NASA Contractor Report 187227

NASA-CR-187227
19920010035

Engine Structures Modeling Software System (ESMOSS)

General Electric Company
Aircraft Engine Business Group
Cincinnati, Ohio

October 1991

Prepared for
Lewis Research Center
Under Contract NAS3-22767

NASA
National Aeronautics and
Space Administration

TABLE OF CONTENTS

LIST OF ILLUSTRATIONS

LIST OF ILLUSTRATIONS (Continued)

LIST OF ILLUSTRATIONS (Continues)

1.0 INTRODUCTION AND SUMMARY

Throughout the aircraft engine industry, the use of computers is fundamental to the design and analysis of jet engine components. Similarly, the use of graphics devices for interactive design and display is widespread. In particular, a tremendous amount of computer software is in existence for the modeling and analysis of engine structures using Finite Element Methodology (FEM). For the most part, this software is in the form of individual unique stand-alone programs capable of handling only a part of the total process. Pre- and post-processing packages have been developed in an attempt to tie the different pieces together, generally through intermediate data files. This has facilitated the process, but the designer/analyst is required to run separate programs, set up files and provide data transfer; in effect, he must manage the system in order to carry out the desired analysis. A means of unifying all of these capabilities and software into a user-oriented software system was needed. The development of the Engine Structures Modeling Software System (ESMOSS) was undertaken to fulfill this need.

The main objective of ESMOSS is the development of a specialized software system for the construction of geometric descriptive and discrete analytical models of engine parts, components and substructures which can be transferred to finite element analysis programs such as NASTRAN. The NASA-Lewis Engine Structures Program is concerned with the development of technology for the rational structural design and analysis of advanced gas turbine engines with emphasis on advanced structural analysis, structural dynamics, structural aspects of aeroelasticity, and life prediction. Fundamental and common to all these developments is the need for geometric and analytical model descriptions at various engine assembly levels.

The software architecture of ESMOSS has been designed and developed in modular form with a central executive module through which the user controls and directs the development of the analytical model. Modules consist of a geometric shape generator; a library of discretizing procedures; an interfacing module for joining models of structural parts into a structure; and engine component and substructure modules. The system has separate modules which provide for input and output data to and from stored files, external analysis programs and terminal devices such as keyboards and graphic displays. The design of ESMOSS and the individual system modules are open-ended with built-in flexibility to allow for future expansion and state-of-the-art developments.

2.0 SURVEY

The first task was to perform a survey of computerized geometric and discretization generators. Both computerized and manual searches were made for techniques and software in the general areas of geometric shape descriptions, model discretization, and mesh refinement. From these literature searches, documents considered to have applicability were ordered from the Aircraft Engine Business Group (AEBG) Technical Information Center. The computerized searches utilized the NASA Scientific and Technical Information Facility, the Defense Technical Information Center, and the Lockheed Information Systems' DIALOG service.

Based on this survey, the recommended method for the storage of geometric shapes was the bounded surface approach. Its ability to store complex shapes, the use of a relational model and compatibility with discretization procedures were definite advantages. The GPRIME system, developed at the David W. Taylor Naval Ship Research and Development Center, had direct applicability but needed some enhancement to support the requirements of the ESMOSS shape generator library. GRPIME includes a geometric language processor for structural modeling and uses cubic B-spline functions for internal geometric representation.

The geometric language approach for shape generation has several advantages. First, the method for creating the geometry is natural and well defined. Points are created from previously defined scalars, curves created from points, and surfaces created from curves. Second, a language which uses easily remembered mnemonics provides the user with an effective and understandable modeling tool. Third, the language can be structured to handle new features and accommodate new construction techniques.

A weakness in GPR1ME is its inability to store the description of a solid object. The six planes which can be used to define a cube are connected only in the mind of the user. This same deficiency is present in most interactive graphics systems. The problem was solved by forming a relational model in addition to the geometry. The original work in this area was Bruce Baumgart's "winged-edge" polyhedra (Stanford University Report, STAN-CS-72-320). In that work, the face of a polyhedron is bounded by edges which intersect at vertex points. Non-planar surfaces can be handled by mapping the surfaces onto the relational model. Using this approach, the cube's surfaces have now been tied together.

The basic discretization approach for ESMOSS is to subdivide the geometric model into regions. The regions, characterized geometrically as face-edge-vertex models, are subsets of the overall geometric model. The finite-element mesh can then be generated region by region and merged or interfaced. No available software was identified during the survey which would satisfy these requirements while being easily integrated into GPR1ME.

3.0 SOFTWARE DEFINITION

Once the survey was completed the next step was the definition of the software system. As with any major software project, it is critical to establish a clearly defined set of specifications and requirements early in the project. This was the objective of the methodology used on ESMOSS.

The primary goals in the development of the software architecture were modularity, open-endedness, portability, and user-friendliness. In addiion to these goals, which are highly visible, the software's efficiency, reliability, and maintainability were equally important. These last three items are difficult to quantify or measure, but the design of the software architecture must be built around them. The General Electric Company's software design philosophy embodies this total set of goals and was applied to ESMOSS. The methodology that was used to achieve these goals is described below.

The methodology which was used is a structured approach to software development. This technique allows for the formal, controlled solution of complex problems and can be broken down into four phases, each of which has a set of associated documentation:

1. Preparation of the software requirements document (i.e., the Statement of Work)
2. Functional analysis of the problem to be solved (i.e., what the software is intended to do)
3. Design of the software system (i.e., how the software will solve the problem)
4. Implementation and testing of the resultant software system.

Item 2 as defined above consists of the definition of the software architecture and items 3 and 4 are considered to be part of the software development process.

The design of the software architecture consists of a functional analysis of the problem to be solved -- the modeling of engine structures. The end product of this analysis effort is the software functional specification. The structured analysis technique which was used is similar to the Integrated Computer-Aided Manufacturing Definition (IDEF-0) which was developed for the U.S. Air Force by SofTech, Inc. and is based on SofTech's Structured Analysis and Design Technique (SADT). This is a graphical methodology which begins by defining the problem in its most general terms, including its inputs and outputs to the outside world, and which provides for a hierarchical decomposition of the problem into more and more precise functions. At each level of decomposition, each function individually defines itself and its interfaces and collectively these functions define the problem. This procedure is continued until the original problem is defined at a sufficiently detailed level to allow easy solution. The upper level diagrams of this functional decomposition which define the system architecture are shown in Appendix A.

ESMOSS is menu driven with overall program control being handled by an executive which provides the following functions:

1. User command processing
2. Error processing and control
3. Data file manipulation
4. User interaction with the various modules.

This architecture allows for both user-friendliness (through the menu driven arrangement) and open-endedness (through the executive control of the execution of the various program modules). Since the executive provides the interface to the file system and supplies that file information to the modules, the executing modules are written independently of the computer operating system.

The user interaction is provided by autonomous input and output modules. This philosophy allows the executive to be device-independent with the specific device characteristics being incorporated into the input and output modules. Using autonomous input and output modules also provides an open-endedness for the addition of any new I/O devices that may be desired later. The output module provides for the display of the input data as well as both graphical display of geometric and discrete models.

The geometric modeling, discretization, and component definition are performed by modules which execute under the control of the program executive. These modules are independent of both file manipulation and user interaction, both of which are provided for them by the executive and input/output modules. This modularity and I/O independence will make adding additional capabilities and techniques to the system very easy should such a need arise at a later time.

Once the functional analysis was completed and approved, the design of the computer software was initiated. As mentioned above, the ESMOSS software development was based on state-of-the-art-techniques in structured design, programming, and testing. These techniques provide a method for transforming the statement of the problem as defined by the functional analysis into a top-down design, the structure of which can be easily implemented, tested, and maintained. Specifically, the techniques used were the structured design

methodology developed and promoted by Yourdon and Constantine. Several of the top level structure charts are shown in Appendix B. Once the structure charts were completed, each subroutine was designed in detail using structured flow charts (i.e. Nassi-Schneiderman charts). The testing philosophy was to test each module individually with the rest of the system simulated. This independent testing of modules allowed for the maximum detection of errors before the modules were integrated.

4.0 EXECUTIVE MODULE

The executive module is structured into four separate functional packages of subroutines as follows:

1. Menu Directory Maintenance - creates the menu directory from an ASCII sequential file. The user defines the menu structure along with the action to be taken and files required for each menu item. This file is then processed into a random access menu directory file which is used during ESMOSS execution.
2. Command Processor - presents menus to the user, receives and interprets responses and determines the action to be taken. The command processor has been designed to be very user-friendly by allowing the input of the menu responses to be either menu number, name or abbreviated name and by allowing the user to enter multiple menu selections on a single line. All menu items will have a provision for giving the user further information via the "HELP" command.
3. File Manager - satisfies the file requirements for module execution and maintains a Logical Unit Table of all currently open files. All interfacing with the file system is done in this area.
4. File Manipulator - allows the user to add to, delete from or interrogate both the Logical Unit Table and the ESMOSS File Directory.

User control in ESMOSS is menu driven, utilizing a command processor. The command processor is designed to work equally well in either a batch or an interactive mode. The menu approach in ESMOSS is hierarchical. After successfully signing on to ESMOSS, the highest-level menu, the master menu, is activated. The user then selects one of the commands associated with the active menu - in this case, the master menu. The command processor has been designed to be very user-friendly in the manner in which it accepts a user command. Each command has a number associated with it. The user can select a command by entering either the command number, the command itself, or a distinguishable abbreviation of the command. Each command can be one to three words long, each word being separated by a blank.

After a command is entered and recognized by the command processor, its corresponding action is taken. This action can be one of two types - execute an ESMOSS module or activate a new menu. With this method, the user can traverse a hierarchy of menus until the desired level to execute a function is reached. In addition to the currently active menu, general-purpose utility commands in a resident menu are always active and can be invoked.

Menus are the basis for the command processor. Each menu includes the following pieces of information: a descriptive title of the menu, a unique menu number, and the number of commands in this menu. Included with the information on the menu is specific information about each of the menu's commands. The command information includes the actual command, the action to be taken, the "Help" message for the command and the file requirements if a module is to be invoked. Menus are input to the ESMOSS system via a sequential file. The maintenance portion of the executive uses the information on this sequential file to set up an indexing structure using a random file. This indexing structure greatly increases the speed of the subsequent user interaction with ESMOSS.

The menu approach taken in ESMOSS offers many advantages. One obvious advantage, in the interactive mode, is the prompting and assistance provided to the user. The user can ask, at any time, for the currently allowable commands to be displayed, and even for a brief description of each of them, if so desired. A major advantage in both the interactive and batch environments is the user-friendliness of the command processor. The use of a menu-driven processor also gives the system designer a great deal of flexibility and allows an open-ended design. The indexed menu structure greatly increases the speed in which commands from the user can be executed. Further, the menus serve as directories for all system functions and facilities, including the ESMOSS libraries.

5.0 GEOMETRIC SHAPE GENERATOR

As discussed in Section 2.0, GPRIME provides the basis for the ESMOSS Geometric Shape Generator Library. Geometric models are constructed of points, curves and surfaces which are defined by the user and stored internally as cubic B-splines. In addition, the original GPRIME capability was expanded to allow the definition of regions which are used to define the relational model needed for discretization. The relational model is the vehicle by which the various geometric entitites are grouped logically as parts of "super elements" which can be discretized. ESMOSS has the capability of creating both six-sided solid regions and surface regions (i.e. two-dimension) with either three or four bounding edges.

The paragraphs that follow discuss the capabilities available for defining the geometric entities:

Points can be defined as follows:

1) Basic Point Definition - defined from the coordinates of the point in rectangular, cylindrical or spherical coordinates.
2) Point Defined from a File of Digitized Data - defined from the coordinates read from a digitized data file.
3) Point Defined by Parametric Reference of a Curve - defined by evaluating a curve at a specified parametric value.
4) Point Defined by Parametric Reference of a Surface - defined by evaluating a surface at a specified parametric value.
5) Point Defined by Intersection - defined by determining the intersection of two curves.

Curves Can Be Defined as Follows:

1) Straight Line - defined by line connecting two points.

2) Circle - defined by specifying its center point and two points on the circumference or by specifying three points on its circumference.

3) Circular Arc - defined by specifying its center and endpoints, or by specifying three points on the arc, the first and last of which are its endpoints.

4) Curve Defined from a File of Digitized Data - defined by fitting a set of digitized data points read from the file.

5) Curve Defined by Intersection - defined by determining the intersection of two surfaces.

6) Curve to Fit a Sequence of Points - defined to fit a set of previously defined points with the number of B-spline functions given.

7) Curve Defined by Joining Two Curves - defined as the union of two parent curves.

8) Curve Defined as a Parabola - defined by specifying a vertex point and two end points or by two endpoints and the tangents at those endpoints.

Surfaces Can be Defined as Follows:

1) Plane Perpendicular to a Coordinate Axis - defined as a plane with given extents perpendicular to a coordinate axis in the rectangular coordinate system.

2) Plane or Warped Surface Defined by Four Points - Defined as a planar or warped quadrilateral patch with four specified corner points.

3) Circular Cylinder - defined as a right circular cylinder, given the endpoints of the cylindrical axis and the radius of the cylinder, or by specifying the endpoints of the cylindrical axis and one point on the circumference of the cylinder at the base.

4) Frustrum of a Circular Cone - defined as the frustrum of a circular cone, given the two end points of the axis of the cone and two points on the circumferences of the top and bottom.

5) Ruled Surface between Two Curves - defined as a surface of straight lines generated by connecting corresponding points on a pair of three - dimensional arcs.

6) Surface Rotation - defined as the rotation of a curve about a coordinate axis thru a given number of degrees.

7) Surface to Fit a Collection of Curves (Cubic B-surface) - defined by fitting or passing through a given number of previously defined curves.

8) Parametric Subsurface Patch - defined as the subregion of a parent surface, bounded by parametric straight lines connecting four corner points.

9) Surface Defined from a File of Digitized Data - defined to fit an ordered set of digitized data points with a given number of B-spline functions in the parametric S direction and a given number of B-spline functions in the parametric T direction.

6.0 DISCRETIZING PROCEDURES LIBRARY

The input to the Discretizing Procedure Library is a completed geometric model. The geometry file contains a regionalized description of the part in the form of cubic B-splines. ESMOSS discretizes the geometric model on a region by region basis. A model may contain up to 200 individual regions, each of which may be discretized (or rediscretized) without affecting the model as a whole. A geometric model can be regionalized using four different region types. Solid models must be divided into six-sided solid regions. Plate, shell or two-dimensional models may be regionalized into four-sided regular regions, four-sided irregular regions or three-sided irregular regions or a combination of all three.

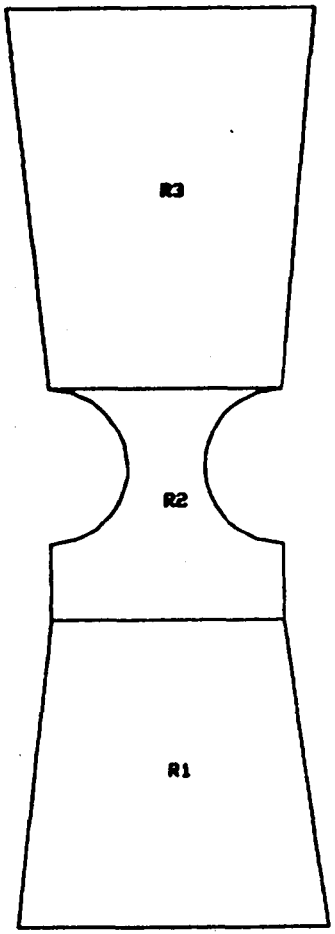
The discretization of a region begins with the assignment of the nodes along the region edges. The user specifies the number of nodes interior to the edge (nodes are automatically assigned to the vertices) and the type of node biasing desired. The bias parameter, which is required with each bias type, determines the magnitude at which edge nodes are to be packed. The parameter is read as a real number and must be greater than zero. The bias parameter is used such that a value of 2 places nodes twice as closely together in the packed section as in the loose section with a linear variation in between. There are four possible biasing options:

1. No bias - produces equally spaced nodes along the edge.
2. End bias - produces a nodal pattern with the nodes packed closer to one end of the edge.
3. Center bias - packs nodes symmetrically toward the center of the edge.
4. Edge bias - packs nodes symmetrically toward the vertices of the edge (the inverse of center bias).

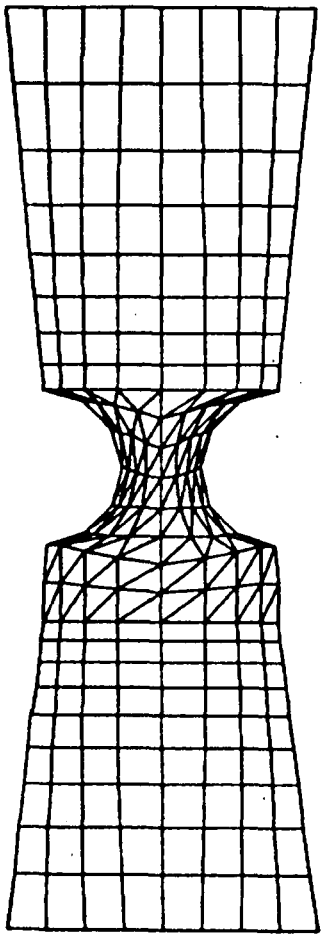
Solid three-dimensional models must be divided into regions composed of 8 vertices (geometric points), 12 edges (geometric curves) and 6 faces (geometric surfaces). Each edge may have its nodes biased independently allowing higher concentrations of nodes in high stress areas, however, opposite edges are required to have equal numbers of nodes. Presently, all solid models are discretized with eight-noded brick elements. For determining interior nodes, ESMOSS employs a three-dimensional cubic Lagrangian shape function to generate the nodal coordinates. This method parameterizes the solid region using 64 sets of coordinates (8 vertex points, 24 edge points and 24 face points generated using the geometric library and 8 interior points found iteratively.

A two-dimensional regular region is a single face bounded by 4 vertices and 4 edges. As with the solid regions, opposite edges must have equal numbers of edge nodes, while biasing remains independent. Element types supported are corner-noded quadrilaterals and triangles. Figure 1a represents a two-dimensional model with three regions labeled R1, R2 and R3. Regions 1 and 3 in Figure 1b have been discretized using quadrilateral elements and region 2 has been discretized using triangles.

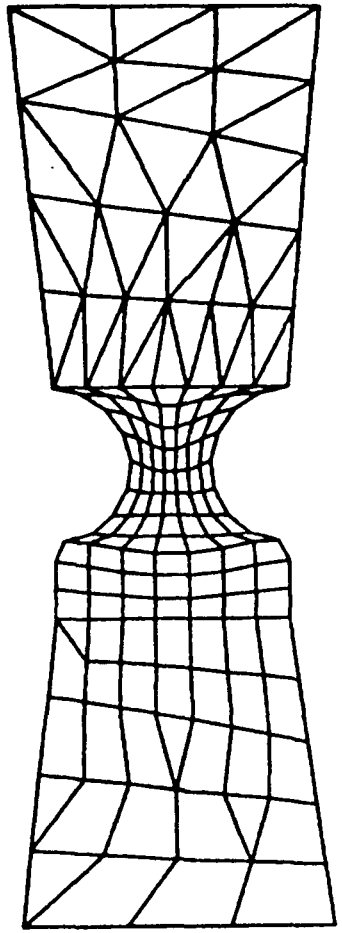
DISCRETIZED QUADRILATERAL ELEMENTS



a.



b.



c.

FIGURE 1

The irregular region option removes the constraint that opposite edges have equal numbers of nodes. Each edge is assigned a number of nodes and the program automatically transitions from an edge of high nodal density to one of low density. This allows the user greater flexibility in placing nodes in areas suspected of having high deformation. Edge biasing is still permitted, but the choice of element type is triangular or "mostly" quadrilateral elements. Highly irregular regions will require at least a few triangular elements mixed among a grid of corner-noded quadrilaterals. ESMOSS supports both three-sided and four-sided irregular regions. The algorithm used for this procedure is based on a paper by Imafuku, Kodera, Sayawaki and Kono - "A Generalized Automatic Mesh Generation Scheme for Finite Element Method" - which appeared in the International Journal for Numerical Methods in Engineering, Vol. 15, 713-731 (1980). An example of this technique is shown in Figure 1c. The objective is to create a fine mesh in Region 2 due to high stress concentrations and have a coarser mesh in Regions 1 and 3 for transitioning to other parts of the model which have lower stresses. As shown in Figure 1c, Region 2 has been discretized using the procedure for discretizing a regular region with quadrilateral elements. Regions 1 and 3 were discretized using the Imafuku method. The procedure gives the user the option to create all triangular elements (as in Region 3) or mostly quadrilateral elements with only as many triangular elements as necessary (as in Region 1).

Once a geometric model has been discretized the finalization process can be executed. The finalization module gives the capability to add the following items to a previously discretized model:

1. Nodal Boundary Conditions
2. Material Properties
3. Element Thicknesses
4. Constraint Equations
5. Loading Data

Once finalized, a model is ready to be processed by the NASTRAN interface which produces an input deck for NASTRAN analysis.

7.0 INTERFACING MODULE

The ESMOSS Interfacing Module allows the user to combine two models into a single model. Both geometric and discrete models may be interfaced. In both cases, the input to the interfacing module is two previosuly completed models. The second model is rotated and translated, as specified by the user, to align it properly with the first model. ESMOSS then automatically renames the entities of the second model so that no duplicate entity names exist. The user is asked to specify starting points for the new names in the second model. Once these procedures have been completed, a new combined model exists which can be further refined for each model type.

The new geometric model will need to have the regions at the intersection redefined so that the ESMOSS discretizing procedures library will generate a discrete model with these models connected. The user can use the definition capabilities of the geometric shape generator library to accomplish this task.

The combined discrete model can be manipulated as follows: ESMOSS has the capability to eliminate common nodes (i.e. nodes that are positioned within a tolerance) and redefine the elements containing the eliminated nodes. Before eliminating common nodes, the user has the option to selectively move nodal positions so that this elimination will take place. In addition, the combined model can be joined by using the constraint equation option of the model finalization procedure in the discretization procedures library.

8.0 DATA INPUT PROCESSOR

The ESMOSS Input Processor has been designed to be open-ended to allow future expansion with the addition of devices such as light pens, data tablets, touch screens, etc. Currently ESMOSS input can come from several sources. In addition to terminal input for interactive mode and the batch input stream for batch mode, ESMOSS allows the user to provide his input from a command file.

A command file is an ASCII file which contains one user input response per line. The command file must contain a response to every ESMOSS prompt in the processing sequence. If the user wishes a particular response to come from the terminal rather than from the command file, the line on the command file that contains that response must contain a "/*SWITCH." Command files can be created off-line using a standard text editor, created as an echo file during an interactive session, or produced by the Recipe Processor which is discussed in Section 10. Once an echo file is created it can be used to reproduce the ESMOSS session by using the created echo file as a command file. Since the echo file is an ASCII file it can be edited to modify the session.

9.0 DATA OUTPUT MODULE

The ESMOSS Data Output Module Processes four types of output.

1. Menu Display
2. User Prompts and Information Messages
3. Error Messages
4. Model Plots

The architecture for the output module was designed around NASA's TSS/370 Graphics Library. The open-endedness of this module is provided via general output device characteristics tables. This allows the addition of new devices supported by TSS/370 to be implemented by setting the appropriate values in these tables.

The display model portion of the output processor operates in two modes. Displaying one entity at a time as the model is being built or plotting a complete model that has been previously created. In addition to plotting, the display model processor provides the following capabilities and options:

1. Displaying a title for the plot.
2. Optional labeling (by name) of lines, points, surfaces and regions as well as nodes and elements for discretized model plots.
3. Windowing the plot which has the effect of zooming a portion of the plot to a larger scale.
4. Rotating the plot around the three axes to change the view orientation.
5. Setting parameters such as color and intensity for plotting devices which support those features.
6. Creating a two-dimensional plot display from the three-dimensional geometric data.

10.0 NASTRAN INTERFACE

ESMOSS has the capability to produce an input deck for analysis codes. The interface to the NASTRAN program is currently implemented and requires a finalized discrete model as input. ESMOSS will automatically produce a subset of the NASTRAN deck by prompting the user for input. This subset includes the following:

- Nodes (GRID)
- Elements
  - Isoparametric triangular elements (CTRIA3)
  - Isoparametric quadrilateral elements (CQUAD4)
  - Six-sided solid elements (CHEXA)
  - Constant strain triangular ring (CTRIARG)
  - Trapezoidal ring (CTRAPRG)
- Element Property Definitions
  - Property definition for CHEXA (PSOLID)
  - Property definition for CTRIA3 and CQUAD4 (PSHELL)
- Materials
  - Elastic material properties (MAT1 and MATT1)
  - Orthotropic material properties (MAT3 and MATT3)
  - Tabular function definition for MATT1 and MATT3 (TABELM1)

Constraints

Single point constraints (SPC)

Multi point constraints (MPC)

Loads

Concentrated load at grid point (FORCE)

Moment at grid point (MOMENT)

Pressure load on face of element (PLOAD4)

Temperatures

Temperature at grid point (TEMP)

For these supported functions NASTRAN defaults are assumed. The user is given the capability of overriding these defaults either from the terminal or from a file. In addition to the supported subset, ESMOSS allows the user to enter any other data either manually or from a file. The input "deck" will actually be in the form of a sequential file which also may be edited manually by the user prior to submission to NASTRAN.

11.0 ENGINE COMPONENT AND SUBSTRUCTURE MODULES

The Recipe Processor provides the ESMOSS user with the capability of creating and discretizing models by means of definable parameters. The process for accomplishing this is as follows:

1. A generic part is defined using the ESMOSS-defined macro language. This definition is called a "recipe" and is created as a standard ASCII sequential file using computer system's editor.
2. ESMOSS is executed to process this recipe using the ESMOSS Recipe Processor. The user is asked to input the defining parameters and the Recipe Processor creates a command file of ESMOSS primative commands which traverse the menu structure, define the necessary geometric entities and discretize the model.
3. The resulting command file is executed to generate the geometric and discrete models.

The Engine Component Modules or "recipes" allow the user to define a part using parameters without having to specifically define all of the geometric points, lines and surfaces which comprise the part. In addition, the recipes provide for discretization of geometric models. These recipes are in the form of sequential files which are processed by the Recipe Processor to create a file of ESMOSS commands which perform the required operations. The recipes are written in a high level macro language developed for ESMOSS. These language elements are given below:

Logical Operators:

AND

OR

NOT

Command Statements:

PRINT - Print character string to terminal

READ - Read real number value from terminal

WRITE - Print value of variable to terminal

INCLUDE - Include contents of another file

DEFINE - Define a macro procedure or recipe

END - End a macro procedure or recipe

Program Control Statements:

REPEAT.. UNTIL - Repeat operation until condition is met

IF THEN ..ELSE..END IF - Process based on condition

Built-in Functions:

ACOS	ASIN	ATAN	ATAN2	COS
COSH	COTAN	SIN	SINH	TAN
TANH	MAX	MIN	ABS	TRUNC
DIM	LOG	LOG10	MOD	RADIAN
SIGN	SQRT	ERF	ERFC	GAMMA
EXP	LGAMMA			

Geometry Procedures:

POINT - Generate commands to define a point (x,y,z)

PARMCRV - Generate commands to define a point from a parametric curve

PARMSRF - Generate commands to define a point from a parametric surface

LINE - Generate commands to define a line

ARC, ARCCF - Generate commands to define a circular arc

MRGCURVE - Generate commands to merge two curves

PARABOLA - Generate commands to define a parabola

CIRCLE, CIRCLEC - Generate commands to define a circle

CVPTFILE - Generate commands to define a curve from a point file

INTRSECT - Generate commands to define a curve as the intersection of two surfaces

RULEDSRF - Generate commands to define a ruled surface

FRUSTRUM - Generate commands to define a frustrum

CYLINDR, CYLINDRP - Generate commands to define a cylinder

PLANE, PLANE4PT - Generate commands to define a plane

SUBSURFC - Generate commands to define a subsurface patch

REVOLVE - Generate commands to define a surface of revolution

REG2D4S - Generate commands to define a two-dimensional, four-sided region

REG3D - Generate commands to define a three-dimensional, six-sided region

DELETEPT - Generate commands to delete a point

DELETECV - Generate commands to delete a curve

SETMODEL - Generate commands to initialize a geometry file

Discretizing Procedures:

DIS2DREG - Generate commands to discretize a regular two-dimensional region

DIS2DIRG - Generate commands to discretize an irregular two-dimensional region

DIS8NBRK - Generate commands to discretize a three-dimensional region with eight-noded brick elements

DISEDGE - Generate commands to define the number of nodes along an edge

BIASCNTR, BIASEDGE, BIASEND - generate commands to bias edge nodes

ENDISCRT - Generate commands to end the discretizing procedure

Parametric definitions were developed and recipes were generated for the following components:

Airfoil

Dovetail/Platform

Disk

Broach

Burner Liner Section

Air-Cooled Turbine Blade

The parametric models used for these recipes follow.

BURNER LINER-PARAMETRIC MODEL

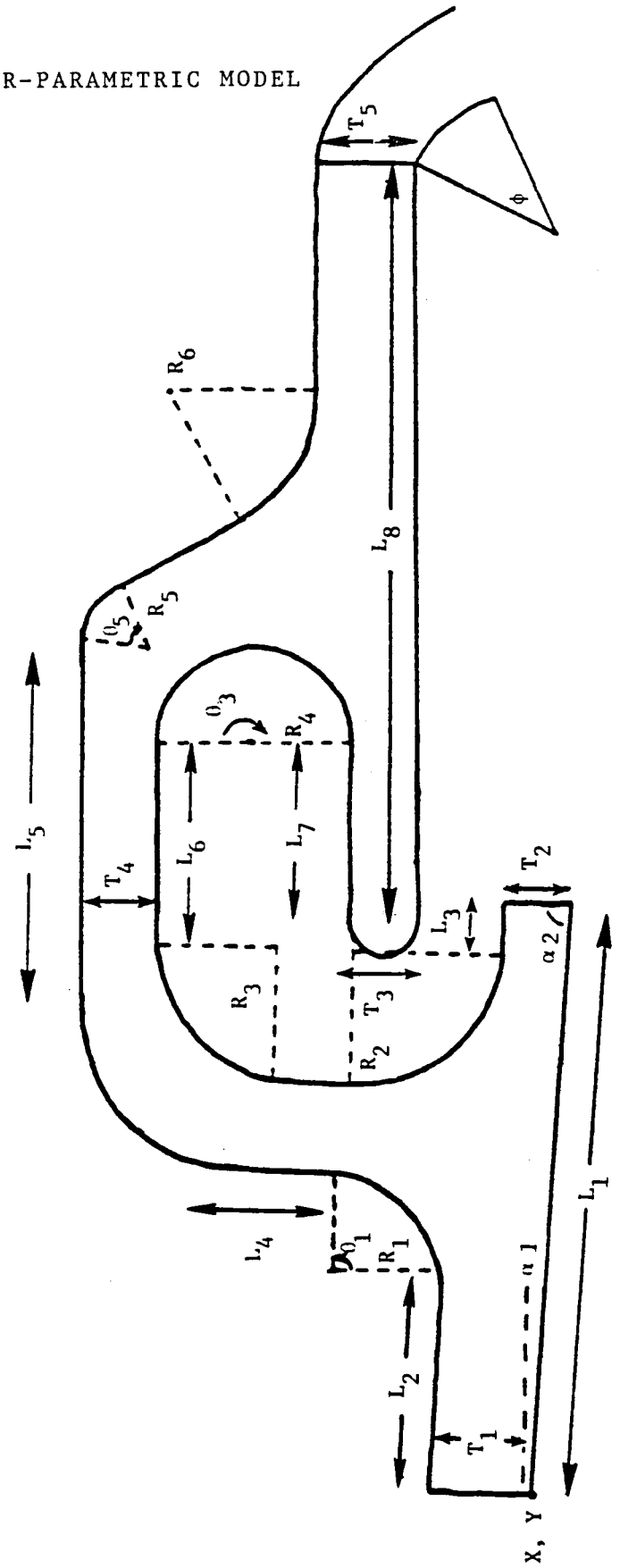


FIGURE 2

BURNER LINER - GEOMETRIC MODEL

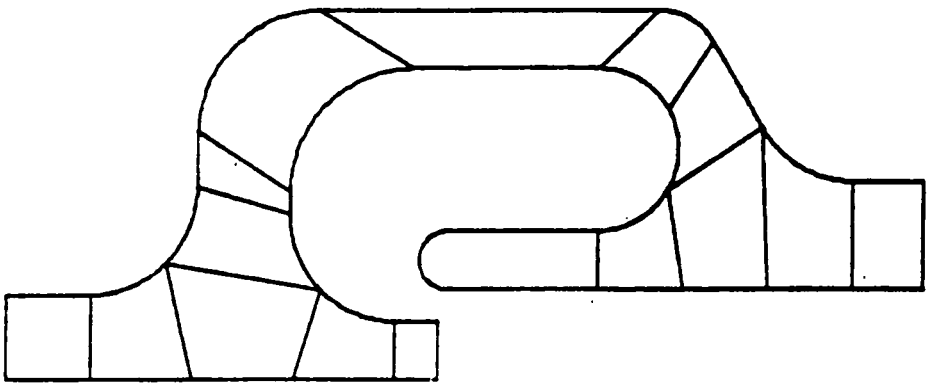


FIGURE 3

BURNER LINER - DISCRETE MODEL

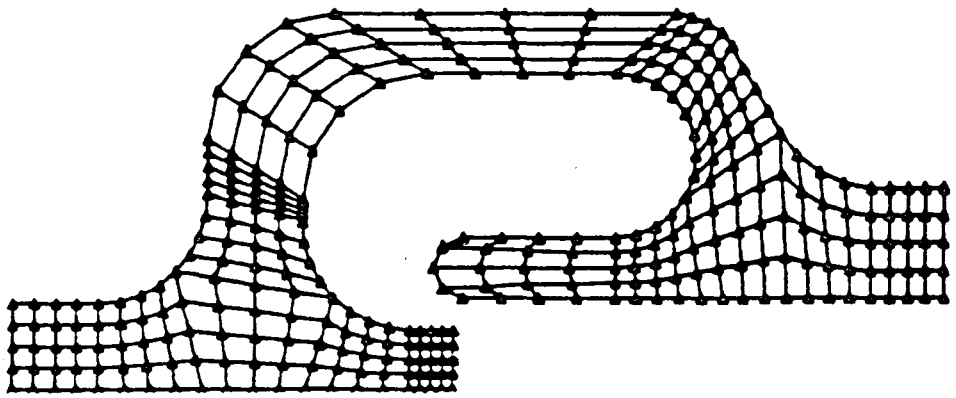
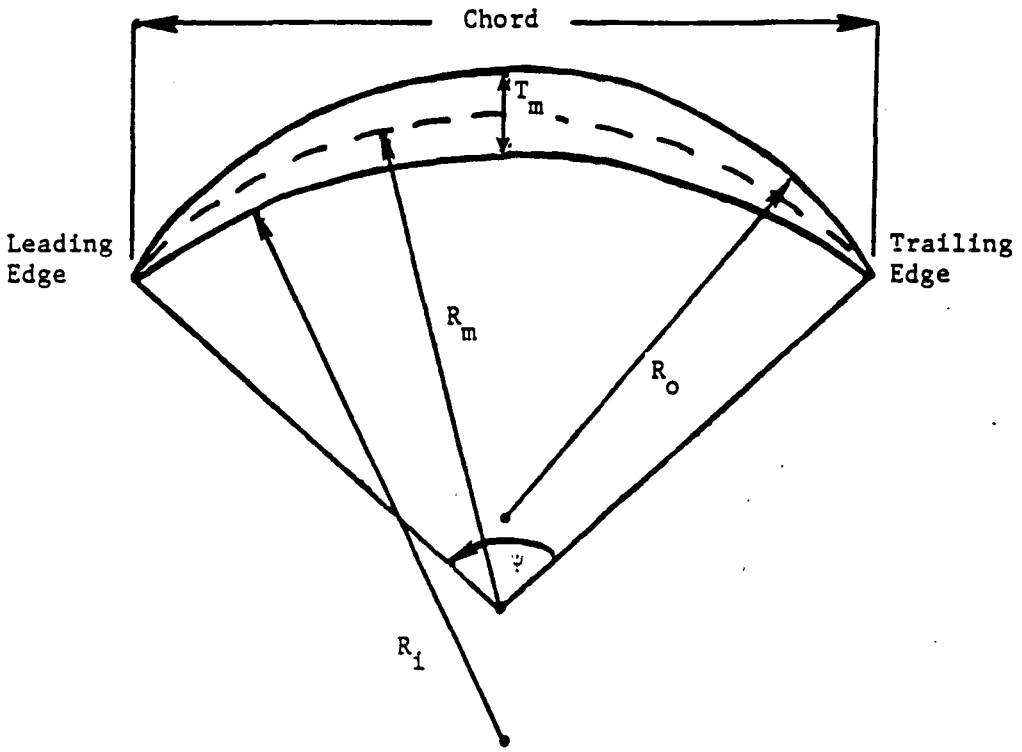


FIGURE 4

AIRFOIL - PARAMETRIC MODEL



- ψ ~ Camber Angle
- C ~ Chord
- t_m/C ~ Maximum Thickness/Chord
- Section Height
- Section Offset

FIGURE 5

AIRFOIL - GEOMETRIC MODEL

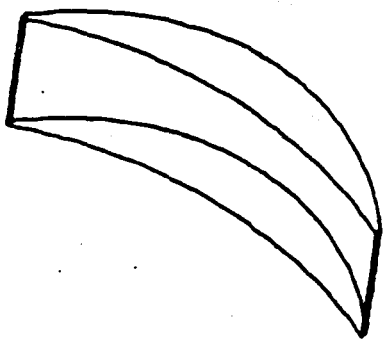


FIGURE 6

AIRFOIL - DISCRETE MODEL

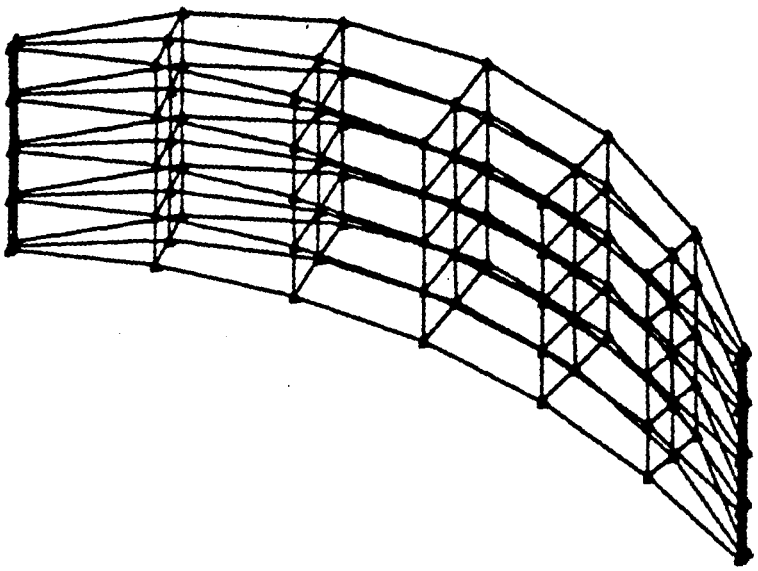
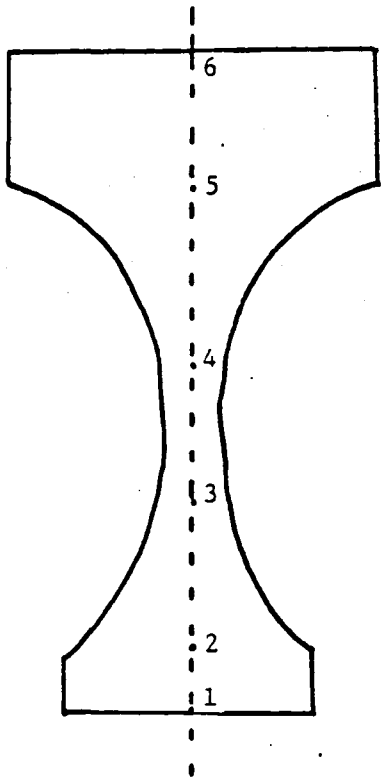


FIGURE 7

DISK - PARAMETRIC MODEL



If Arc Radius = 0, Connect by Straight Line from Prior Point

If Arc Radius > 0, Connect by Circular Arc from Prior Point

FIGURE 8

DISK - GEOMETRIC MODEL

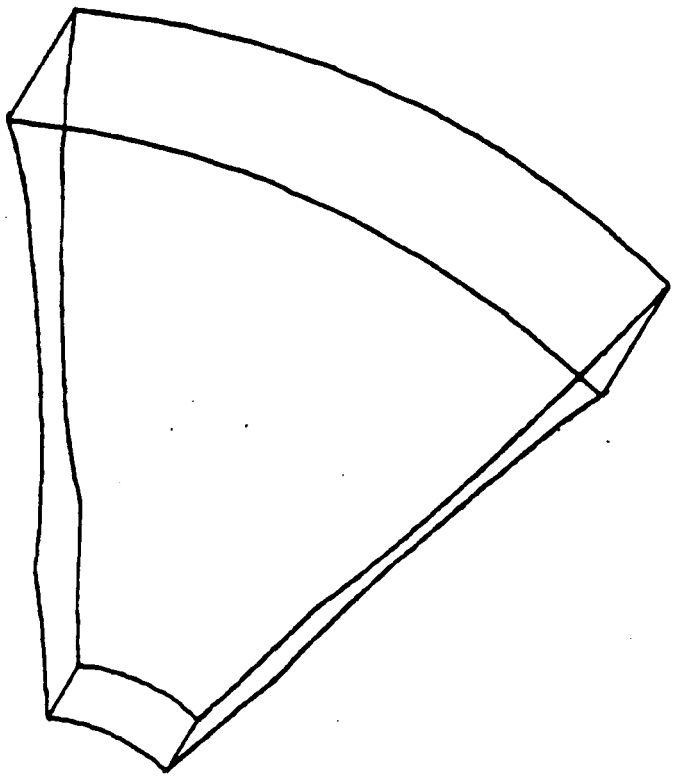


FIGURE 9

DISK - DISCRETE MODEL

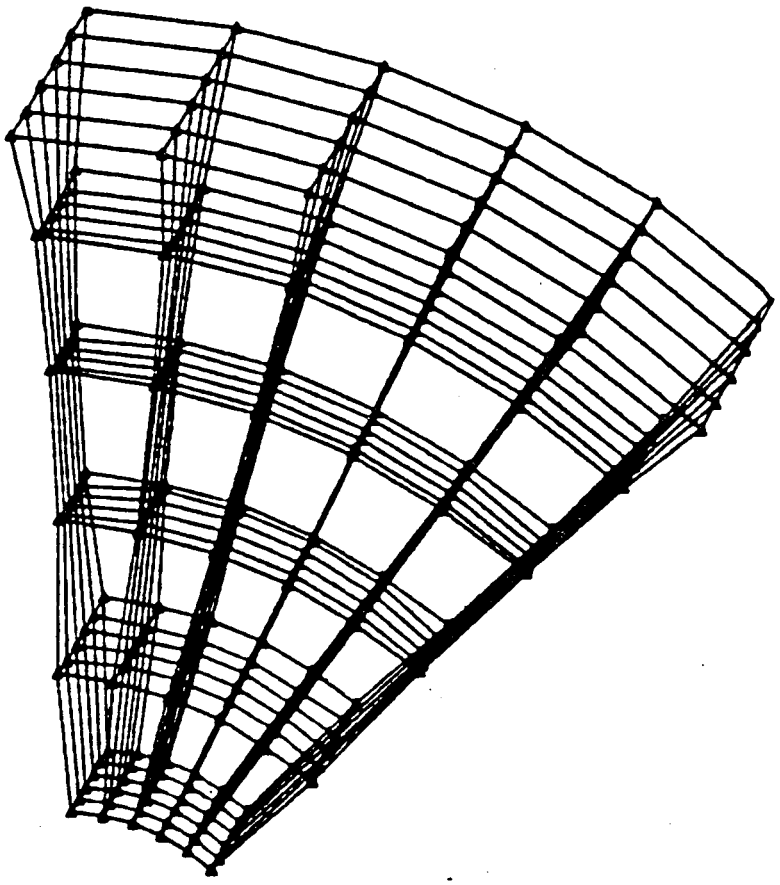
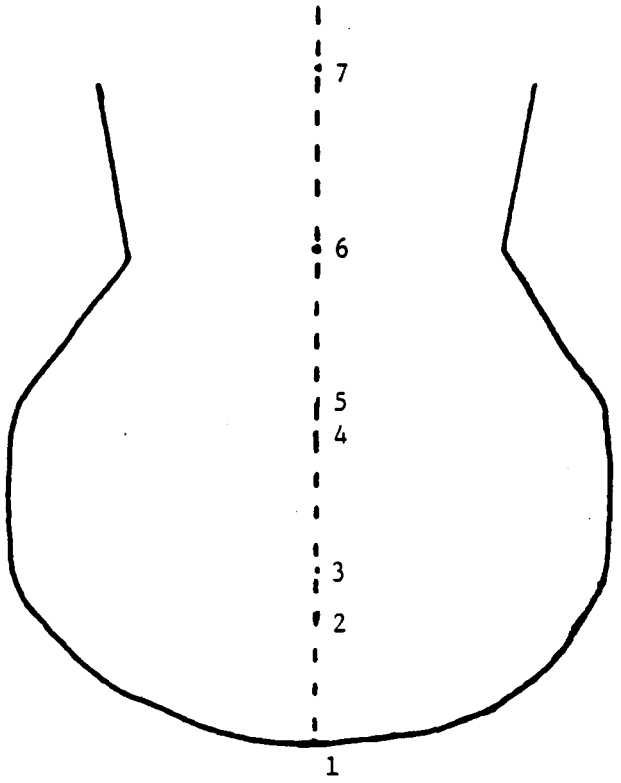


FIGURE 10

BROACH - PARAMETRIC MODEL



Arc Radius = 0, Connect by Straight Line with Prior Point

Arc Radius < 0, Connect by Circular Arc with Prior Point

FIGURE 11

BROACH - GEOMETRIC MODEL

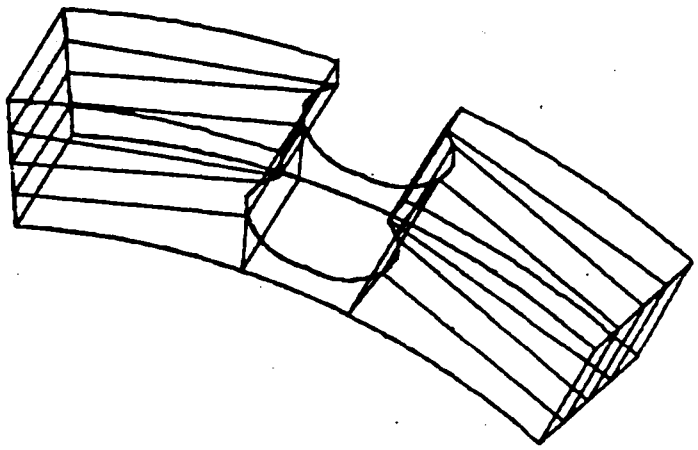


FIGURE 12

BROACH - DISCRETE MODEL

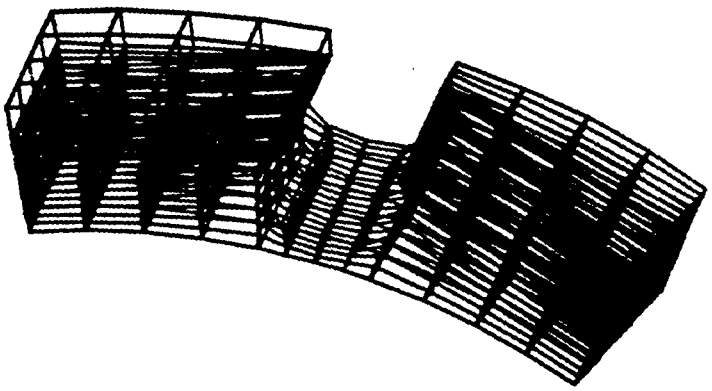


FIGURE 13

DOVETAIL/PLATFORM - PARAMETRIC MODEL

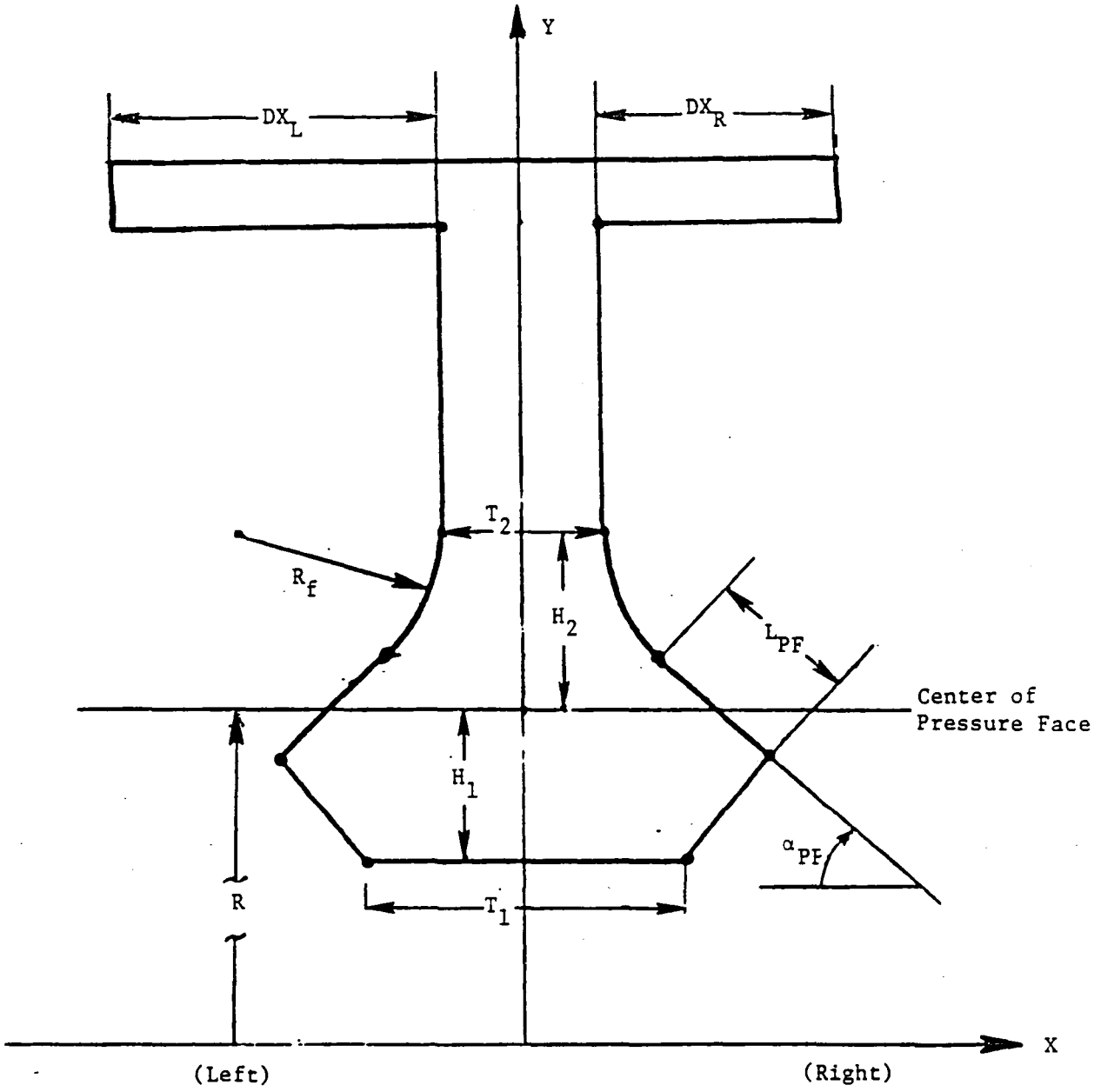


FIGURE 14

DOVETAIL/PLATFORM - PARAMETRIC MODEL

R	RADIUS TO CENTER OF PRESSURE FACE
L_{PF}	LENGTH OF PRESSURE FACE
α_{PF}	ANGLE OF PRESSURE FACE
H_1	DISTANCE TO DOVETAIL BOTTOM
T_1	THICKNESS AT DOVETAIL BOTTOM
H_2	DISTANCE TO MIN NECK
T_2	THICKNESS AT MIN NECK
R_F	FILLET RADIUS FROM PRESSURE FACE TO MIN NECK
DX_R	PLATFORM WIDTH (RIGHT)
DX_L	PLATFORM WIDTH (LEFT)

FIGURE 15

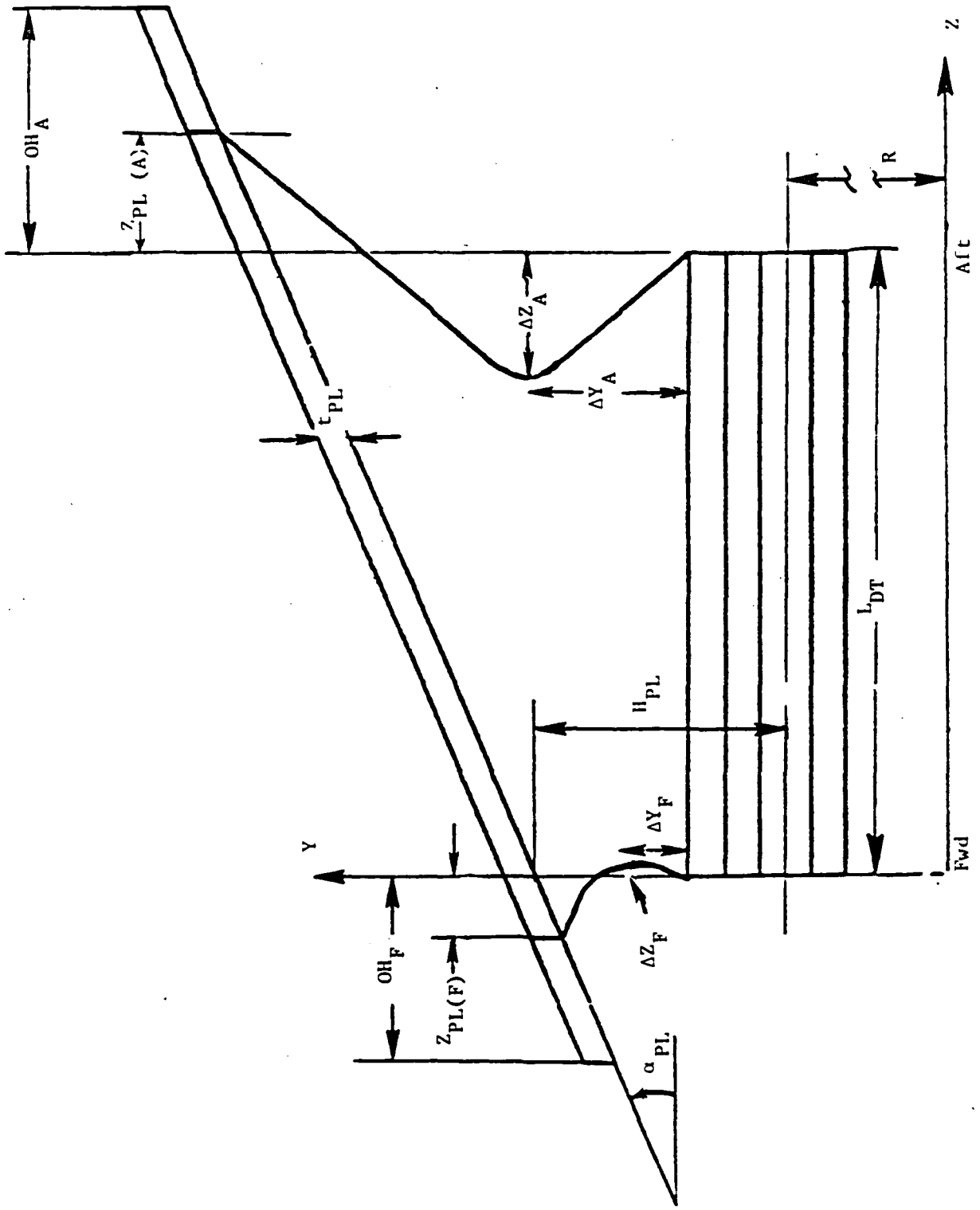


FIGURE 16

DOVETAIL/PLATFORM - PARAMETRIC MODEL

T_{PL}	THICKNESS OF PLATFORM
H_{PL}	HEIGHT OF PLATFORM AT Z=0.
α_{PL}	ANGLE OF PLATFORM
OH_F	PLATFORM OVERHANG (FWD)
OH_A	PLATFORM OVERHANG (AFT)
L_{DT}	LENGTH OF DOVETAIL (AXIAL)
$Z_{PL(F)}$	OFFSET FOR FORWARD PLATFORM DOVETAIL INTERSECTION
$Z_{PL(A)}$	OFFSET FOR AFT PLATFORM DOVETAIL INTERSECTION
ΔY_A	AXIAL OFFSET FOR AFT DOVETAIL CUTOUT
ΔZ_A	RADIAL OFFSET FOR AFT DOVETAIL CUTOUT
ΔY_F	AXIAL OFFSET FOR FORWARD DOVETAIL CUTOUT
ΔZ_F	RADIAL OFFSET FOR FORWARD DOVETAIL CUTOUT

FIGURE 17

DOVETAIL/PLATFORM - GEOMETRIC MODEL

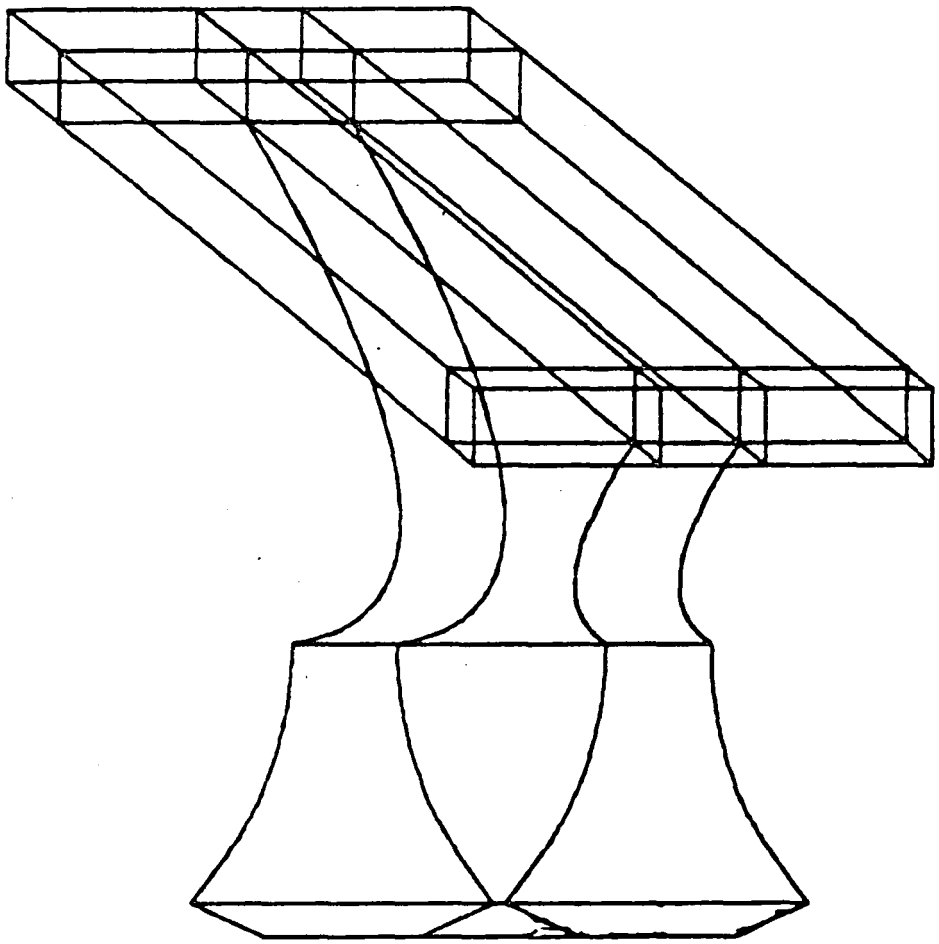


FIGURE 18

DOVETAIL/PLATFORM - DISCRETE MODEL

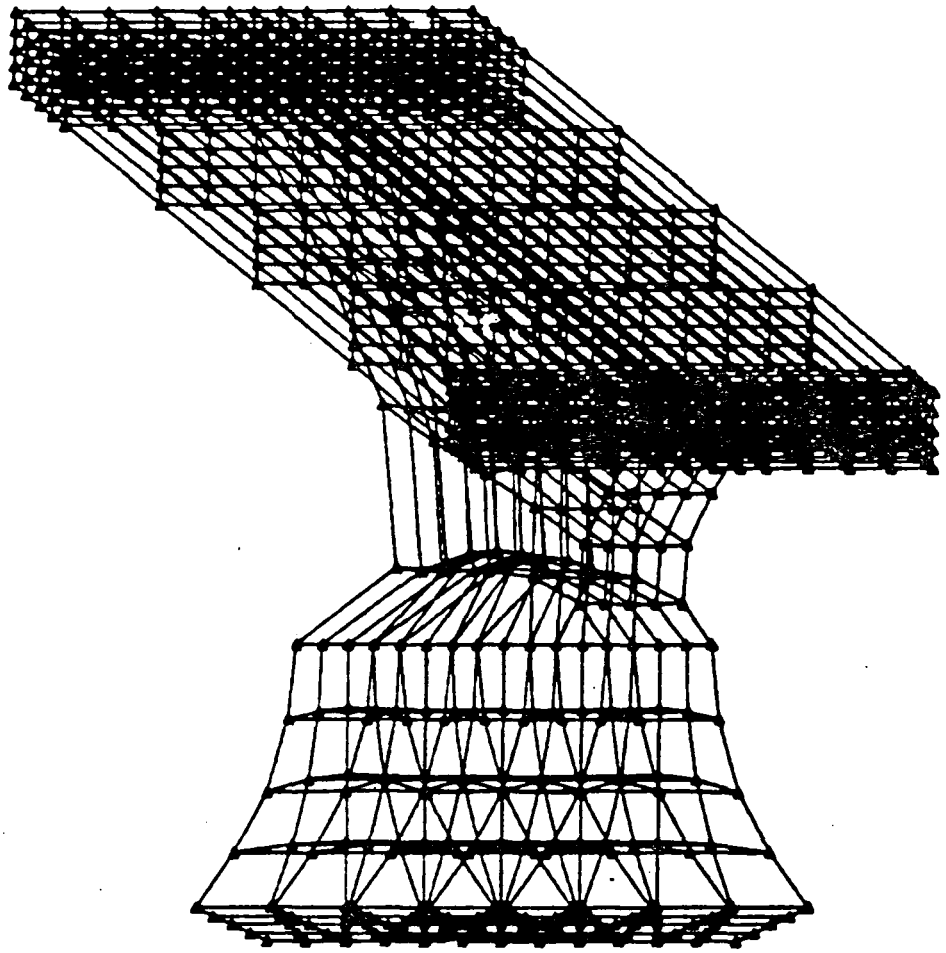


FIGURE 19

AIR COOLED TURBINE BLADE - MODELING REGIONS

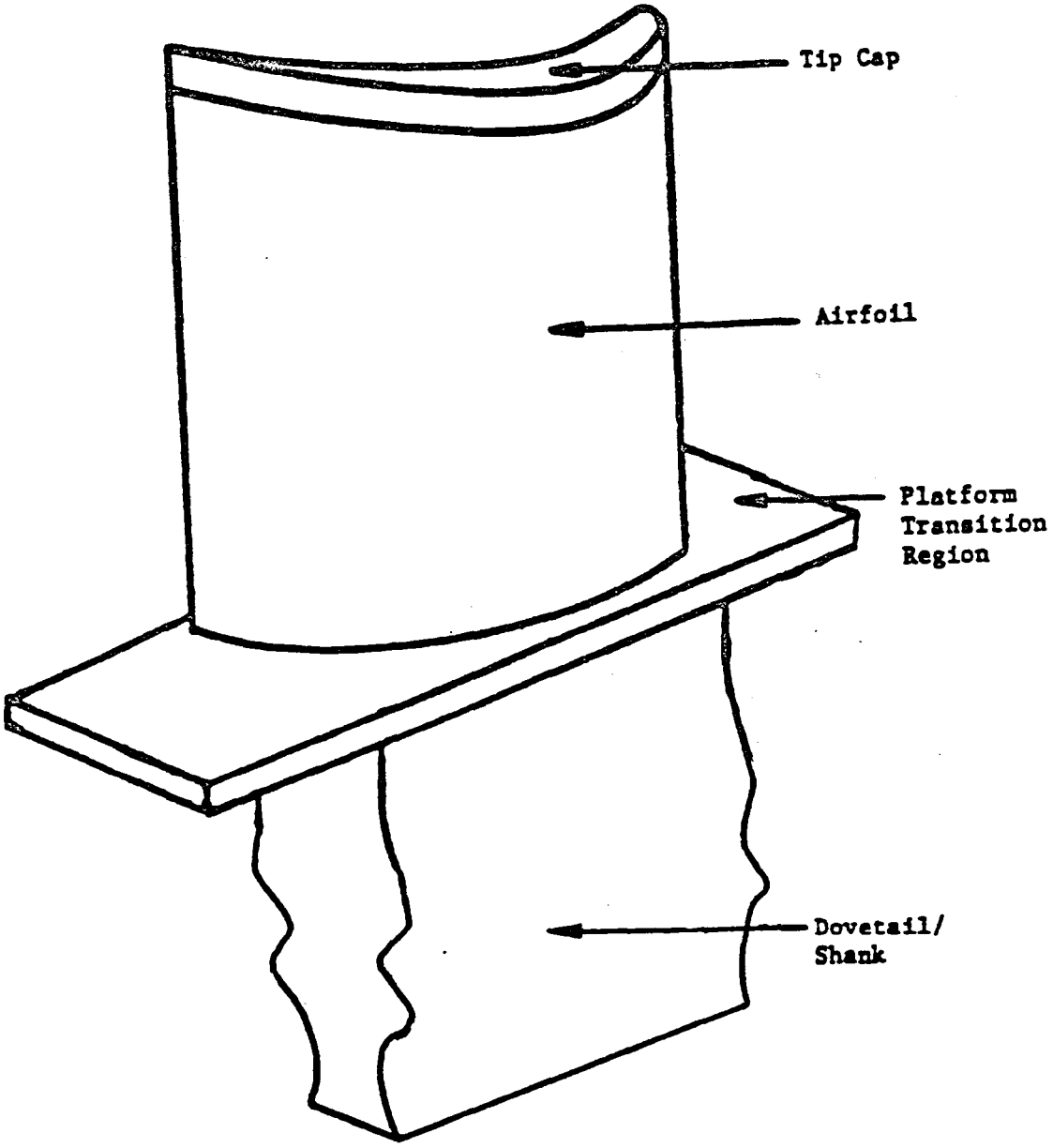
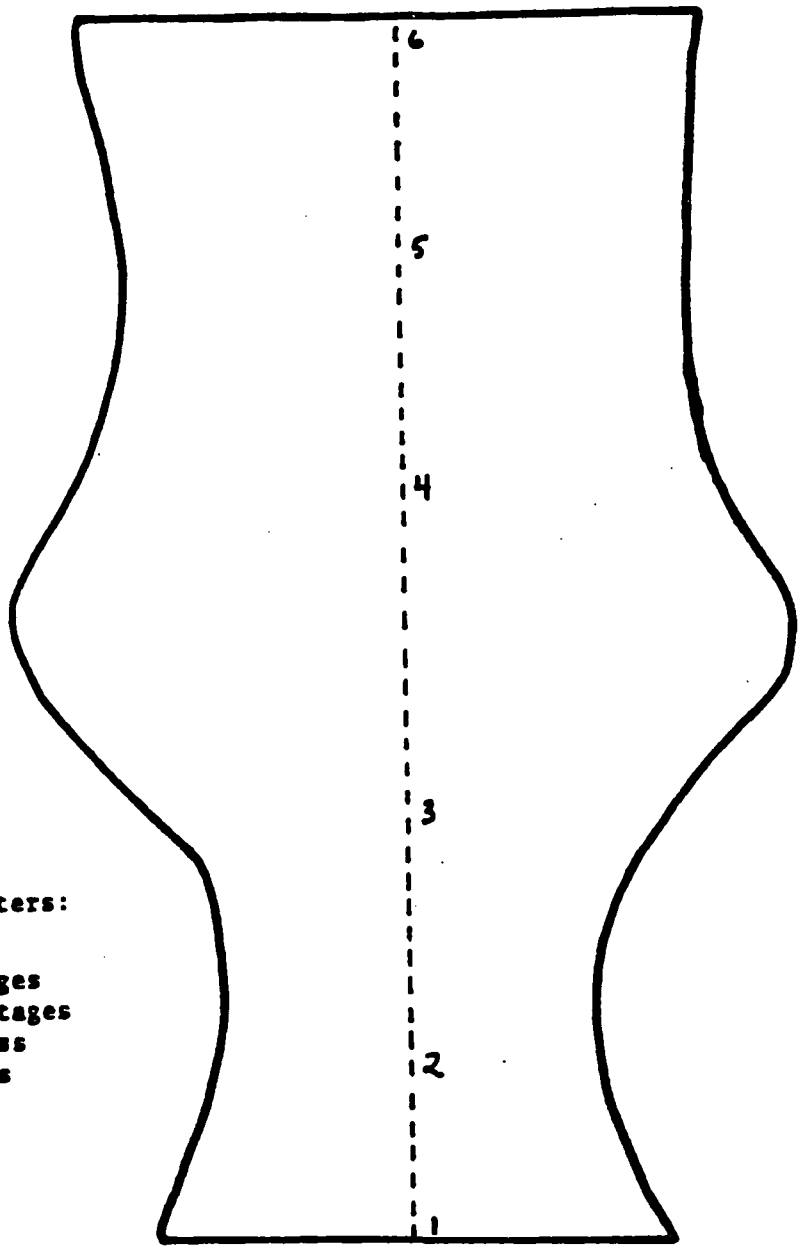


FIGURE 20

AIR COOLED TURBINE BLADE - DOVETAIL



Input Parameters:

Thikness
Width at Stages
Arc Radius Stages
Wall Thickness
Rib Thickness

FIGURE 21

AIR COOLED TURBINE BLADE - DOVETAIL

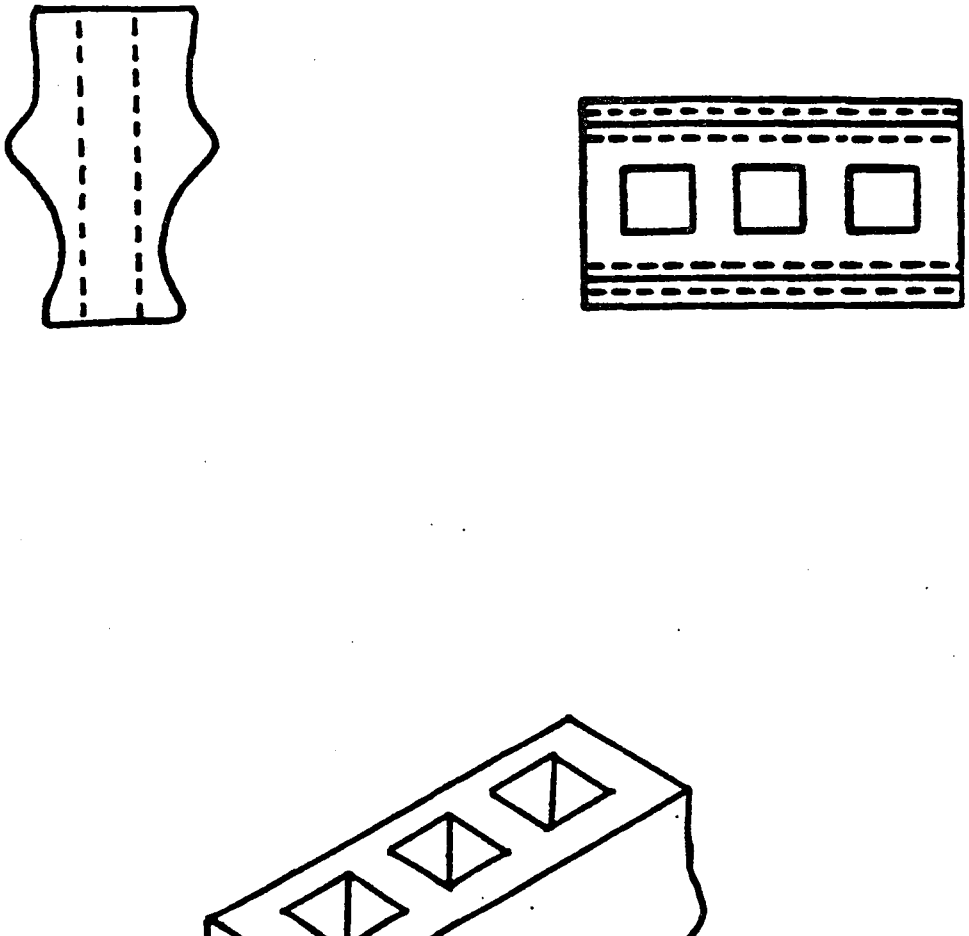


FIGURE 22

AIR COOLED TURBINE BLADE - PLAT FORM (BOTTOM)

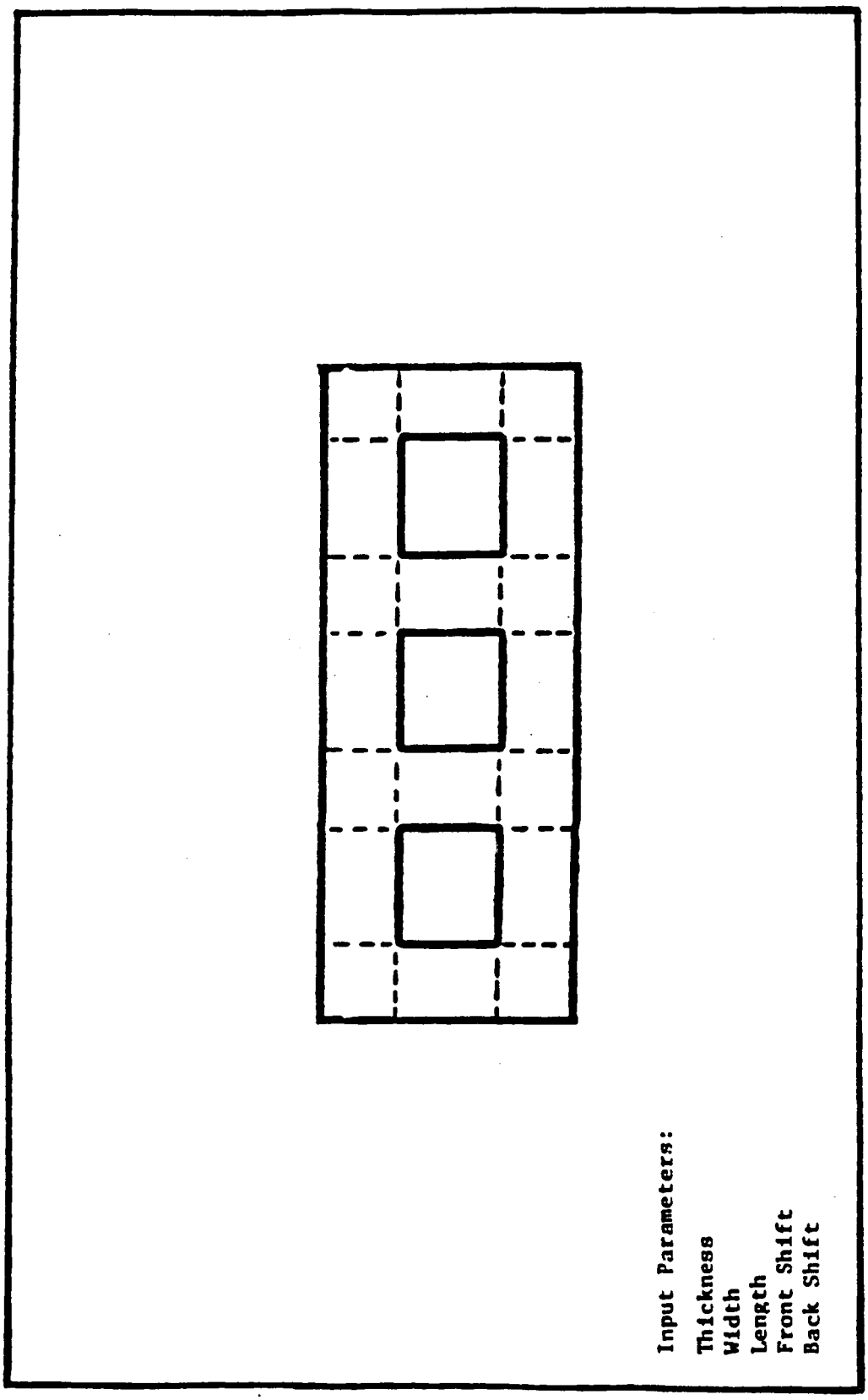


FIGURE 23

AIR COOLED TURBINE BLADE - PLATFORM (TOP)

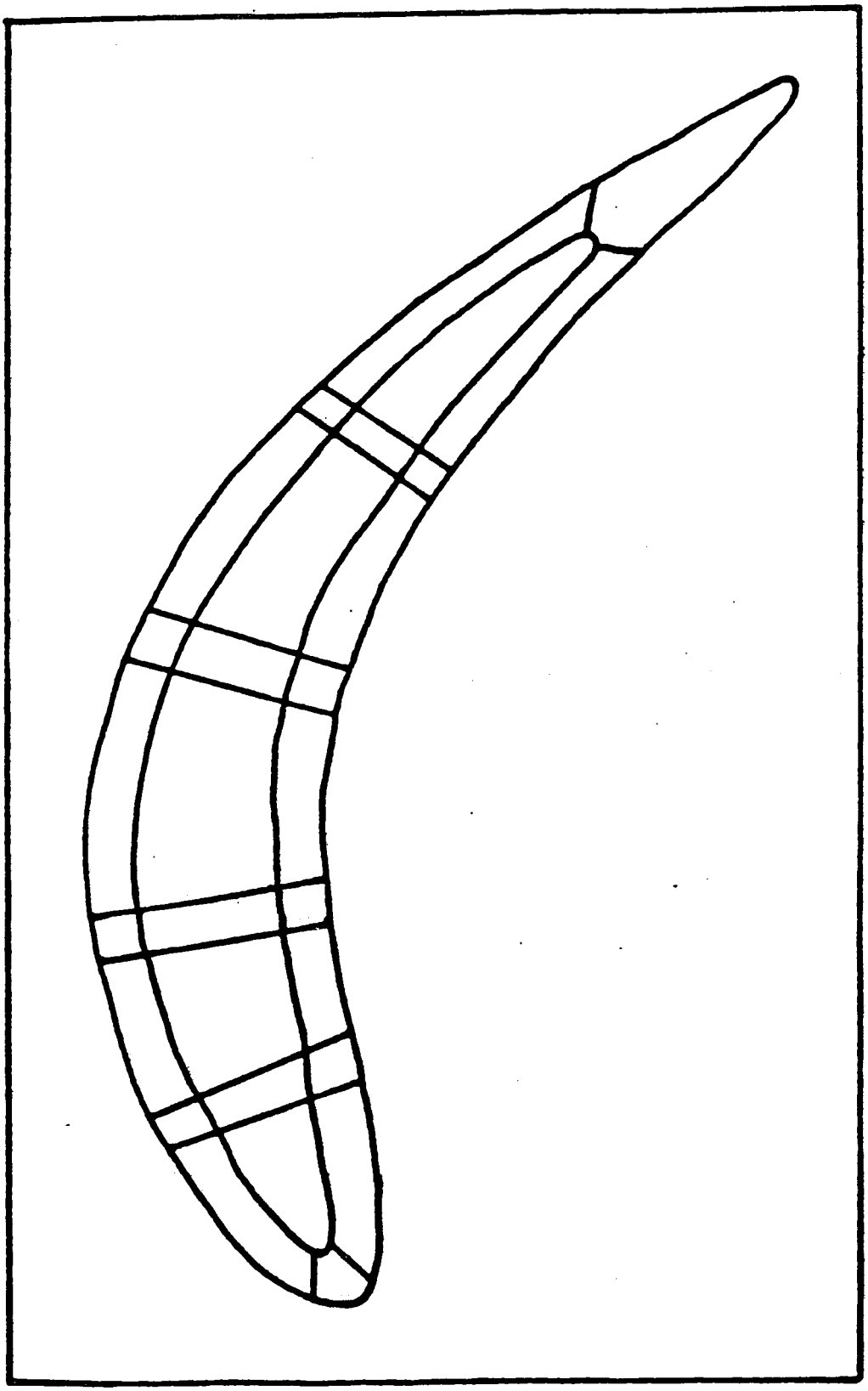


FIGURE 24

AIR COOLED TURBINE BLADE - AIRFOIL

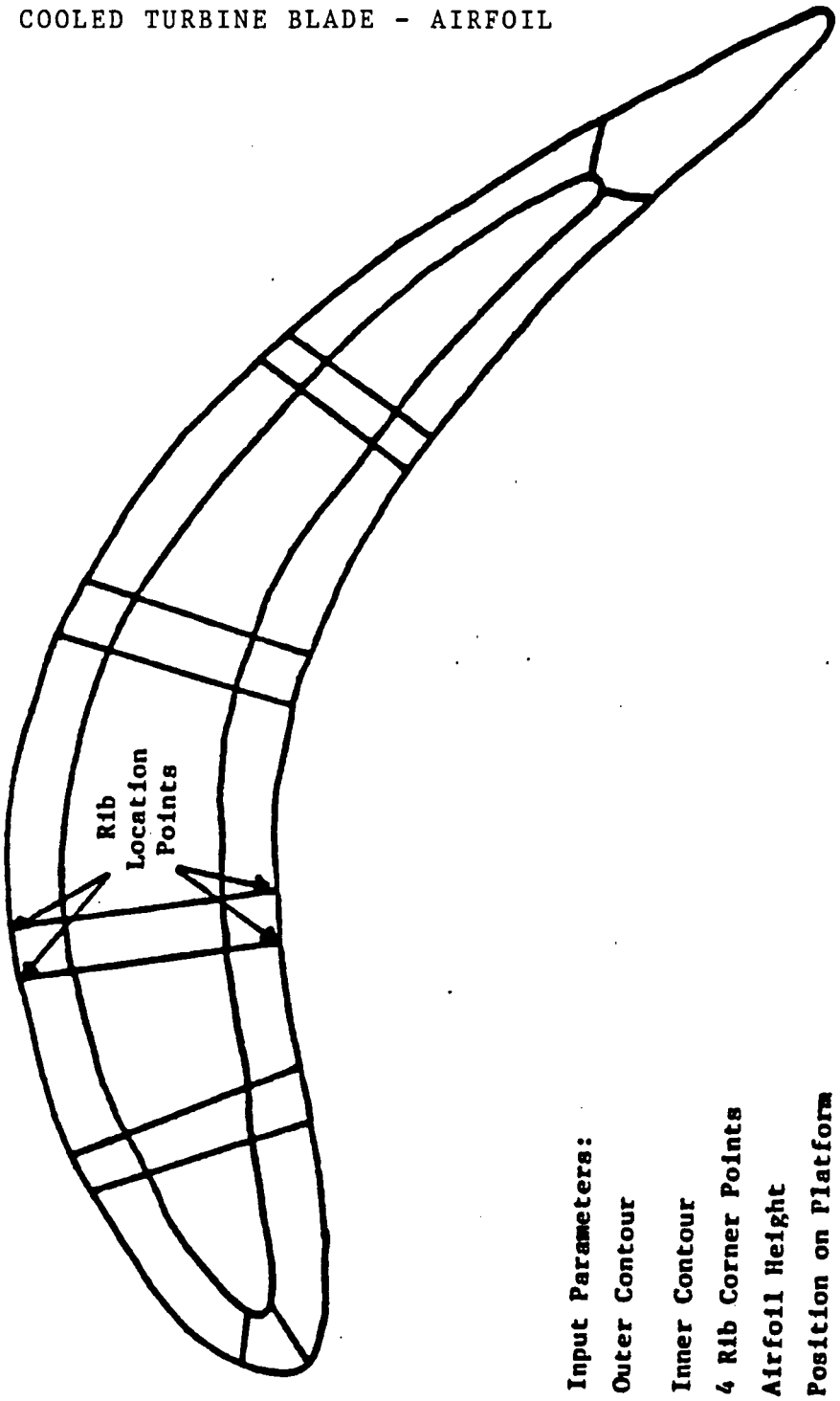
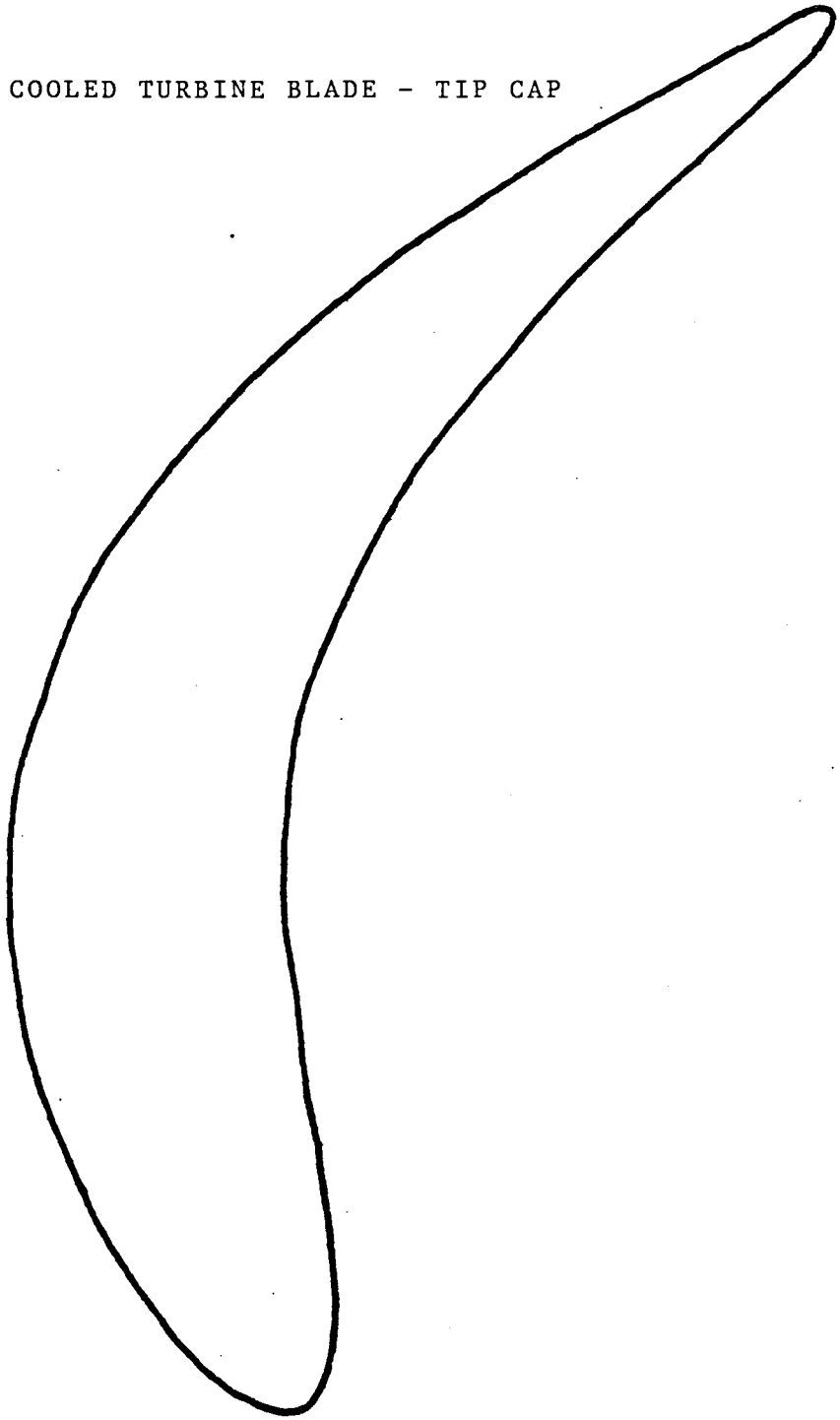


FIGURE 25

AIR COOLED TURBINE BLADE - TIP CAP



Input Parameters:
Thickness

FIGURE 26

AIR COOLED TURBINE BLADE - GEOMETRIC MODEL

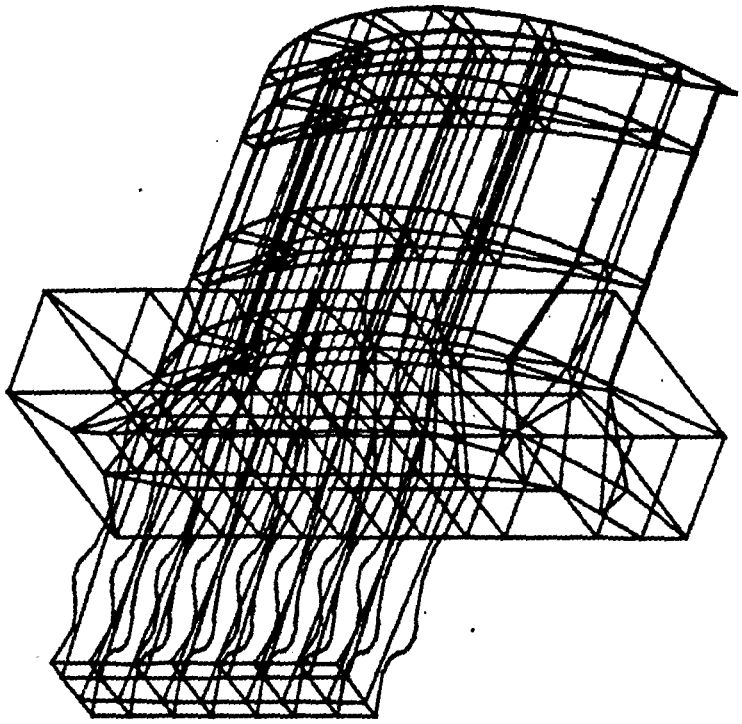


FIGURE 27

AIR COOLED TURBINE BLADE - DISCRETE MODEL

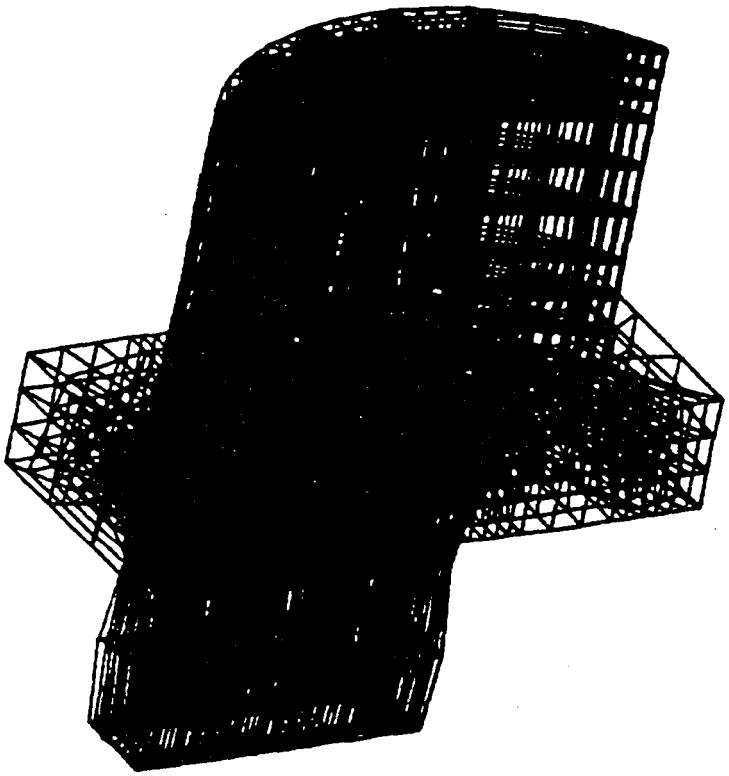
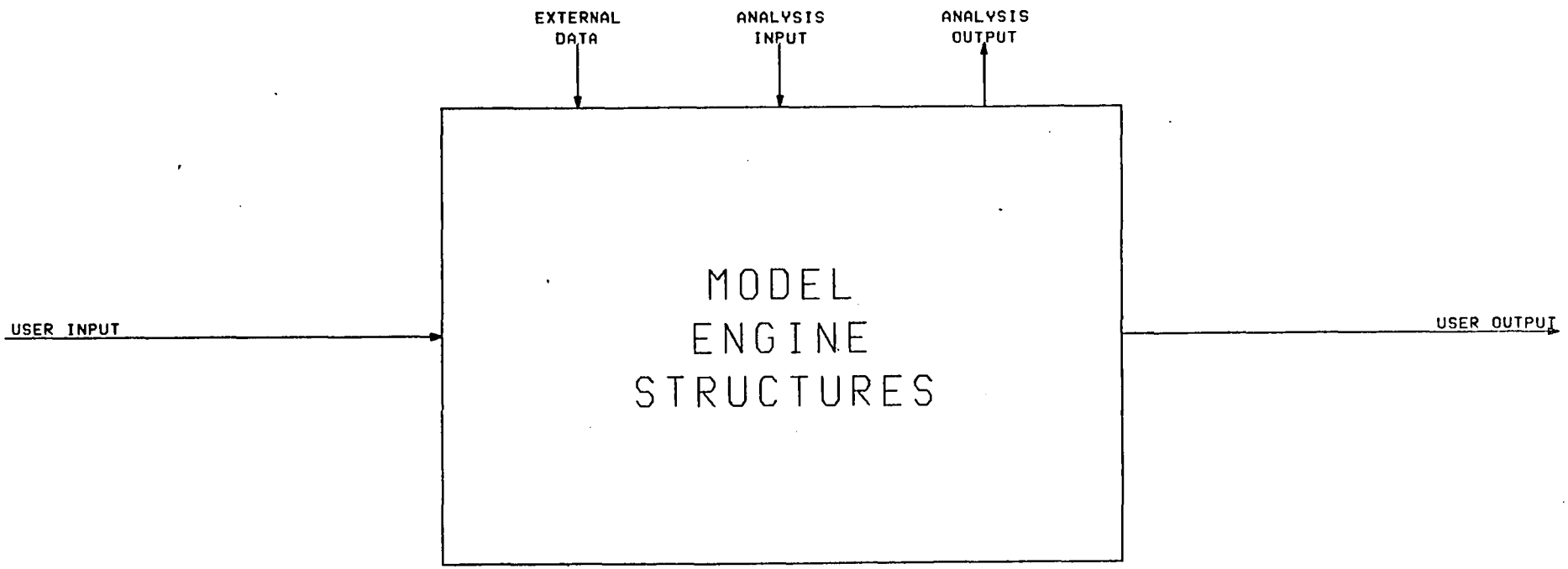


FIGURE 28

APPENDIX A

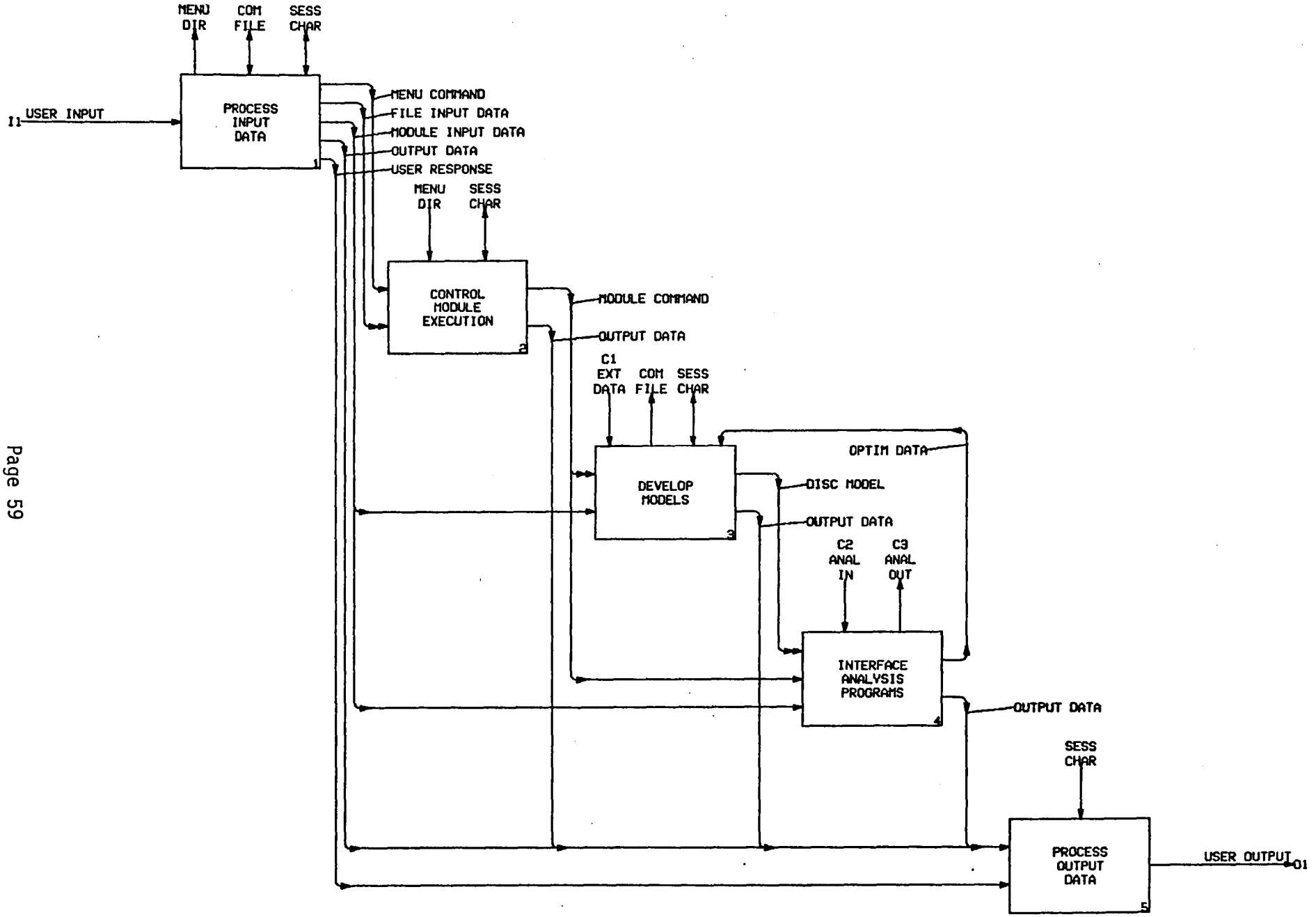
Analysis Diagrams



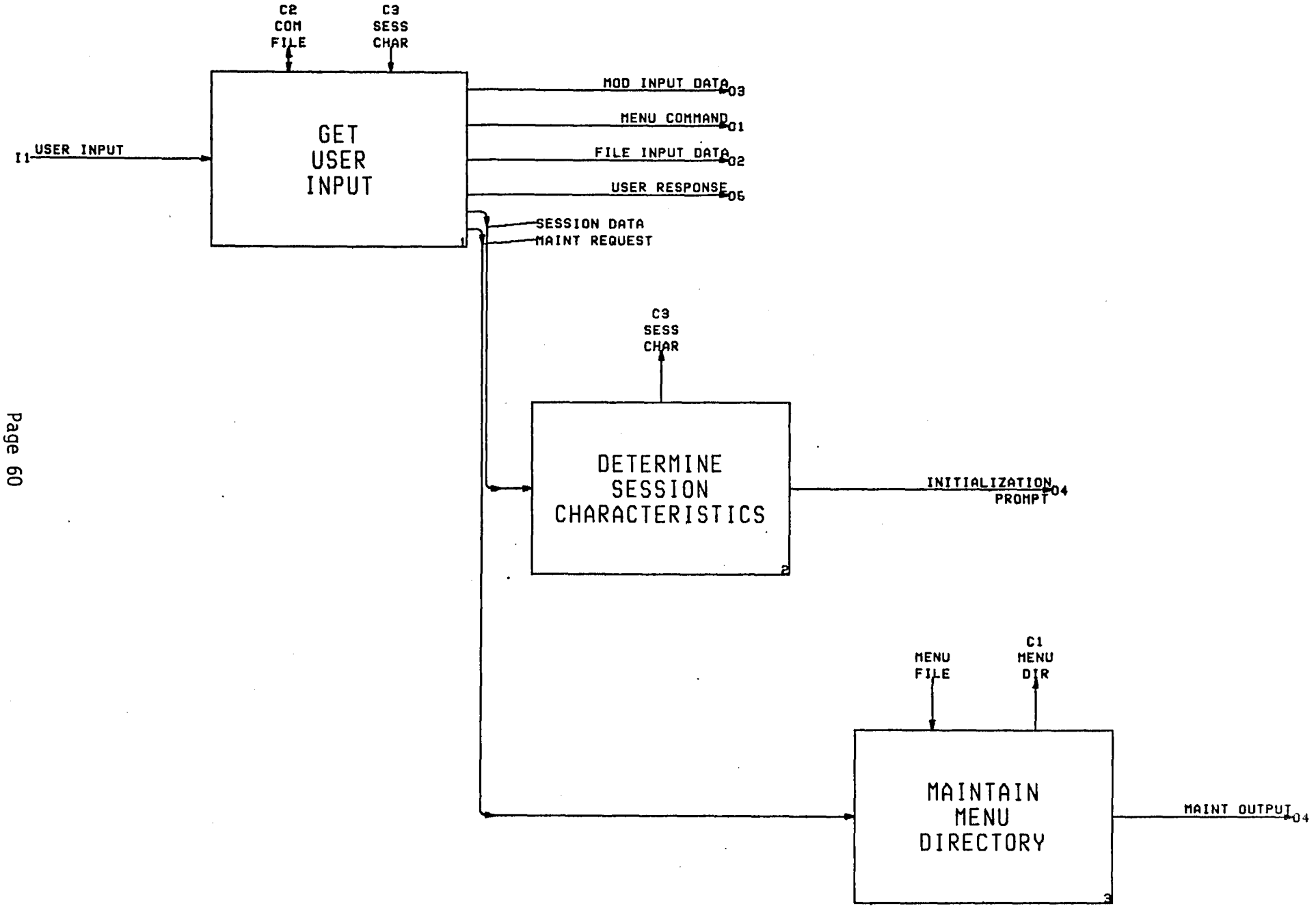
ESMOSS - CONTEXT

A-0

08/17/82



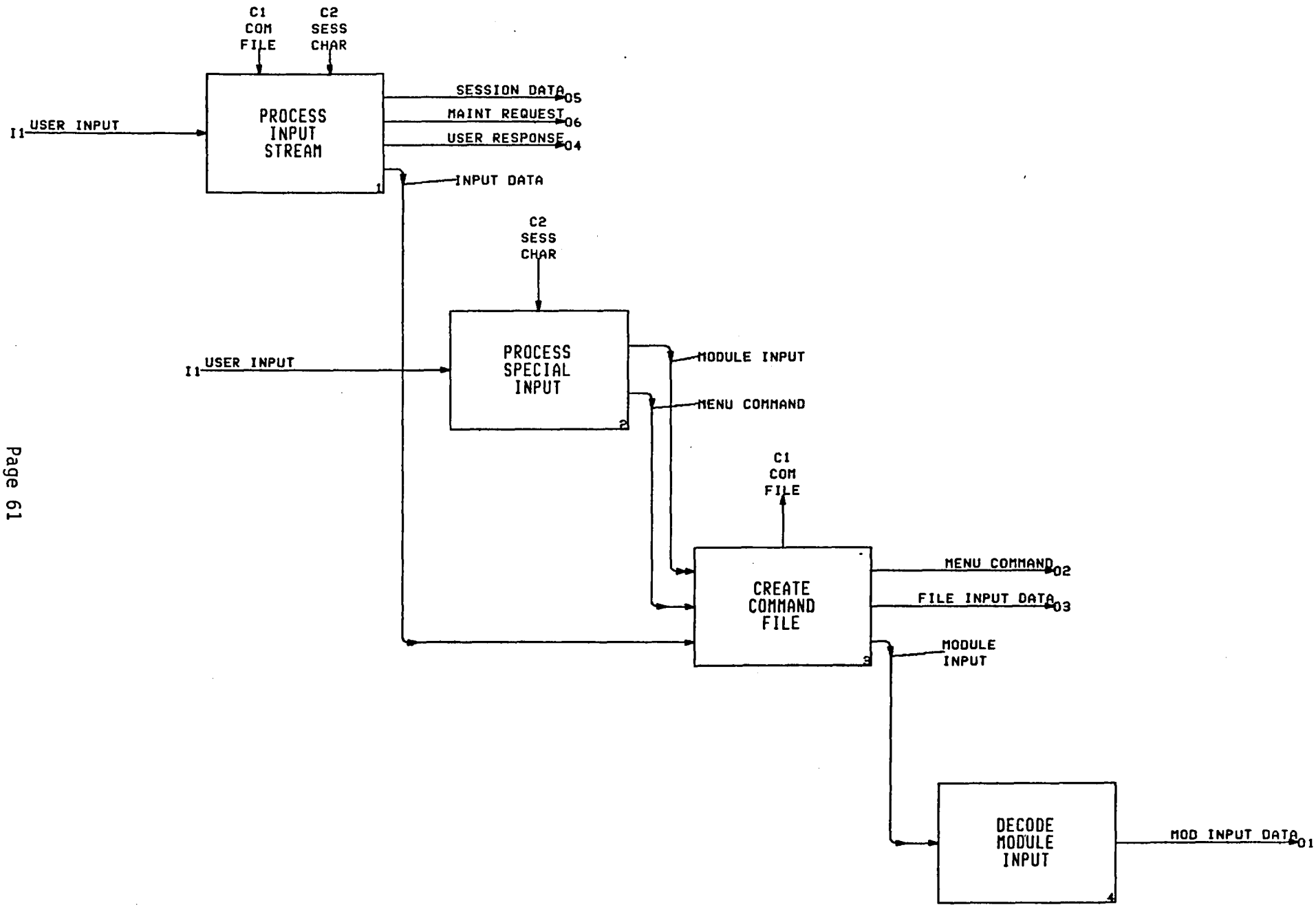
MODEL ENGINE STRUCTURES A0



PROCESS INPUT DATA

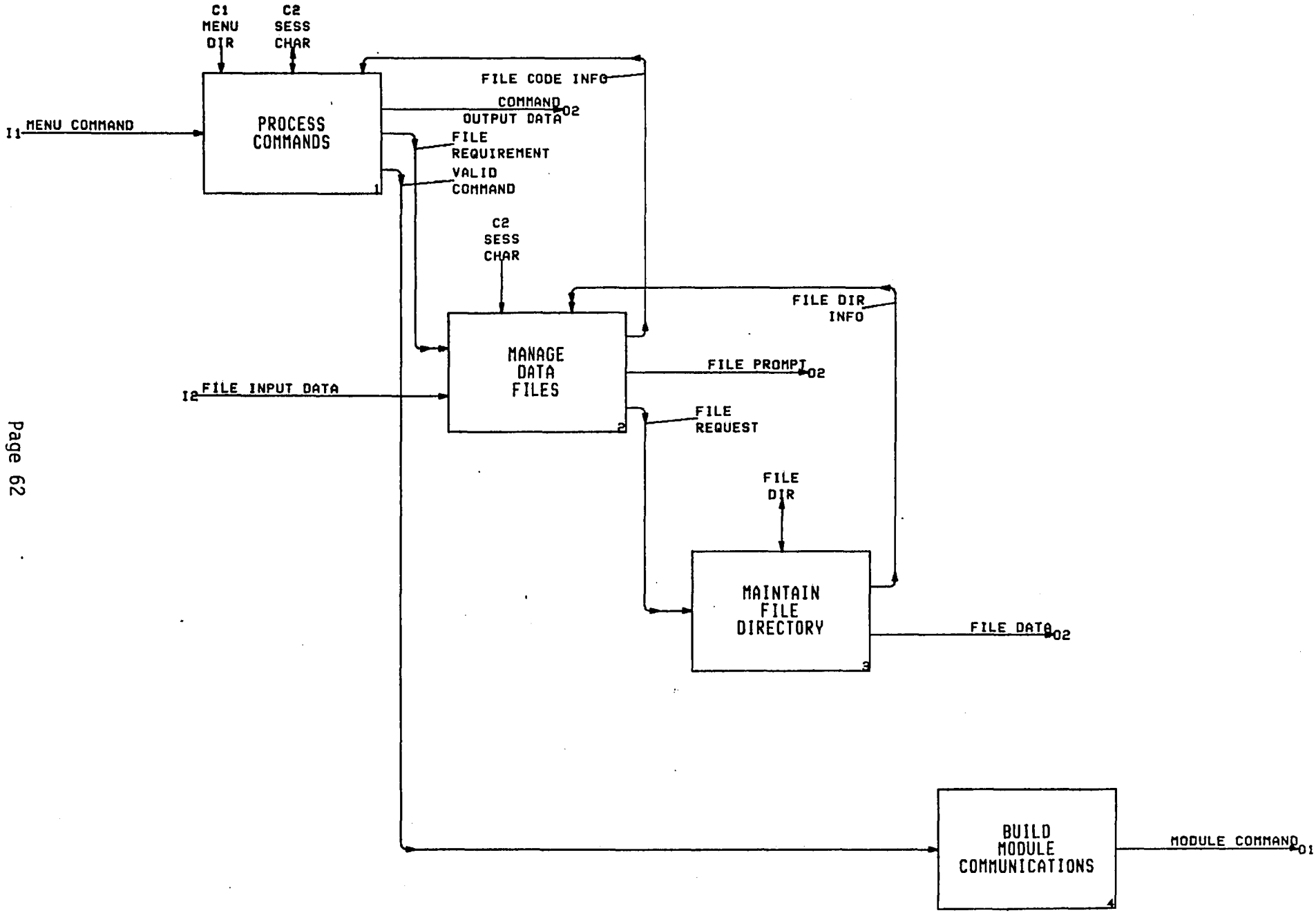
A1

08/17/82



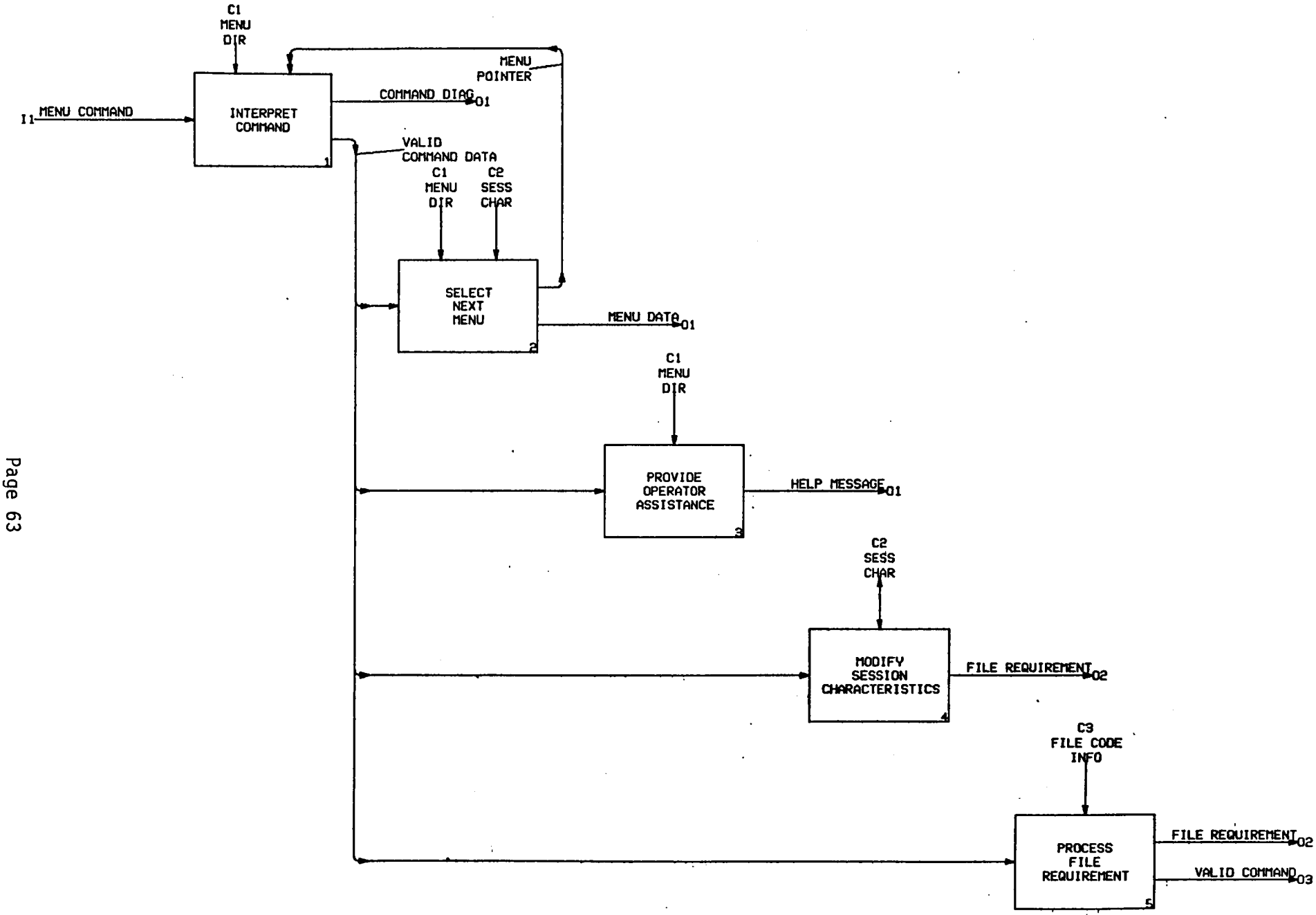
GET USER INPUT

A1.1



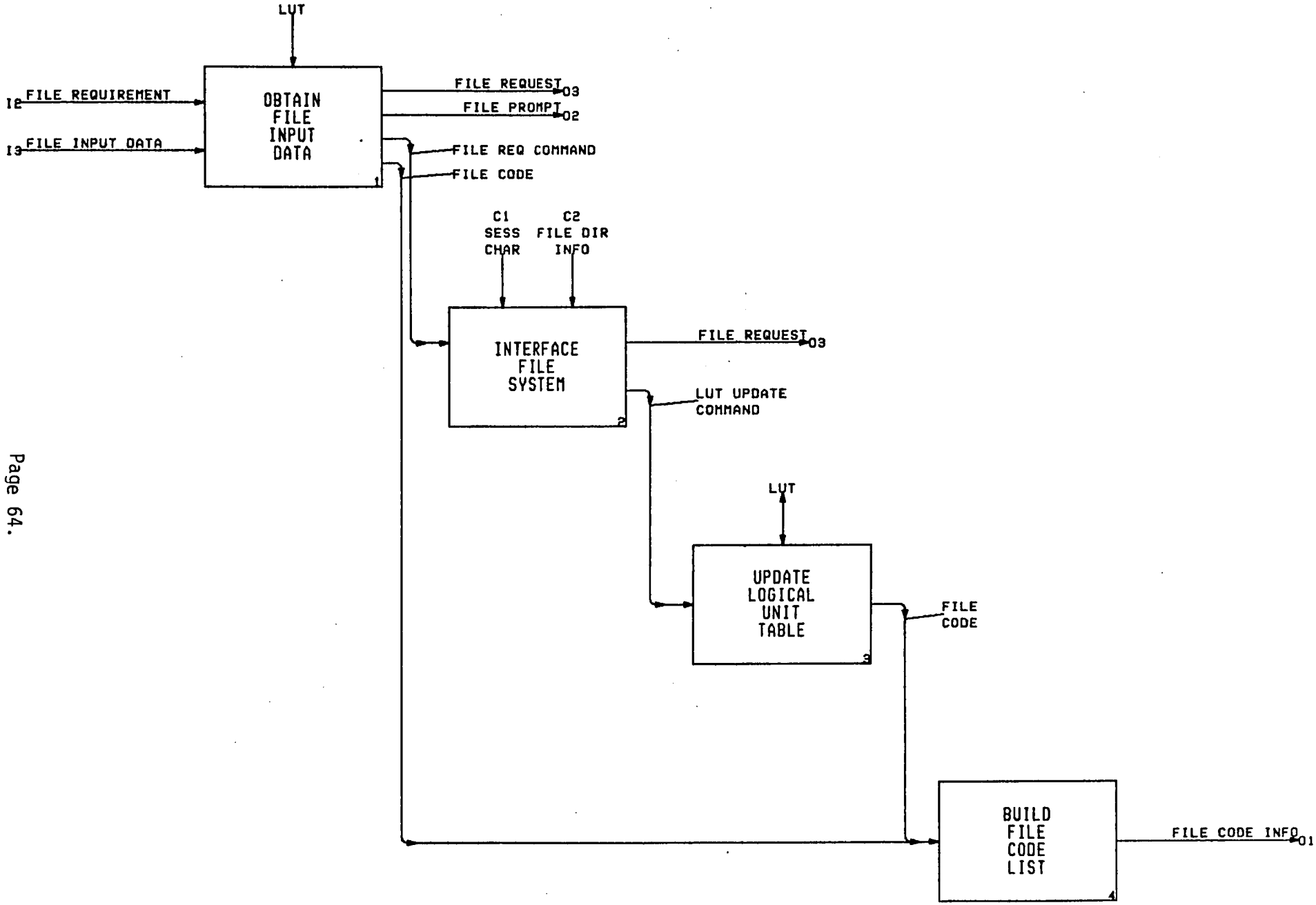
CONTROL MODULE EXECUTION A2

08/19/82



PROCESS COMMANDS

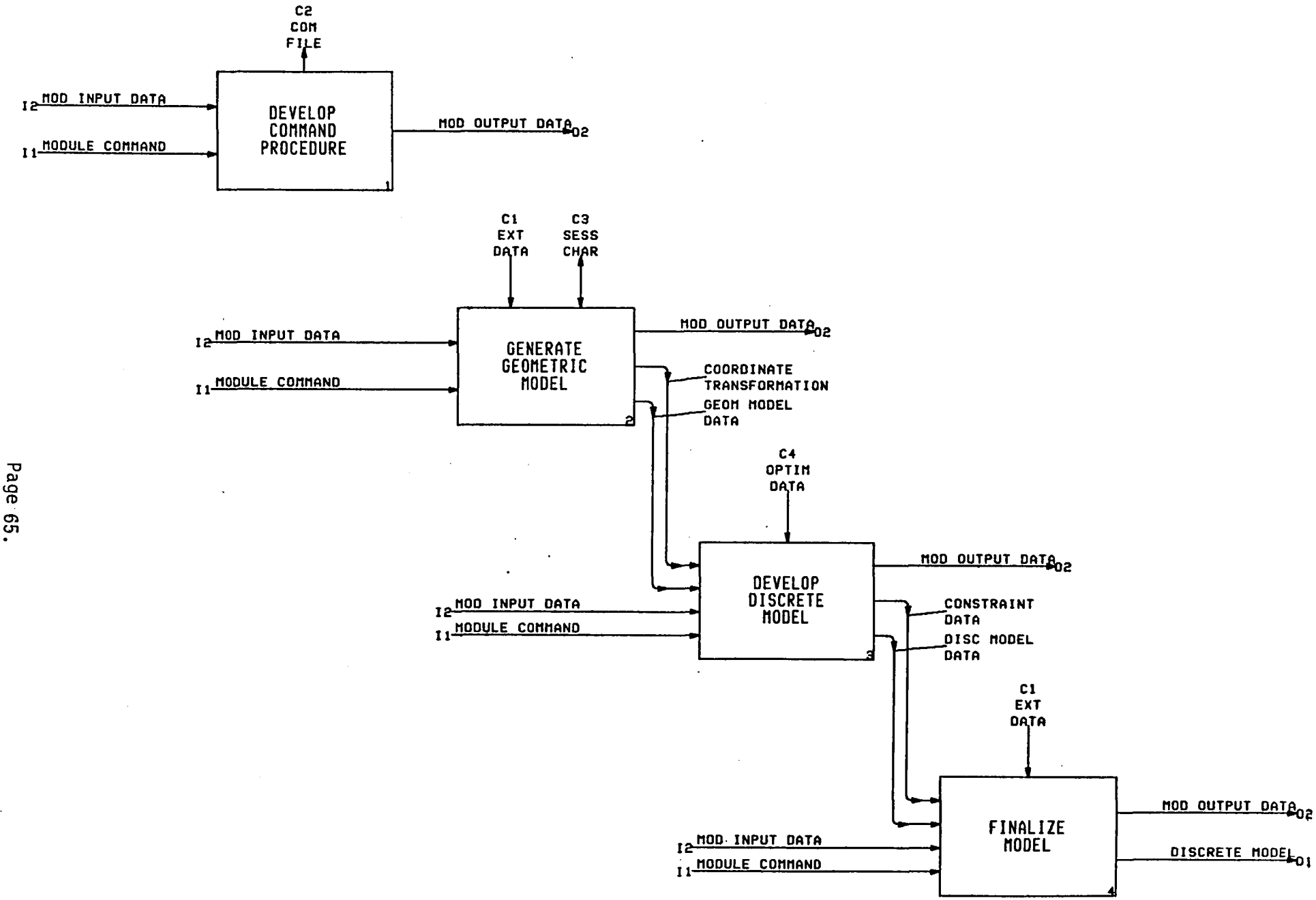
A2.1



MANAGE DATA FILES

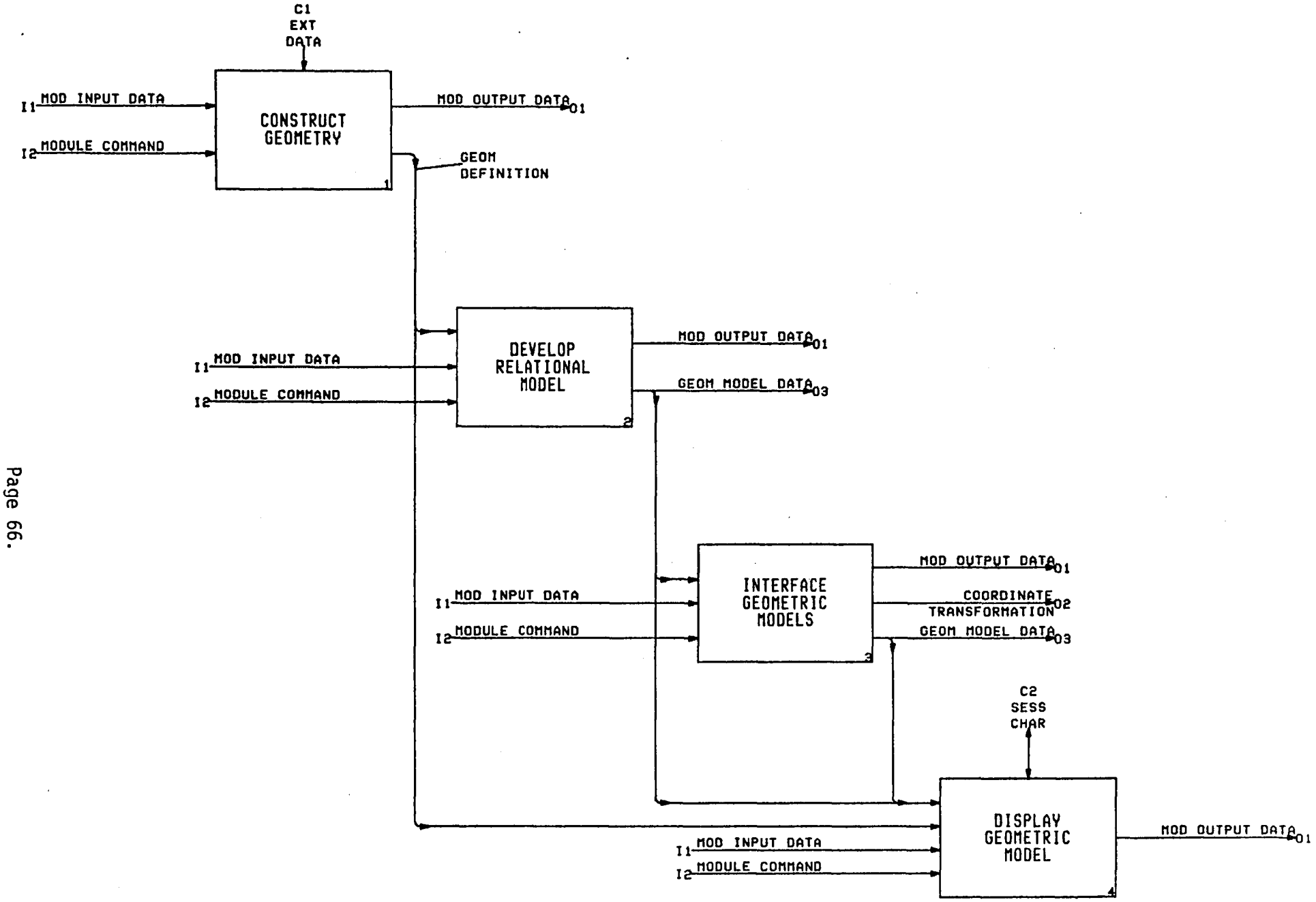
A2.2

08/19/82



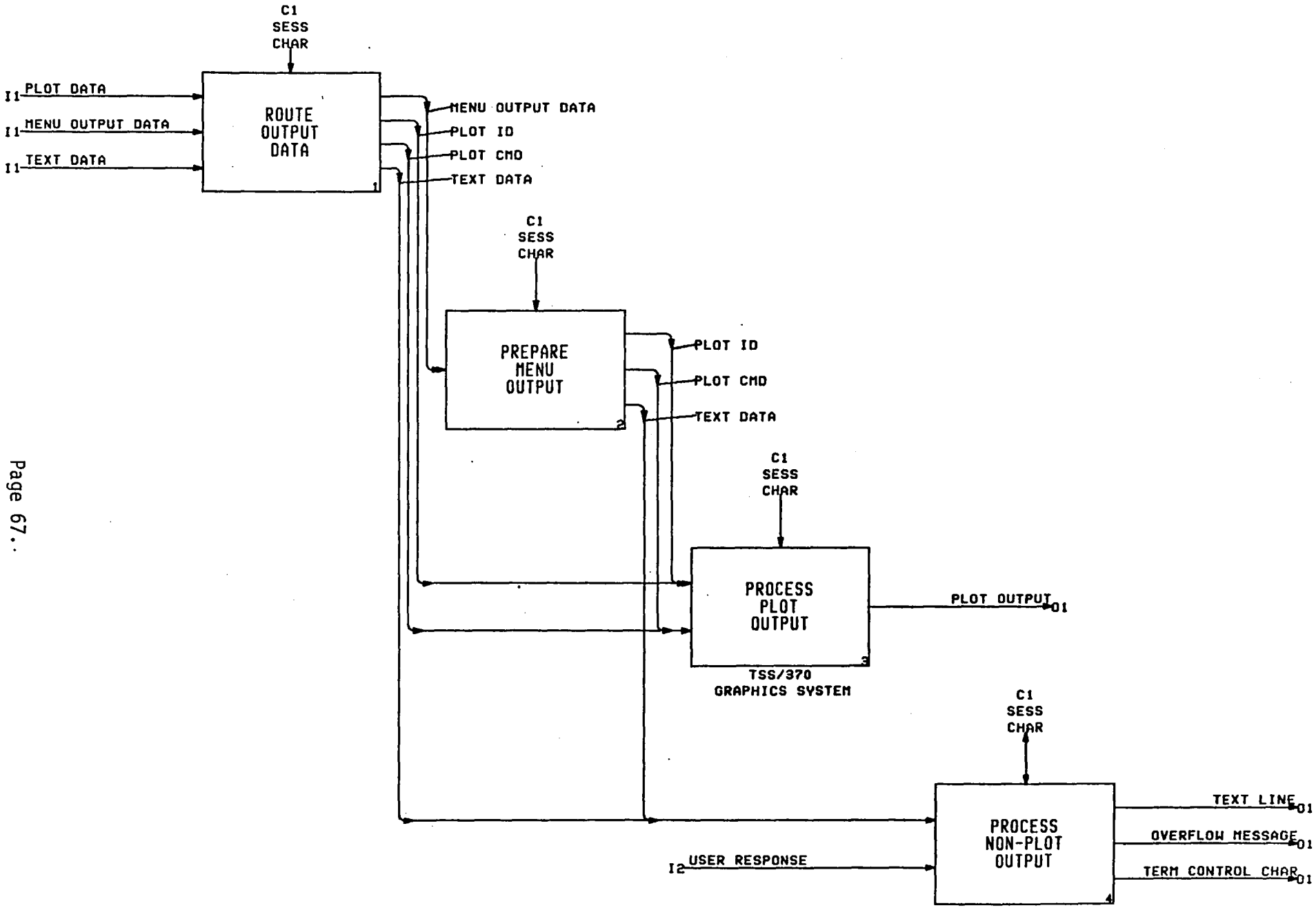
DEVELOP MODELS

A3



GENERATE GEOMETRIC MODEL A3.2

09/13/82



PROCESS OUTPUT DATA

APPENDIX B

Structure Charts

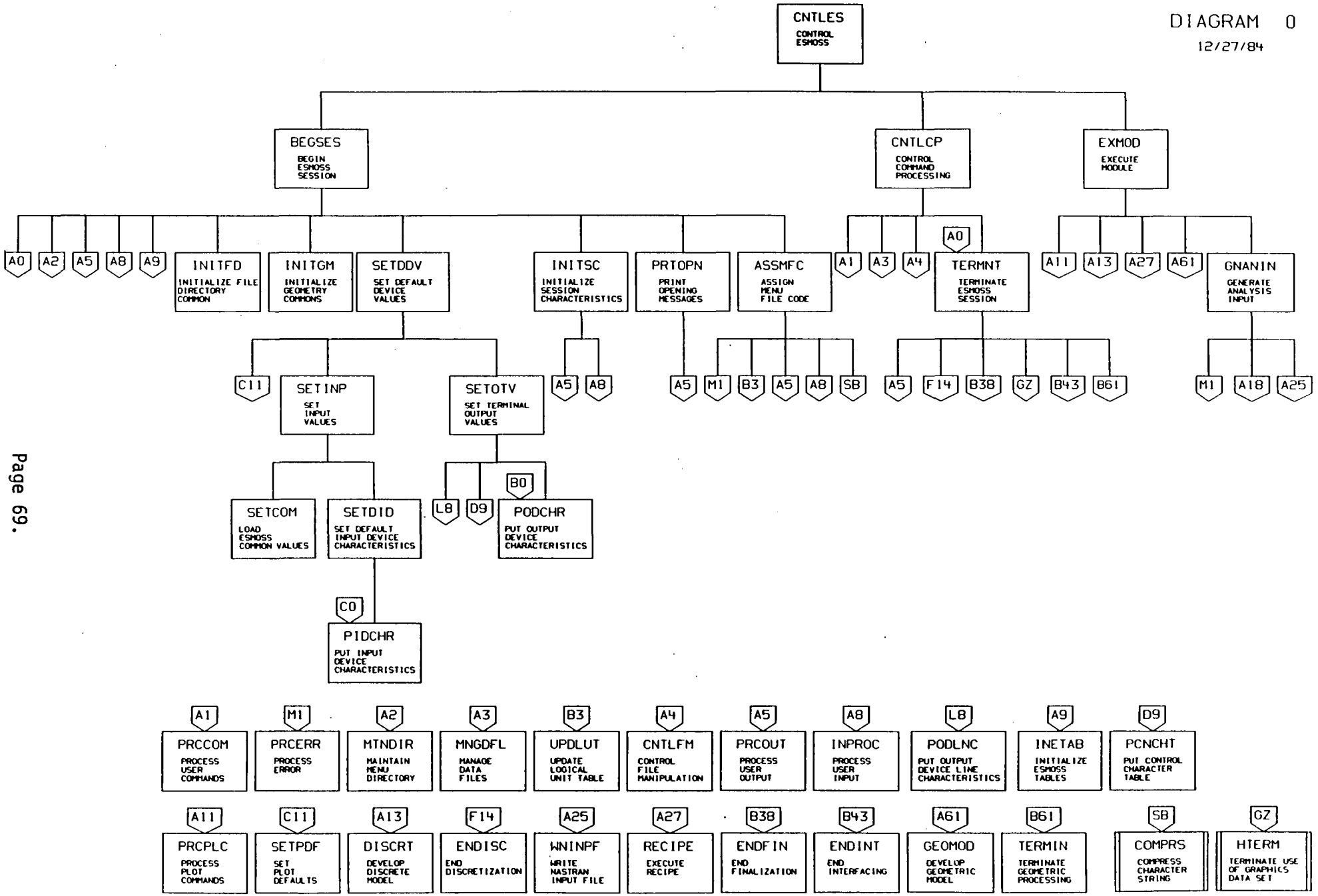


DIAGRAM 0
12/27/84
CNTLES
CONTROL ESMOSS
BEGSES
BEGIN ESMOSS SESSION
CNTLCP
CONTROL COMMAND PROCESSING
EXMOD
EXECUTE MODULE
A0
A2
A5
A8
A9
INITFD
INITIALIZE FILE DIRECTORY COMMON
INITGM
INITIALIZE GEOMETRY COMMONS
SETDDV
SET DEFAULT DEVICE VALUES
INITSC
INITIALIZE SESSION CHARACTERISTICS
PRTOPN
PRINT OPENING MESSAGES
ASSMFC
ASSIGN MENU FILE CODE
A1
A3
A4
A0
TERMNT
TERMINATE ESMOSS SESSION
A11
A13
A27
A61
GNANIN
GENERATE ANALYSIS INPUT
C11
SETINP
SET INPUT VALUES
SETOTV
SET TERMINAL OUTPUT VALUES
A5
A8
A5
M1
B3
A5
A8
SB
A5
F14
B38
GZ
B43
B61
M1
A18
A25
SETCOM
LOAD ESMOSS COMMON VALUES
SETDID
SET DEFAULT INPUT DEVICE CHARACTERISTICS
L8
D9
B0
PODCHR
PUT OUTPUT DEVICE CHARACTERISTICS
C0
PIDCHR
PUT INPUT DEVICE CHARACTERISTICS
A1
PRCCOM
PROCESS USER COMMANDS
M1
PRCERR
PROCESS ERROR
A2
MTNDIR
MAINTAIN MENU DIRECTORY
A3
MNGDFL
MANAGE DATA FILES
B3
UPDLUT
UPDATE LOGICAL UNIT TABLE
A4
CNTLFM
CONTROL FILE MANIPULATION
A5
PRCOUT
PROCESS USER OUTPUT
A8
INPROC
PROCESS USER INPUT
L8
PODLNC
PUT OUTPUT DEVICE LINE CHARACTERISTICS
A9
INETAB
INITIALIZE ESMOSS TABLES
D9
PCNCHT
PUT CONTROL CHARACTER TABLE
A11
PRCPLC
PROCESS PLOT COMMANDS
C11
SETPDF
SET PLOT DEFAULTS
A13
DISCRT
DEVELOP DISCRETE MODEL
F14
ENDISC
END DISCRETIZATION
A25
WNINPF
WRITE NASTRAN INPUT FILE
A27
RECIPE
EXECUTE RECIPE
B38
ENDFIN
END FINALIZATION
B43
ENDINT
END INTERFACING
A61
GEOMOD
DEVELOP GEOMETRIC MODEL
B61
TERMIN
TERMINATE GEOMETRIC PROCESSING
SB
COMPRS
COMPRESS CHARACTER STRING
GZ
HTERM
TERMINATE USE OF GRAPHICS DATA SET

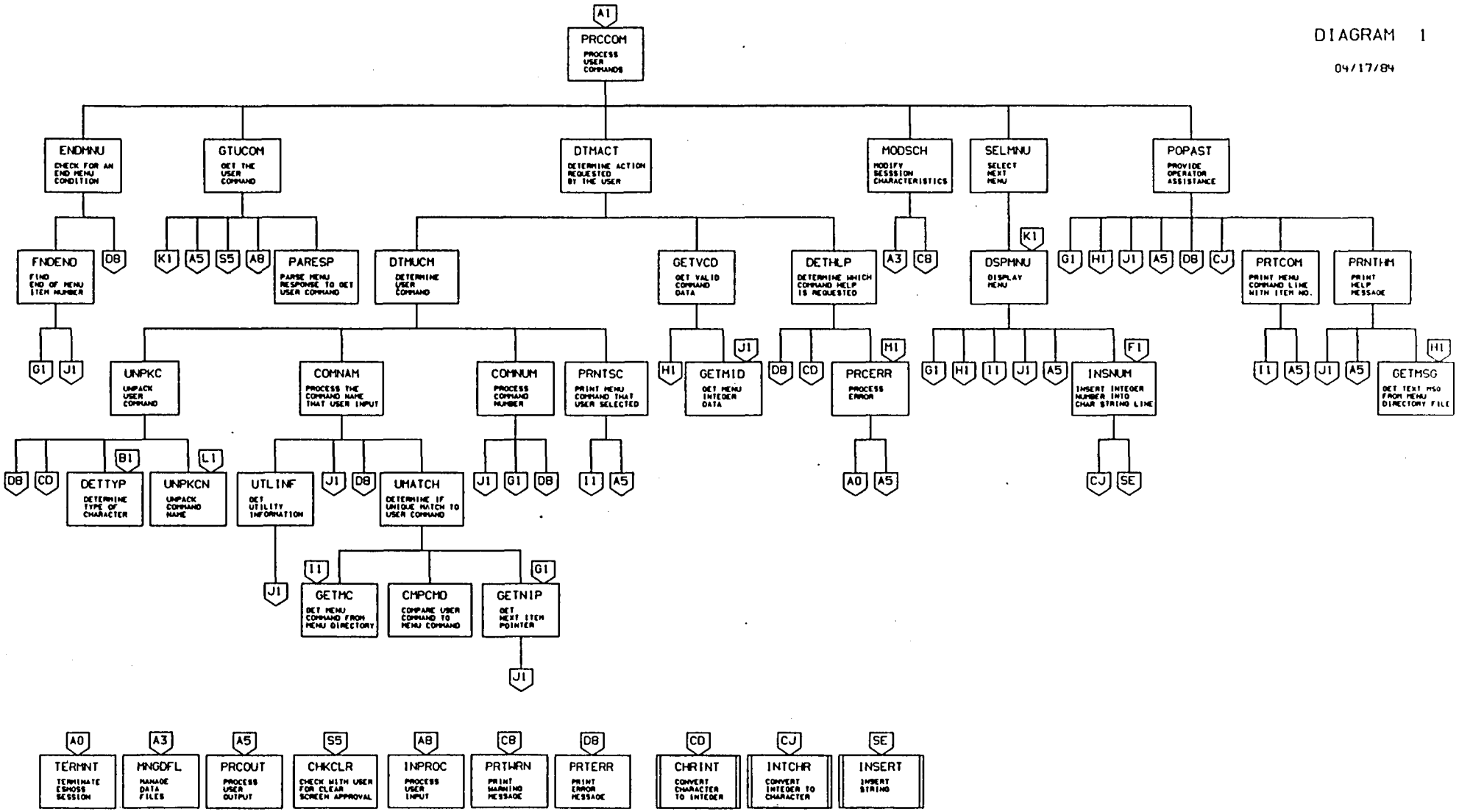
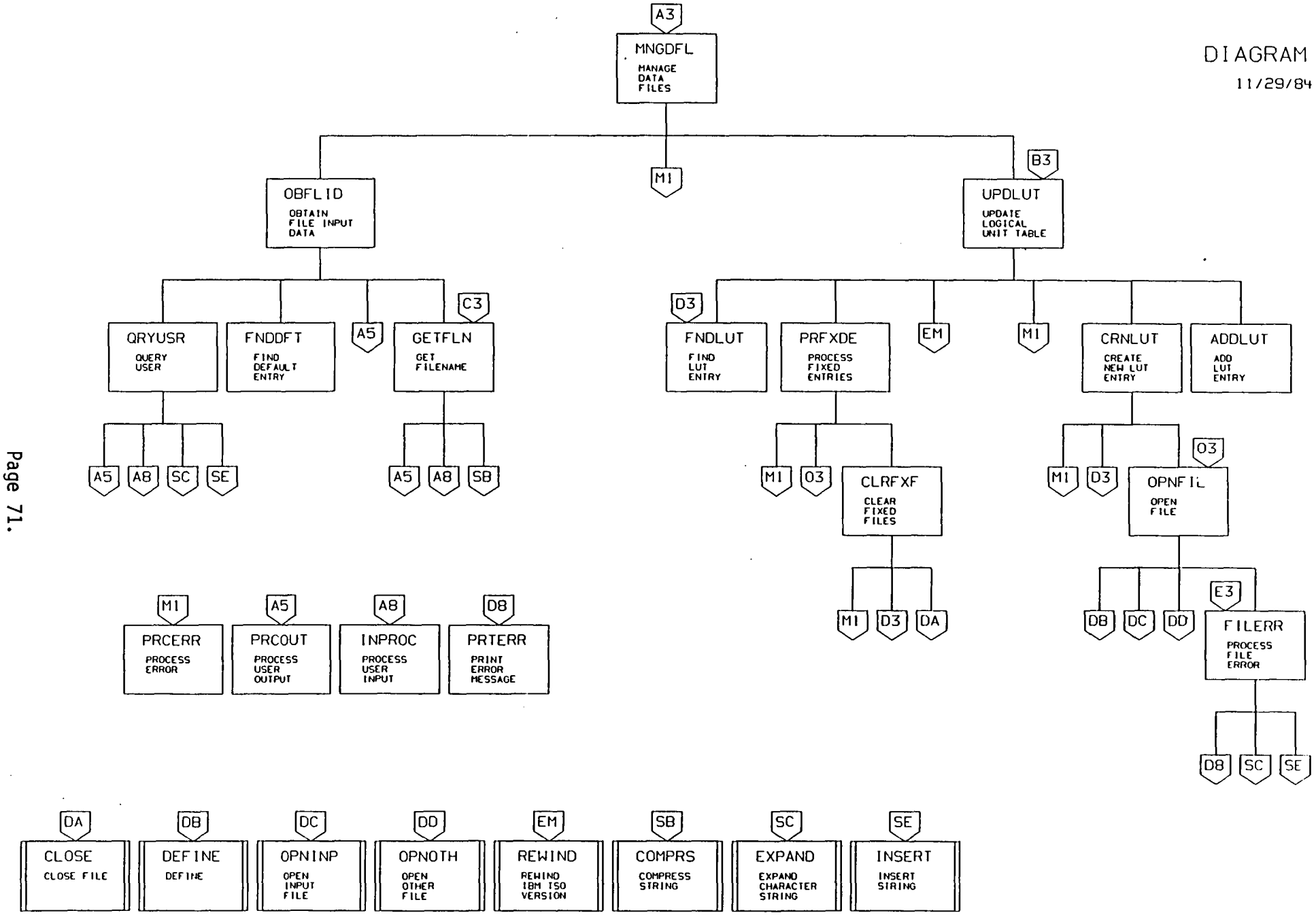

A1
PRCCOM
PROCESS USER COMMANDS
ENDMNU
CHECK FOR AN END MENU CONDITION
GTUCOM
GET THE USER COMMAND
DTMACT
DETERMINE ACTION REQUESTED BY THE USER
MODSCH
MODIFY SESSION CHARACTERISTICS
SELMNU
SELECT NEXT MENU
POPAST
PROVIDE OPERATOR ASSISTANCE
FNDEND
FIND END OF MENU ITEM NUMBER
D8
K1
A5
S5
A8
PARESP
PARSE MENU RESPONSE TO GET USER COMMAND
DTMUCM
DETERMINE USER COMMAND
GETVCD
GET VALID COMMAND DATA
DETHLP
DETERMINE WHICH COMMAND HELP IS REQUESTED
A3
C8
DSPMNU
DISPLAY MENU
G1
H1
J1
CJ
PRTCOM
PRINT MENU COMMAND LINE WITH ITEM NO.
PRNTHM
PRINT HELP MESSAGE
UNPKC
UNPACK USER COMMAND
COMNAM
PROCESS THE COMMAND NAME THAT USER INPUT
COMNUM
PROCESS COMMAND NUMBER
PRNTSC
PRINT MENU COMMAND THAT USER SELECTED
GETMID
GET MENU INTEGER DATA
PRCERR
PROCESS ERROR
M1
I1
F1
INSNUM
INSERT INTEGER NUMBER INTO CHAR STRING LINE
GETMSG
GET TEXT MSG FROM MENU DIRECTORY FILE
H1
C0
B1
L1
DETTYP
DETERMINE TYPE OF CHARACTER
UNPKCN
UNPACK COMMAND NAME
UTLINF
GET UTILITY INFORMATION
UMATCH
DETERMINE IF UNIQUE MATCH TO USER COMMAND
GETMC
GET MENU COMMAND FROM MENU DIRECTORY
CMPCMD
COMPARE USER COMMAND TO MENU COMMAND
GETNIP
GET NEXT ITEM POINTER
A0
TERMNT
TERMINATE EMOSS SESSION
A3
MNGDFL
MANAGE DATA FILES
A5
PRCOUT
PROCESS USER OUTPUT
S5
CHKCLR
CHECK WITH USER FOR CLEAR SCREEN APPROVAL
A8
INPROC
PROCESS USER INPUT
C8
PRTWRN
PRINT WARNING MESSAGE
D8
PRTERR
PRINT ERROR MESSAGE
C0
CHRINT
CONVERT CHARACTER TO INTEGER
CJ
INTCHR
CONVERT INTEGER TO CHARACTER
SE
INSERT
INSERT STRING

DIAGRAM 3

11/29/84



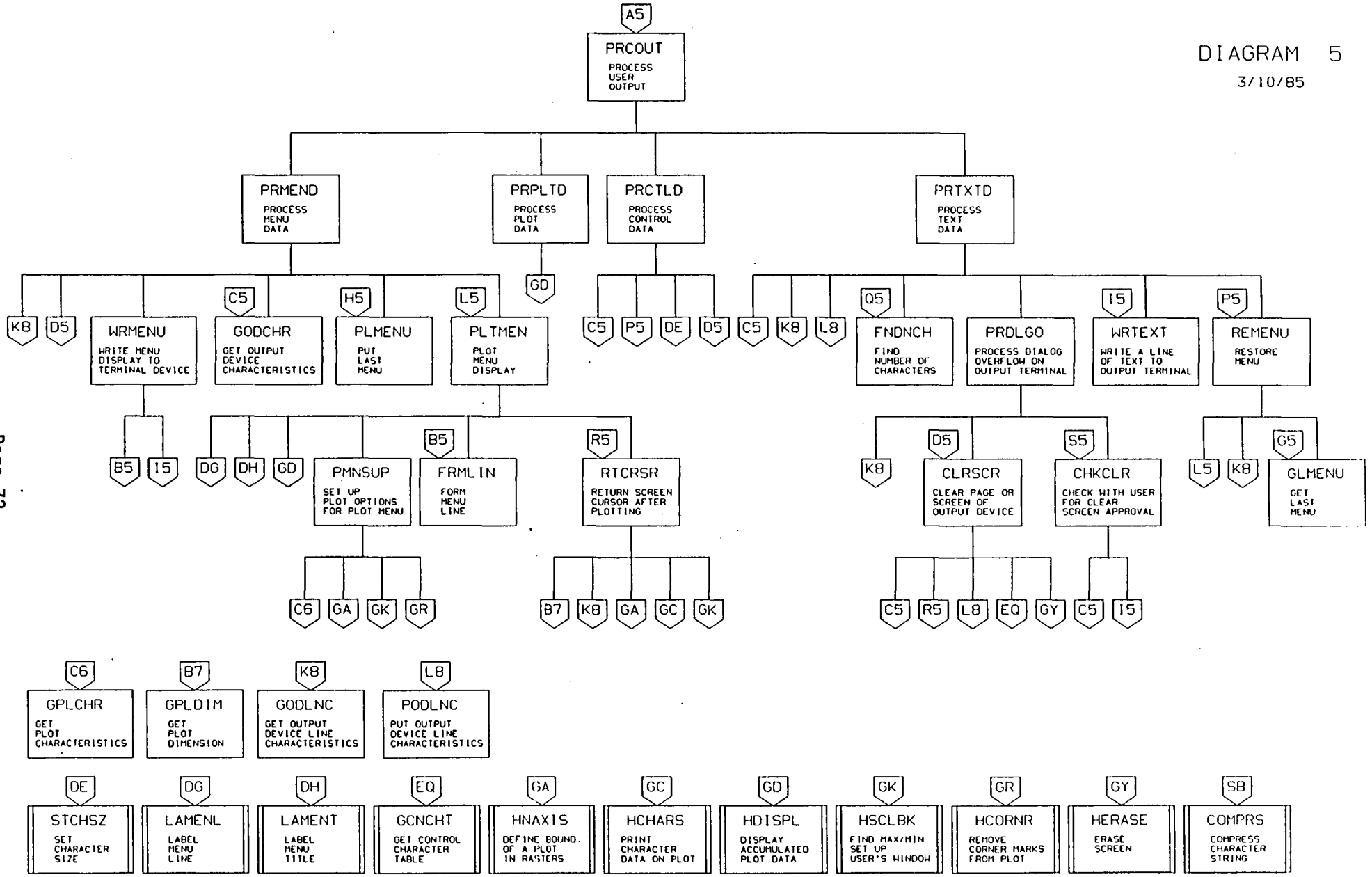
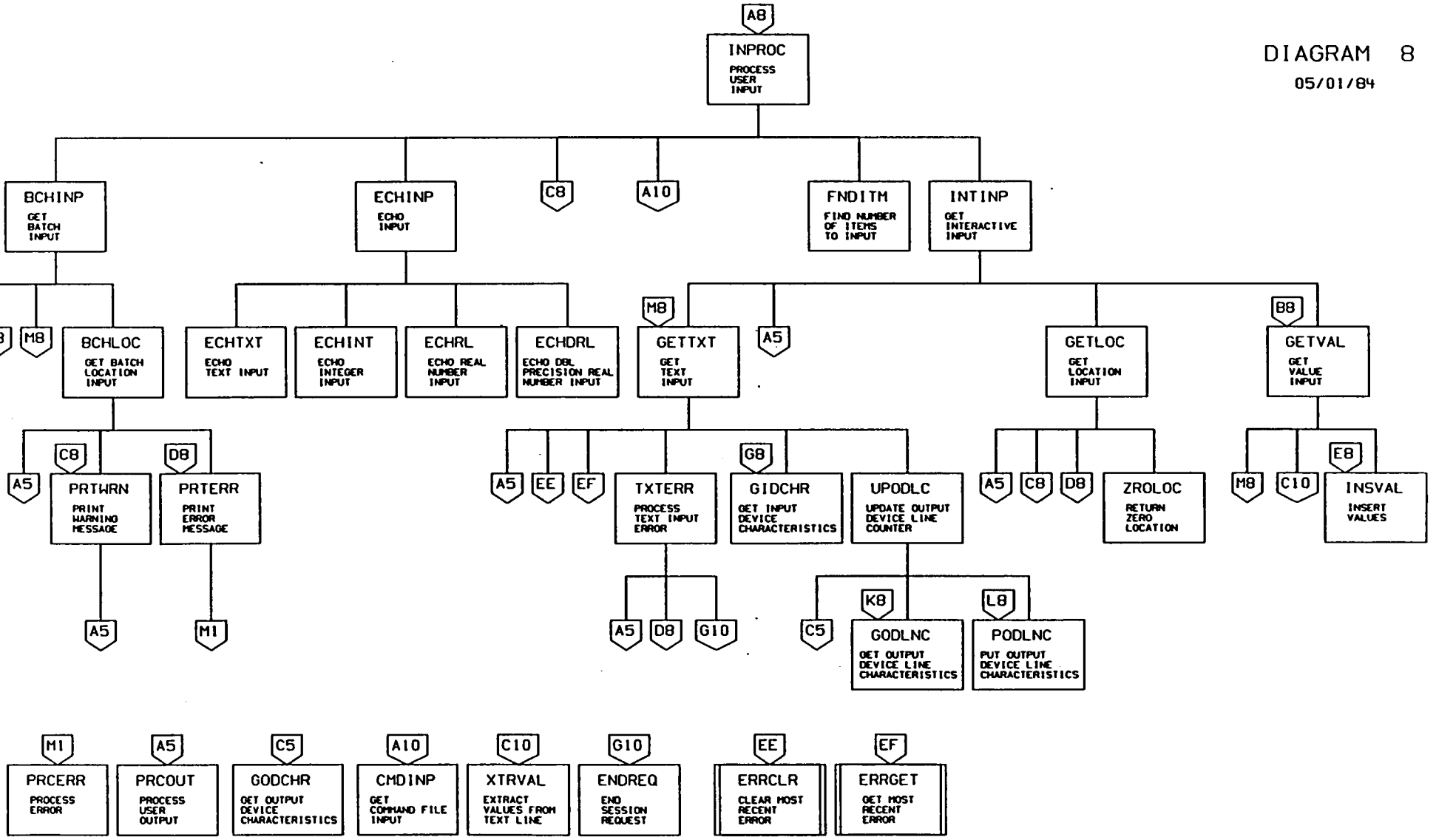
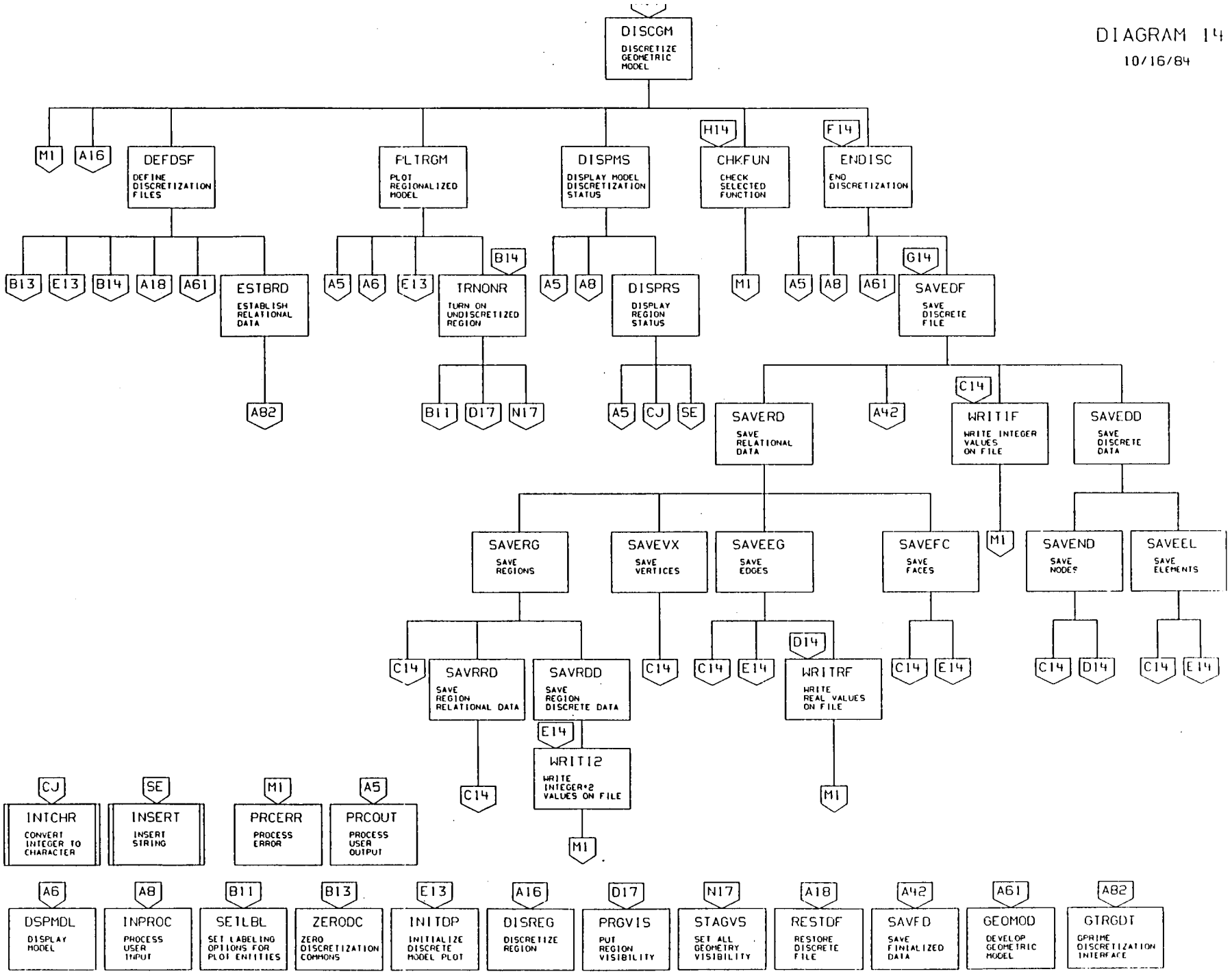


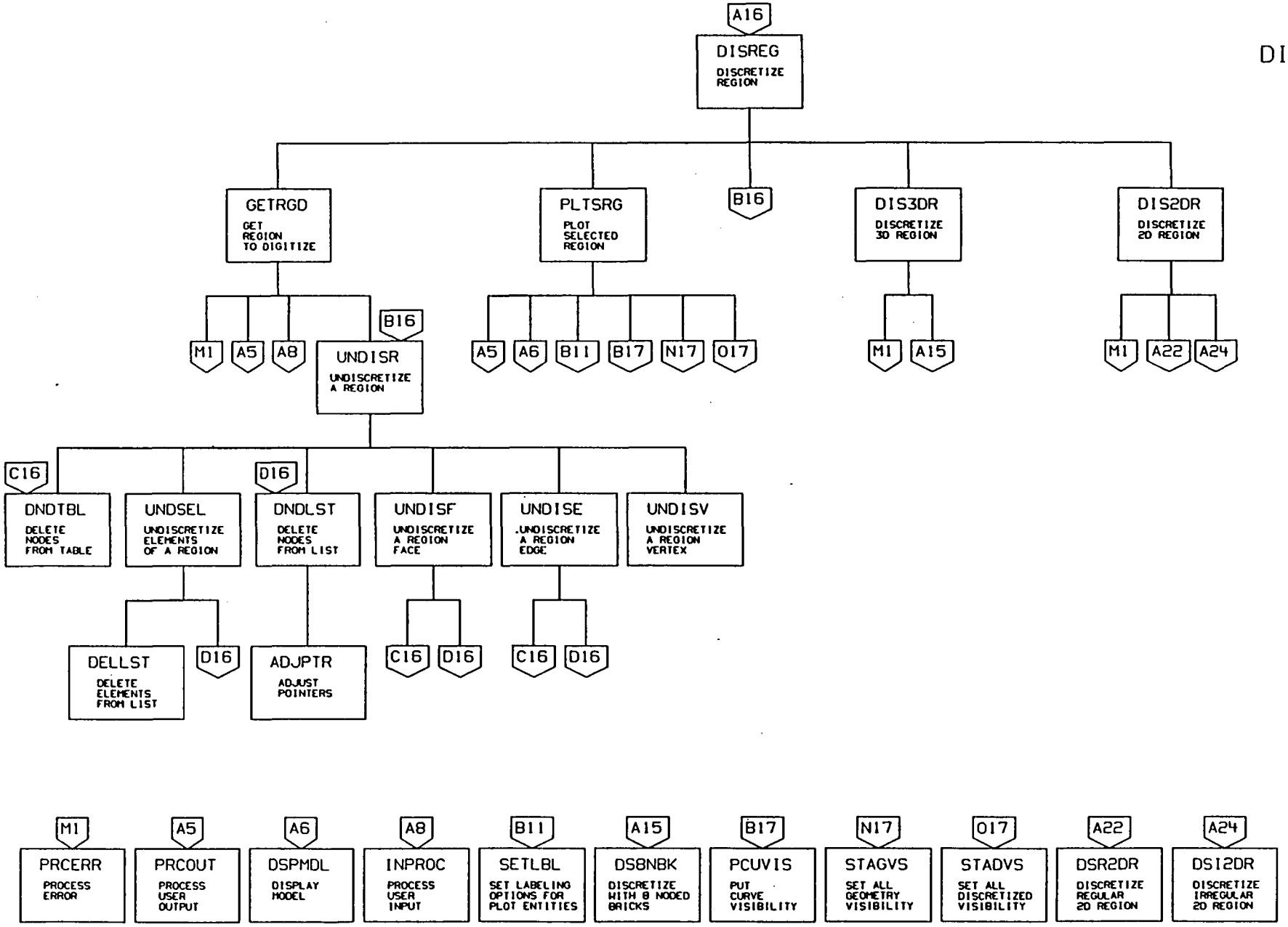
DIAGRAM 5
3/10/85
A5
PRCOUT
PROCESS USER OUTPUT
PRMEND
PROCESS MENU DATA
PRPLTD
PROCESS PLOT DATA
PRCTLD
PROCESS CONTROL DATA
PRTXTD
PROCESS TEXT DATA
K8
D5
WRMENU
WRITE MENU DISPLAY TO TERMINAL DEVICE
C5
GODCHR
GET OUTPUT DEVICE CHARACTERISTICS
H5
PLMENU
PUT LAST MENU
L5
PLTMEN
PLOT MENU DISPLAY
GD
C5
P5
DE
D5
C5
K8
L8
Q5
FNDNCH
FIND NUMBER OF CHARACTERS
PRDLGO
PROCESS DIALOG OVERFLOW ON OUTPUT TERMINAL
I5
WRTEXT
WRITE A LINE OF TEXT TO OUTPUT TERMINAL
P5
REMENU
RESTORE MENU
B5
I5
DG
DH
GD
PMNSUP
SET UP PLOT OPTIONS FOR PLOT MENU
B5
FRMLIN
FORM MENU LINE
R5
RTCRSR
RETURN SCREEN CURSOR AFTER PLOTTING
K8
D5
CLRSCR
CLEAR PAGE OR SCREEN OF OUTPUT DEVICE
S5
CHKCLR
CHECK WITH USER FOR CLEAR SCREEN APPROVAL
L5
K8
G5
GLMENU
GET LAST MENU
C6
GA
GK
GR
B7
K8
GA
GC
GK
C5
R5
L8
EQ
GY
C5
I5
C6
GPLCHR
GET PLOT CHARACTERISTICS
B7
GPLDIM
GET PLOT DIMENSION
K8
GODLNC
GET OUTPUT DEVICE LINE CHARACTERISTICS
L8
PODLNC
PUT OUTPUT DEVICE LINE CHARACTERISTICS
DE
STCHSZ
SET CHARACTER SIZE
DG
LAMENL
LABEL MENU LINE
DH
LAMENT
LABEL MENU TITLE
EQ
GCNCHT
GET CONTROL CHARACTER TABLE
GA
HNAXIS
DEFINE BOUND. OF A PLOT IN RASTERS
GC
HCHARS
PRINT CHARACTER DATA ON PLOT
GD
HDISPL
DISPLAY ACCUMULATED PLOT DATA
GK
HSCLBK
FIND MAX/MIN SET UP USER'S WINDOW
GR
HCORNR
REMOVE CORNER MARKS FROM PLOT
GY
HERASE
ERASE SCREEN
SB
COMPRS
COMPRESS CHARACTER STRING

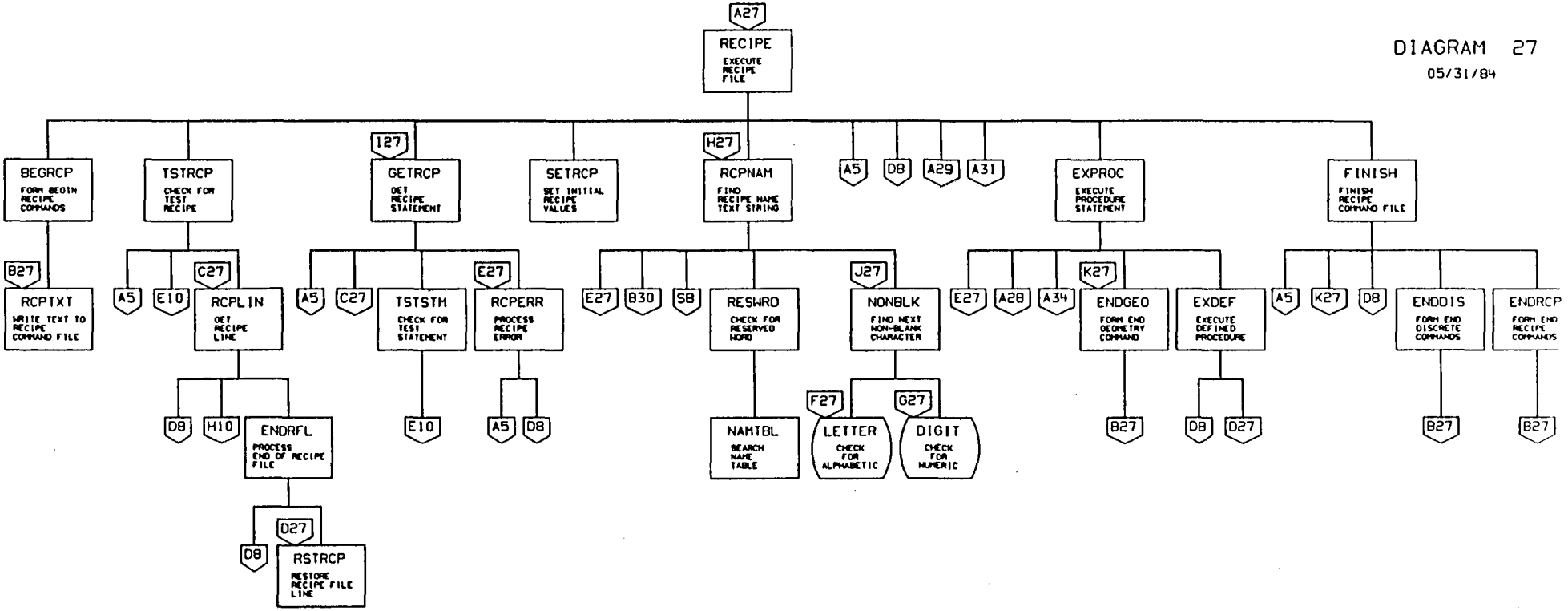
DIAGRAM 8

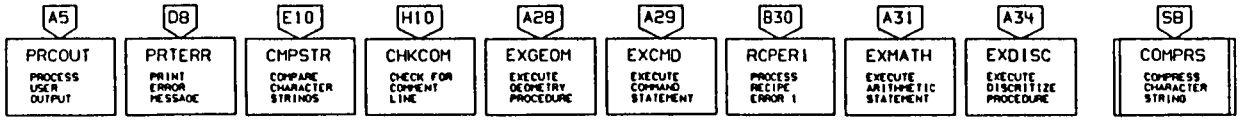
05/01/84

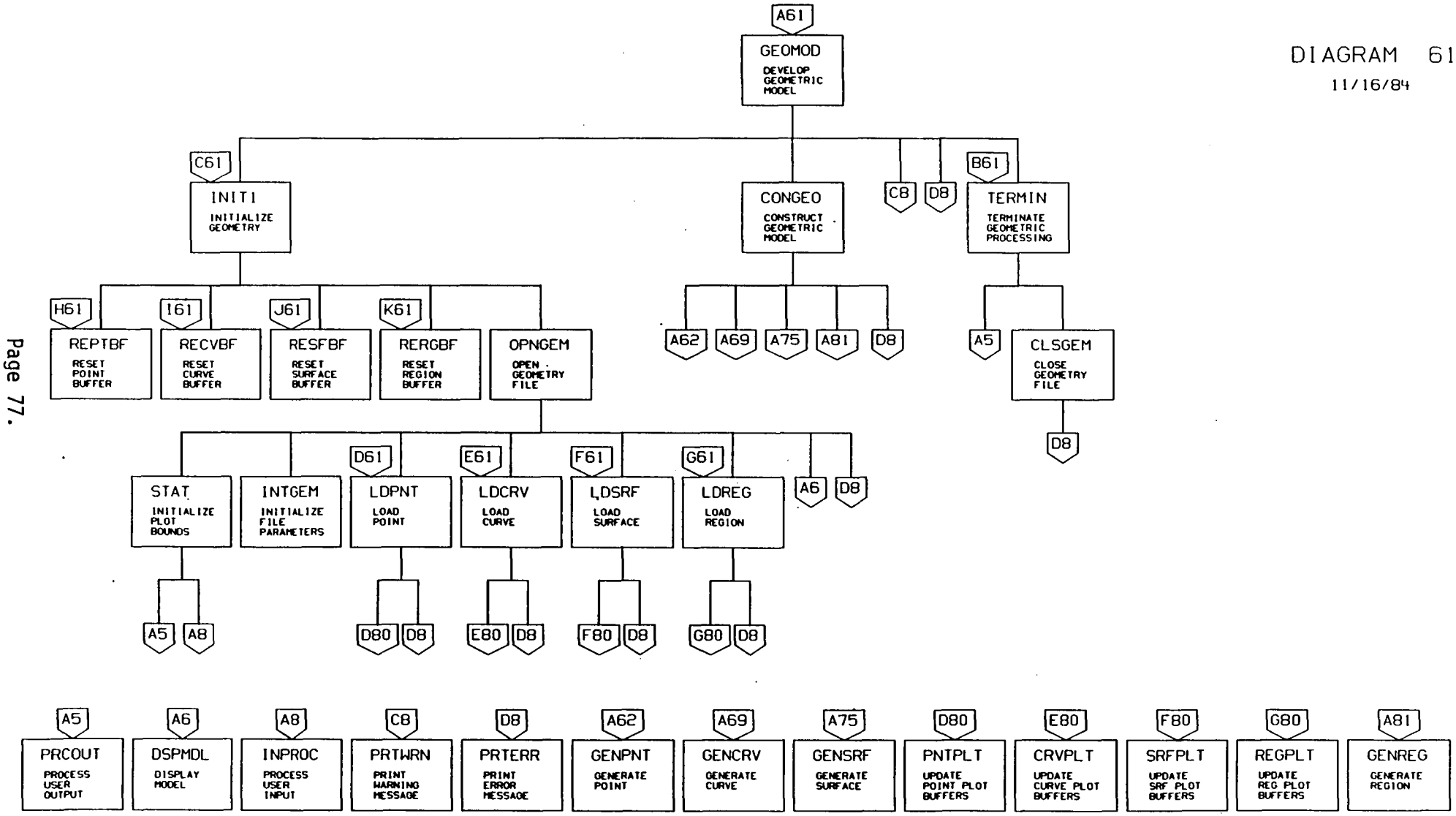


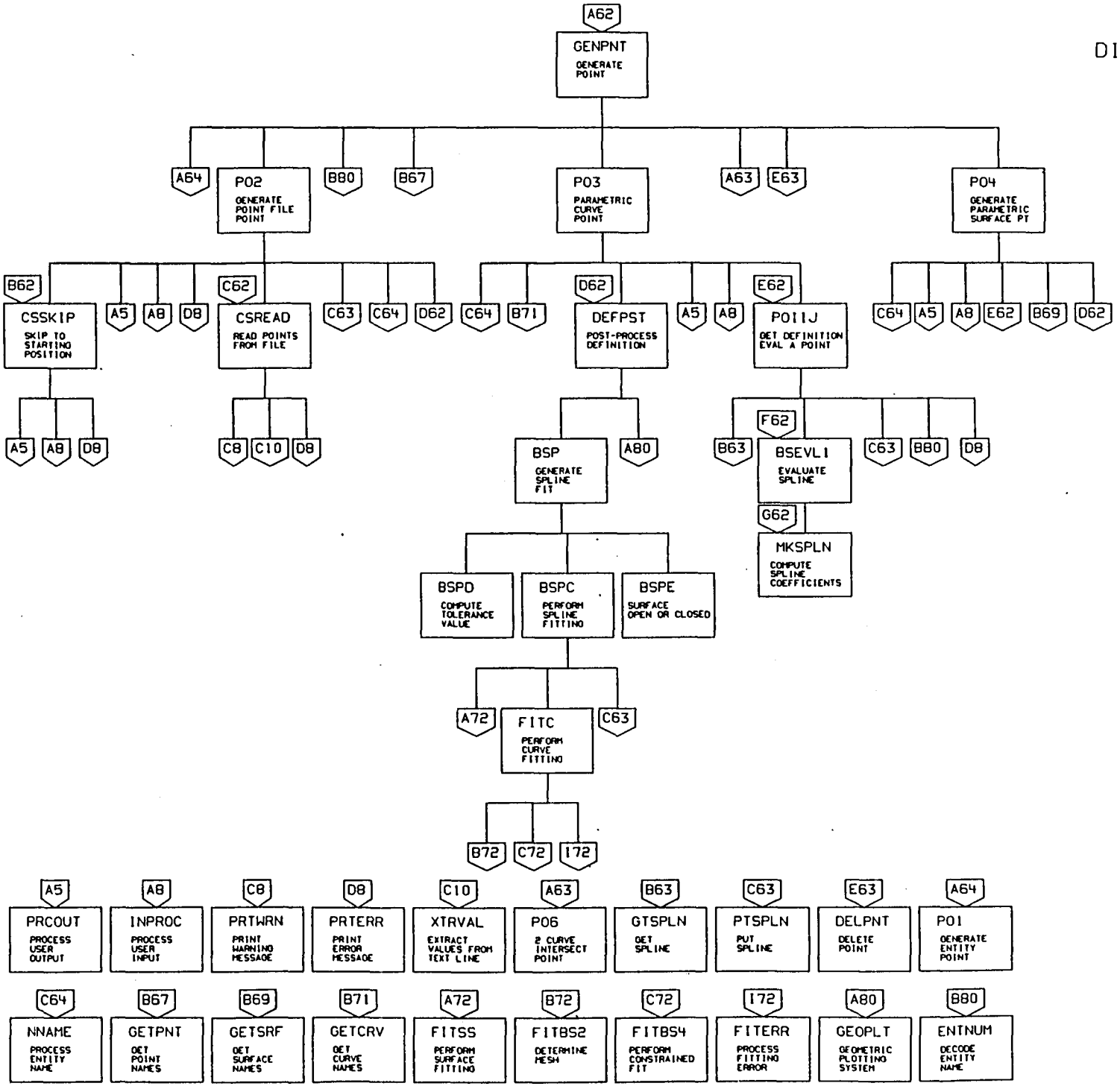

DISCGM
DISCRETIZE GEOMETRIC MODEL
M1
A16
DEFDSF
DEFINE DISCRETIZATION FILES
PLTRGM
PLOT REGIONALIZED MODEL
DISPMS
DISPLAY MODEL DISCRETIZATION STATUS
H14
CHKFUN
CHECK SELECTED FUNCTION
F14
ENDISC
END DISCRETIZATION
B13
E13
B14
A18
A61
ESTBRD
ESTABLISH RELATIONAL DATA
A82
A5
A6
E13
B14
TRNONR
TURN ON UNDISCRETIZED REGION
B11
D17
N17
A5
A8
DISPRS
DISPLAY REGION STATUS
A5
CJ
SE
M1
A5
A8
A61
G14
SAVEDF
SAVE DISCRETE FILE
SAVERD
SAVE RELATIONAL DATA
A42
C14
WRITIF
WRITE INTEGER VALUES ON FILE
SAVEDD
SAVE DISCRETE DATA
SAVERG
SAVE REGIONS
SAVEVX
SAVE VERTICES
SAVEEG
SAVE EDGES
SAVEFC
SAVE FACES
M1
SAVEND
SAVE NODES
SAVEEL
SAVE ELEMENTS
C14
SAVRRD
SAVE REGION RELATIONAL DATA
SAVRDD
SAVE REGION DISCRETE DATA
C14
C14
E14
D14
WRITRF
WRITE REAL VALUES ON FILE
C14
E14
C14
D14
C14
E14
E14
WRITI2
WRITE INTEGER*2 VALUES ON FILE
C14
M1
M1
CJ
INTCHR
CONVERT INTEGER TO CHARACTER
SE
INSERT
INSERT STRING
M1
PRCERR
PROCESS ERROR
A5
PRCOUT
PROCESS USER OUTPUT
A6
DSPMDL
DISPLAY MODEL
A8
INPROC
PROCESS USER INPUT
B11
SETLBL
SET LABELING OPTIONS FOR PLOT ENTITIES
B13
ZERODC
ZERO DISCRETIZATION COMMONS
E13
INITDP
INITIALIZE DISCRETE MODEL PLOT
A16
DISREG
DISCRETIZE REGION
D17
PRGVIS
PUT REGION VISIBILITY
N17
STAGVS
SET ALL GEOMETRY VISIBILITY
A18
RESTDF
RESTORE DISCRETE FILE
A42
SAVFD
SAVE FINIALIZED DATA
A61
GEOMOD
DEVELOP GEOMETRIC MODEL
A82
GTRGDT
GPRIME DISCRETIZATION INTERFACE


A16
DISREG
DISCRETIZE REGION
GETRGD
GET REGION TO DIGITIZE
PLTSRG
PLOT SELECTED REGION
B16
DIS3DR
DISCRETIZE 3D REGION
DIS2DR
DISCRETIZE 2D REGION
M1
A5
A8
B16
UNDISR
UNDISCRETIZE A REGION
A5
A6
B11
B17
N17
O17
M1
A15
M1
A22
A24
C16
D16
DNDTBL
DELETE NODES FROM TABLE
UNDSEL
UNDISCRETIZE ELEMENTS OF A REGION
DNDLST
DELETE NODES FROM LIST
UNDISF
UNDISCRETIZE A REGION FACE
UNDISE
UNDISCRETIZE A REGION EDGE
UNDISV
UNDISCRETIZE A REGION VERTEX
DELLST
DELETE ELEMENTS FROM LIST
D16
ADJPTR
ADJUST POINTERS
C16
D16
C16
D16
M1
PRCERR
PROCESS ERROR
A5
PRCOUT
PROCESS USER OUTPUT
A6
DSPMDL
DISPLAY MODEL
A8
INPROC
PROCESS USER INPUT
B11
SETLBL
SET LABELING OPTIONS FOR PLOT ENTITIES
A15
DS8NBK
DISCRETIZE WITH 8 NODED BRICKS
B17
PCUVIS
PUT CURVE VISIBILITY
N17
STAGVS
SET ALL GEOMETRY VISIBILITY
O17
STADVS
SET ALL DISCRETIZED VISIBILITY
A22
DSR2DR
DISCRETIZE REGULAR 2D REGION
A24
DSI2DR
DISCRETIZE IRREGULAR 2D REGION


A27
RECIPE
EXECUTE RECIPE FILE
I27
H27
A5
D8
A29
A31
BEGRCP
FORM BEGIN RECIPE COMMANDS
TSTRCP
CHECK FOR TEST RECIPE
GETRCP
GET RECIPE STATEMENT
SETRCP
SET INITIAL RECIPE VALUES
RCPNAM
FIND RECIPE NAME TEXT STRING
EXPROC
EXECUTE PROCEDURE STATEMENT
FINISH
FINISH RECIPE COMMAND FILE
B27
RCPTXT
WRITE TEXT TO RECIPE COMMAND FILE
A5
E10
C27
RCPLIN
GET RECIPE LINE
A5
C27
TSTSTM
CHECK FOR TEST STATEMENT
E27
RCPERR
PROCESS RECIPE ERROR
E27
B30
S8
RESWRD
CHECK FOR RESERVED WORD
J27
NONBLK
FIND NEXT NON-BLANK CHARACTER
E27
A28
A34
K27
ENDGEO
FORM END GEOMETRY COMMAND
EXDEF
EXECUTE DEFINED PROCEDURE
A5
K27
D8
ENDDIS
FORM END DISCRETE COMMANDS
ENDRCP
FORM END RECIPE COMMANDS
D8
H10
ENDRFL
PROCESS END OF RECIPE FILE
E10
A5
D8
NAMTBL
SEARCH NAME TABLE
F27
G27
LETTER
CHECK FOR ALPHABETIC
DIGIT
CHECK FOR NUMERIC
B27
D8
D27
B27
B27
D8
D27
RSTRCP
RESTORE RECIPE FILE LINE


A5
PRCOUT
PROCESS USER OUTPUT
D8
PRTERR
PRINT ERROR MESSAGE
E10
CMPSTR
COMPARE CHARACTER STRINGS
H10
CHKCOM
CHECK FOR COMMENT LINE
A28
EXGEOM
EXECUTE GEOMETRY PROCEDURE
A29
EXCMD
EXECUTE COMMAND STATEMENT
B30
RCPERI
PROCESS RECIPE ERROR 1
A31
EXMATH
EXECUTE ARITHMETIC STATEMENT
A34
EXDISC
EXECUTE DISCRITIZE PROCEDURE
S8
COMPRS
COMPRESS CHARACTER STRING


A61
GEOMOD
DEVELOP GEOMETRIC MODEL
C61
INITI
INITIALIZE GEOMETRY
CONGEO
CONSTRUCT GEOMETRIC MODEL
C8
D8
B61
TERMIN
TERMINATE GEOMETRIC PROCESSING
H61
REPTBF
RESET POINT BUFFER
I61
RECVBF
RESET CURVE BUFFER
J61
RESFBF
RESET SURFACE BUFFER
K61
RERGBF
RESET REGION BUFFER
OPNGEM
OPEN GEOMETRY FILE
A62
A69
A75
A81
D8
A5
CLSGEM
CLOSE GEOMETRY FILE
D8
STAT
INITIALIZE PLOT BOUNDS
INTGEM
INITIALIZE FILE PARAMETERS
D61
LDPNT
LOAD POINT
E61
LDCRV
LOAD CURVE
F61
LDSRF
LOAD SURFACE
G61
LDREG
LOAD REGION
A6
D8
A5
A8
D80
D8
E80
D8
F80
D8
G80
D8
A5
PRCOUT
PROCESS USER OUTPUT
A6
DSPMDL
DISPLAY MODEL
A8
INPROC
PROCESS USER INPUT
C8
PRTWRN
PRINT WARNING MESSAGE
D8
PRTERR
PRINT ERROR MESSAGE
A62
GENPNT
GENERATE POINT
A69
GENCRV
GENERATE CURVE
A75
GENSRF
GENERATE SURFACE
D80
PNTPLT
UPDATE POINT PLOT BUFFERS
E80
CRVPLT
UPDATE CURVE PLOT BUFFERS
F80
SRFPLT
UPDATE SRF PLOT BUFFERS
G80
REGPLT
UPDATE REG PLOT BUFFERS
A81
GENREG
GENERATE REGION


A62
GENPNT
GENERATE POINT
A64
P02
GENERATE POINT FILE POINT
B80
B67
P03
PARAMETRIC CURVE POINT
A63
E63
P04
GENERATE PARAMETRIC SURFACE PT
B62
CSSKIP
SKIP TO STARTING POSITION
A5
A8
D8
C62
CSREAD
READ POINTS FROM FILE
C63
C64
D62
C64
B71
D62
DEFPST
POST-PROCESS DEFINITION
A5
A8
E62
P01IJ
GET DEFINITION EVAL A POINT
C64
A5
A8
E62
B69
D62
A5
A8
D8
C8
C10
D8
BSP
GENERATE SPLINE FIT
A80
F62
B63
BSEVL1
EVALUATE SPLINE
C63
B80
D8
G62
MKSPLN
COMPUTE SPLINE COEFFICIENTS
BSPD
COMPUTE TOLERANCE VALUE
BSPC
PERFORM SPLINE FITTING
BSPE
SURFACE OPEN OR CLOSED
A72
FITC
PERFORM CURVE FITTING
C63
B72
C72
I72
A5
PRCOUT
PROCESS USER OUTPUT
A8
INPROC
PROCESS USER INPUT
C8
PRTWRN
PRINT WARNING MESSAGE
D8
PRTERR
PRINT ERROR MESSAGE
C10
XTRVAL
EXTRACT VALUES FROM TEXT LINE
A63
P06
2 CURVE INTERSECT POINT
B63
GTSPLN
GET SPLINE
C63
PTSPLN
PUT SPLINE
E63
DELPNT
DELETE POINT
A64
P01
GENERATE ENTITY POINT
C64
NNAME
PROCESS ENTITY NAME
B67
GETPNT
GET POINT NAMES
B69
GETSRF
GET SURFACE NAMES
B71
GETCRV
GET CURVE NAMES
A72
FITSS
PERFORM SURFACE FITTING
B72
FITBS2
DETERMINE MESH
C72
FITBS4
PERFORM CONSTRAINED FIT
I72
FITERR
PROCESS FITTING ERROR
A80
GEOPLT
GEOMETRIC PLOTTING SYSTEM
B80
ENTNUM
DECODE ENTITY NAME

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. AGENCY USE ONLY (*Leave blank*)	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	October 1991	Final Contractor Report

4. TITLE AND SUBTITLE
Engine Structures Modeling Software System (ESMOSS)

5. FUNDING NUMBERS
WU-505-63-5B
C-NAS3-22767

6. AUTHOR(S)

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
General Electric Company
Aircraft Engine Business Group
Cincinnati, Ohio 45215

8. PERFORMING ORGANIZATION REPORT NUMBER
None

9. SPONSORING/MONITORING AGENCY NAMES(S) AND ADDRESS(ES)
National Aeronautics and Space Administration
Lewis Research Center
Cleveland, Ohio 44135-3191

10. SPONSORING/MONITORING AGENCY REPORT NUMBER
NASA CR-187227

11. SUPPLEMENTARY NOTES
Project Manager, C.C. Chamis, Structures Division, NASA Lewis Research Center, (216) 433-3252

12a. DISTRIBUTION/AVAILABILITY STATEMENT
Unclassified - Unlimited
Subject Category 39

12b. DISTRIBUTION CODE

13. ABSTRACT (*Maximum 200 words*)

ESMOSS is the development of a specialized software system for the construction of geometric descriptive and discrete analytical models of engine parts, components and substructures which can be transferred to finite element analysis programs such as NASTRAN. The NASA-Lewis Engine Structures Program is concerned with the development of technology for the rational structural design and analysis of advanced gas turbine engines with emphasis on advanced structural analysis, structural dynamics, structural aspects of aeroelasticity, and life prediction. Fundamental and common to all these developments is the need for geometric and analytical model descriptions at various engine assembly levels which are generated using ESMOSS.

14. SUBJECT TERMS
Computer code; Geometric modeling; Engine parts; Finite element; Structural analysis; Dynamics aeroplasticity

15. NUMBER OF PAGES
80

16. PRICE CODE
A05

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

End of Document