IMCMA system was that any off-the-shelf codes used in the system would be used without modification. For use as IMCMA KSs, this required that the codes use ASCII text files and piped commands to receive input information, and that IMCMA read FORTRAN-generated binary output files. Approximately one-half of the total processing time required for an initial analysis of an MCM device is expended in these interfaces.
- **The Sandia codes are inflexible.** A significant amount of code was added to IMCMA to work within the limits of the Sandia codes. Because of limitations in the ways the Sandia codes can be used, representations of "pseudo-components," components, and materials must all be used at appropriate times during the mesh-generation process.
- **The Sandia codes add a system-administrative burden.** The Sandia codes are large and require an extensive library of routines to install, build, and maintain.

- **The Sandia codes are not universally available.** The Sandia codes are available for governmental and educational users only. They are not readily available to commercial users. This makes the IMCMA system unavailable for evaluation by many of the manufacturers and developers of multichip modules.

In this effort, we replaced the activities performed by the Sandia codes with redesigned, tightly-integrated IMCMA KSs. We also designed and implemented a 3D-viewer for displaying analysis results (complementing the existing 2D display facility) and extended and enhanced the existing graphical user interface. As a result of this effort, the IMCMA 2.0 prototype is smaller, faster, more flexible, more portable, more available, and more self contained than its predecessor. In this report, we describe these activities and our results.

Acknowledgments

The IMCMA project is a cooperative effort involving a number of individuals from Rome Laboratory, the Mechanical Engineering and Computer Science Departments at the University of Massachusetts at Amherst, and Blackboard Technology Group, Inc. Without the friendly and open interchange of ideas and expertise among everyone involved with the IMCMA effort, the research described in this report would not have been accomplished.

Rome Laboratory provided the expertise in microelectronic devices and finite-element-based reliability assessment. Rome Laboratory also alpha and beta-tested the developing IMCMA 2.0 system.

From the UMass Mechanical Engineering Department, Professor Ian Grosse and his students, Prasanna Katragadda, Sandeepan Bhattacharya, Krishna Darbha, Mike Sheehy, Sarang Kulkarni, and Shankar Raman developed and maintained the FEECAP finite-element analysis code used in IMCMA and also used IMCMA for submodeling and in their design-of-experiments (DOVE) shell. They often served as unwitting alpha and beta testers as well.

At Blackboard Technology Group, Inc., Raymond de Lacaze implemented the integrated 3D mesh-generation code and designed and implemented the 3D graphics viewer.

Table of Contents

1 Introduction

Early detection of design-related reliability problems can significantly decrease the development, manufacture, and testing costs of multichip microelectronic modules (MCMs). For example, the Design Analysis Branch (ERSD) of the Electromagnetic Reliability Directorate at Rome Laboratory has successfully used the finite-element method for reliability assessment of MCM components to predict failure modes and assess their mechanical reliability [3,4]. Detailed simulation studies of microelectronic devices has been used to successfully predict the location and magnitude of critical thermal and physical stresses [5,6,7,8,9,10]. These studies can be used to predict the reliability and failure modes of the device.

Detecting design-related reliability problems using detailed simulation studies is labor intensive, and requires significant expertise to build and, if necessary, remodel critical regions of an initial finite-element model of MCM devices on the computer. Typically, it may take an engineer several weeks to construct and analyze an initial model of a microelectronic device on the computer. The numerical results from the initial model will often indicate critical regions of the device which must be remodeled in order to obtain an accurate model of these regions. For example, remeshing is needed to resolve meshing problems due to violation of geometric transitioning constraints or due to basic limitations of the mesh generator. Increasing the productivity of design experts will significantly reduce the development effort, lead time, and cost associated with new MCMs.

Prior to IMCMA

Prior to IMCMA, an expert designer had to decide how to model the individual components, how to represent them so they can be used by analysis tools (such as an automated mesh generator), how to interpret the results and potential failure of these tools, and how to reformulate the model for further analysis, if needed. The existing analysis tools lack high-level models of microelectronic devices and loading conditions. Based on low-level geometric representations, these tools required a skilled expert to translate the high-level model of the device into representations that the analysis tools can utilize. Similarly, the

expert had to relate the detailed analysis results back to the high-level model, understanding the implications of the detailed results to overall device reliability. Conversion between the many different representations used in the various analysis tools was left to the design engineer.

The goal of IMCMA is to reduce modeling effort and expert operator requirements through several techniques:

- Use of a high-level representation of devices as the interface between the designer and the analysis tools. The designer would define the device in terms of its components and the analysis system would use these high-level definitions to develop a detailed representation of the device. For example, a MCM might consist of a number of uniform rectangular chips and capacitors mounted on the substrate in a symmetrical pattern. The designer would specify the chips, capacitors, and the pattern, and the system would develop all the detailed submodels of critical features of the device.
- By capturing the expertise of experienced designers in an intelligent assistant for MCM device analysis, much of the labor-intensive aspects of reliability analysis can be automated and made available to less experienced designers [11]. Instead of the expert designer deciding how to use the tools to effectively model devices, the analysis system would develop and implement a modeling strategy for analyzing the device. The system would monitor the accuracy of the modeling process, detecting modeling problems such as idealizations resulting in singularities in the finite-element solution, violations of mesh transitioning constraints, and poorly structured meshes. Once detected, the system would reformulate the model or remesh until an acceptable model is developed.

These techniques form the basis of the IMCMA system.

A Brief History of IMCMA

The IMCMA effort was started in the Spring of 1992 as a set of cooperative efforts by the Mechanical Engineering and Computer

Science Departments at the University of Massachusetts and Rome Laboratory. At the end of a one-year effort, a basic IMCMA prototype was in place, which used a suite of FORTRAN-based finite-element mesh generation codes from Sandia National Laboratories and a FORTRAN-based finite-element mesh-analysis program developed by the Mechanical Engineering Department at the University of Massachusetts. Nine knowledge sources (KSs) were completed that took a high-level device specification through model simplification, finite-element generation, and thermal analysis [2,1].[1]

Additional research starting in 1993 developed a Graphical User Interface (GUI) for the IMCMA 1.0 prototype [12] and submodeling and reliability assessment was added [13,14].[2]

The current effort (1994–1995) focuses on eliminating the problems related to the integration of the Sandia mesh-generation codes and on extending and enhancing the GUI developed in the 1993 effort. Before detailing this effort, we provide a brief overview of the IMCMA architecture.

2 An Overview of the IMCMA Prototype

A blackboard system, based on the blackboard problem-solving paradigm [15,16], was selected as the basis for the IMCMA architecture. A blackboard system performs problem solving by using three basic components (Figure 1):

- A *blackboard*, that is a global database containing input data, partial solutions, and other data that are in various problem-solving states.
- *Knowledge sources* (KSs) which are independent modules that contain the knowledge needed to solve the problem, and that can be widely diverse in representation and in

[1]The 1992–1993 prototype system will be referred to as IMCMA 1.0 in this report.

[2]The augmented 1993–1994 prototype system will be referred to as IMCMA 1.5 in this report.

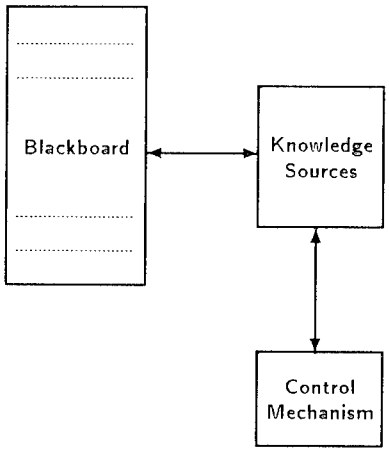


Figure 1 Basic Components of the Blackboard Model

inference techniques. KS modularity facilitates application development and simplifies maintenance and enhancement.

- A *control mechanism*, that is separate from the individual KSs and that makes dynamic decisions about which KS is to be executed next.

The power of the blackboard approach lies in its ability to:

- Organize problem-solving knowledge into multiple independent knowledge modules
- Allow the knowledge in each module to be represented differently
- Allow different problem-solving techniques in each knowledge module
- Flexibly apply knowledge modules in the most appropriate way to efficiently solve the problem

The blackboard model offers a powerful problem-solving architecture that is suitable when:

- Many diverse, specialized knowledge representations are needed. KSs can be developed in the most appropriate representation for the data they are to handle.

- An integration framework is needed that allows for heterogeneous problem-solving representations and expertise.
- The application uses large-grained modularity for design, implementation, and maintenance.
- The application involves many developers.
- Uncertain knowledge or limited data inhibits absolute determination of a solution.
- Multilevel reasoning or flexible, dynamic control of problem-solving activities is required in an application.

The IMCMA prototype is implemented using Blackboard Technology Group, Inc.'s GBB™ framework, a toolkit for rapidly developing and delivering high-performance blackboard applications. GBB provides the infrastructure for a blackboard-based application such as IMCMA, as well as a graphical user interface toolkit and graphical monitoring and inspection tools.

IMCMA 1.0 Knowledge Sources

The original IMCMA prototype (Version 1.0), developed in the 1992–1993 effort, contained nine KSs (Table 1).

Processing proceeded among these KSs as shown in Figure 2.

IMCMA 2.0 Knowledge Sources

The IMCMA 2.0 system developed in this effort contains 7 KSs (Table 2).[3]

GBB is a trademark of Blackboard Technology Group, Inc.

[3]The `create-model` KS was created for the IMCMA 1.5 prototype to separate the blackboard objects representing the device (components and materials) from those used in *modeling* the device (model components and model materials) [12]. The `create-model` KS was not developed as part of this effort.

`input-model`	GBB	Reads a device specification file and builds the component and material objects specified therein
`adjust-model`	GBB	Simplifies the geometry of the device to facilitate rapid analysis
`find-symmetry`	GBB/CLIPS	Identifies 2D (XY) geometric symmetries in the device
`complete-model`	GBB	Builds the component objects that are based on the adjusted model
`generate-mapmesh-regions`	GBB	Generates the coarsest possible 2D mesh for the adjusted device
`generate-2d-mesh`	GBB/FORTRAN	Generates a 2D mesh from the mapmesh regions and mesh density specifications
`extrude-component`	GBB/FORTRAN	Edits the 2D mesh for a specific component and extrudes that component into a 3D mesh
`combine-3d-meshes`	GBB/FORTRAN	Combines all the component 3D meshes into a single 3D mesh
`analyze-3d-mesh`	GBB/FORTRAN	Analyzes the combined 3D mesh

Table 1 IMCMA 1.0 Knowledge Sources

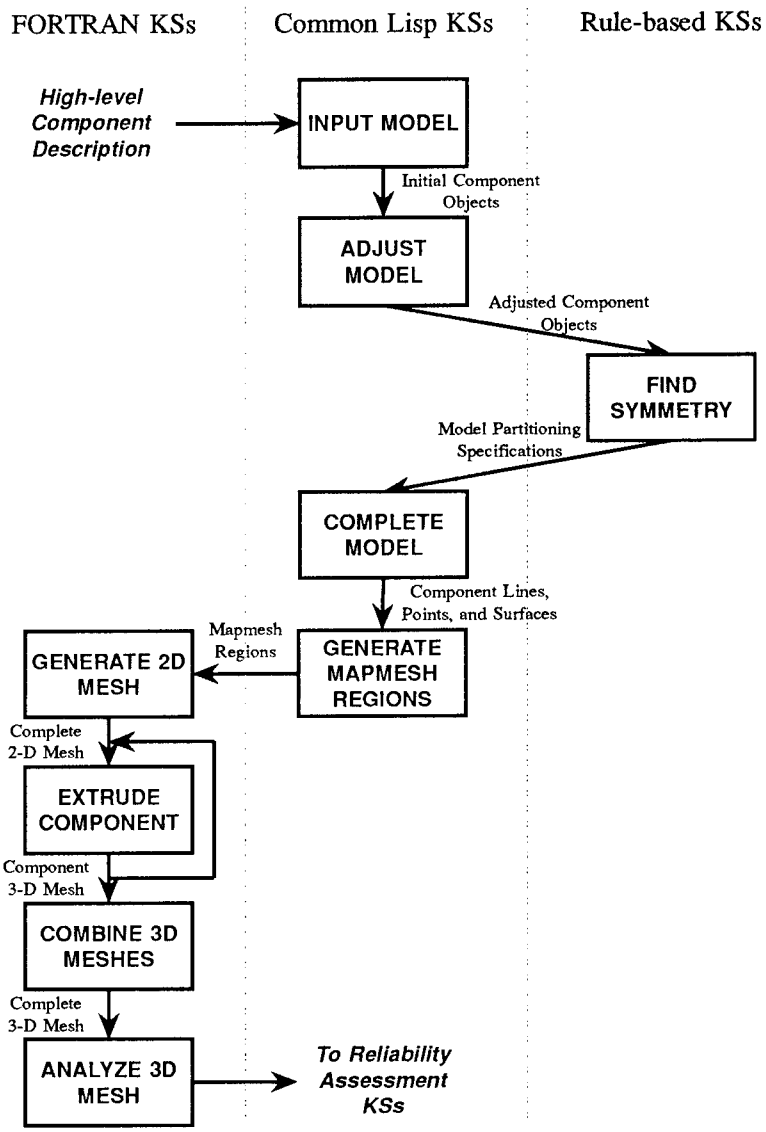


Figure 2 Original IMCMA KS Processing

input-model	GBB	Reads a device specification file and builds the component and material objects specified therein
create-model	GBB	Builds model component and material objects based on the component and material objects representing the device
adjust-model	GBB	Simplifies the geometry of the device to facilitate rapid analysis
complete-model	GBB	Builds the component objects that are based on the adjusted model
generate-3d-component-mesh	GBB	Generates the 3D mesh for a component, merging the mesh with that of components previously meshed
complete-3d-mesh	GBB	Adds insulating chip sides where appropriate and links prescribed points and surfaces to the completed 3D mesh
analyze-3d-mesh	GBB/FORTRAN	Analyzes the combined 3D mesh

Table 2 IMCMA 2.0 Knowledge Sources

In IMCMA 2.0, the four Version 1.0 KSs: `generate-mapmesh-regions`, `generate-2d-mesh`, `extrude-component`, and `combine-3d-meshes`; have been replaced with a two new KSs: `generate-3d-component-mesh` and `complete-3d-mesh`. The `find-symmetry` KS was eliminated due to the lack of substantial symmetry in today's typical MCM devices. In addition to these major KS changes, the remaining KSs, as well as the blackboard structure and object definitions, were enhanced as part of this effort.

Processing proceeds among these KSs as shown in Figure 3.

3 3D Mesh Generation in IMCMA 2.0

As with IMCMA 1.0, a 3D mesh consists of a set of 3D nodes (vertices) and a set of 3D elements (rectangular prisms). The goal

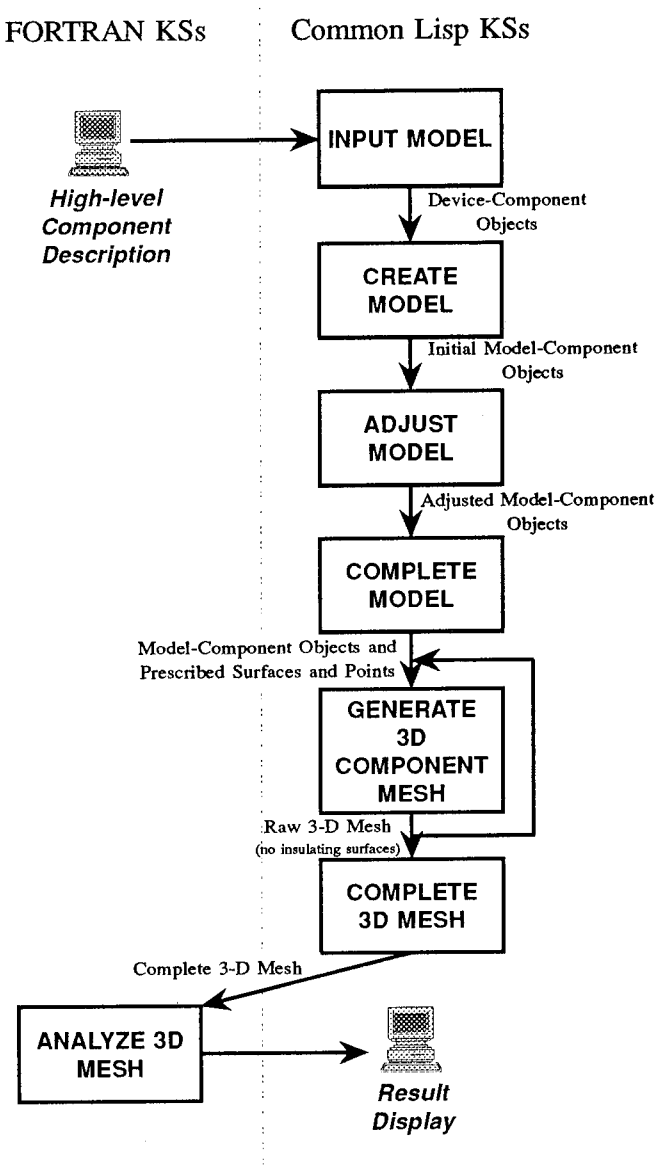


Figure 3 IMCMA 2.0 KS Processing

of the 3D-mesh-generation process is to produce a mesh with a relatively small number of elements that is well suited to analysis.

Basic Mesh-Generation Strategy

Integrated 3D mesh generation in IMCMA 2.0 is performed in two steps. In the first step, a raw mesh (one with no insulating surfaces) is generated based on the extents of the modelled components as well as on the specified number of default xy elements and the number of default z elements. The second step adds insulting surfaces to the mesh by splitting appropriate nodes into two nodes, with the same coordinates, but attached to different elements.

The `GENERATE-3D-COMPONENT-MESH` KS

The `generate-3d-component-mesh` KS is called once for each modelled component to generate a 3D mesh for the component. An important part of mesh generation is locating and using 3D nodes from the already created 3D meshes of other components that share a common boundary with the component.

Prior to the generation of component 3D meshes, the `complete-model-ks` KS has computed the x and y values where element boundaries must extend vertically through the modelled device. These values are stored in the variables `*x-values*` and `*y-values*`, respectively.

For the model component, M, the `generate-3d-component-mesh` KS does the following:

- Computes an ordered list of z coordinates, $vals_z$ for M on the M's z extent of the component and the desired number of xy planes.
- Computes an ordered list of x coordinates, $vals_x$, the subset of `*x-values*`, that are within the x extent of M.
- Computes an ordered list of y coordinates, $vals_y$, the subset of `*y-values*`, that are within the y extent of M.

- For each consecutive pair of z coordinates, (z_1, z_2), of M (from $vals_z$):
    - Generate two xy planes of 3D-nodes. Each node in the first plane has z value z_1, and each node in the second plane has z value z_2. The pairs of (x, y) values to be used in each plane is simply the cartesian product of $vals_x$ and $vals_y$.
    - For each x_i in $vals_x$ (except the last one):
        * For each y_i in $vals_y$ (except the last one), unless we are in a well[4] of M, generate a 3D-element whose minimum node has coordinates (x_i, y_i, z_1) and whose maximum node has coordinates (x_{i+1}, y_{i+1}, z_2).

The COMPLETE-3D-MESH KS

The `complete-3d-mesh` KS performs the following activities on the newly generated 3D mesh:

- Adds insulating surfaces (nodes) to the 3D mesh where chip sides are adjacent to other non-chip components.
- Determines the positions of 3D nodes and elements (whether they are on the top, bottom, left, right, front, or back surfaces of their components). These positions are used by the graphical display routines, but they are determined once by this KS and cached with the 3D node and element objects.
- Links prescribed surfaces and point sources to elements using GBB's advanced object retrieval capabilities.

The last two activities are enhanced versions of the IMCMA 1.0 prototype, the addition of insulating surfaces is new to 2.0.

[4] A *well* is an empty volume removed or molded into a component to make room for another component. For example, a substrate component may have a number of wells cut into it to allow chip components to be placed with their top surfaces aligned with the top surface of the substrate. IMCMA automatically models wells as needed when component volumes overlap.

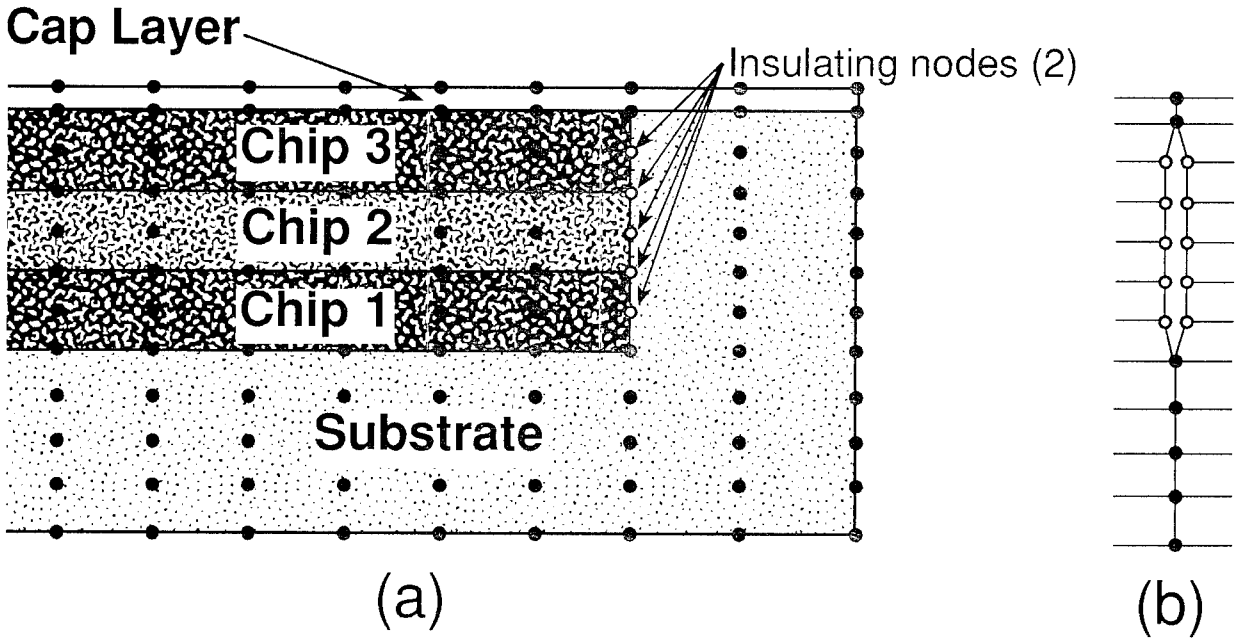


Figure 4 A Device Modeled with Insulating Chip Surfaces
(modelled by disjoint elements and nodes)

Insulating Chip Surfaces

A device showing some of the complexity of modeling insulating chip surfaces is shown in Figure 4. In this device, three chip components are stacked vertically in a well cut into the substrate component. The chip stack and substrate are covered by a "cap layer" component. To best model insulating chip sides, only the bottom surface nodes of Chip 1 should be shared with the Substrate and only the top surface nodes of Chip3 should be shared with the Substrate and Cap Layer. Note further that if the Cap Layer was removed, the top surface nodes of Chip3 should no longer be shared with the Substrate, as there is no longer a top surface connection with the Substrate via the Cap Layer.

The **complete-3d-mesh-ks** contains knowledge about how to best represent insulating chip sides under various geometric situations. The overall approach to adding the insulating layers is as follows:

1. For each node N, of each relevant side S, of each model chip

M, if N is also attached to some other component, M', and M' shares some surface area with M, then:

(a) split N, by creating a new node N', with the same coordinates as N
(b) replace node N in component M with N', so that N is no longer attached to M.

More specifically, for each model chip, M, do the following:

1. Let L be the set of lateral nodes of M, that are neither bottom nodes nor top nodes.
2. For each node N in L, if N is also attached to at least one other component, M', and M' laterally overlaps M (M' shares some lateral surface area with M), then:
   (a) we split N, by creating another node N', with the same coordinates
   (b) then replace node N in component M with N', so that N is no longer attached to M.
3. For each peripheral node of the top side of M, do the same as in step 2, with the exception that, if there is another component, T, above M, that is such that, N is a node of T, and T overlaps M', then do not split N. In the case that T does not overlap M', we split N (as in step 2), but node is replaced with the new node N' in both M and T.

IMCMA 1.0 and 2.0 Comparison

The integrated mesh-generation algorithms used in IMCMA 2.0 were cross checked with those generated by the IMCMA 1.0 system. When given the same device specification and constraints, the IMCMA 2.0 algorithms produced a mesh that was identical to that produced by the IMCMA 1.0 prototype. The time required to generate the mesh in IMCMA 2.0 was reduced by about an order of magnitude over IMCMA 1.0 and no intermediate 2D objects were required.

Table 3 shows the names and counts of the blackboard objects created by IMCMA 1.0 in performing a high-level analysis of a ficticious MCM device. Although based on real technology, this

```
MODEL
   2D-MODEL
      COMPONENT-2D-SURFACES  11 Units: (11 COMPONENT-2D-SURFACE)
      COMPONENT-2D-LINES     44 Units: (44 COMPONENT-2D-LINE)
      COMPONENT-2D-POINTS    43 Units: (43 COMPONENT-2D-POINT)
      MAPMESH-2D-REGIONS    272 Units: (272 MAPMESH-2D-REGION)
      MAPMESH-2D-LINES      577 Units: (577 MAPMESH-2D-LINE)
      MAPMESH-2D-POINTS     306 Units: (306 MAPMESH-2D-POINT)
      2D-ELEMENTS          1088 Units: (1088 2D-ELEMENT)
      2D-NODES             1155 Units: (1155 2D-NODE)
   3D-MODEL
      COMPONENTS             22 Units: (10 PRESCRIBED-FLUX-SURFACE,
                                         1 PRESCRIBED-TEMPERATURE-SURFACE,
                                        11 COMPONENT)
      COMPONENT-3D-LINES    132 Units: (132 COMPONENT-3D-LINE)
      COMPONENT-3D-POINTS    88 Units: (88 COMPONENT-3D-POINT)
      COMPONENT-3D-SURFACES  66 Units: (66 COMPONENT-3D-SURFACE)
      3D-ELEMENTS          1352 Units: (1352 3D-ELEMENT)
      3D-NODES             2704 Units: (2704 3D-NODE)
   MATERIALS                  2 Units: (2 MATERIAL)
CONTROL-SHELL
   KSS                        9 Units: (9 KS)
   KSAS                      19 Units: (19 KSA)
```

Table 3 IMCMA 1.0 Blackboard (analyzing test device)

```
MODEL
   3D-MODEL
      COMPONENTS             22 Units: (11 MODEL-COMPONENT,
                                        10 MODEL-PRESCRIBED-FLUX-SURFACE,
                                         1 MODEL-PRESCRIBED-TEMPERATURE-SURFACE)
      3D-ELEMENTS          2014 Units: (2014 3D-ELEMENT)
      3D-NODES             3746 Units: (3746 3D-NODE)
   MATERIALS                  2 Units: (2 MODEL-MATERIAL)
DEFINED
   COMPONENTS                12 Units: (1 DEVICE, 1 SUBSTRATE, 10 CHIP)
   MATERIALS                  2 Units: (2 MATERIAL)
CONTROL-SHELL
   KSS                        7 Units: (7 KS)
   KSAS                       Empty
```

Table 4 IMCMA 2.0 Blackboard (analyzing test device)

"Test MCM" device does not physically exist and was created solely for purposes of demonstration, debugging, and illustration. The device-description file for this device is shown in Appendix B.

Table 4 shows the names and counts of the blackboard objects

created by IMCMA 2.0 in performing this same high-level analysis. The elimination of the Version 1.0 `2D-MODEL` blackboard objects has been balanced by the use of more elements in the Version 2.0 3D mesh.

4 The 3D Viewer

As part of this effort, a 3D graphics display mode was added to the existing IMCMA graphics system. Although the linked top-, front-, right-side views provided by the IMCMA Version 1.5 prototype GUI (Figure 5)are well suited to detailed viewing of the analysis results, there are times when an overall 3D view of the results is useful. As part of this effort, an integrated 3D-viewer mode was added to the 2D display. This 3D Viewer provides a "flying eye" view of the analysis results of the entire device or any selected subset of the device components. Figure 6 shows the 3D Viewer displaying the results of analyzing the Test MCM device. Figure 7 shows the 3D Viewer displaying only the chip components.

For top performance, the 3D Viewer uses a specialized display algorithm that avoids the expense of hidden-surface elimination by determining which surfaces of which 3D elements are visible without considering occluding elements. These surfaces are drawn, from back to front (in terms of the view frame of reference) so that any occluded surfaces will be overdrawn as needed (Figure 8). Since the element surfaces are relatively small and of uniform size, this approach results in a fast and accurate redisplay.[5]

[5]However, this approach will not work for a component-level 3D display. Component surfaces do not share the relatively small, uniform size properties of element surfaces. For a proper 3D display of component surfaces, a standard hidden-surface-elimination algorithm must be used.

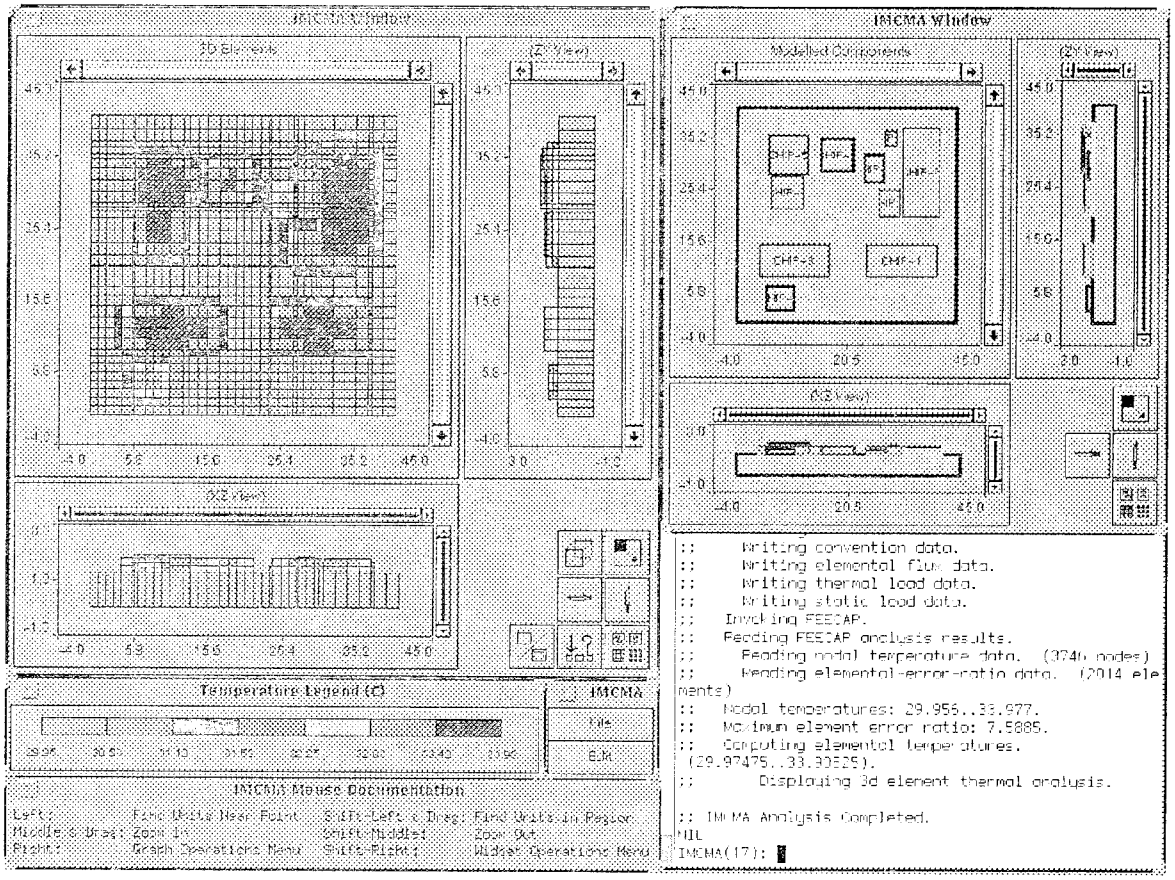


Figure 5 The IMCMA 2.0 System (showing Test MCM device)

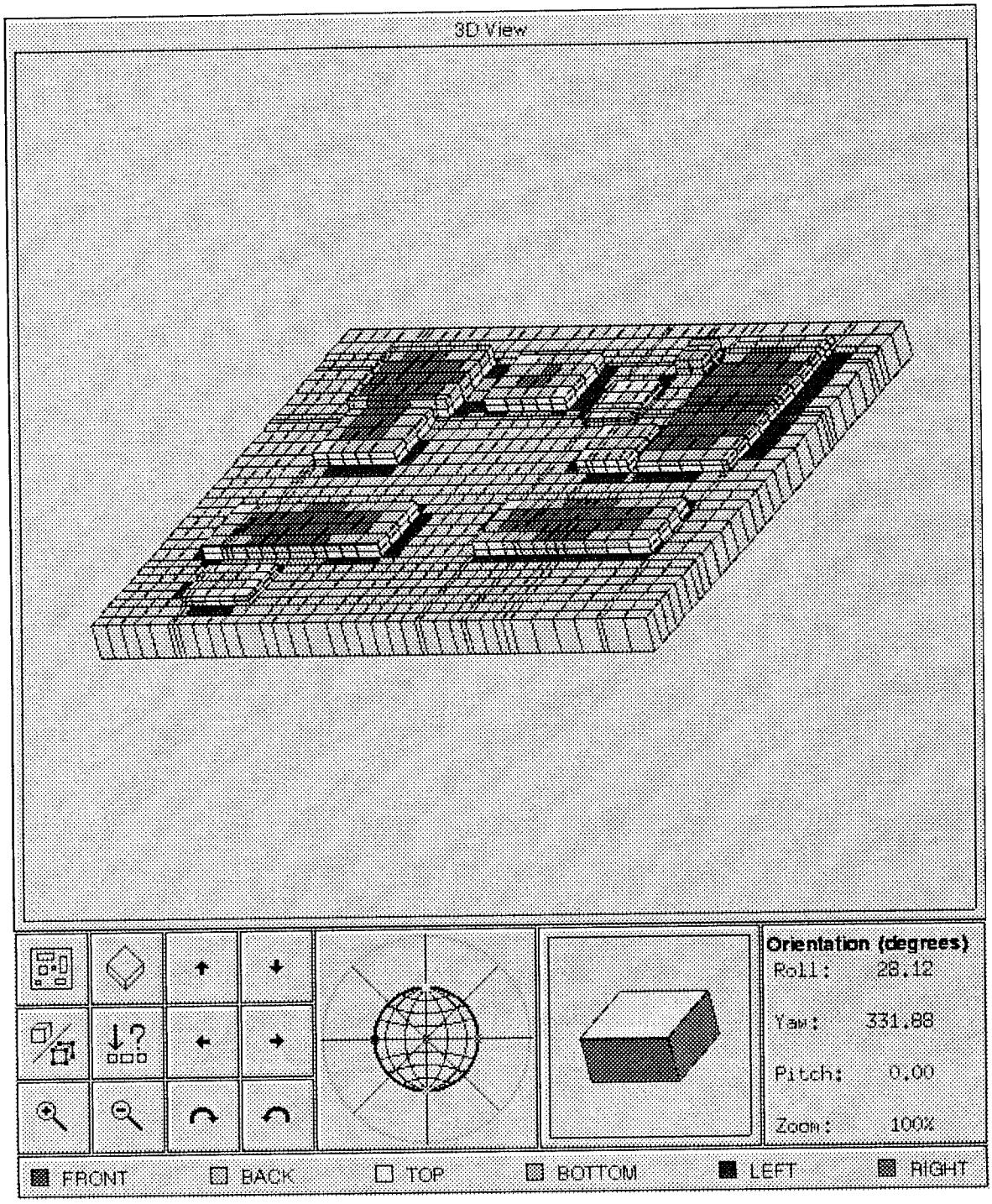


Figure 6 The 3D Viewer

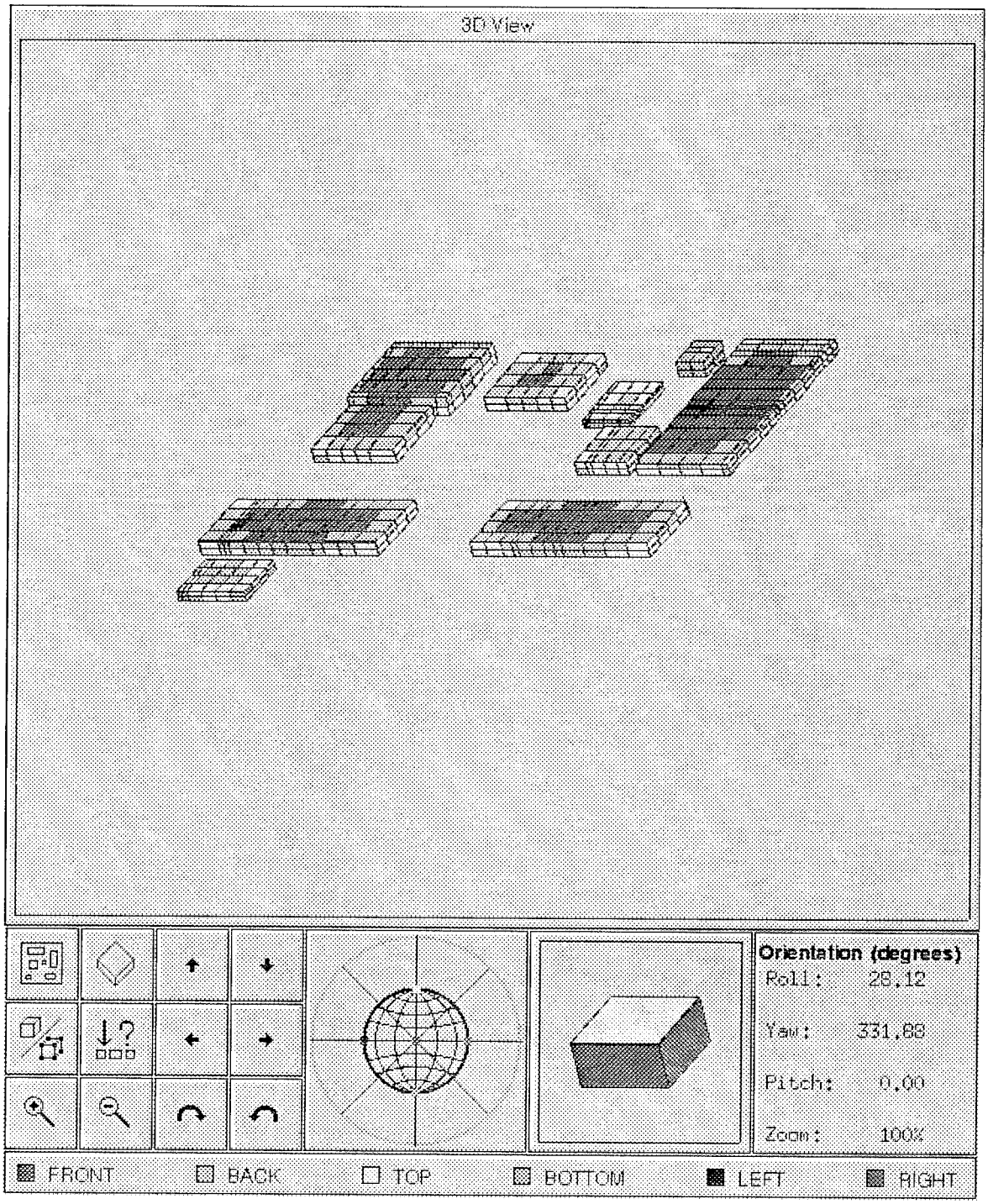


Figure 7 The 3D Viewer Showing Only Chip Components

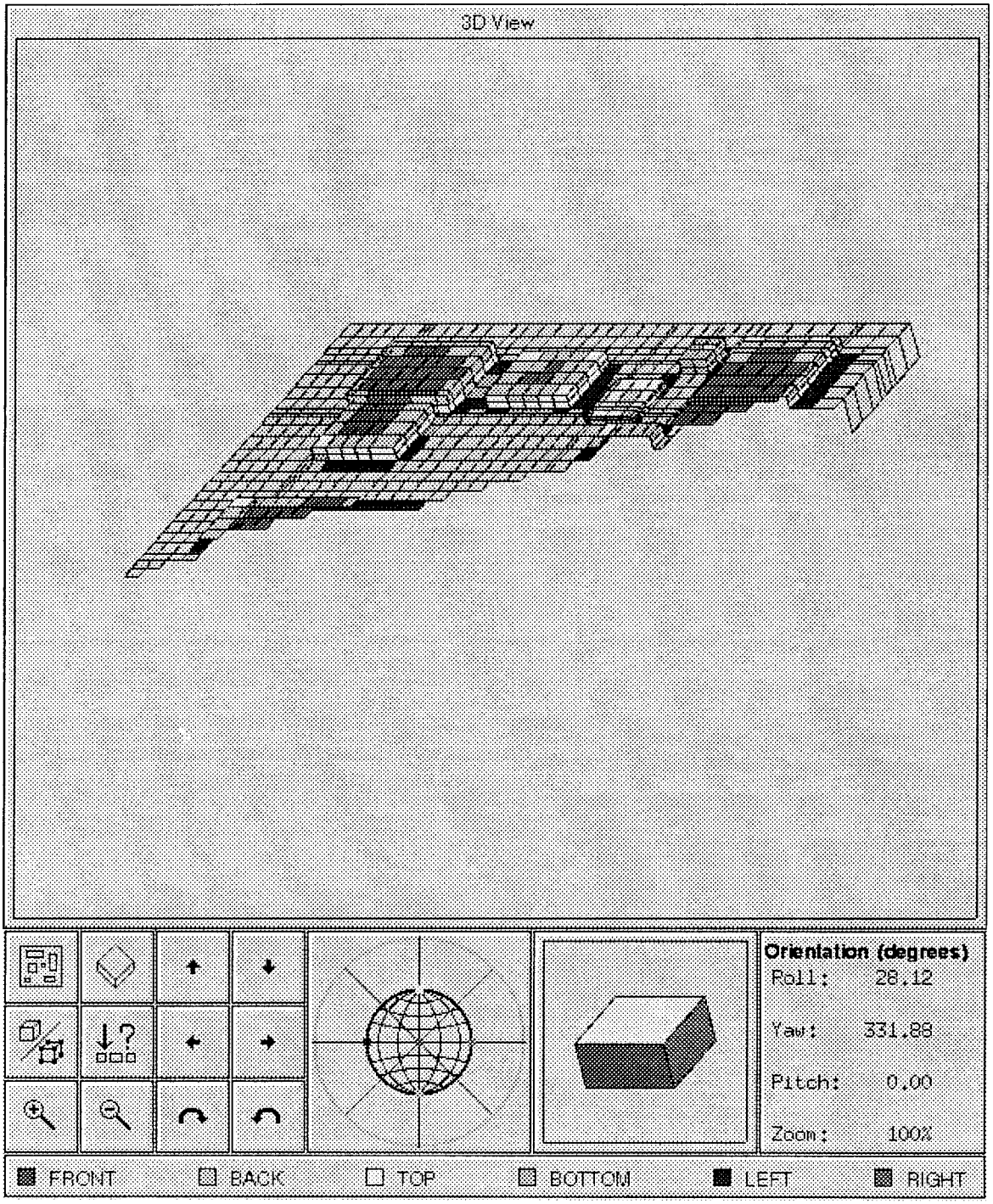


Figure 8 The 3D Viewer Display Algorithm (in progress)

5 Lessons Learned

Although significant progress had been made in the IMCMA 1.0 and 1.5 prototypes, some subtle issues arose as the IMCMA 2.0 prototype progressed toward field trials. Some of these issues included:

Attaches

The most natural way of specifying and representing chip and substrate attaches went through several iterations during this effort. During course "what-if" modeling, attaches may be left out from the device specification and analysis. Later, when attaches are added to the existing device geometry, the device components positions and sizes must be respecified to provide space for the attaches. In IMCMA 2.0, we wanted to allow attach geometries to be easily added, deleted, or modified by the user.

A seemingly obvious approach of adding the attaches by shifting components upward becomes problematic when overlapping layers of components are present in the device. (Adding a chip attach to a single "hot" chip covered by a component layer overlaying a number of chips would require identical attach geometries to be added to all the chips.)

Because the attach height is very small relative to the heights of other components, we choose to model an attache by "borrowing" its vertical height from its parent chip or substrate. This approach allows individual attaches to be easily added, removed, or changed. If detailed modeling is required, the actual component and attach heights are simply added and specified as the "height" of the component.

The attach-borrowing representation is used consistently through the IMCMA GUI dialogs. However, the blackboard-object representation and the device-description file always use actual component extents. Converting to and from the attach-borrowing representation is performed automatically by object methods written on the defined substrate and chip blackboard-unit classes.

Numerical accuracy

In IMCMA 1.0, we had to deal with the sensitivity of the various FORTRAN-based mesh-generation tools to numeric precision. GBB represents floating point values using double precision, while the FORTRAN-based tools used a mixture of single and double-precision. An example of the problems stemming from this difference was determining which 3D-element objects (produced by FORTRAN codes) corresponded to which 3D component objects (created within GBB). After meshing, some 3D-elements extended slightly outside the component, while other neighboring 3D-elements incorrectly extended inside the component. In IMCMA 1.0, we adopted a strategy of adding a numeric tolerance to GBB's retrieval operations to cope with these differences.

With the integrated meshing code in IMCMA 2.0, these differences among meshing tools were eliminated and along with it, the need for adding a tolerance to GBB's retrieval operations. However, numerical accuracy issues still arose, particularly with the approach of "borrowing" attach height described above.

To illustrate the problem, consider an imaginary component with a height of 0.05 units. If we borrow 0.02 units for an attach, the remaining space for the component is 0.03 units. However, due to limited numerical precision the computation $0.05 - 0.02$ yields 0.30000001. Further geometric-reasoning computations using the 0.30000001 value can result in components that are not quite attached or extremely thin "virtual" components that should not be present.

We could always resort to the use of a numeric tolerance in retrieval operations used in IMCMA 1.0, but that would reintroduce some of the limitations that we sought out to avoid.

In IMCMA 2.0, we took advantage of the fact that addition of (non-negative) floating point values did not introduce inaccuracies to avoid subtraction wherever possible. For example, rather than determining the *height* as the $z_{max} - z_{min}$, z_{min} and the *height* would be retained as values with z_{max} calculated as $z_{min} + height$. Where this was not possible (such as when "borrowing" attach heights), the results of subtractions were immediately rounded to the appropriate number of decimal places, based on the numeric tolerance value supplied by the user.

As a result of this approach, none of the downstream geometric calculations in IMCMA 2.0 require any use of numeric tolerance.

Power dissipation surfaces

Power dissipation surfaces are idealized heat-producing surfaces used to model the thermally active portions of active MCM components. Power dissipation surfaces differ from more traditional thermal flux surfaces, in that they do not have a predefined direction for thermal flow associated with them. In other words, heat flow is not always outward from a power dissipation surface if there are hotter regions nearby.

Prior to IMCMA 2.0, power dissipation surfaces could be associated with entire chip faces. Because the underlying mesh analysis program (FEECAP) does not handle power dissipation surfaces, these were represented by converting each power dissipation surface to a set of point heat sources with values summing to that of the power dissipation surface.

In IMCMA 2.0, power dissipation surfaces could be defined for any portion of a chip face (including overlapping power dissipation surfaces). The approach used was again to convert each power dissipation surface to a set of point heat sources. In this case, the value assigned to each point heat source was apportioned based on the surface area of the power dissipation surface in the neighborhood of the point heat source. This approach is a modeling approximation, but one that is sufficiently accurate for analysis purposes in IMCMA 2.0.

IMCMA as a Blackboard Application

How does the IMCMA 2.0 prototype relate to other blackboard-based applications?

Processing in IMCMA 2.0 is linear, without any event-based, opportunistic "focus of attention" decisions that are often characteristic of blackboard applications. Since IMCMA 2.0 is performing a single activity: generating and analyzing a finite-element mesh from a high-level device description; the

problem-solving steps required are unchanging.[6] Does this lack of advanced control mean that the choice of a blackboard architecture for IMCMA was not a good one? Definitely not.

The integration advantages of the blackboard approach that were exploited in the development of IMCMA 1.0 and 1.5 were equally beneficial in adding the integrated meshing KSs in IMCMA 2.0. The blackboard object representations and non-direct invocation of other KSs allowed the Sandia-based KSs to be switched in and out with the developing integrated KSs, facilitating both development and later comparison testing with the Sandia KSs.[7]

GBB's multidimensional blackboard representation and retrieval facilities were used heavily in the integrated meshing KS algorithms. As each component is meshed, the resulting nodes must be merged with nodes already created for other components. GBB's blackboard-retrieval machinery provided an efficient mechanism for performing this merging operation. For example, 4266 individual 3D-node retrieval operations are performed when meshing the MCM Test device. Without GBB, indexing and retrieval facilities for performing this merging would have had to be developed.

One of the important results of using the blackboard approach for IMCMA, is the ease with which it can be modified and extended. Just as the Sandia tools were replaced with integrated meshing KSs in this effort, other aspects of IMCMA (such as the FEECAP-based analysis KS) could also be replaced by different tools. So, for example, a commercial finite-element solver could be substituted for the FEECAP-based KS by merely providing the input/output interface as part of a new "analyze-mesh" KS.[8]

The internal design of IMCMA can also support extension to new finite-element techniques. Although the mesh-generation code requires rectilinear elements, additional nodes can be added to elements to assist in analysis.

---

[6] An expanded IMCMA system which considers a number of design-related characteristics would exhibit more opportunistic control as different characteristics were explored based on the specifics of the design.

[7] For a time, both sets of KSs were present and could be switched from one to the other by simply enabling and disabling the individual KSs.

[8] If desired, the two solvers could both be run, each updating different slots in the mesh objects on the blackboard. The results of each solver could then be compared.

In addition to maintenance and enhancement of existing capabilities, the blackboard approach used in IMCMA also facilitates to addition of new capabilities. Possibilities include electro-magnetic analysis, manufacturability analysis, etc. The IMCMA 2.0 prototype is an extensible base application ready for this type of enhancement.

6 Next Steps

Although the IMCMA 2.0 prototype system is substantially improved over the earlier prototypes and is now ready for field trials, several useful enhancements would further improve the 2.0 prototype.

Node Renumbering

The speed of the 3D-mesh analysis performed by FEECAP (and other finite-element solvers) is effected by the *bandwidth* of the 3D-mesh. In a low-bandwidth mesh, geometrically adjacent nodes have node numbers that are near one another. The mesh-generation approach used in IMCMA 2.0 produces component meshes with low bandwidth, but the boundaries between adjacent components often have large spreads among node numbers.

Node renumbering is applied as a preprocessing step in some finite-element solvers. Node renumbering is not performed by the FEECAP codes. Because the mesh objects are linked in the IMCMA blackboard representation, adding a node renumbering phase is straightforward, requiring:

- implementation of a mesh-traversal algorithm to assign "new" node numbers (represented in an additional slot in the node objects)
- a lookup scheme (using a temporary vector or the blackboard retrieval facilities) for locating the appropriate 3D-node object when storing the analysis results for each node

If mesh-analysis times start to become an issued (which has not been the case with the devices modelled to date), node renumbering would be an low-labor enhancement to reduce the computational cost of mesh analysis.

Libraries

Support for libraries of materials and components are not provided in the IMCMA 2.0 prototype. A library facility was part of the original IMCMA design and adding a facility to the prototype would require straightforward extension of the representations and capabilities already in place. In IMCMA 2.0, the blackboard objects representing the device (as defined) are kept separate from the blackboard objects used to model the components. Similarly, the blackboard objects representing material properties are separate from those used in modeling the materials. Functions for creating modeling objects from device-definition objects (with modifications, where appropriate) are part of the IMCMA 2.0 prototype.

A library facility for components and materials would require adding library material and component objects that would serve as sources for the device definition materials and components. Library materials and components would be stored on a separate "library" blackboard and copied onto the device-definition blackboard, perhaps with modifications, as appropriate. Newly defined materials and components could also be copied out of the device-definition blackboard and placed into the library. The IMCMA GUI would be modified to provide these operations, as well as loading and storing of public and private material and component libraries.

Since the utility of a library rests in the ease with which items can be found, locating appropriate library materials and components (by properties other than name) is likely to be important. The retrieval facilities provided by GBB are a good starting point for allowing the user to quickly locate candidate library objects for use in an MCM design.

Additional Meshing Intelligence

The IMCMA 2.0 prototype uses a relatively simple approach to developing the mesh, where an initial phase determines the elemental boundaries that must extend through the entire device (the XY mesh-element boundaries and minimal Z boundaries) and a second phase adds additional Z mesh-element boundaries which are localized to individual components. This approach is based on the 2.5D nature of the MCM devices we are modeling. This simple approach works well, even with devices where components are stacked vertically one on top of the other.

If MCM devices start to be developed where components are aligned vertically as well as horizontally, a more intelligent meshing approach will be needed. Similarly, if insulating surfaces are possible on arbitrary sides of components (rather than the current left, right, front, and back sides), additional analysis will be needed during the initial and secondary mesh-determination phases.

The design of IMCMA 2.0 is well suited to adding this additional meshing intelligence. Once created, planes of meshes can be shifted as a unit; allowing for revision of the generated mesh. However, the most efficient technique would be to attempt to determine prior to meshing exactly where the complete and partial planes of mesh-element sides will be loaded in the device.

7 Summary of Accomplishments

To summarize, the following research activities were performed in this contract:

- Integrated, GBB-based, mesh-generation KSs were developed to replace the existing FORTRAN-based 2D and 3D mesh-generation KSs.
- The integrated mesh-generation KSs were validated against the existing Sandia-code KSs to insure their accuracy.
- A 3D graphics display facility was designed and implemented.

- The existing graphical user interface was extended and enhanced to improve the ease of using the IMCMA prototype system.

As a result of this effort, the IMCMA 2.0 prototype can now perform a course analysis of an MCM device in several minutes, and the prototype is now available for field testing.

8 References

[1] Ian R. Grosse, Prasanna Katragadda, and Anagha Jog. Knowledge sources for an intelligent mcm analyzer. Final Report RL-TR-93-123, Rome Laboratory, Griffiss Air Force Base, New York, June 1983.

[2] Daniel D. Corkill. An architecture for intelligent multichip module reliability analysis. Final Report RL-TR-94-71, Rome Laboratory, Griffiss Air Force Base, New York, April 1984.

* [3] W. J. Bocchi, J. A. Collins, and D. J. Holzhauer. Thermal stress analysis of integrated circuits using finite element methods. Technical Report RADC-TR-84-100, Rome Air Development Center, Rome, NY, April 1984.

[4] W. J. Bocchi. Finite element modeling and thermal simulations of transistor integrated circuits. Technical Report RADC-TR-89-176, Rome Air Development Center, Rome, NY, October 1989.

[5] J. H. Lau, D. W. Rice, and P. A. Avery. Elastoplastic analysis of surface mount solder joints. *IEEE Transactions on Components, Hybrids and Manufacturing Technology*, CHMT-10(3):346–357, September 1986.

[6] P. A. Engel and C. K. Lim. Stress analysis in electronic packaging. *Finite Element Analysis and Design*, 4(1):9–18, June 1988.

[7] J. H. Lau and C.G. Harkins. Thermal stress analysis of SOIC packages and interconnections. *IEEE Transactions on Components, Hybrids and Manufacturing Technology*, 11(4):380–389, 1988.

*NOTE: Although this report references * limited document above, no limited information has been extracted. Distribution limited to USGO Agencies and their contractors.

[8] J. C. Glaser and M. P. Juaire. Thermal and structural analysis of a PLCC device for surface mount processes. *Journal of Electronic Packaging*, 3(3):172–178, September 1989.

[9] S. Kawai. Structural design of plastic ic packages. *JSME International Journal, Series 1—Solid Mechanics, Strength of Materials*, 32(3):320–330, July 1989.

[10] J. H. Lau. Thermal stress analysis of SMT PQFP packages and interconnections. *Journal of Electronic Packaging*, 3(1):2–8, March 1989.

[11] George Turklyyah and Stevern J. Fenves. Feasibility study of a knowledge-based finite-element modeling assistant. Final report, Department of Civil Engineering, Carnegie Mellon University, Pittsburgh, PA 15213, February 1988.

[12] Daniel D. Corkill and Venkat S. Manakkal. An enhanced architecture for intelligent multichip module reliability analysis. Final report, Rome Laboratory, Griffiss Air Force Base, New York, 1985.

[13] Ian R. Grosse, Michael Sheehy, Prasanna Katragadda, and Shankar Raman. Intelligent finite element submodeling of multichip modules for reliability analysis. Final Report RL-TR-94-218, Rome Laboratory, Griffiss Air Force Base, New York, December 1984.

[14] Ian R. Grosse, Prasanna Katragadda, Sandeepan Bhattacharya, and Sarang Kulkarni. Intelligent computer based reliability assessment of multichip modules. Final Report RL-TR-94-217, Rome Laboratory, Griffiss Air Force Base, New York, December 1984.

[15] Lee D. Erman, Frederick Hayes-Roth, Victor R. Lesser, and D. Raj Reddy. The Hearsay-II speech-understanding system: Integrating knowledge to resolve uncertainty. *Computing Surveys*, 12(2):213–253, June 1980.

[16] Daniel D. Corkill. Blackboard systems. *AI Expert*, 6(9):40–47, September 1991.

Appendices

A IMCMA Test MCM Device-Description File

The device-description file for the Test MCM example (with the chips located on top of the substrate) is shown below. This is the same TEST MCM device used in the IMCMA 1.0 report [2].

```
;;;; -*- Mode:COMMON-LISP; Package:IMCMA; Base:10 -*-
;;;; *-* File: DIAMOND: /usr/users/cork/newimcma/examples/get.lisp *-*
;;;; *-* Edited-By: Cork *-*
;;;; *-* Last-Edit: Wednesday, August 31, 1994  17:14:10 *-*
;;;; *-* Machine: GRANITE (Explorer II, Microcode 489) *-*

;;;; ********************************************************************
;;;; ********************************************************************
;;;; *
;;;; *                              TEST MCM DEVICE
;;;; *
;;;; ********************************************************************
;;;; ********************************************************************
;;;
;;; Written by:  Dan Corkill
;;;              Blackboard Technology Group, Inc.
;;; Modified by: Prasanna Katragadda
;;;              ME, UMASS,  04/01/93
;;; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;;;
;;;  11-18-92 File created.
;;;
;;; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *

(in-package "IMCMA")

;; This must come first:

(define-device "TEST MCM"
  :filename "test-mcm"
  :size (40.64 40.64 2.955))
```

```
;; Materials come next:

(defmaterial :silicon
  :min-error-tolerance .1
  :max-error-tolerance .1
  :tk (0.1256 0.1256 0.1256)
  :alpha (0.233e-05 0.233e-05 0.233e-05)
  :beta 0
  :reference-temperature 0)

(defmaterial :Al2O3
  :min-error-tolerance .1
  :max-error-tolerance .1
  :tk (0.025 0.025 0.025)
  :alpha (0.8e-05 0.8e-05 0.8e-05)
  :beta 0
  :reference-temperature 0)

;; Now the device:

;; The aluminum-oxide carrier:

(defcomponent :SUBSTRATE-1 :substrate
  :size (40.64 40.64 1.27)
  :prescribed-temperature-surfaces ((:bottom 30))
  :material :Al2O3)

;; The Chips:

(defcomponent :CHIP-1 :chip
  :size (13.0010 5.9890 .516)
  :x (+ (/ 40.64 2.0) 10.1451)
  :y (+ (/ 40.62 2.0) -8.4118)
  :z (+ 1.27)
  :xy-alignment :centered
  :prescribed-flux-surfaces ((:top 0.08))
  :material :silicon)

(defcomponent :CHIP-2 :chip
  :size (6.0000 6.0000 .516)
  :x (+ (/ 40.64 2.0) -1.8409)
  :y (+ (/ 40.62 2.0) 11.6080)
  :z (+ 1.27)
  :xy-alignment :centered
  :prescribed-flux-surfaces ((:top 0.08))
  :material :silicon)

(defcomponent :CHIP-3 :chip
  :size (12.9887 5.9890 .516)
  :x (+ (/ 40.64 2.0) -9.6919)
  :y (+ (/ 40.62 2.0) -8.4363)
  :z (+ 1.27)
  :xy-alignment :centered
  :prescribed-flux-surfaces ((:top 0.08))
  :material :silicon)
```

```
(defcomponent :CHIP-4 :chip
  :size (6.0000 6.0000 .516)
  :x (+ (/ 40.64 2.0) -11.0530)
  :y (+ (/ 40.62 2.0) 4.4870)
  :z (+ 1.27)
  :xy-alignment :centered
  :prescribed-flux-surfaces ((:top 0.08))
  :material :silicon)

(defcomponent :CHIP-5 :chip
  :size (7.2430 7.2440 .669)
  :x (+ (/ 40.64 2.0) -10.8227)
  :y (+ (/ 40.62 2.0) 11.6080)
  :z (+ 1.27)
  :xy-alignment :centered
  :prescribed-flux-surfaces ((:top 0.08))
  :material :silicon)

(defcomponent :CHIP-6 :chip
  :size (6.7090 16.7800 .429)
  :x (+ (/ 40.64 2.0) 13.6987)
  :y (+ (/ 40.62 2.0) 8.4359)
  :z (+ 1.27)
  :xy-alignment :centered
  :prescribed-flux-surfaces ((:top 0.08))
  :material :silicon)

(defcomponent :CHIP-7 :chip
  :size (4.9090 4.6160 .361)
  :x (+ (/ 40.64 2.0) -12.3240)
  :y (+ (/ 40.62 2.0) -15.6350)
  :z (+ 1.27)
  :xy-alignment :centered
  :prescribed-flux-surfaces ((:top 0.08))
  :material :silicon)

(defcomponent :CHIP-8 :chip
  :size (3.8500 4.9500 .480)
  :x (+ (/ 40.64 2.0) 7.8190)
  :y (+ (/ 40.62 2.0) 2.4290)
  :z (+ 1.27)
  :xy-alignment :centered
  :prescribed-flux-surfaces ((:top 0.08))
  :material :silicon)

(defcomponent :CHIP-9 :chip
  :size (3.4190 5.1230 .264)
  :x (+ (/ 40.64 2.0) 5.0069)
  :y (+ (/ 40.62 2.0) 9.0850)
  :z (+ 1.27)
  :xy-alignment :centered
  :prescribed-flux-surfaces ((:top 0.08))
  :material :silicon)
```

```
(defcomponent :CHIP-10 :chip
  :size (2.0090 2.8490 .567)
  :x (+ (/ 40.64 2.0) 8.0830)
  :y (+ (/ 40.62 2.0) 14.7080)
  :z (+ 1.27)
  :xy-alignment :centered
  :prescribed-flux-surfaces ((:top 0.08))
  :material :silicon)

;;; ------------------------------------------------------------------------
;;;   End of File
;;; ------------------------------------------------------------------------
```

B IMCMA Trace Output

As the IMCMA 2.0 system is operating, it describes what is occurring to the user in the form of brief trace messages. An example of the trace output for an analysis of the Test MCM example (Appendix A) is included below.

```
;; Starting IMCMA:
;;   Clearing the blackboard database.
;;   Setting event printers.
;;   Creating KSs.
;;   Starting control shell.
```

Once the blackboard system and knowledge sources have been initialized, the INPUT-MODEL-KS KS is triggered to read the device description from the device-definition file. This trace is from a "batch-mode" analysis, where an existing device is being analyzed. If the device was being created interactively using the GUI, INPUT-MODEL-KS would not be called and the IMCMA analysis would start with CREATE-MODEL-KS (below).

```
;; Executing INPUT-MODEL-KS:
;;   Loading model from /usr/users/cork/imcma/examples/test-mcm.lisp.
;; Loading /usr/users/cork/imcma/examples/test-mcm.lisp.
;;   Defining device TEST MCM.
;;   Instantiating the defined and model blackboards.
;;   Defining material SILICON.
;;   Defining material AL2O3.
;;   Defining component SUBSTRATE-1 (SUBSTRATE).
;;   Defining component CHIP-1 (CHIP).
;;   Defining component CHIP-2 (CHIP).
;;   Defining component CHIP-3 (CHIP).
;;   Defining component CHIP-4 (CHIP).
;;   Defining component CHIP-5 (CHIP).
;;   Defining component CHIP-6 (CHIP).
;;   Defining component CHIP-7 (CHIP).
;;   Defining component CHIP-8 (CHIP).
;;   Defining component CHIP-9 (CHIP).
;;   Defining component CHIP-10 (CHIP).
;;   Model loaded.
```

Now that the device description has been read from the device-definition file, model objects are created on the blackboard.

```
;; Executing CREATE-MODEL-KS:
;;   Modeling component CHIP-6 (CHIP).
;;   Modeling component CHIP-7 (CHIP).
;;   Modeling component CHIP-1 (CHIP).
;;   Modeling component CHIP-8 (CHIP).
;;   Modeling component CHIP-2 (CHIP).
;;   Modeling component CHIP-9 (CHIP).
;;   Modeling component CHIP-3 (CHIP).
;;   Modeling component CHIP-10 (CHIP).
;;   Modeling component CHIP-4 (CHIP).
;;   Modeling component CHIP-5 (CHIP).
;;   Modeling component SUBSTRATE-1 (SUBSTRATE).
;;   Modeling material SILICON (1).
;;   Modeling material AL2O3 (2).
```

Modelled objects are adjusted (shifted) to allow a courser mesh to be used (if shifting has been specified).

```
;; Executing ADJUST-MODEL-KS:
;;   Actual XY Adjust: 0.0.
;;   Actual Z Adjust:  0.0.
```

The model is completed by: determining what extra planes of nodes are needed for a good mesh and creating the point and surface sources from the device description.

```
;; Executing COMPLETE-MODEL-KS:
;;   Refining mesh in X dimension based on average element-edge size
;;     (adding 16 planes of nodes).
;;   Refining mesh in Y dimension based on average element-edge size
;;     (adding 13 planes of nodes).
;;   Making prescribed surfaces and point sources for
;;          component CHIP-6 (1 surface, 0 points).
;;   Making prescribed surfaces and point sources for
;;          component CHIP-7 (1 surface, 0 points).
;;   Making prescribed surfaces and point sources for
;;          component CHIP-1 (1 surface, 0 points).
;;   Making prescribed surfaces and point sources for
;;          component CHIP-8 (1 surface, 0 points).
;;   Making prescribed surfaces and point sources for
;;          component CHIP-2 (1 surface, 0 points).
;;   Making prescribed surfaces and point sources for
;;          component CHIP-9 (1 surface, 0 points).
;;   Making prescribed surfaces and point sources for
;;          component CHIP-3 (1 surface, 0 points).
;;   Making prescribed surfaces and point sources for
;;          component CHIP-10 (1 surface, 0 points).

;;   Making prescribed surfaces and point sources for
;;          component CHIP-4 (1 surface, 0 points).
;;   Making prescribed surfaces and point sources for
;;          component CHIP-5 (1 surface, 0 points).
```

```
;;   Making prescribed surfaces and point sources for
;;          component SUBSTRATE-1 (1 surface, 0 points).
```

Next, the individual components are meshed, one at a time. Each new component's mesh is merged with the existing mesh as part of the generation process.

```
;; Executing GENERATE-3D-COMPONENT-MESH-KS for CHIP-6.
;;   Created 140 elements.

;; Executing GENERATE-3D-COMPONENT-MESH-KS for CHIP-7.
;;   Created 48 elements.

;; Executing GENERATE-3D-COMPONENT-MESH-KS for CHIP-1.
;;   Created 120 elements.

;; Executing GENERATE-3D-COMPONENT-MESH-KS for CHIP-8.
;;   Created 40 elements.

;; Executing GENERATE-3D-COMPONENT-MESH-KS for CHIP-2.
;;   Created 40 elements.

;; Executing GENERATE-3D-COMPONENT-MESH-KS for CHIP-9.
;;   Created 40 elements.

;; Executing GENERATE-3D-COMPONENT-MESH-KS for CHIP-3.
;;   Created 120 elements.

;; Executing GENERATE-3D-COMPONENT-MESH-KS for CHIP-10.
;;   Created 12 elements.

;; Executing GENERATE-3D-COMPONENT-MESH-KS for CHIP-4.
;;   Created 40 elements.

;; Executing GENERATE-3D-COMPONENT-MESH-KS for CHIP-5.
;;   Created 84 elements.

;; Executing GENERATE-3D-COMPONENT-MESH-KS for SUBSTRATE-1.
;;   Created 1330 elements.
```

Once the complete mesh has been generated, additional nodes and elements are added to model insulated side surfaces of chip components. In this device with the chips on top, no insulating surfaces are created.

This KS also determines which nodes and elements are on exterior surfaces of the device (used solely by the GUI displays) and links surface and point objects to elements and nodes.

```
;; Executing COMPLETE-3D-MESH-KS:
;;   Adding nodes for insulating surfaces.
```

```
;;      Adding nodes for CHIP-3 (0 nodes)
;;      Adding nodes for CHIP-6 (0 nodes)
;;      Adding nodes for CHIP-10 (0 nodes)
;;      Adding nodes for CHIP-7 (0 nodes)
;;      Adding nodes for CHIP-4 (0 nodes)
;;      Adding nodes for CHIP-1 (0 nodes)
;;      Adding nodes for CHIP-5 (0 nodes)
;;      Adding nodes for CHIP-8 (0 nodes)
;;      Adding nodes for CHIP-2 (0 nodes)
;;      Adding nodes for CHIP-9 (0 nodes)
;;    Determining 3D-element positions (1330 top elements).
;;    Determining 3D-node positions (1638 top nodes).
;;    Linking prescribed convection surfaces to 3D elements (0 links).
;;    Linking prescribed flux surfaces to 3D elements (342 links).
;;    Linking prescribed temperature surfaces to 3D elements/nodes (1404 links).
;;    Linking power dissipation surfaces to 3D nodes (0 links).
;;    Linking point heat sources to 3D elements (0 links).
;;    Linking point static loads to 3D elements (0 links).
```

Finally, the FORTRAN-based FEECAP codes are run by creating an input file, invoking FEECAP on this file, and then reading the result file produced by FEECAP. Elemental temperatures (used in the GUI) are computed from the individual node temperatures.

```
;; Executing ANALYZE-3D-MESH-KS:
;;   Writing FEECAP input file.
;;     Assigning prescribed temperature surfaces.
;;     Writing nodal data.
;;     Writing material data.
;;     Writing elemental data.
;;     Writing convention data.
;;     Writing elemental flux data.
;;     Writing thermal load data.
;;     Writing static load data.
;;   Invoking FEECAP.
;;   Reading FEECAP analysis results.
;;     Reading nodal temperature data.  (3746 nodes)
;;     Reading elemental-error-ratio data.  (2014 elements)
;;   Nodal temperatures: 29.956..33.977.
;;   Maximum element error ratio: 7.5885.
;;   Computing elemental temperatures.
 (29.97475..33.90825).
;; IMCMA Analysis Completed.
nil
```

C Running IMCMA 2.0

This appendix provides a quick overview of the steps used to run the IMCMA 2.0 prototype system.[9] The IMCMA graphical user interface (GUI) is activated by running the function `(imcma-gui)`.[10] When the IMCMA GUI is running, the IMCMA *main menu* will be visible:

Figure 9 The IMCMA Main Menu

Selecting the **File** button brings up the following *file menu*:

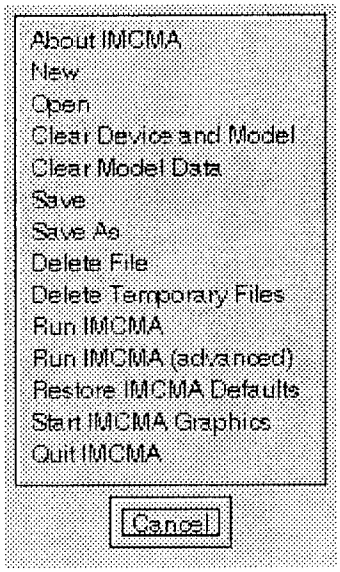


Figure 10 The File Operations Dialog

[9]This appendix assumes the IMCMA 2.0 system has been correctly installed and loaded into GBB.

[10]Depending upon how IMCMA has been installed, this function may be automatically called as part of starting the IMCMA/GBB image.

Selecting the **Open** item brings up the *Open File* dialog (Figure 11) for selecting an existing device definition file. Once the desired file has been selected and the **Open** button clicked with the mouse, the device definition will be loaded into IMCMA.

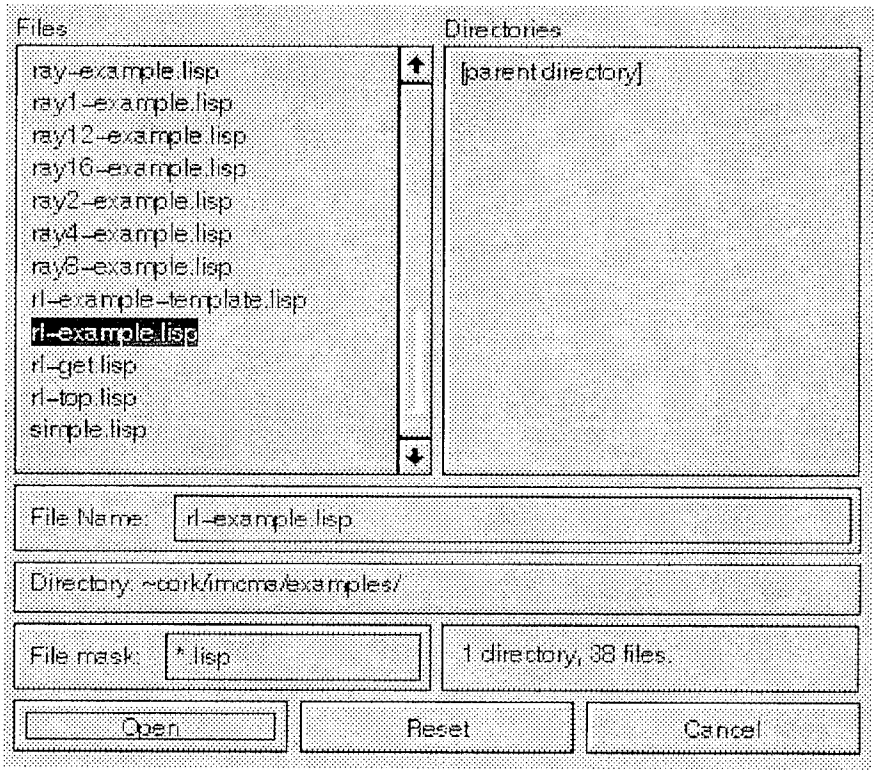


Figure 11 The Open File Dialog

At any point, the IMCMA graphics facility can be started using the **Start IMCMA Graphics** item in the **File** menu. This will start in the standard 2D display mode.[11]

To perform an IMCMA analysis, use the **Run IMCMA** item in the **File** menu. This will bring up the following dialog:

Clicking the Run button initiates IMCMA analysis of the device.[12]

[11]The **IMCMA Mouse Documentation** window provides help in determining what can be done using the IMCMA graphics facility.

[12]For a more detailed description of the IMCMA GUI, see [12].

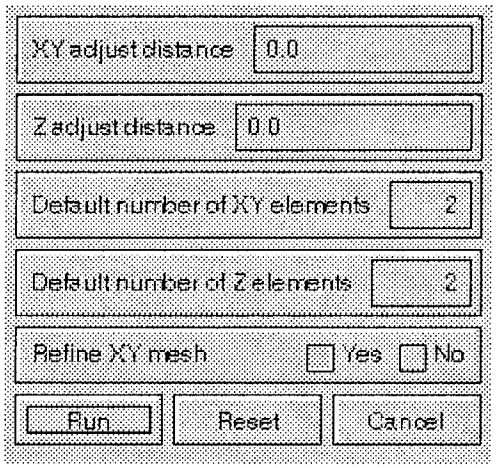


Figure 12 The Run IMCMA Dialog

D Glossary

Blackboard (GBB)	A space or another GBB blackboard
GBB	Product trademark for Blackboard Technology Group's generic blackboard framework
FEECAP	UMass finite-element analysis package
IMCMA	Intelligent Multichip Module Analyst system
KS	Knowledge source
KSA	An activation of a knowledge source
Link	A bidirectional relationship between two GBB blackboard units
MCM	Multichip module
Space (GBB)	A blackboard level or plane
UMass	University of Massachusetts
Unit (GBB)	A blackboard object

Rome Laboratory

Customer Satisfaction Survey

RL-TR-_____

Please complete this survey, and mail to RL/IMPS, 26 Electronic Pky, Griffiss AFB NY 13441-4514. Your assessment and feedback regarding this technical report will allow Rome Laboratory to have a vehicle to continuously improve our methods of research, publication, and customer satisfaction. Your assistance is greatly appreciated.
Thank You

_____

_____

Organization Name:_____(Optional)

Organization POC: _____(Optional)

Address:_____

1. On a scale of 1 to 5 how would you rate the technology developed under this research?

5-Extremely Useful 1-Not Useful/Wasteful

Rating_____

Please use the space below to comment on your rating. Please suggest improvements. Use the back of this sheet if necessary.

2. Do any specific areas of the report stand out as exceptional?

Yes___ No_____

If yes, please identify the area(s), and comment on what aspects make them "stand out."

3. Do any specific areas of the report stand out as inferior?

Yes___ No___

If yes, please identify the area(s), and comment on what aspects make them "stand out."

4. Please utilize the space below to comment on any other aspects of the report. Comments on both technical content and reporting format are desired.

*U.S. GOVERNMENT PRINTING OFFICE: 1995-610-126-20028