FINAL REPORT

DUCT FLOW NONUNIFORMITIES FOR SPACE SHUTTLE MAIN ENGINE (SSME)

30 December 1987

Contract NAS8-35592

NATIONAL AERONAUTICS AND SPACE ADMINISTRATION
MARSHALL SPACE FLIGHT CENTER, AL 35812

By

Lockheed
Missiles & Space Company, Inc.
Huntsville Engineering Center
4800 Bradford Blvd., Huntsville, AL 35807
FOREWORD

This report was prepared by personnel of the Computational Mechanics Section of Lockheed's Huntsville Engineering Center. It constitutes final documentation of efforts performed under Contract NAS8-35592 for NASA-Marshall Space Flight Center.

The NASA-MSFC Contracting Officer's Representative for this research study was Dr. P.K. McConnaughey, ED32.
CONTENTS

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>FOREWORD</td>
<td>ii</td>
</tr>
<tr>
<td>1</td>
<td>INTRODUCTION</td>
</tr>
<tr>
<td>2</td>
<td>TECHNICAL APPROACH</td>
</tr>
<tr>
<td>2.1</td>
<td>HGM Computational Grid Code</td>
</tr>
<tr>
<td>2.2</td>
<td>SSME Fuel-Side Preburner Analysis</td>
</tr>
<tr>
<td>3</td>
<td>RESULTS</td>
</tr>
<tr>
<td>3.1</td>
<td>Three-Duct HGM Grid Code</td>
</tr>
<tr>
<td>3.1.1</td>
<td>HGM Geometry</td>
</tr>
<tr>
<td>3.1.2</td>
<td>HGM Grid</td>
</tr>
<tr>
<td>3.1.3</td>
<td>Computer Code</td>
</tr>
<tr>
<td>3.1.4</td>
<td>Code Implementation</td>
</tr>
<tr>
<td>3.2</td>
<td>Preburner Analysis</td>
</tr>
<tr>
<td>3.2.1</td>
<td>Inflow and Outflow Boundary Treatment</td>
</tr>
<tr>
<td>3.2.2</td>
<td>Chemistry Model</td>
</tr>
<tr>
<td>4</td>
<td>SUMMARY AND CONCLUSIONS</td>
</tr>
<tr>
<td>5</td>
<td>REFERENCES</td>
</tr>
</tbody>
</table>

Appendixes

<table>
<thead>
<tr>
<th>Appendix</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>Input Guide for HGM Grid Code</td>
<td>A-1</td>
</tr>
<tr>
<td>B</td>
<td>HGM Grid Code Input Listing</td>
<td>B-1</td>
</tr>
<tr>
<td>C</td>
<td>HGM Grid Code Listing</td>
<td>C-1</td>
</tr>
</tbody>
</table>
1. INTRODUCTION

Future research development, as well as production activities in space by U.S. federal and private agencies, will depend on the Space Shuttle and its derivative versions as a principal space transportation system. This dependence requires improved designs or techniques to extend the life, upgrade performance, reduce weight, lower operational costs, and generally improve the functional capability of the main propulsion system. The engines for this main propulsion system are advanced high pressure engines operating on oxygen and hydrogen. A need therefore exists to investigate, develop, and define basic concepts in support of the main propulsion system improvements. One basic area that has been investigated is the hot gas flow nonuniformities that occur within the manifold, duct work, and main injector. Nonuniformities result from highly distorted and mismatched flows within the ducts which create severe environments for the system components, thus limiting their useful life.

Development and verification tests of the Space Shuttle Main Engine (SSME) and results of computational modeling have shown that the three gas transfer tubes have an uneven flow distribution with large areas of separated flow. The outer transfer tubes each carry approximately twice the amount of gas as the center tube. This causes the energy of the gas to be much higher in the outer tubes. Flow from the tubes impinge upon the main injector liquid oxygen posts which bend under the static load of the gas flow, the bending being more pronounced in line with the outer tubes. To alleviate this phenomenon and to keep the posts cooler, shields, linking pairs of posts in the outermost row, were installed. The design alteration enhanced the injector life, however, LOX post failures show that this change alone is insufficient to grant specified life at equal to or greater rated power levels. Incorporation of the shields also affects circumferential flow in the annulus and degrades the engine performance and necessitates higher operating temperatures in the
turbines. To improve the SSME design and for future use in the design of new
generation rocket engines, more experimental data are needed. These data,
combined with previous test data and integrated into analytical methods, can
be used to establish a base for design of high energy hot gas flow systems.

In addition the preburners of the SSME still experience serious problems
related also to the cracking of LOX posts and the frisbee in the inlet mani-
 folds. Operation of the fuel preburner creates flows which cause turbine
blade cracking in the high pressure fuel turbopump. The most likely cause of
these problems is the extreme thermal environments caused by start-up and
shut-down of the engine. A detailed transient analysis of these flows is
necessary in order to help alleviate these problems.

This report describes results of efforts by personnel of the Computational
Mechanics Group at the Lockheed-Huntsville Engineering Center to assist the
computational staff of NASA-MSFC in developing analytical capabilities for
modeling hot gas flow on the fuel side of the SSME. The primary objective was
to develop and deliver a computer code which produces a computational grid for
the three-duct Hot Gas Manifold (HGM) on the fuel side of the SSME. A
secondary objective was to provide input for an accurate computation of the
flow characteristics inside the fuel side preburner during the start-up
transient of the SSME.
2. TECHNICAL APPROACH

2.1 HGM COMPUTATIONAL GRID CODE

Nearly all computational codes which are available to numerically solve the three-dimensional fluid flow equations are designed to be applied to a well structured grid model of the flow region. To perform the calculations, generalized independent variables are introduced which transform the physical coordinates, \((x, y, z)\), into general curvilinear coordinates, \((n_1, n_2, n_3)\). Thus the physical domain must be gridded as a single or series of hexahedral zones described by eight corner points, 12 edges, and six surfaces. Such an arbitrary zone is shown in Fig. 2-1.

Fig. 2-1 Hexahedral Element Showing Local Intrinsic Coordinates
The approach used in the current study was to provide an algebraic grid generation code which would produce the Cartesian coordinates \((x, y, z)\), for points along the lines of constant \(n_1\), \(n_2\), and \(n_3\). Basic mathematical techniques taken from analytic geometry and vector algebra were employed to describe a hexahedral zone in terms of piecewise continuous analytic functions which represent the zonal edges and surfaces.

An intrinsic curvilinear coordinate system can be produced by mapping a unit cube onto the simply connected hexahedral zone. What is needed is a transformation function that will map a unit cube in \((n_1, n_2, n_3)\) space univalently onto the hexahedral volume of interest thus producing the required intrinsic coordinate system. A procedure which produces the desired result is referred to as either the method of transfinite interpolation or multi-variate blending function interpolation (Ref. 1). A brief description of this method follows.

Let \(F(n_1, n_2, n_3)\) be a vector-valued functional representing the region \(R\) of interest in curvilinear space. Then as \((n_1, n_2, n_3)\) range over \(R\), \(F\) traces out the region in Euclidean space \((x, y, z)\). Also let \(\phi, \psi, \text{ and } \lambda\) be blending functions which obey the cardinality conditions:

\[
\phi_i(n_i=1) = \begin{cases} 
1 & \text{if } i = 1 \\
0 & \text{if } i \neq 1 
\end{cases}
\]

\[
\psi_j(n_j=m) = \begin{cases} 
1 & \text{if } j = m \\
0 & \text{if } j \neq m 
\end{cases}
\]

\[
\lambda_k(n_k=n) = \begin{cases} 
1 & \text{if } k = n \\
0 & \text{if } k \neq n 
\end{cases}
\]

The simplest form of blending functions meeting these conditions are

\[
\phi_0(n_1) = 1 - n_1 \\
\psi_0(n_2) = 1 - n_2 \\
\lambda_0(n_3) = 1 - n_3
\]

\[
\phi_1(n_1) = n_1 \\
\psi_1(n_2) = n_2 \\
\lambda_1(n_3) = n_3
\]

Then a trilinearly blended interpolant of \(F\), which will map a unit cube onto the region \(R\) is given by
\[
U(n_1, n_2, n_3) = \begin{bmatrix}
x(n_1, n_2, n_3) \\
y(n_1, n_2, n_3) \\
z(n_1, n_2, n_3)
\end{bmatrix}
\]

\[
= (1-n_1)F(0, n_2, n_3) + n_1 F(1, n_2, n_3)
\]

\[
+ (1-n_2)F(n_1, 0, n_3) + n_2 F(n_1, 1, n_3)
\]

\[
+ (1-n_3)F(n_1, n_2, 0) + n_3 F(n_1, n_2, 1)
\]

\[
- (1-n_1)(1-n_2)F(0, 0, n_3) - (1-n_1)n_2 F(0, 1, n_3)
\]

\[
- n_1(1-n_2)F(1, 0, n_3) - n_1 n_2 F(1, 1, n_3)
\]

\[
- (1-n_1)n_3 F(0, n_2, 0) - (1-n_1)n_3 F(0, n_2, 1)
\]

\[
- n_1(1-n_3)F(1, n_2, 0) - n_1 n_3 F(1, n_2, 1)
\]

\[
- (1-n_2)(1-n_3)F(n_1, 0, 0) - (1-n_2)n_3 F(n_1, 0, 1)
\]

\[
- n_2(1-n_3)F(n_1, 1, 0) - n_2 n_3 F(n_1, 1, 1)
\]

\[
+ (1-n_1)(1-n_2)(1-n_3)F(0, 0, 0) + (1-n_1)(1-n_2)n_3 F(0, 0, 1)
\]

\[
+ (1-n_1)n_2(1-n_3)F(0, 1, 0) + (1-n_1)n_2 n_3 F(0, 1, 1)
\]

\[
+ n_1(1-n_2)(1-n_3)F(1, 0, 0) + n_1(1-n_2)n_3 F(1, 0, 1)
\]

\[
+ n_1 n_2(1-n_3)F(1, 1, 0) + n_1 n_2 n_3 F(1, 1, 1)
\]

where

\[
F(0, n_2, n_3)\] represents a surface with \(n_1 = 0\), etc.

\[
F(0, 0, n_3)\] represents an edge with \(n_1 = n_2 = 0\), etc.

\[
F(0, 0, 0)\] represents a point with \(n_1 = n_2 = n_3 = 0\), etc.
In two dimensions this equation performs a bilinear interpolation over an arbitrary region consisting of four distinct corners simply connected by four edges, where

\[
\begin{align*}
F(0, n_2) &= \left[ \begin{array}{c} x \\ y \end{array} \right] \text{ along } \text{EDGE}_4 & F(0, 0) &= \left[ \begin{array}{c} x \\ y \end{array} \right] \text{ at POINT}_1 \\
F(1, n_2) &= \left[ \begin{array}{c} x \\ y \end{array} \right] \text{ along } \text{EDGE}_2 & F(0, 1) &= \left[ \begin{array}{c} x \\ y \end{array} \right] \text{ at POINT}_4 \\
F(n_1, 0) &= \left[ \begin{array}{c} x \\ y \end{array} \right] \text{ along } \text{EDGE}_1 & F(1, 0) &= \left[ \begin{array}{c} x \\ y \end{array} \right] \text{ at POINT}_2 \\
F(n_1, 1) &= \left[ \begin{array}{c} x \\ y \end{array} \right] \text{ along } \text{EDGE}_3 & F(1, 1) &= \left[ \begin{array}{c} x \\ y \end{array} \right] \text{ at POINT}_3
\end{align*}
\]

and the interpolation equation for \( F \) could be rewritten as

\[
U = \left[ \begin{array}{c} x \\ y \end{array} \right] = (1-n_1)\text{EDGE}_4 + n_1\text{EDGE}_2 + (1-n_2)\text{EDGE}_1 + n_2\text{EDGE}_3
\]

\[
- (1-n_1)(1-n_2)\text{POINT}_1 - (1-n_1)n_2\text{POINT}_4
\]

\[
- n_1(1-n_2)\text{POINT}_2 - n_1n_2\text{POINT}_3
\]

In this equation \( \text{EDGE}_4 \) represents a vector-valued functional along edge four, etc. Examination of this equation shows that it performs linear interpolations between \( \text{EDGE}_1 \) and \( \text{EDGE}_3 \) and between \( \text{EDGE}_2 \) and \( \text{EDGE}_4 \), hence the term bilinear interpolation. Hence, if the functional for each edge can be derived and is analytic, a grid or mesh of node points can be generated by substituting values of \( n_1 \) and \( n_2 \) into the equation.

In three dimensions the general equation above performs trilinearly blended interpolation over an arbitrary region consisting of eight distinct corner points simply connected by 12 edges, where
\[ F(0, n_2, n_3) = \left[ \frac{\chi}{\lambda} \right] \text{ on } \text{SIDE}_5, \text{ etc.} \]
\[ F(0, 0, n_3) = \left[ \frac{\chi}{\lambda} \right] \text{ along } \text{EDGE}_5, \text{ etc.} \]
\[ F(0, 0, 0) = \left[ \frac{\chi}{\lambda} \right] \text{ at POINT}_1, \text{ etc.} \]

and which can be rewritten as

\[
U = \left[ \frac{\chi}{\lambda} \right] \\
= (1-n_1)\text{SIDE}_5 + n_1\text{SIDE}_6 + (1-n_2)\text{SIDE}_2 + n_2\text{SIDE}_4 + (1-n_3)\text{SIDE}_1 + n_3\text{SIDE}_3 \\
- (1-n_1)(1-n_2)\text{EDGE}_5 - (1-n_1)n_2\text{EDGE}_8 - n_1(1-n_2)\text{EDGE}_6 - n_1n_2\text{EDGE}_7 \\
- (1-n_1)(1-n_3)\text{EDGE}_4 - (1-n_1)n_3\text{EDGE}_{12} - n_1(1-n_3)\text{EDGE}_2 - n_1n_3\text{EDGE}_{10} \\
- (1-n_1)(1-n_3)\text{EDGE}_1 - (1-n_2)n_3\text{EDGE}_9 - n_2(1-n_3)\text{EDGE}_3 - n_2n_3\text{EDGE}_{11} \\
+ (1-n_1)(1-n_2)(1-n_3)\text{POINT}_1 + (1-n_1)(1-n_2)n_3\text{POINT}_5 + (1-n_1)n_2(1-n_3)\text{POINT}_4 \\
+ (1-n_1)n_2n_3\text{POINT}_8 + n_1(1-n_2)(1-n_3)\text{POINT}_2 + n_1(1-n_2)n_3\text{POINT}_6 \\
+ n_1n_2(1-n_3)\text{POINT}_3 + n_1n_2n_3\text{POINT}_7
\]

where \text{SIDE}_i, \text{EDGE}_j, \text{POINT}_k represent vector-valued functionals on the surfaces, along the edges, and at the corner points, respectively.
This equation reduces to the previous two-dimensional analog along any flat surface or along any surface in which a straight line can be drawn between any two opposing edges such that the line lies entirely within the surface.

With the general transformation, any point in local coordinates $\eta_1$, $\eta_2$, $\eta_3$ can be related to the physical Cartesian coordinates $x$, $y$, $z$. The entire grid of discrete points is generated in the HGM code using this concept. This general interpolant can accommodate any stretching function for concentrating points near walls or regions of large gradients. Furthermore, the edges of the hexahedral can be segmented allowing another means of grid spacing control.

### 2.2 SSME FUEL-SIDE PREBURNER ANALYSIS

Operation of the fuel preburner is initiated by opening propellant control valves and igniting the flow with the augmented spark ignited (ASI). The configuration of the FPB is shown in Fig. 2-2. The flow passage from the fuel preburner oxidizer valve (FPOV) into the oxidizer manifold above the interpropellant plate is shown in this figure. The oxidizer flow to the ASI is supplied from a bleed line from the FPOV. The valves which control this flow are shown schematically in Fig. 2-3; note that there is not a separate fuel control valve for each preburner. Flows from the fuel and oxidizer manifolds enter the FPB combustor through the injector face plate, one third of the layout of which is shown in Fig. 2-2. The preburner liner extension shown in Fig. 2-2 slides inside of the sealer groove as shown in the upper left and right sides of Fig. 2-5. Thus, the preburned gases flow, via the dome at the bottom of the chamber, onto the turbine blades, then following the path through the 180-deg turnaround duct to the fuel bowl and transfer ducts of the HGM and onto the LOX posts and into the main combustion chamber.

The presence of the baffles shown in Figs. 2-2 and 2-4 create symmetry planes for computational analysis but remove the simplicity of axis symmetry. In addition, any model geometry should include the influence of the large nut at the bottom of the preburner chamber which holds the dome in place.
Fig. 2-2 Fuel Preburner Configuration
Fig. 2-3  Schematic of Control Valves for Preburner Configuration
Fig. 2-4  Face Plate Showing One-Third of Preburner
Distribution of Injection Holes and Elements
Fig. 2-5  Cross-Section of Lower Half Fuel Side Preburner Chamber Indicating at Top Left and Right Procedure for Insertion of Upper Preburner Liner Extension
Our approach is to use the symmetry created by the baffles and model only one-third of the geometry, and the large nut on top of the dome should be incorporated into the computational grid. Furthermore, in order that the most accurate analysis of the preburner be made for portions of the startup transient stage of its operation, a thorough study was made of the average chamber parameters for the first few seconds of operation. Data were obtained from the latest Digital Transient Model (DTM) results. The DTM is a one-dimensional lumped parameter model of the entire SSME which was originally developed by Rocketdyne and modified by D.C. Seymour of the Performance Analysis Branch of the Propulsion Lab at NASA-MSFC, to more closely match actual engine test data obtained at MSFC. These data were used to obtain, as accurate as possible, the face plate inlet conditions and downstream chamber exit conditions for use in the boundary treatment of a computational analysis.
3. RESULTS

3.1 HGM COMPUTATIONAL GRID CODE

3.1.1 HGM Geometry

A schematic representation showing the construction of the geometry which must be modeled is presented in Fig. 3-1. The 180-deg turnaround duct is displayed in a cutaway fashion to show the turn. Only half of the manifold is represented since, due to the plane of symmetry which divides the center transfer duct in half, only half need be computationally modeled.

Fig. 3-1 Schematic Showing Components of HGM Geometry to be Modeled
The geometry was generated in five separate pieces or zones. The five zones are shown in Fig. 3-2. Each zone has a general hexahedral shape and in Fig. 3-2 most of the eight corner points of each are clearly marked. Figure 3-3 shows the Cartesian coordinate system relative to which the position of each node in the grid is referenced. Except for the hole perimeters in the outer wall of Zone 3 and the fairing at the entrances to Zones 4 and 5 the edges of each zone can be described as piecewise continuous segments composed of either straight lines or circular arcs. In addition, excluding the two special regions previously mentioned, the surfaces of each zone can be generated by rotating an edge about the appropriate axis. For Zones 1, 2, and 3 the X axis is the axis of revolution.

3.1.2 HGM Grid

Zone 1, the turnaround duct (TAD), is composed of 94 nodes in the streamwise direction, 72 nodes in the circumferential direction (0 to 180 deg), and 21 nodes across the duct between inner and outer surfaces. Two perspectives showing this nodal distribution are shown in Fig. 3-4.

Zones 2 and 3 make up the bowl section of the HGM and contain 59 nodes in the x-direction, from bowl entrance to rear of the bowl, 107 nodes in the circumferential direction and 21 nodes between the inner and outer surfaces. Distribution of nodes in the bowl is presented in Figs. 3-5 through 3-7.

Zones 4 and 5 comprise the right and half of the middle transfer duct portions of the HGM. The right duct has been generated with 29 nodes along the duct axis and 27 x 20 nodes in a cross section. The similar distribution in Zone 5 is 29 x 27 x 13. Surface and cross-section grids for these two zones are given in Figs. 3-8 and 3-9. A closeup of the surface mesh in the vicinity of the fairing for each transfer duct is contained in Fig. 3-10.
Fig. 3-2  Schematic Showing Five Zones into Which the Three-Duct HGM was Subdivided
Fig. 3-3 Cartesian Coordinate System Relative to Which Position all Nodes are Referred
Fig. 3-4 Grid Plots of TAD Internal Grid (Bottom) and of Part of the Grid Distribution on Inner and Outer Surfaces
Fig. 3-5 Grid Plots for Inner and Outer Surfaces of Zones 2 and 3 as well as internal grid at the common plane of intersection.
Fig. 3-7 Grid Plot of Computational Mesh on Outer Surface of Zone 3
Fig. 3-8  Surface and Cross-Section Node Distribution for Top Transfer Duct
Fig. 3-9  Middle Transfer Duct Surface  
and Cross-Section Nodal Distribution
Fig. 3-10  Expanded View of Surface Mesh for Region of Fairing on Both Transfer Ducts
The internal grid resolution presented in the previous figures is adequate for a laminar viscous computation but would need to be modified for application to a turbulent computation. The procedure for doing this will be described in Section 3.1.4.

3.1.3 Computer Code

A source listing of the three-duct HGM geometry code is provided in Appendix C. A concise input guide and an input listing for the grids displayed in Figs. 3-4 through 3-9 is also included in Appendixes A and B, respectively.

In Fig. 3-11 a primary calling sequence flow chart is shown. A brief explanation of the function of each subroutine follows:

- INITIAL - Reads the first two lines of the input file, initializes coefficient arrays, and defines logical unit numbers and counters.

- INPUT - Reads the remainder of the input file and sets all parameters to be used in remaining subroutines.

- GRID - The controlling subroutine for the generation of the spatial coordinates of each node in each zone.

- ETABC - Calculates the values of $\eta_1$, $\eta_2$, $\eta_3$ along the I, J, and K directions for each hexahedral shaped section of each zone.

- EDGE - Determines the Cartesian coordinates for nodes along each edge of each side of each section of each zone using the bilinear/trilinear interpolation scheme.

- SURFACE - Determines the Cartesian coordinates of nodes on a three-dimensional surface using the trilinear interpolation scheme. Here, all outer and internal surface nodes are calculated from the previously determined edge distribution.

- OUTPUT - Provides printed output and stores geometry in File 20 for use in plotting or as input to an integration code.
Fig. 3-11 Primary Calling Sequence
The output to File 20 is in the format to be input as a multi-grid geometry file to the PLOT3D plotting code.

The code is written in standard FORTRAN. However, the listing provided in Appendix C is a VAX 11/785 version and could contain some generic VAX statements which would have to be translated if used on other than a VAX.

3.1.4 Code Implementation

To implement the HGM geometry code as it appears in the listing of Appendix C, the input listing of Appendix B must correspond to logical Unit 5. Unit 6 must be assigned to the written output and Unit 20 to the geometry file which will contain the x, y, z coordinates for each node in each zone.

Each separate input file is labeled to indicate the zone being described by the following card images. These zone labels correspond to those shown in Fig. 3-2. Because of the number of nodes in each zone it is recommended that each zone be run separately. In fact as shown in Appendix B, each zone input is a separate file since all contain card types 1, 2, and 3. If more than one zone is to be run together (e.g., Zones 2 and 3), only one set of card types 1, 2, and 3 should appear at the top of the input file.

A detailed description of the input file to the code is provided in Appendix A. Modifications to the geometry can be facilitated by studying the input guide while observing both the input files and the detailed grid pictures presented in Figs. 3-3 through 3-10. Redistributions of nodes can be accomplished by making minor modifications to the input files. For example for a turbulent computation if the nodes in Zone 4 near the wall require redistribution closer to the wall then a change would need to be made to card type 9 on line 11 of the input file for that zone. The 7.0 appearing in the $\eta_1$ and $\eta_3$ positions could be changed to 10.0 (see page A-10). If more nodes were desired in a cross plane then the 27 and 29 on card type 8 could both be increased.
Dimensioning in the program has been kept to a minimum. The largest
dimensioned arrays in the bulk of the code are NODENUM(10000) and
NODE(5,10000). At the very end of all computations the PLOT3D file is gen-
erated. In this subroutine the x, y, z coordinate arrays are each dimensioned
to 155,000. The 10,000 corresponds to twice the maximum number of nodes in a
plane perpendicular to the marching direction (η direction input on card 7) for creating the geometry. The 155,000 must be equal to or greater than the
number of nodes in the largest zone, which is the TAD. In Zones 1, 2, and 3 the direction of η₁ is from TAD entrance to bowl back wall; the η₂ direction is from inner to outer wall; and η₃ is directed from side opposite transfer
ducts circumferentially. In zones 4 and 5, η₃ increases in the streamwise
direction from bowl outward, and η₁ x η₂ form the cross planes in each duct.

Note that the code is designed to output each zone of the HGM so that
each has one cross plane in common with the preceding zone(s). This must be remembered for incorporating the grid into a flowfield solver code. The
geometry must be integrated in a multi-block or multi-zone fashion. If the computer available has large enough core memory or if it is a large virtual
machine then Zones 2 and 3 can easily be combined into one larger zone since
at all common planes the nodal positions match exactly. However, the TAD zone
matches exactly to Zone 2 but only every other circumferential plane in Zone 3 matches with a corresponding plane in the TAD. This mismatch can be corrected
by making appropriate minor modifications to the input file for Zone 1.

3.2 PREBURNER ANALYSIS

An accurate analysis of the fluid dynamic environment in the preburner
during the startup transient portion of its operation demands careful treat-
ment of the inflow and outflow boundary conditions as well as the most up-to-
date chemistry model for the reacting components in the flow. Results of a
comprehensive study of these two items is presented in the following two
sections.
3.2.1 Inflow and Outflow Boundary Treatment

Data were obtained in the form of printed output from the most recent DTM analysis of the SSME startup transient conditions. This included injector fuel and oxidizer flow rates, ignition fuel, and oxidizer flow rates. From this, average chamber temperature and static pressure data tables were generated. These data are displayed graphically in Figs. 3-12 through 3-16. At any instant in time during the 5-sec startup process, these data provide information necessary to specify mass and composition flow rates at the inlet plane as well as downstream static pressure at the exit plane for performing a flowfield computation.

During any selected time interval, downstream static pressure would be obtained from a table look-up of the data shown in Fig. 3-16. Total flow rates are obtainable from the remaining data, but the specific strategy for distributing the mass flow over the face plate inlet plane requires careful analyses. Observation of Fig. 2-4 reveals that there is clearly a nonuniform distribution of mass injection. A practical approach which allows for a reasonable approximation to the physical situation would be to divide the inlet computational plane into 18 regions; 17 concentric annular regions forming the injection elements and holes and one ignition region. The 17 injection regions correspond to the 17 rings of holes and elements and holes.

Nine of these annular regions contain only hydrogen injection while in the remaining eight both hydrogen and oxygen injection occurs, and within each region fluid is injected nonuniformly. To most accurately model each region fluid should be injected with the proper momentum from computational nodes which form an area which is approximately the same as the effective area for that region. The remaining nodes in that region must have a zero velocity fixed boundary condition. The effective area is less than the actual combined areas of holes and elements.
Fig. 3-12 Fuel Preburner Flow Rates Taken from Digital Transient Model (DTM)

Fig. 3-13 Fuel Preburner Flow Rates from DTM During Early Transient
Fig. 3-14 Fuel Preburner Igniter Flow Rates from Digital Transient Model (DTM)

Fig. 3-15 Fuel Preburner Igniter Flow Rates from DTM During Early Transient
Fig. 3-16 Fuel Preburner Average Chamber Pressure and Temperature During Start-Up Transient (Taken from DTM)
The mass flow rate, $\frac{\Delta m}{\Delta t}$, through an area $A$ of fluid having mass density of $\rho$ and inlet speed $V$ can be expressed as

$$\frac{\Delta m}{\Delta t} = \rho VA$$

The ratio of dynamic pressure to pressure loss through the opening defines a discharge coefficient $C_d$ given by

$$C_d = \frac{1}{2} \rho \frac{V^2}{\Delta p}$$

Combining these two equations one expression for the product of $A$ and $C_d$ is obtained, which corresponds to the effective area.

$$AC_d = \frac{\Delta m/\Delta t}{0.67 (\rho \Delta p)^{1/2}}$$

In this equation the 0.67 includes unit conversion factors so that $AC_d$ is determined in square inches. Data obtained from the SSME Power Balance Program, Revision F, provide a $\Delta p$ for the entire face plate at FPL. Combining this with the appropriate mass flow rate at FPL one obtains an $AC_d$ for the entire face plate of 3.32 in$^2$. This should be equal to the sum of the $AC_d$ products for all of the fuel holes and the annular injection elements. The actual combined hole area is 0.792 in$^2$ and that of the combined annular fuel elements is 3.59 in$^2$. Assuming a $C_d$ of 1.0 for the holes we thus determine $C_d$ for the fuel elements to be 0.704. Because of the geometry of the oxygen injection path through the center of the injection elements it is reasonable to expect that the $C_d$ for these holes is essentially unity.

With this information an effective area which should be modeled by the computational grid for each annular inlet plane region can be compiled and is presented in Table 3-1. Note that if a mass flow rate is computed for each region and is input uniformly over that part of the grid the stream will have the incorrect momentum. It is absolutely necessary that the areas of the
Table 3-1 EFFECTIVE AREAS (IN SQUARE INCHES) OF EACH ANNULAR ZONE (360 DEGREES) OF PREBURNER INJECTOR FACE-PLATE FOR COMPUTATIONAL MODELING OF INLET PLANE

<table>
<thead>
<tr>
<th>Zone</th>
<th>Fuel Elements</th>
<th>Holes</th>
<th>Baffles</th>
<th>Oxidizer Injection Area</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.54639</td>
<td>0.075745</td>
<td>0.0093</td>
<td>0.335934</td>
</tr>
<tr>
<td>2</td>
<td>0.48568</td>
<td>0.041849</td>
<td>0.0093</td>
<td>0.298608</td>
</tr>
<tr>
<td>3</td>
<td>0.42497</td>
<td>0.036270</td>
<td>0.0093</td>
<td>0.261282</td>
</tr>
<tr>
<td>4</td>
<td>0.36426</td>
<td>0.030690</td>
<td>0.0093</td>
<td>0.223956</td>
</tr>
<tr>
<td>5</td>
<td>0.30355</td>
<td>0.037599</td>
<td>0.0093</td>
<td>0.186630</td>
</tr>
<tr>
<td>6</td>
<td>0.24284</td>
<td>0.030750</td>
<td>0.0093</td>
<td>0.149304</td>
</tr>
<tr>
<td>7</td>
<td>0.24284</td>
<td>0.029243</td>
<td>0.0093</td>
<td>0.149304</td>
</tr>
<tr>
<td>8</td>
<td>0.18213</td>
<td>0.025620</td>
<td>0.00853</td>
<td>0.111978</td>
</tr>
<tr>
<td>9</td>
<td>0.12142</td>
<td>0.015373</td>
<td>0.00615</td>
<td>0.074652</td>
</tr>
<tr>
<td>10</td>
<td>0.12142</td>
<td>0.020500</td>
<td>0.0214</td>
<td>0.074652</td>
</tr>
<tr>
<td>11</td>
<td>0.12142</td>
<td>0.020500</td>
<td>0.0214</td>
<td>0.074652</td>
</tr>
<tr>
<td>12</td>
<td>0.12142</td>
<td>0.020500</td>
<td>0.0214</td>
<td>0.074652</td>
</tr>
<tr>
<td>13</td>
<td>0.12142</td>
<td>0.020500</td>
<td>0.0214</td>
<td>0.074652</td>
</tr>
<tr>
<td>14</td>
<td>0.12142</td>
<td>0.020500</td>
<td>0.0214</td>
<td>0.074652</td>
</tr>
<tr>
<td>15</td>
<td>0.12142</td>
<td>0.020500</td>
<td>0.0214</td>
<td>0.074652</td>
</tr>
<tr>
<td>16</td>
<td>0.12142</td>
<td>0.020500</td>
<td>0.0214</td>
<td>0.074652</td>
</tr>
<tr>
<td>17</td>
<td>0.12142</td>
<td>0.020500</td>
<td>0.0214</td>
<td>0.074652</td>
</tr>
</tbody>
</table>
regions of mass injection in each ring on the inlet plane closely match the appropriate effective area. Thus many of the inlet plane nodes in the grid will be "turned off."

3.2.2 Chemistry Model

During the startup transient it is not clear how much and where combustion or burning occurs. It cannot be assumed that during this time interval the chemistry is completed within a short distance of the face plate. Much research was performed to ascertain the most appropriate chemistry model to employ for computing a realistic flow-kinetic picture for this problem.

It was determined that a non-equilibrium chemistry model be used to model the details of the kinetics. The eight reactive species are $H_2$, $O_2$, $H_2O$, $H_2O_2$, $H$, $O$, $OH$, and $HO_2$. A 20-reaction set was selected, eight of which apply to ignited hot portions of the flow field and 12 of which model ignition and cool flows. Tables 3-2 and 3-3 list these reactions along with their rate coefficients. Literature references are also included for completeness.

A reduced set of reactions can be substituted for these 20-reactions for a less complete treatment. How much of the physics is lost, if any, can only be discovered by computing a test problem using both sets. This has not been done.

The reduced set is obtained as follows:

1. Use Reactions 1 through 6 in Table 3-2
2. Use Reaction 9 in Table 3-3
3. Rewrite Reaction 10 as

\[
H + O_2 + M \rightarrow 0.5 H_2 + O_2 + M
\]

The eighth reaction is forward only, and the rate coefficient should be set equal to two thirds that for Reaction 10. Thus the minimum recommended chemistry treatment should contain eight reactions and six species.
Table 3-2 KINETIC REACTIONS (1 THROUGH 8)

<table>
<thead>
<tr>
<th>N Body Reaction (Reversible)</th>
<th>Rate Coefficient ((cm^3\text{ particle}^{-1})(N^{-1})\text{ sec}^{-1}) units</th>
<th>Ref.</th>
</tr>
</thead>
<tbody>
<tr>
<td>OH + H(_2) → H(_2)O + H</td>
<td>(1.7 \times 10^{-16}T^{1.6}\exp(-3,300/RT))</td>
<td>1, 2</td>
</tr>
<tr>
<td>H + O(_2) → OH + O</td>
<td>(2.5 \times 10^{-7}T^{-0.9}\exp(-17,330/RT))</td>
<td>3, 4</td>
</tr>
<tr>
<td>O + H(_2) → OH + H</td>
<td>(1.8 \times 10^{-20}T^{2.8}\exp(-5,920/RT))</td>
<td>4</td>
</tr>
<tr>
<td>OH + OH → H(_2)O + O</td>
<td>(1.4 \times 10^{-20}T^{2.7}\exp(1,800/RT))</td>
<td>5, 6</td>
</tr>
<tr>
<td>H + OH + M → H(_2)O + M</td>
<td>(2.4 \times 10^{-26}T^{-2})</td>
<td>7, 8</td>
</tr>
<tr>
<td>H + H + M → H(_2) + M</td>
<td>(2.0 \times 10^{-30}T^{-1}) (M \neq H(_2))</td>
<td>9, 2</td>
</tr>
<tr>
<td>H + O + M → OH + M</td>
<td>(2.8 \times 10^{-31}T^{-0.6}) (M = H(_2))</td>
<td>2</td>
</tr>
<tr>
<td>O + O + M → O(_2) + M</td>
<td>(4.0 \times 10^{-29}T^{-1})</td>
<td>10</td>
</tr>
</tbody>
</table>

Table 3-3 KINETIC REACTIONS (9 THROUGH 20)

<table>
<thead>
<tr>
<th>N Body Reaction (Reversible)</th>
<th>Rate Coefficient ((cm^3\text{ particle}^{-1})(N^{-1})\text{ sec}^{-1}) units</th>
<th>Ref.</th>
</tr>
</thead>
<tbody>
<tr>
<td>H(_2) + O(_2) → OH + OH</td>
<td>(2.8 \times 10^{-11}\exp(-48,100/RT))</td>
<td>12</td>
</tr>
<tr>
<td>H + O(_2) + M → HO(_2) + M</td>
<td>(2.0 \times 10^{-30}T^{-0.8})</td>
<td>13</td>
</tr>
<tr>
<td>HO(_2) + OH → H(_2)O + O(_2)</td>
<td>(1.6 \times 10^{-11}\exp(1,000/RT))</td>
<td>14</td>
</tr>
<tr>
<td>HO(_2) + H → H(_2) + O(_2)</td>
<td>(4.2 \times 10^{-11}\exp(-700/RT))</td>
<td>9, 2</td>
</tr>
<tr>
<td>HO(_2) + H → OH + OH</td>
<td>(1.9 \times 10^{-10}\exp(-1,000/RT))</td>
<td>16</td>
</tr>
<tr>
<td>HO(_2) + O → OH + O(_2)</td>
<td>(3.3 \times 10^{-11}\exp(400/RT))</td>
<td>17</td>
</tr>
<tr>
<td>HO(_2) + HO(_2) → H(_2)O(_2) + O(_2)</td>
<td>(3.2 \times 10^{-12}\exp(2,300/RT))</td>
<td>15</td>
</tr>
<tr>
<td>HO(_2) + H(_2) → H(_2)O(_2) + H</td>
<td>(3.2 \times 10^{-12}\exp(-21,000/RT))</td>
<td>15</td>
</tr>
<tr>
<td>H(_2)O(_2) + OH → HO(_2) + H(_2)O</td>
<td>(3.7 \times 10^{-12}\exp(-500/RT))</td>
<td>18</td>
</tr>
<tr>
<td>H(_2)O(_2) + H → OH + H(_2)O</td>
<td>(1.7 \times 10^{-11}\exp(-3,600/RT))</td>
<td>2</td>
</tr>
<tr>
<td>H(_2)O(_2) + O → OH + HO(_2)</td>
<td>(8.7 \times 10^{-13}\exp(-3,600/RT))</td>
<td>20</td>
</tr>
<tr>
<td>H(_2)O(_2) + M → OH + OH + M</td>
<td>(1.2 \times 10^{-7}\exp(-46,300/RT))</td>
<td>19</td>
</tr>
</tbody>
</table>
4. SUMMARY AND CONCLUSIONS

A three-duct SSME Hot Gas Manifold geometry code has been developed for use by the computational mechanics staff of NASA-MSFC. This report describes the methodology of the program, makes recommendations on its implementation, and provides an input guide, and input deck listing, and a source code listing. The code listing is strewn with an abundance of comments to assist the user in following its development and logic. A working source deck will be provided to NASA-MSFC on the EADS network upon request.

A thorough analysis has been made of the proper boundary conditions and chemistry kinetics necessary for an accurate computational analysis of the flow environment in the SSME fuel side preburner chamber during the initial startup transient. Pertinent results have been presented to facilitate incorporation of these findings into an appropriate CFD code. The computation must be a turbulent computation, since the flow field turbulent mixing will have a profound effect on the chemistry. Because of the additional equations demanded by the chemistry model it is recommended that for expediency a simple algebraic mixing length model be adopted.

Performing this computation for all or selected time intervals of the startup time will require an abundance of computer CPU time regardless of the specific CFD code selected.
5. REFERENCES


6. Wagner, G., and R. Zellner, "Temperature Dependence of the Reaction OH + OH → H₂O + O," Ber. Bunsenges, Phys. Chem., Vol. 85, 1981, pp. 1122-1128. (As reviewed in Ref. 5, these authors' results for Reaction 4 were written as k₄ = 9.06 x 10⁻¹³ exp( ) over the range 250-2000 K. This expression has been refit here to the format k = ATⁿ exp -(E/RT) for use in computations. The other reviews examined to not incorporate data beyond 1977.)


8. Troe, J., "Atom and Radical Recombination Reactions," in Ann. Rev. Phys. Chem., Vol. 29, Annual Reviews, Palo Alto, 1978, pp. 223-250. (Low pressure recombination results in Ar from Ref. 8 - reviewed further in Ref. 8 have been utilized here to obtain the recommended expression.)

5. REFERENCES (Continued)

10. Schofield, K., "Evaluated Chemical Kinetic Rate Constants for Various Gas Phase Reactions," J. Phys. Chem., Reference Data, Vol. 2, 1973, pp. 25-84. (Schofield's recommendation that \( k = 2 \times 10^{-32} \) over the 1000-3000 K range has been refit here to a \( T^{-1} \) dependence anchored at 2000 K. There is no more recent input for this poorly known rate coefficient.)


14. Data reviewed in Ref. 15 by Kaufman and Sherwell indicate a preferred room temperature value \( f 6.6 \times 10^{-11} \) at low pressures and \( 10.7 \times 10^{-11} \) at pressures nearer latm; also, flame studies and shock tube work give lower results \( (2.0 \times 10^{-11} \text{ at } 2130 \text{ K}) \) at higher temperatures. These data are reconciled by the interim expression recommended here.


16. The expression recommended for \( k_{12} + k_{13} \) at room temperature, i.e., \( 4.8 \times 10^{-11} \), utilizing the indicated volume for \( k_{12} \).


18. The expression given here for \( k_{17} \) is based on a simple Arrhenius fit to the value \( k_{17} = 1.6 \times 10^{-12} \) at room temperature recommended by Kaufman and Sherwell (Ref. 15 and the measured ratio of \( 3.2 \pm 0.6 \) for \( k_{17}/k_1 \) at 773 K by Baldwin et al. (Ref. 19), yielding \( k_{17} = 2.7 \times 10^{-12} \) at 773 K, utilizing the recommended value for \( k_1 \).
5. REFERENCES (Concluded)


20. The expression given here for \( k^g \) is based on a simple Arrhenius fit to the mean of the values for \( k_{19} \) (2.3 \( \times \) 10\(^{-19} \) and 1.7 \( \times \) 10\(^{-19} \)) at room temperature recommended by Baulch et al. (Ref. 21) and by DeMore et al. (Ref. 17), and the measured ratio of 10.4 for \( k_{19}/k_2 \) at 773 K by Baldwin et al. (Ref. 19) yielding \( k_{19} = 8.3 \times 10^{-14} \) at 773 K, utilizing the recommended value for \( k_1 \).

Appendix A
GEOMETRY INPUT GUIDE
Appendix A

INTRODUCTION

The geometry input guide is presented in two sections: (1) a definition of terminology commonly used for inputting and describing the geometry, and (2) a summary of card types used to input the geometry and a detailed description of the associated parameters and their input values.

We begin with an overview of how to apply the program. The flowfield domain is divided into zones in order to simplify the input necessary to describe the complicated geometry. Each zone contains its own coordinate system. A zone consists of at least one section. A multiple section zone is used to describe Zones 4 and 5. Sections may be added in only one internal coordinate direction which is called the marching direction since this is the direction in which the grid is created. Each section is described using points, edges, and surfaces. An edge may consist of from one to ten segments. A segment or a surface may require additional input depending on its type.

The second section presents a detailed description of the input parameters. Each card type is listed in the order of input with its associated parameters. Each parameter is identified as to its usage in the program with the options of each shown. Reference to Fig. A-1 or Table A-1 may be necessary to explain some of the input parameters and their order of input. All of the card types are not necessarily input for a specific zone.

Card type 9 may be used when other than an equal distribution of nodes is desired, etc. Whereas card type 13 is necessary if there will be more than one segment per edge. And cards type 10 and 11 are used if additional information is needed to describe a segment or a surface. Certain of the input parameters on early cards dictate which of the later cards are read in.
Fig. A-1 General Hexahedral: Numbering of Points, Edges, and Surfaces
DEFINITION OF TERMINOLOGY

Edge
An edge consists of from one to five segments. Four edges are used to describe a 2D section. Twelve edges are used to describe a 3D section.

Map
The geometry maps a point from $\eta$ space into real space. When describing a surface mapping we could say map $= 2$ refers to a planar $\eta$ space surface being mapped onto a cylindrical surface in real space.

Node
At each intersection of $\eta$ coordinates a node is generated by the program forming the grid which will describe the flowfield domain.

Point
The corners of a section are called "points." The location and initial flow directions are input for each point. There are eight points.

Section
A zone may be further subdivided into sections. Each zone consists of at least one section. A zone may only be subdivided into sections in the direction in which the grid is generated (marching direction). Points, edges or a surface in common with a previous section within a zone are not reinput. Sectioning is done primarily to reduce the amount of input. Each section is described using points, edges, and surfaces.

Segment
An edge is subdivided into as many as five segments. A segment may be a straight line, a circular arc, a helical coil, a trigonometric function of angle or length, a cubic spline, or user defined.

Surface
A three-dimensional section will consist of six surfaces which form a generalized hexahedron. A surface may be planar, cylindrical, an edge of revolution, or user defined.

Zone
The flowfield domain may be subdivided into zones. Zones are generated independently and are the fundamental building block. Each zone contains its own $\eta$ coordinate system.

$\eta_1, \eta_2, \eta_3$
Localized coordinate directions within a zone. These coordinates describe a cube in $\eta$ space. $0 \leq \eta_i \leq 1$
## SUMMARY OF CARDS

<table>
<thead>
<tr>
<th>Card Type</th>
<th>Parameter List/Format</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Flowfield Parameters</strong></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>ITITLE(I), I=1,80 (20A4)</td>
</tr>
<tr>
<td>2</td>
<td>NZONE, MAPTEN, INCHES (8I5)</td>
</tr>
<tr>
<td>3</td>
<td>IWRTI, IWRTC, IWRTN (3I5)</td>
</tr>
<tr>
<td><strong>Zone Parameter</strong></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>NSECT (I5)</td>
</tr>
<tr>
<td><strong>Section Parameters</strong></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>MAPEDGE(I), I=1,12 (12I5) or (3(4I10))</td>
</tr>
<tr>
<td>6</td>
<td>MAPSIDE(I), I=1,6 (6I5)</td>
</tr>
<tr>
<td>7</td>
<td>MARCH (I5)</td>
</tr>
<tr>
<td>8</td>
<td>(NMBRNDNS(I), I=1,3), (ISTRTCH(I), I=1,3) (6I5)</td>
</tr>
<tr>
<td>9</td>
<td>STRETCH(I), I=1,3 (3E10.4)</td>
</tr>
<tr>
<td>10</td>
<td>[(COEFE(I,K,J), I=1,8), K=1,5], J=1,4(2D) OR 12(3D) (8E10.4)</td>
</tr>
<tr>
<td>11</td>
<td>[COEFS(I,J), I=1,8], J=1,6 (8E10.4)</td>
</tr>
<tr>
<td>12</td>
<td>(POINT(I,J), 8=15), J=1, 8 (8E10.4)</td>
</tr>
<tr>
<td>13</td>
<td>[(SEGMAX(I,K,J), I=1,5), ETAMAX(K,J), K=1,4] J=1, 12 (6E10.4)</td>
</tr>
</tbody>
</table>
CARD AND VARIABLE DESCRIPTIONS

Flowfield Parameters

CARD TYPE 1

Problem Identification Label

ITITLE
Alphanumeric information used for identifying the flowfield geometry. Columns 1-80 are read and printed only.

CARD TYPE 2

Problem Option Controls Flags

NZONE
The number of zones into which the flowfield geometry is divided. The maximum number of zones is 99.

MAPTEN
This option determines the maximum number of segments which will be input per edge.

= 0 Five segments per edge Format(12I5)
= 1 Ten segments per edge Format(3(4I10))

INCHES
This option specifies the dimensions of the coordinates being input. The output data will be written in feet for compatibility with the INTEGRATION program.

= 0 Dimensions in feet
= 1 Dimensions in inches

CARD TYPE 3

Output Print Options

IWRTI
This option may be used when either grid plots or nodal printout are insufficient to determine the cause of an error in the geometry. Intermediate debug printout may be generated in subroutines EDGE and SURFACE. Subroutine EDGE prints the following information: node number, edge number, position on the edge, \( \eta \) value at this position on the edge, a ratio which gives the relative location on a segment, and the maximum \( \eta \) value for this segment of the edge. Subroutine SURFACE prints the following information: node number, surface number, and position on the surface. It also prints the following information for an edge of revolution: node number, surface number, position on the edge of revolution to be revolved, a vector tangent to the edge at this position, edge number, the distance along the axis of revolution from the origin of the relative coordinate system to a perpendicular from the axis to the point on the revolved edge, and the radius along this perpendicular from the axis to the revolved point on the edge. IWRTI should be chosen carefully to restrict the amount of printout.
= 0     No debug print
= N > 0  Intermediate error printout for every N^{th} node
= N > Total number of nodes - Prints the number of nodes stored plane.

**IWRTC**

This option controls the printout of connectivities for all nodes specified for MATING. Incorrect connectivities result in various types of errors in the INTEGRATION program and a person familiar with it should be consulted if this type of error is suspected.

= 0     Do not print mated connectivities
= 1     Print mated connectivities

**IWRTN**

This option controls the amount of nodal printout for a run. The nodal information will be printed will be node number and position. For large problems with many thousands of nodes, IWRTN should be chosen carefully to restrict the amount of paper produced. A plot of the grid is usually more instructive than a massive nodal printout for locating errors in the code.

= 0     No printout
= N > 0  Print nodal point information every N^{th} node

**Zone Parameter**

**CARD TYPE 4**

**NSECT**

The number of sections in this zone. A zone may only be sectioned in the \( \eta \) direction which corresponds to MARCH. There is no limit to the number of sections within a zone.

**Section Parameters**

**CARD TYPE 5**

**MAPEDGE(I), I=1, 12**

These are packed integer flags that specify which edge shape functions will be used for the current section. The edges are input in numerical order. The edge numbers are defined according to Fig. A-1. The user should study this figure before inputting the geometry.
Each of the edges may consist of up to ten segments with each of these segments having its own shape function. The value of MAPEDGE(I) can consist of up to ten integers packed into one word MAPEDGE(I). MAPTEN specifies the maximum number of segments per edge. The edge shape function indicators for each segment are input in chronological order of increasing n for each edge with the final packed integer being right adjusted. For example, if MAPEDGE(4) = 112, then edge 4 consists of three segments: the first segment is type 1; the second segment is type 1; and the third segment is type 2. If only one segment describes an edge, then only one indicator is used, right adjusted.

A library of edge shape function indicators for the HGM GEOMETRY program follow. If any edge shape function other than a linear segment is specified, then edge coefficients (COEFE(I)) must be input. Card type 1 is used to define the analytical function describing a segment.

= 1 Linear segment
= 2 Circular arc (input COEFE(I))
= 3 Edge of revolution (input COEFE(I))
= 4 Special segment (input COEFE(I))
= 5 Special segment (input COEFE(I))

CARD TYPE 6 Surface Shape Function Indicators Format(6I5)

MAPSIDE(I),I=1,6

These are integer flags that specify which surface shape functions will be used for the current section. These flags are input only for three-dimensional problems since two-dimensional geometries are defined completely by the edge functions. The surfaces are input in numerical order. The surface numbers are defined in Fig. A-1. The user should study this figure before inputting the geometry. An edge of revolution requires the input of surface coefficients (COEFS(I)) on card type 15 to define a relative origin on the axis, the axis of revolution, and the direction of revolution.

= 1 Planar surface
= 2 Cylindrical surface
= 3 Special surface (user defined)
= 4 Edge of revolution (input COEFS(I))
= 5 Holes in bowl surface
= 6 Hole side of duct surface
= 7 Duct outer surface
CARD TYPE 7  Node Numbering Sequence Specs  Format(6I5)

MARCH(I5) The value of MARCH determines the node generation and hence the
node numbering sequence. (default = 1)

The numbering sequence corresponding to follows.

= 1 \eta_3, \eta_2, \eta_1 (default)
= 2 \eta_1, \eta_3, \eta_2
= 3 \eta_2, \eta_1, \eta_3

CARD TYPE 8  Node Distribution Parameters  Format(6I5)

NMBRND(I), I=1 3

NMBRND(I) is the number of nodes in the \eta_I direction for
the current section. The limit is 200 nodes in any coordinate
direction. This may be changed in the program by respecifying
the ETAS(3,200) array.

ISTRCH(I), I=1 3

This option gives the user control over the node distribution
in each of the coordinate directions.

= 0 Uniform spacing
= 1 Input actual \eta_I values for NMBRND(I) nodes
  (input ETAS(I))
= 2 Decrease spacing in \eta_I direction. Input a stretching
  factor greater than 0.0 in STRETCH(I).
= 3 Increase spacing in \eta_I direction. Input a stretching
  factor greater than 0.0 in STRETCH(I).
= 4 Double stretching. Input a stretching factor greater than
  0.0 in STRETCH(I). Use an odd number of nodes.
= 5 Decrease spacing in \eta_I direction. Input minimum grid
  spacing as a percentage of the total length in STRETCH(I).
= 6 Increase spacing in \eta_I direction. Input minimum grid
  spacing as a percentage of the total length in STRETCH(I).
= 7 Double stretching. Input minimum grid spacing as a
  percentage of the total length in STRETCH(I). Use an odd
  number of nodes.

If ISTRCH(I) = 1, input a set of cards type 13 for each \eta
direction to be input.
If ISTRCH(I) \geq 2, input card type 12.
CARD TYPE 9  Option for Stretching Function  
(input when ISTRCH(I) ≥ 2)  

ISTRCH(I).I=1, 3

This parameter is input for each coordinate direction designated for stretching by ISTRCH(I) ≥ 2.

Example: Several stretching functions will be demonstrated using 21 points for comparison. Note, that total length = 10.0 for ISTRCH = 6 and 7.

ISTRCH = 3

STRETCH = 2.0

STRETCH = 4.0

STRETCH = 6.0

STRETCH = 8.0

STRETCH = 10.0

ISTRCH = 4

STRETCH = 2.0

A-9

LOCKHEED-HUNTSVILLE ENGINEERING CENTER
CARD TYPE 14  Coefficients for Edge Shape Functions
            Format(8E10.4)
            (input for each segment in MAPEDGE(I) > 1)

COEFE(I),I=1,8

These coefficients are used to describe the edge shape
functions for each segment of the current section. The
coefficients for each segment are input on separate cards in
the same order as the indicators on card type 5.
Map Type Input Parameters

2 Circular Arc COEFE(I), I=1,3 are the x, y, and z coordinates of the center of the arc.

3 Edge of Revolution COEFE(I), I=1,3 are coordinates of the center of the arc.
COEFE(I), I=4,6 are components of the unit vector along the axis. Direction according to right hand rule when a vector is revolved from point 1 to point 2.

CARD TYPE 15 Coefficients for Surface Shape Functions Format(8E10.4)
(input when MAPSIDE(I) = 4 and IDIM = 3)

COEFS(I), I=1,8

These are the coefficients defining the surface shape functions for each surface formed by an edge of revolution in the current section. Each surface which has MAPSIDE(I) = 4 on card type 8 is input on a separate card in the same order as they occur on card type 8.

Map Type Input Parameters

4 Surface of Revolution Surface formed by revolving an edge about an axis.
COEFS(I), I=1,3 a point on the axis of revolution which becomes the origin of a local coordinate system. This point must lie outside of the projection of the edge onto the axis of revolution.
COEFS(I), I=4,6 are components of the unit vector along the axis of revolution in the direction of increasing n.
COEFS(7) indicates the η direction in which the edge will revolve.

CARD TYPE 16 Coordinates of Points and flow direction Format(5E10.4)

POINT(I,J), J = Point number

These parameters are the coordinates and flow direction at each corner of a general hexahedral(3D). Figure A-1 shows this configuration with the points numbered from 1 to 8. There are eight cards of type 16 to be input.
POINT(1,J) - the x coordinate of point J
POINT(2,J) - the y coordinate of point J
POINT(3,J) - the z coordinate of point J
POINT(4,J) - the flow angle $\theta$ at point J
POINT(5,J) - the flow angle $\phi$ at point J

Important Note: All cards type 12 are not input consecutively. They are grouped with cards type 13. See Table A-1 for the exact sequence of card types 11 and 13.

CARD TYPE 17 Segment Extremals for Edges Format(6E10.4)

SEGMAX(1,K,J), K = Segment Number, J = Edge Number

Each edge may be segmented up to five times. Therefore, cards type 13 are repeated for each successive segment of edge J. Each segment must be input on a separate card type 13. The extremal for the final segment of an edge is not to be input since this point is already defined by the POINT(I) input. The number of cards type 13 for each edge will thus be one less than the number of segments on that edge. In particular, if an edge consists of only one segment, no cards of type 13 are input for that edge.

See Table 1 for the input order of card types 12 and 13. Each POINT(I) is input on a single card, followed by up to five cards containing the extremals.

SEGMAX(1,K,J) - The extremal x coordinate for the Kth segment of edge J.
SEGMAX(2,K,J) - The extremal y coordinate for the Kth segment of edge J.
SEGMAX(3,K,J) - The extremal z coordinate for the Kth segment of edge J.
SEGMAX(4,K,J) - The extremal flow angle $\theta$ for the Kth segment of edge J.
SEGMAX(5,K,J) - The extremal flow angle $\phi$ for the Kth segment of edge J.

ETAMAX(K,J), K = Segment Number, J = Edge Number

The maximum value of the $\eta_T$ coordinate on the Kth segment of edge J. Input a negative value when defining a fold line.
<table>
<thead>
<tr>
<th>Order</th>
<th>Card Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>12</td>
<td>Point 1</td>
</tr>
<tr>
<td>2</td>
<td>13</td>
<td>Extremals for each segment of Edge 1</td>
</tr>
<tr>
<td>3</td>
<td>12</td>
<td>Point 2</td>
</tr>
<tr>
<td>4</td>
<td>13</td>
<td>Extremals for each segment of Edge 2</td>
</tr>
<tr>
<td>5</td>
<td>12</td>
<td>Point 3</td>
</tr>
<tr>
<td>6</td>
<td>13</td>
<td>Extremals for each segment of Edge 3</td>
</tr>
<tr>
<td>7</td>
<td>12</td>
<td>Point 4</td>
</tr>
<tr>
<td>8</td>
<td>13</td>
<td>Extremals for each segment of Edge 4</td>
</tr>
<tr>
<td>9</td>
<td>12</td>
<td>Point 5</td>
</tr>
<tr>
<td>10</td>
<td>13</td>
<td>Extremals for each segment of Edge 5</td>
</tr>
<tr>
<td>11</td>
<td>12</td>
<td>Point 6</td>
</tr>
<tr>
<td>12</td>
<td>13</td>
<td>Extremals for each segment of Edge 6</td>
</tr>
<tr>
<td>13</td>
<td>12</td>
<td>Point 7</td>
</tr>
<tr>
<td>14</td>
<td>13</td>
<td>Extremals for each segment of Edge 7</td>
</tr>
<tr>
<td>15</td>
<td>12</td>
<td>Point 8</td>
</tr>
<tr>
<td>16</td>
<td>13</td>
<td>Extremals for each segment of Edge 8</td>
</tr>
<tr>
<td>17</td>
<td>13</td>
<td>Extremals for each segment of Edge 9</td>
</tr>
<tr>
<td>18</td>
<td>13</td>
<td>Extremals for each segment of Edge 10</td>
</tr>
<tr>
<td>19</td>
<td>13</td>
<td>Extremals for each segment of Edge 11</td>
</tr>
<tr>
<td>20</td>
<td>13</td>
<td>Extremals for each segment of Edge 12</td>
</tr>
</tbody>
</table>
Notes on Input Sequence of Cards Type 4 Through 13

- Card types 5-13 are repeated for each section in a zone. Sections can be added only in the direction of MARCH.

- Card type 4 is repeated for each zone.

- Cards type 9-13 are not re-input for points, edges and surfaces in common with the preceding section. Since these have already been input, the code transfers the entries from one section to another.

- The first section of each new zone, however, must be input since zones are considered to be independent.
Appendix B
HGM GRID CODE INPUT LISTING
## Appendix B

### ZONE 1 (TAD) FOR 3-DUCT HGM GEOMETRY

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>3</th>
<th>1</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>10000</td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**ZONE 1**

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>311</th>
<th>1</th>
<th>1311</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>4</th>
<th>1</th>
<th>4</th>
<th>4</th>
<th>4</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>94</th>
<th>21</th>
<th>72</th>
<th>0</th>
<th>4</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.0</td>
<td>8.0</td>
<td>0.0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1.35</td>
<td>0.0</td>
<td>-6.0</td>
<td>0.0</td>
<td>-1.0</td>
<td>0.0</td>
</tr>
<tr>
<td>1.35</td>
<td>0.0</td>
<td>-5.875</td>
<td>0.0</td>
<td>-1.0</td>
<td>0.0</td>
</tr>
<tr>
<td>3.31</td>
<td>0.0</td>
<td>0.0</td>
<td>1.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>6.07</td>
<td>0.0</td>
<td>0.0</td>
<td>1.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>6.07</td>
<td>0.0</td>
<td>0.0</td>
<td>1.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>3.31</td>
<td>0.0</td>
<td>0.0</td>
<td>1.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>1.35</td>
<td>0.0</td>
<td>6.0</td>
<td>0.0</td>
<td>1.0</td>
<td>0.0</td>
</tr>
<tr>
<td>1.35</td>
<td>0.0</td>
<td>5.875</td>
<td>0.0</td>
<td>1.0</td>
<td>0.0</td>
</tr>
<tr>
<td>6.07</td>
<td>0.0</td>
<td>0.0</td>
<td>1.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>6.07</td>
<td>0.0</td>
<td>0.0</td>
<td>1.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>6.07</td>
<td>0.0</td>
<td>0.0</td>
<td>1.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>6.07</td>
<td>0.0</td>
<td>0.0</td>
<td>1.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>3.31</td>
<td>0.0</td>
<td>0.0</td>
<td>-5.55</td>
<td>18.0</td>
<td></td>
</tr>
<tr>
<td>1.35</td>
<td>0.0</td>
<td>-5.55</td>
<td>18.0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1.35</td>
<td>0.0</td>
<td>-6.45</td>
<td>56.0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4.02</td>
<td>0.0</td>
<td>-6.5</td>
<td>81.0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6.07</td>
<td>0.0</td>
<td>-6.5</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6.07</td>
<td>0.0</td>
<td>-7.434</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1.35</td>
<td>0.0</td>
<td>-4.525</td>
<td>18.0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1.35</td>
<td>0.0</td>
<td>-7.225</td>
<td>55.0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4.02</td>
<td>0.0</td>
<td>-7.434</td>
<td>81.0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3.31</td>
<td>0.0</td>
<td>-4.525</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3.31</td>
<td>0.0</td>
<td>5.55</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6.07</td>
<td>0.0</td>
<td>6.5</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6.07</td>
<td>0.0</td>
<td>7.434</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3.31</td>
<td>0.0</td>
<td>4.525</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1.35</td>
<td>0.0</td>
<td>5.55</td>
<td>18.0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1.35</td>
<td>0.0</td>
<td>6.45</td>
<td>56.0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4.02</td>
<td>0.0</td>
<td>6.5</td>
<td>81.0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1.35</td>
<td>0.0</td>
<td>4.525</td>
<td>18.0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1.35</td>
<td>0.0</td>
<td>7.225</td>
<td>55.0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4.02</td>
<td>0.0</td>
<td>7.434</td>
<td>81.0</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### ZONE 2 (1/2 BOWL) FOR 3-DUCT HGM GEOMETRY

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>3</th>
<th>1</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>10000</td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**ZONE 2**

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>11111111</th>
<th>1</th>
<th>121111</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>3</td>
<td>3</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>11111111</th>
<th>1</th>
<th>121111</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>4</td>
<td>1</td>
<td>4</td>
</tr>
</tbody>
</table>

| 3 |   |   |   |   |   |   |

| B-1 |   |   |   |   |   |   |

LOCKHEED-HUNTSVILLE ENGINEERING CENTER
ZONE 3 (1/2 BOWL) FOR 3-DUCT HGM GEOMETRY

ZONE 3

1 11111111 1 1211111 1
1 3 3 3 3 3 3 3
1 11111 1 121 1
1 4 4 5 5 4 4
3 0 1 1 1
4.0 20.0 46.0
13.0 41.0 58.0
3.0 4.0 4.0
3.0 4.0 4.0 3.0
59 21 72 0 4 0
0.0 8.0 0.0
9.445 0.0 0.0
0.0 0.0 0.0 1.0 0.0 0.0
0.0 0.0 0.0 1.0 0.0 0.0
0.0 0.0 0.0 1.0 0.0 0.0
0.0 0.0 0.0 1.0 0.0 0.0
9.445 0.0 0.0
<table>
<thead>
<tr>
<th>Zone 4 (Top T-D) for 3-Duct GGM Geometry</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
</tr>
<tr>
<td>10000</td>
</tr>
<tr>
<td>3</td>
</tr>
<tr>
<td>4</td>
</tr>
<tr>
<td>4</td>
</tr>
<tr>
<td>7</td>
</tr>
</tbody>
</table>

ZONE 4

| 27 | 20 | 29 | 4 | 0 | 4 |
| 7.0 | 0.0 | 7.0 |
| 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 |
| 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 |
| 9.445 | 5.564 | 0.0 | -0.25506 | -0.092215 | -0.96252 |
| 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 |
| 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 |
| 9.445 | 5.564 | 0.0 | -0.25506 | -0.092215 | -0.96252 |
| 9.445 | 5.564 | 0.0 | 0.25506 | 0.092215 | 0.96252 |
| 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 |
| 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 |
| 9.445 | 5.564 | 0.0 | 0.25506 | 0.092215 | 0.96252 |
| 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 |
| 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 |
| 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 |
| 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 |
| 8.691001 | 7.428427 | 2.549918 |

LOCKHEED-HUNTSVILLE ENGINEERING CENTER
ZONE 5 (MID T-D) FOR 3-DUCT HGM GEOMETRY

<table>
<thead>
<tr>
<th>Zone</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

ZONE 5

B-4

LOCKHEED-HUNTSVILLE ENGINEERING CENTER
<p>| | | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>7.0</td>
<td>0.0</td>
<td>5.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>7.32766</td>
<td>0.0</td>
<td>0.37334</td>
<td>-0.40139</td>
<td>0.0</td>
<td>-0.91591</td>
<td></td>
</tr>
<tr>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>7.32766</td>
<td>0.0</td>
<td>0.37334</td>
<td>-0.40139</td>
<td>0.0</td>
<td>-0.91591</td>
<td></td>
</tr>
<tr>
<td>7.32766</td>
<td>0.0</td>
<td>0.37334</td>
<td>0.40139</td>
<td>0.0</td>
<td>0.91591</td>
<td></td>
</tr>
<tr>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>9.040005</td>
<td>1.695449</td>
<td>7.695033</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>11.889400</td>
<td>1.695342</td>
<td>7.307725</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>12.3459</td>
<td>1.5014</td>
<td>8.6854</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>12.5633</td>
<td>1.5014</td>
<td>9.6966</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8.233353</td>
<td>0.0</td>
<td>7.796413</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>12.667614</td>
<td>0.0</td>
<td>7.201867</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>12.98742</td>
<td>0.0</td>
<td>8.405000</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>9.39706</td>
<td>0.0</td>
<td>9.97844</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>1</td>
<td>3</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>1</td>
<td>1</td>
<td>4</td>
<td>4</td>
<td></td>
<td></td>
</tr>
<tr>
<td>27</td>
<td>25</td>
<td>16</td>
<td>4</td>
<td>0</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>7.0</td>
<td>0.0</td>
<td>5.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>7.32766</td>
<td>0.0</td>
<td>0.37334</td>
<td>-0.40139</td>
<td>0.0</td>
<td>-0.91591</td>
<td></td>
</tr>
<tr>
<td>7.32766</td>
<td>0.0</td>
<td>0.37334</td>
<td>0.40139</td>
<td>0.0</td>
<td>0.91591</td>
<td></td>
</tr>
<tr>
<td>7.32766</td>
<td>0.0</td>
<td>0.37334</td>
<td>0.40139</td>
<td>0.0</td>
<td>0.91591</td>
<td></td>
</tr>
<tr>
<td>7.32766</td>
<td>0.0</td>
<td>0.37334</td>
<td>-0.40139</td>
<td>0.0</td>
<td>-0.91591</td>
<td></td>
</tr>
<tr>
<td>7.32766</td>
<td>0.0</td>
<td>0.37334</td>
<td>0.40139</td>
<td>0.0</td>
<td>0.91591</td>
<td></td>
</tr>
<tr>
<td>7.32766</td>
<td>0.0</td>
<td>0.37334</td>
<td>0.40139</td>
<td>0.0</td>
<td>0.91591</td>
<td></td>
</tr>
<tr>
<td>7.32766</td>
<td>0.0</td>
<td>0.37334</td>
<td>-0.40139</td>
<td>0.0</td>
<td>-0.91591</td>
<td></td>
</tr>
<tr>
<td>9.39706</td>
<td>0.0</td>
<td>9.97844</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>14.7473</td>
<td>1.8845</td>
<td>13.3654</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>11.8507</td>
<td>1.8845</td>
<td>14.6346</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>15.55268</td>
<td>0.0</td>
<td>13.01283</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>11.04640</td>
<td>0.0</td>
<td>14.98767</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>1</td>
<td>3</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>1</td>
<td>1</td>
<td>4</td>
<td>4</td>
<td></td>
<td></td>
</tr>
<tr>
<td>27</td>
<td>4</td>
<td>16</td>
<td>4</td>
<td>0</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>7.0</td>
<td>0.0</td>
<td>5.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>7.32766</td>
<td>0.0</td>
<td>0.37334</td>
<td>-0.40139</td>
<td>0.0</td>
<td>-0.91591</td>
<td></td>
</tr>
<tr>
<td>7.32766</td>
<td>0.0</td>
<td>0.37334</td>
<td>0.40139</td>
<td>0.0</td>
<td>0.91591</td>
<td></td>
</tr>
<tr>
<td>7.32766</td>
<td>0.0</td>
<td>0.37334</td>
<td>0.40139</td>
<td>0.0</td>
<td>0.91591</td>
<td></td>
</tr>
<tr>
<td>7.32766</td>
<td>0.0</td>
<td>0.37334</td>
<td>-0.40139</td>
<td>0.0</td>
<td>-0.91591</td>
<td></td>
</tr>
<tr>
<td>7.32766</td>
<td>0.0</td>
<td>0.37334</td>
<td>0.40139</td>
<td>0.0</td>
<td>0.91591</td>
<td></td>
</tr>
<tr>
<td>7.32766</td>
<td>0.0</td>
<td>0.37334</td>
<td>0.40139</td>
<td>0.0</td>
<td>0.91591</td>
<td></td>
</tr>
<tr>
<td>7.32766</td>
<td>0.0</td>
<td>0.37334</td>
<td>-0.40139</td>
<td>0.0</td>
<td>-0.91591</td>
<td></td>
</tr>
<tr>
<td>14.9963</td>
<td>1.8845</td>
<td>13.9344</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>12.0997</td>
<td>1.8845</td>
<td>15.2036</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>15.80235</td>
<td>0.0</td>
<td>13.58253</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>11.29607</td>
<td>0.0</td>
<td>15.55737</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

LOCKHEED-HUNTSVILLE ENGINEERING CENTER
Appendix C

HGM GRID CODE LISTING
Appendix C

**********MAIN**********

PROGRAM HGMGEOM(INPUT,OUTPUT,FILE19,FILE20,TAPE5,TAPE6=OUTPUT, &
TAPE19=FILE19,TAPE20=FILE20)

C
C

C

C

PROGRAM HGMGEOM

C DEVELOPED BY THE COMPUTATIONAL MECHANICS SECTION
LOCKHEED ENGINEERING CENTER, HUNTSVILLE, ALABAMA.
L.A. NICHOLSON

C
C
C COMMON /COUNTER/ NODESA,V,NODETOT,NBNODES,NPLANE
C COMMON /INITA/ IDIM,MAPTEN,INCHES
C COMMON /ZONING/ ISECT,NSECT,IZONE,NMBRNDS(3)
C COMMON /UNITS/ NU5,NU6,NU19,NU20,NU21

C
C---INITIALIZE PROGRAM
C
CALL INITIAL(NZONE)
C
C
C ZONE
C
DO 300 IZONE=1,NZONE
C
READ(NU5,1000) NSECT
C
C
C SECTION
C
DO 200 ISECT=1,NSECT
C
C---READ INPUT FOR EACH SECTION
C
CALL INPUT
C
C---GENERATE GRID FOR EACH SECTION
C
200 CALL GRID
C
CALL REWRITE
C
300 CONTINUE
C
C STOP
C
C---FORMAT STATEMENTS
C
1000 FORMAT(I5)
C
END

C-1

LOCKHEED-HUNTSVILLE ENGINEERING CENTER
SUBROUTINE INITIAL(NZONE)

COMMON /COEFF/ COEFE(8,10,12),COEFS(8,6),NMBRSEG(12)
COMMON /COUNTER/ NODESAV,NODETOT,NBNODES,NPLANE
COMMON /HEADER/ ITITLE(20),LINE
COMMON /INITA/ IDIM,MAPTEN,INCHES
COMMON /INITB/ IBOX(4,6),IRAY(4,3),NRAY,IPT(12),IPT2(12)
COMMON /INITC/ PI,RADDEG
COMMON /IWRITE/ IWRTC,IWRTI,IWRTN
COMMON /NODNMBR/ NODENUM(10000),TOL(3)
COMMON /UNITS/ NU5,NU6,NU19,NU20,NU21

DIMENSION NAME(20)

DATA PI /3.141592654/
DATA RADDEG /57.29577951/
DATA IPT1 /1,2,4,1,1,2,3,4,5,6,8,5/
DATA IPT2 /2,3,3,4,5,6,7,8,6,7,7,8/
DATA IRAY /1,3,9,11,4,2,12,10,5,6,8,7/
DATA IBOX /4,1,2,3,5,1,6,9,12,9,10,11,8,3,7,11,5,4,8,12,6,2,7,10/

DO 40 I=1,8

C---DYNAMIC DIMENSIONER INPUT

C---INITIALIZE

NU5 = 5
NU6 = 6
NU19 = 19
NU20 = 20
NU21 = 21
NPLANE = 0
NBNODES = 0
NINODES = 0
NODETOT = 0
NODESAV = 0
DO 30 J=1,6
   30 COEFS(I,J) = 0.0

DO 40 J=1,10
   DO 40 K=1,12
   40 COEFS(I,J,K) = 0.0

PRINT HEADER
WRITE(NU5,1120)

NZONE = NUMBER OF ZONES TO BE USED TO GENERATE GRID
IDIM = 2 FOR 2D
       = 3 FOR 3D
MATING = 0 NO ZONE MATING
IWRIT = 0 NO INTERMEDIATE PRINT
IWRTR = 0 NO CONNECTIVITY PRINTOUT
IWRTN = 0 PRINT ONLY BOUNDARY NODES
MAPTEN = N PRINT EVERY NTH INTERIOR NODE
         = 0 FIVE SEGMENTS PER EDGE
         = 1 TEN SEGMENTS PER EDGE
INCHES = 0 COORDINATES INPUT IN FEET
         = 1 COORDINATES INPUT IN INCHES

READ(NU5,1000) ITITLE
READ(NU5,1010) NZONE,IDIM,
               MAPTEN,INCHES

READ(NU5,1010) IWRIT,IWRTR,IWRTN

NRAY = 2*(IDIM - 1)
WRITE(NU5,1130) ITITLE
WRITE(NU6,1140) NZONE,IDIM,
                MAPTEN,INCHES

REWIND(NU20)

DRAW PICTURE
CALL PICTURE(IDIM)
RETURN

FORMAT STATEMENTS
1000 FORMAT(20A4)
1010 FORMAT(1615)
1120 FORMAT(1H1
         3 / 40X,34H
         4 / 40X,34H
         5 / 40X,32H
1130 FORMAT( // 5X,20A4)
1140 FORMAT( // 21H GEOMETRY PARAMETERS:
             1 // 45H NZONE IDIM
             2 MAPTEN INCHES /, 8110 )
SUBROUTINE INPUT

COMMON /COEFF/ COEFS(8,10,12),COEFS(8,6), NMBrSEG(12)
COMMON /COUNTER/ NODESAV,NODETOT,NBnodes,Nplane
COMMON /HEADER/ ITITLE(20),LINE
COMMON /HOTGAS/ IHGM,ETA11,ETA12,ETA31,ETA32,ETA33,
& STR11,STR12,STR13,STR31,STR32,STR33,STR34
COMMON /INITA/ IDIM,MAPten,INRCHES
COMMON /INITB/ IBOX(4,6),IRAY(4,3),NRAY,IPt1(12),IPt2(12)
COMMON /INPUT/ EDGE(6,12),POINT(6,8),SIDE(6,6)
COMMON /IWRTA/ IWRTA,IWRTI,IWRTN
COMMON /MAPING/ MAPSIDE(6),MAPSEG(10,12)
COMMON /MARCHS/ MARCH,INDEX(3)
COMMON /MAXIMUM/ ETAMAX(10,12),SEGMAX(6,10,12)
COMMON /SPACING/ ISTRITCH(3),STRETCH(3),ETAS(3,200),ETA(3),DETA(3)
COMMON /SPLIT/ NSPLIT,ISURF(10),IBC(10),
& ISTART(10),INC1(10),ISTART2(10),INC2(10)
COMMON /UNITS/ NU5,NU6,NU19,NU20,NU21
COMMON /ZONING/ ISECT,NSECT,IZONE,NMBRND(3)

DIMENSION IFLAGP(8),IFLAGE(12),IFLAGS(6)
DIMENSION LSTRTCH(3),LBCINPT(6),LMAPS(6)
DIMENSION MAPEDGE(12),STRTCH(3),LNMBRND(3)

MAPEDGE(I) INDICATES TYPE OF GEOMETRY FOR EDGE I
  = 1 LINEAR
  = 2 CIRCULAR ARC
  = 3 EDGE OF REVOLUTION

MAPSIDE(I) TYPE OF GEOMETRY FOR SURFACE I
  = 1 FLAT SURFACE
  = 4 EDGE OF REVOLUTION
  = 5,6,7 SPECIAL SURFACE SYBROUTINE

MARCH ETA DIRECTION IN WHICH COMPUTATION IS TO ADVANCE

NMBRND(I) NUMBER OF NODES IN EACH ETA(I) DIRECTION

ISTRITCH(I) = 0 NO STRETCHING IN ETA(I) DIRECTION
  = 1 INPUT N VALUES OF ETA(I)
  = 2 TIGHTEN GRID IN ETA(I) DIRECTION
  = 3 LOOSEN GRID IN ETA(I) DIRECTION

WRITE(NU6,1100) ITITLE,ISECT,NSECT,IZONE

IF(ISeCT.EQ.1) GO TO 100

SAVE INPUT PARAMETERS IN COMMON WITH PREVIOUS SECTION
DO 20 I=1,6
20  LMAPS(I) = MAPSIDE(I)

READ INPUT PARAMETERS

100 CONTINUE

READ(NU5,1030) MAPEDGE
READ(NU5,1000) MAPSIDE
READ(NU5,1000) MARCH,NSPLIT,IHGM

IF(IHGM.EQ.1) THEN
  READ(NU5,1020) ETA11,ETA12,ETA13
  READ(NU5,1020) ETA31,ETA32,ETA33
  READ(NU5,1020) STR11,STR12,STR13
  READ(NU5,1020) STR31,STR32,STR33,STR34
END IF

C---READ INPUT FOR GENERALIZED BOUNDARY CONDITIONS
READ(NU5,1000) NMBRNDX,ISTRCH

C---PRINT OUT INPUT VARIABLES
WRITE(NU6,3000) MAPEDGE
WRITE(NU6,3010) MAPSIDE
WRITE(NU6,3020) MARCH
WRITE(NU6,3040) NMBRNDX,ISTRCH

C---INITIALIZE FLAGS TO CHECK IF POINT, EDGE, OR SURFACE HAS BEEN INPUT
DO 130 I=1,8
130  IFLAGP(I) = 0
DO 140 I=1,12
140  IFLAGE(I) = 0
DO 150 I=1,6
150  IFLAGS(I) = 0

C---SET ORDER OF EXECUTION
INDEX(1) = MARCH
INDEX(3) = 3

DO 160 I=2,IDIM
160  IF(INDEX(I).GT.IDIM) INDEX(I) = 1
INDEX(I) = INDEX(I-1) + 1

C---INITIALIZE ETA
ETA(1) = 0.0
ETA(2) = 0.0
ETA(3) = 0.0
IF(I$E$E$.EQ.1 ) GO TO 300

C---TRANSFER INPUT PARAMETERS IN COMMON WITH PREVIOUS SECTION

C---TRANSFER POINTS

DO 200 I=1,NRAX
   L = IRAf(I,MARCH)
   NEWPT = IPT1(L)
   LASTPT = IPT2(L)
   IFLAGP(NEWPT) = 1

DO 200 J=1,6
   POINT(J,NEWPT) = POINT(J,LASTPT)

C---DETERMINE WHICH TWO SURFACES ARE IN COMMON

NEWSRF = 10 + (MARCH - 6)*MARCH
LASTSRF = 9 + (MARCH - 7)*MARCH/2

C---NUMBER OF EDGES IN COMMON WITH PREVIOUS SECTION

MEDGES = 3*IDIM - 5

DO 230 I=1,MEDGES

C---DETERMINE WHICH EDGES ARE IN COMMON WITH PREVIOUS SECTION

NEWEDG = IBOX(I, NEWSRF)
LASTEDG = IBOX(I,LASTSRF)

IFLAGA(NEWEDG) = 1

C---TRANSFER NUMBER OF SEGMENTS

NMBRSEG(NEWEDG) = NMBRSEG(LASTEDG)

C---TRANSFER SEGMENT MAPPING

DO 210 J=1,NMBRSEG(LASTEDG)
   MAPSEG(J,NEWEDG) = MAPSEG(J,LASTEDG)

C---TRANSFER EDGE COEFFICIENTS

DO 210 K=1,8
   COEFE(K,J, NEWEDG) = COEFE(K,J,LASTEDG)

DO 210 K=1,8
   COEFE(K,J,LASTEDG) = 0.0

IF(NMBRSEG(LASTEDG).EQ.1) GO TO 230

C---TRANSFER ETA MAXIMUMS

DO 220 J=1,NMBRSEG(LASTEDG) - 1
   ETAMAX(J, NEWEDG) = ETAMAX(J,LASTEDG)
ETAMAX(J,LASTEDG) = 0.0
C
---TRANSFER SEGMENT MAXIMUMS
C
DO 220 K=1,6
SEGMAX(K,J,NEWEDG) = SEGMAX(K,J,LASTEDG)
220 SEGMAX(K,J,LASTEDG) = 0.0
C
---TRANSFER SURFACE MAPPING
C
MAPSIDE(NEWSRF) = LMAPS(LASTSRF)
C
IFLAGS(NEWSRF) = 1
C
---TRANSFER SURFACE COEFFICIENTS
C
DO 240 I=1,8
COEFS(I, NEWSRF) = COEFS(I, LASTSRF)
240 COEFS(I, LASTSRF) = 0.0
C
---TRANSFER NUMBER OF NODES AND STRETCHING DATA FOR ETA_2 AND ETA_3
C
DO 250 I=2,IDIM
J = INDEX(I)
NMBRNS(J) = LNMBRND(J)
STRETCH(J) = STRETCH(J)
I 250 I = LSTRETCH(J)
C
---READ INPUT PARAMETERS FOR STRETCHING FUNCTIONS
C
DO 310 I=1,IDIM
STRETCH(I) = 0.0
C
IF(ISTRETCH(I).LE.1) GO TO 310
C
READ(NU5,1020) STRETCH
C
GO TO 320
C
310 CONTINUE
C
---COMPUTE STRETCHING FUNCTION PARAMETER B USING NEWTON-RAPHSON
C
DO 370 I=1,IDIM
B = STRETCH(I)
DS = B
TNODE = REAL(NMBRNS(I))
C
---DECREASING OR INCREASING SPACING (INPUT MINIMUM SPACING)
C
IF(ISTRETCH(I).EQ.5 .OR. ISTRETCH(I).EQ.6) THEN
B = SQRT(1.0 - (TNODE - 1.0)*DS)/(TNODE - 1.0)
C
DO 330 ITER=1,10
C
ARG1 = B*(TNODE - 1.)
ARG2 = B*(TNODE - 2.)
C
C-7

LOCKHED-HUNTSVILLE ENGINEERING CENTER
EXPA1 = EXP(ARG1)
EXPA2 = EXP(ARG2)

TANH1 = (EXPA1 - 1./EXPA1)/(EXPA1 + 1./EXPA1)
TANH2 = (EXPA2 - 1./EXPA2)/(EXPA2 + 1./EXPA2)

PSI = DS - (1.0 - TANH2/TANH1)

TANHPI = 1.0 - TANH1**2
TANHP2 = 1.0 - TANH2**2

PSIP = (1.0/TANH1)*(TANHP2*(TNODE - 2.) - (TANH2/TANH1)*TANHPI*(TNODE - 1.))

IF(PSIP.EQ.0.) THEN
    WRITE(NU6,5100)
    STOP
END IF

B0 = B
B = B0 - PSI/PSIP
DBF = (B - B0)/B0

IF(ABS(DBF).LE.0.001) GO TO 340

END IF

C---DOUBLE STRETCHING (INPUT MINIMUM SPACING)
IF(ISTRTCH(I).EQ.7) THEN

B = SQRT(1.0 - (TNODE - 1.0)*DS)/(TNODE - 1.0)

DO 350 ITER=1,10

ARG1 = B*(TNODE - 1.)
ARG3 = B*(TNODE - 3.)

EXPA1 = EXP(ARG1)
EXPA3 = EXP(ARG3)

TANH1 = (EXPA1 - 1./EXPA1)/(EXPA1 + 1./EXPA1)
TANH3 = (EXPA3 - 1./EXPA3)/(EXPA3 + 1./EXPA3)

PSI = DS - 0.5*(1.0 - TANH3/TANH1)

TANHPI = 1.0 - TANH1**2
TANHP3 = 1.0 - TANH3**2

PSIP = (0.5/TANH1)*(TANHP3*(TNODE - 3.0) - (TANH3/TANH1)*TANHPI*(TNODE - 1.0))

IF(PSIP.EQ.0.) THEN
    WRITE(NU6,5000)
END IF

C-8

LOCKHEED-HUNTSVILLE ENGINEERING CENTER
STOP
END IF

B0 = B
B = B0 - PSI/PSIP
DBF = (B - B0)/B0

IF(ABS(DBF).LE.0.001) GO TO 360

370 STRETCH(I) = B

C---TRANSFER STRETCHING IN COMMON WITH PREVIOUS SECTION
C IF(ISECT.GT.1) THEN
C DO 380 I=2,IDIM
  380 STRETCH(INDEX(I)) = STRETCH(INDEX(I))
C END IF
C
C READ INPUT PARAMETERS FOR ARBITRARY GRID SPACING (ETAS)
C DO 410 I=1,IDIM
  IF(ISTRCH(I).NE.1) GO TO 410
  IF(ISECT.GT.1 .AND. I.NE.MARCH) GO TO 410
  READ(NU5,1020) (ETAS(I,J),J=1,NMBRnds(I))
C 410 CONTINUE
C
C READ INPUT PARAMETERS FOR EDGE COEFFICIENTS
C WRITE(NU6,1100) ITITLE,ISECT,NSECT,IZONE
WRITE(NU6,2040)
  LINE = 6
  II = 0
C
C---TOTAL NUMBER OF EDGES
C NEDGES = 8*IDIM - 12
C DO 540 I=1,NEDGES
C IF(IFLAGE(I).EQ.1) GO TO 540
C ITOTAL = 1
  MAP = MAPEDGE(I)
C---DETERMINE THE NUMBER OF SEGMENTS ON AN EDGE
C DO 500 J=1,10
NMBRSEG(I) = J
MAP = MAP/10
C
IF(MAP.EQ.0) GO TO 510
C
500 ITOTAL = ITOTAL*10
C
510 MAP = MAPEDGE(I)
C
DO 530 J=1,NMBRSEG(I)
C
DO 520 K=1,8
520 COEFE(K,J,I) = 0.0
C
C DETERMINE THE MAPPING FOR EACH SEGMENT
C
MAPSEG(J,I) = MAP/ITOTAL
MAP = MAP - MAPSEG(J,I)*ITOTAL
C
C READ IN THE EDGE COEFFICIENTS FOR EACH SEGMENT
C
IF(MAPSEG(J,I).LE.1) GO TO 530
C
READ(NU5,1020) (COEFE(K,J,I),K=1,8)
C
IF(I.NE.II) THEN
C
LINE = LINE + 2
C
IF(LINE.GE.60) THEN
C
WRITE(NU6,1100) ITITLE,ISECT,NSECT,IZONE
WRITE(NU6,2040)
C
LINE = 8
C
END IF
C
WRITE(NU6,2050) I,J,(COEFE(K,J,I),K=1,8)
C
ELSE
C
LINE = LINE + 2
C
IF(LINE.GE.60) THEN
C
WRITE(NU6,1100) ITITLE,ISECT,NSECT,IZONE
WRITE(NU6,2040)
C
LINE = 8
C
END IF
C
WRITE(NU6,2060) J,(COEFE(K,J,I),K=1,8)
C
END IF
C
II = I
C
530 ITOTAL = ITOTAL/10
C 540 CONTINUE
C --------
C READ INPUT PARAMETERS FOR SURFACE COEFFICIENTS
C --------
C
C LINE = LINE + 5
C
IF(LINE.GE.60) THEN
   WRITE(NU6,1100) ITITLE,ISECT,NSECT,IZONE
   LINE = 6
END IF

WRITE(NU6,3060)
C
DO 610 I=1,6
C
IF(IFLAGS(I).EQ.1) GO TO 610
C
DO 600 J=1,8
600 COEFS(J,I) = 0.0
C
IF(MAPSIDE(I).LE.2) GO TO 610
C
READ(NU5,1020) (COEFS(J,I),J=1,8)
C
LINE = LINE + 2
C
IF(LINE.GE.60) THEN
   WRITE(NU6,1100) ITITLE,ISECT,NSECT,IZONE
   WRITE(NU6,3060)
   LINE = 8
END IF

WRITE(NU6,3065) I,(COEFS(J,I),J=1,8)
C
610 CONTINUE
C --------
C READ INPUT DATA FOR CORNER POINTS AND SEGMENT END POINTS
C --------
C
LINE = LINE + 5
C
IF(LINE.GE.60) THEN
   WRITE(NU6,1100) ITITLE,ISECT,NSECT,IZONE
   LINE = 6
END IF

WRITE(NU6,3070)
DO 750 I=1,NEDGES
   IF(I.GT.8 .OR. IFLAGP(I).EQ.1) GO TO 700
C---READ IN CORNER POINTS
C   READ(NU5,1020) (POINT(J,I),J=1,3)
C   LINE = LINE + 2
C   IF(LINE.GE.60) THEN
C      WRITE(NU6,1100) ITITLE,ISECT,NSECT,IZONE
C      WRITE(NU6,3070)
C      LINE = 8
C      END IF
C   WRITE(NU6,3080) I,(POINT(J,I),J=1,3)
C 700 IF(IFLAGE(I).EQ.1 .OR. NMBRSEG(I).EQ.1) GO TO 750
C---READ IN SEGMENT MAXIMUMS
C   DO 740 J=1,NMBRSEG(I) - 1
C      READ(NU5,1020) (SEGMAX(K,J,I),K=1,3),ETAMX
C      LINE = LINE + 2
C      IF(LINE.GE.60) THEN
C      WRITE(NU6,1100) ITITLE,ISECT,NSECT,IZONE
C      WRITE(NU6,3070)
C      LINE = 8
C      END IF
C   WRITE(NU6,3090) I,J,(SEGMAX(K,J,I),K=1,3),ETAMX
C   DO 720 N=1,3
C      DO 720 L=1,4
C 720 IF(IRA5f(L,N) .EQ.I) GO TO 730
C      CONTINUE
C C---CONVER T NODE NUMBER TO ETA VALUE
C   ETAMAX(J,I) = (ETAMX - 1.0)/(NMBRNDs(N) - 1.0)
C   CONTINUE
C 740 CONTINUE
C 750 CONTINUE
C---
C---
C-12

LOCKHEED-HUNTSVILLE ENGINEERING CENTER
1000 FORMAT(16I5)
1010 FORMAT(6I10)
1020 FORMAT(8E10.0)
1030 FORMAT(4I10)

C

1100 FORMAT(1H1,10X,20A4,13X,8H SECTION,I2,3H OF,I3,9H FOR ZONE,I3)
1110 FORMAT(/ 1H FIXED ETA_I1,8H VALUES: // (10(3X,F10.7)))
2040 FORMAT(/ 25H EDGE SHAPE COEFFICIENTS:
1 // 42H EDGE SEGMENT COEFF_1 COEFF_2,
2 // 42H COEFF_3 COEFF_4 COEFF_5,
3 // 42H COEFF_6 COEFF_7 COEFF_8 )
2050 FORMAT(/ 2X,I2,6X,I1,3X,8(1X,F13.7))
2060 FORMAT(/ 10X,I1,3X,8(1X,F13.7))
2080 FORMAT(/ 7H POINT_I2,1H_:12X,2(7X,F13.7),9X,F7.2)
2090 FORMAT(/ 7H EDGE_I2,1H_:7X,I3,2X,2(7X,F13.7),9X,F7.2,9X,F9.2)

C

3000 FORMAT(/ 32H EDGE SHAPE FUNCTION INDICATORS:
1 // 40H EDGE_1 EDGE_2 EDGE_3 EDGE_4,
2 // 40H EDGE_5 EDGE_6 EDGE_7 EDGE_8,
3 // 40H EDGE_9 EDGE_10 EDGE_11 EDGE_12 / 12I10 )
3010 FORMAT(/ 35H SURFACE SHAPE FUNCTION INDICATORS:
1 // 45H SURFACE_1 SURFACE_2 SURFACE_3,
2 // 45H SURFACE_4 SURFACE_5 SURFACE_6
3 // 6(10X,I5))
3020 FORMAT(/ 26H MARCHING DIRECTION = ETA_I1)
3040 FORMAT(/ 17H NUMBER OF NODES:
1 // 30H ETA_1 ETA_2 ETA_3 / 3I10
2 // 29H NODAL STRETCHING INDICATORS:
3 // 30H ETA_1 ETA_2 ETA_3 / 3I10 )
3050 FORMAT(/ 32H STRETCHING FUNCTION PARAMETERS:
1 // 45H ETA_1 ETA_2 ETA_3
2 // 3(2X,F13.7) )
3060 FORMAT(/ 28H SURFACE SHAPE COEFFICIENTS:
1 // 42H SURFACE COEFF_1 COEFF_2,
2 // 42H COEFF_3 COEFF_4 COEFF_5,
3 // 42H COEFF_6 COEFF_7 COEFF_8 )
3065 FORMAT(/ 4X,I1,9X,8(1X,F13.7))
3070 FORMAT(/ ' COORDINATES:','
1 //10X,' SEGMENT X Y Z ETAMAX')
3080 FORMAT(/ 7H POINT_I2,1H_:8X,3(6X,F13.7))
3090 FORMAT(/ 7H EDGE_I2,1H_:3X,I3,2X,3(6X,F13.7)
5000 FORMAT(1HO,'DEAD IN INPUT')
5100 FORMAT(1HO,'DEAD IN INPUT')

C

END

C******************************************************************************
C******************************************************************************
C******************************************************************************

C SUBROUTINE GRID

C GRID CONTROLS THE GENERATION OF THE SPATIAL COORDINATES OF EACH NODE
C IN THE GRID.

C******************************************************************************
C******************************************************************************
C******************************************************************************
COMMON /COUNTER/ NODESAV,NODETOT,NBNODES,NPLANE
COMMON /HEADER/ ITITLE(20),LINE
COMMON /INITA/ IDIM,MAPTEN,INCHES
COMMON /INITB/ IBOX(4,6),IRAY(4,3),NRAY,IPT1(12),IPT2(12)
COMMON /INPUTA/ EDGE(6,12),POINT(6,8),SIDE(6,6)
COMMON /INPUTBC/ ISIDE{3)
COMMON /IWRITE/ IWRTC,IWRTI,IWRTN
COMMON /MAPING/ MAPSIDE(6),MAPSEG(10,12)
COMMON /MARCHS/ MARCH,INDEX(3)
COMMON /NODNMBR/ NODENUM(10000),TOL(3)
COMMON /OUT/ NODE(5,10000)
COMMON /PARTIAL/ DEDN(3,12),DSDN(3,2),SNORMAL(3,6)
COMMON /SPACING/ ISTRCH(3),STRETCH(3),ETAS(3,200),ETA(3),DETA(3)
COMMON /UNITS/ NU5,NU6,NU19,NU20,NU21
COMMON /ZONING/ ISECT,NSECT,IZONE,NMBRNDS(3)

C
DIMENSION E(26)
DIMENSION IFACE1(3,3),IFACE2(3,3),ISIDES(6)
DIMENSION TANGENT(3),VECTOR(3),DIRECT(3)

C
DATA IFACE1 /0,1,2,1,0,5,2,5,0/
DATA IFACE2 /0,3,4,3,0,6,4,6,0/
DATA ISIDES /3,4,1,2,6,5/
C
INIT = 1
C
DO 30 I=1,IDIM
C
IF(STRETCH(I).LT.1.0) STRETCH(I) = 1.0
C
DETA(I) = 1.0/(NMBRNDS(I) - 1.0)
C
FACTOR = NMBRNDS(I)*10.0
IF(ISTRTCH(I).NE.O) FACTOR = NMBRNDS(I)*20.0
C
DO 20 J=1,NRAY
C
L = IRAY(J,I)
C
C COMPARE X, Y, AND Z BETWEEN END POINTS OF AN EDGE
C
DO 20 K=1,IDIM
C
DELTA = ABS(POINT(K,IPT1(L)) - POINT(K,IPT2(L)))/FACTOR
C
IF(DELTA.LE.0.0) GO TO 20
IF(DELTA.LT.TOL(K)) TOL(K) = DELTA
C
20 CONTINUE
30 CONTINUE
C
TOL(1) = .005
TOL(2) = .005
TOL(3) = .005
C
C DETERMINE SURFACES
C
ISIDE1 = IFACE1(MARCH,INDEX(2))
ISIDE2 = IFACE2(MARCH,INDEX(2))
ISIDE3 = IFACE1(MARCH,INDEX(3))

C-14

LOCKHEED-HUNTSVILLE ENGINEERING CENTER
ISIDE4 = IFACE2(MARCH,INDEX(3))
ISIDE5 = IFACE1(INDEX(2),INDEX(3))
ISIDE6 = IFACE2(INDEX(2),INDEX(3))

C --- MAXIMUM NUMBER OF NODES IN A PLANE
C
MAXPL = NMBRNDs(INDEX(2)) * NMBRNDs(INDEX(3))
C
INDEX(1) = MARCH
C
C AXIS
C
----------
DO 700 IAXIS = 1, NMBRNDs(INDEX(1))
C
IF(IAXIS.EQ.1 .AND. ISECT.GT.1) GO TO 700
C
C --- SEPERATE BOUNDARY CONDITIONS AND DETERMINE ETA
C
CALL ETABC(MARCH, INDEX(1), IAXIS)
C
C --- CALCULATE COORDINATES AND DERIVATIVES FOR POINTS ON EDGES
C
CALL EDGES(INIT, INDEX(1), ETA(INDEX(1)))
C
C --- NUMBER OF NODES TO BE STORED
C
NODSTOR = NODESAV
C
----------
C ROW
C
----------
DO 500 JAXIS = 1, NMBRNDs(INDEX(2))
C
C --- SEPERATE BOUNDARY CONDITIONS AND DETERMINE ETA
C
CALL ETABC(MARCH, INDEX(2), JAXIS)
C
C --- CALCULATE COORDINATES AND DERIVATIVES FOR POINTS ON EDGES
C
CALL EDGES(INIT, INDEX(2), ETA(INDEX(2)))
C
C --- CALCULATE COORDINATES AND SURFACE NORMAL FOR POINTS ON SURFACES
C
CALL SURFACE(INIT, ISIDE1)
C
CALL SURFACE(INIT, ISIDE2)
C
----------
C COLUMN
C
----------
DO 400 KAXIS = 1, NMBRNDs(INDEX(3))
C
C --- SEPERATE BOUNDARY CONDITIONS AND DETERMINE ETA
C
CALL ETABC(MARCH, INDEX(3), KAXIS)
C
C --- CALCULATE COORDINATES AND DERIVATIVES FOR POINTS ON EDGES
C
CALL EDGES(INIT, INDEX(3), ETA(INDEX(3)))
C
C---CALCULATE COORDINATES AND SURFACE NORMAL FOR POINTS ON SURFACES
C
CALL SURFACE(INIT,ISIDE3)
C
CALL SURFACE(INIT,ISIDE4)
C
CALL SURFACE(INIT,ISIDE5)
C
CALL SURFACE(INIT,ISIDE6)
C
C---ETA COEFFICIENTS FOR TRI-LINEAR INTERPOLATION
C
E(1) = 1.0 - ETA(3)
E(2) = ETA(3)
E(3) = 1.0 - ETA(2)
E(4) = ETA(2)
E(5) = 1.0 - ETA(1)
E(6) = ETA(1)
E(7) = E(5)*E(3)
E(8) = E(5)*ETA(2)
E(9) = ETA(1)*E(3)
E(10) = ETA(1)*ETA(2)
E(11) = E(5)*E(1)
E(12) = E(5)*ETA(3)
E(13) = ETA(1)*E(1)
E(14) = ETA(1)*ETA(3)
E(15) = E(3)*E(1)
E(16) = E(3)*ETA(3)
E(17) = ETA(2)*E(1)
E(18) = ETA(2)*ETA(3)
E(19) = E(5)*E(3)*E(1)
E(20) = E(5)*E(3)*ETA(3)
E(21) = E(5)*ETA(2)*E(1)
E(22) = E(5)*ETA(2)*ETA(3)
E(23) = ETA(1)*E(3)*E(1)
E(24) = ETA(1)*E(3)*ETA(3)
E(25) = ETA(1)*ETA(2)*E(1)
E(26) = ETA(1)*ETA(2)*ETA(3)
C
C---INCREMENT NODE COUNTERS
C
INIT = 0
NODESAV = NODESAV + 1
C
NODNUM = NODESAV + NODETOT
C
C---CALCULATE THE COORDINATES
C
DO 340 L=1,3
340 NODE(L,NODESAV) = E(1)*SIDE(L,1) + E(2)*SIDE(L,3)
   + E(3)*SIDE(L,2) + E(4)*SIDE(L,4)
   + E(5)*SIDE(L,5) + E(6)*SIDE(L,6)
   + E(7)*EDGE(L,5) + E(8)*EDGE(L,8) + E(9)*EDGE(L,6) + E(10)*EDGE(L,7)
   + E(11)*EDGE(L,12) + E(12)*EDGE(L,12) + E(13)*EDGE(L,2) + E(14)*EDGE(L,10)
   + E(15)*EDGE(L,1) + E(16)*EDGE(L,9) + E(17)*EDGE(L,3) + E(18)*EDGE(L,11)
   + E(19)*POINT(L,1) + E(20)*POINT(L,5)
   + E(21)*POINT(L,4) + E(22)*POINT(L,8)
   + E(23)*POINT(L,2) + E(24)*POINT(L,6)
   + E(25)*POINT(L,3) + E(26)*POINT(L,7)
C
C-16

LOCKHEED-HUNTSVILLE ENGINEERING CENTER
C---INTERMEDIATE ERROR PRINT
C     IF(IWRTI.EQ.0 .OR. MOD(NODNUM,IWRTI).NE.0) GO TO 350
C
C     LINE = LINE + 27
C
C     IF(LINE.GE.60) THEN
C
C     WRITE(NU6,1100) ITITLE,ISECT,NSECT,IZONE
C
C     LINE = 28
C
C     END IF
C
C     WRITE(NU6,1110) NODNUM
C
C     1 ,E(1),(SIDE(L,1),L=1,3)
C     2 ,E(3),(SIDE(L,2),L=1,3)
C     3 ,E(2),(SIDE(L,3),L=1,3)
C     4 ,E(4),(SIDE(L,4),L=1,3)
C     5 ,E(5),(SIDE(L,5),L=1,3)
C     6 ,E(6),(SIDE(L,6),L=1,3)
C
C     WRITE(NU6,1120) E(15),(EDGE(L,1),L=1,3)
C     1 ,E(13),(EDGE(L,2),L=1,3)
C     2 ,E(17),(EDGE(L,3),L=1,3)
C     3 ,E(11),(EDGE(L,4),L=1,3)
C     4 ,E(7),(EDGE(L,5),L=1,3)
C     5 ,E(9),(EDGE(L,6),L=1,3)
C
C     WRITE(NU6,1130) E(10),(EDGE(L,7),L=1,3)
C     1 ,E(8),(EDGE(L,8),L=1,3)
C     2 ,E(16),(EDGE(L,9),L=1,3)
C     3 ,E(14),(EDGE(L,10),L=1,3)
C     4 ,E(7),(EDGE(L,11),L=1,3)
C     5 ,E(9),(EDGE(L,12),L=1,3)
C
C     WRITE(NU6,1140) E(19),(POINT(L,1),L=1,3)
C     1 ,E(23),(POINT(L,2),L=1,3)
C     2 ,E(25),(POINT(L,3),L=1,3)
C     3 ,E(21),(POINT(L,4),L=1,3)
C     4 ,E(20),(POINT(L,5),L=1,3)
C     5 ,E(24),(POINT(L,6),L=1,3)
C     6 ,E(26),(POINT(L,7),L=1,3)
C     7 ,E(22),(POINT(L,8),L=1,3)
C
C
C     350 CONTINUE
C
C     NODE(4,NODESAV) = THETA
C     NODE(5,NODESAV) = PHI
C
C 400 CONTINUE
C 500 CONTINUE
C
C-----------------------------------------------
C PLANE OUTPUT
C-----------------------------------------------
C     IF(IAXIS.EQ.1) GO TO 700
C
C     IF(NODSTOR.NE.0) THEN
C
C-----------------------------------------------
C-17
C
LOCKHED-HUNTSVILLE ENGINEERING CENTER
C---PRINT AND STORE DATA
C CALL OUTPUT(NU20,NODSTOR)
C
C---TRANSFER DATA SECOND PLANE TO FIRST PLANE
C DO 630 I=1,NODESAV
C DO 600 J=1,5
600 NODE(J,I) = NODE(J,I + NODSTOR)
C 630 CONTINUE
C END IF
C
C---TRANSFER NODE NUMBERS FROM SECOND PLANE TO FIRST PLANE
C DO 650 L=1,MAXPL
650 NODENUM(L) = NODENUM(L + MAXPL)
C 700 CONTINUE
C
C SECTION OUTPUT
C
C IF(ISECT.NE.NSECT) RETURN
C
C---PRINT AND STORE DATA
C NODSTOR = NODESAV
C
C IF(NODESAV.NE.0) CALL OUTPUT(NU20,NODSTOR)
C NODESTAV = 0
C
C RETURN
C
C---FORMAT STATEMENTS
C
1100 FORMAT(1H1,10X,20A4,13X,8H SECTION,12,3H OF,13,9H FOR ZONE,I3)

1110 FORMAT( 8H NODE =,,I10
 1 / 41X,2H X,13X,2H Y,12X,2H Z,20X,15H FLOW DIRECTION
 2 / 8H E( 1) =,F13.7,15H SIDE(1) =,6(1X,F13.7)
 3 / 8H E( 3) =,F13.7,15H SIDE(2) =,6(1X,F13.7)
 4 / 8H E( 2) =,F13.7,15H SIDE(3) =,6(1X,F13.7)
 5 / 8H E( 4) =,F13.7,15H SIDE(4) =,6(1X,F13.7)
 6 / 8H E( 5) =,F13.7,15H SIDE(5) =,6(1X,F13.7)
 7 / 8H E( 6) =,F13.7,15H SIDE(6) =,6(1X,F13.7))

1120 FORMAT( 8H E(15) =,F13.7,15H EDGE( 1) =,6(1X,F13.7)
 1 / 8H E(13) =,F13.7,15H EDGE( 2) =,6(1X,F13.7)
 2 / 8H E(17) =,F13.7,15H EDGE( 3) =,6(1X,F13.7)
 3 / 8H E(11) =,F13.7,15H EDGE( 4) =,6(1X,F13.7)
 4 / 8H E( 7) =,F13.7,15H EDGE( 5) =,6(1X,F13.7)
 5 / 8H E( 9) =,F13.7,15H EDGE( 6) =,6(1X,F13.7))

1130 FORMAT( 8H E(10) =,F13.7,15H EDGE( 7) =,6(1X,F13.7)
 1 / 8H E( 8) =,F13.7,15H EDGE( 8) =,6(1X,F13.7)
 2 / 8H E(16) =,F13.7,15H EDGE( 9) =,6(1X,F13.7)
C-18
LOCKHED-HUNTSVILLE ENGINEERING CENTER
INTERPOLATES ALONG EACH EDGE OF EACH SIDE OF ZONE AS PART OF THE 

BI/TRI-LINEAR INTERPOLATION SCHEME

COMMON /COEFS/ COEFS(8,10,12),COEFS(8,6),NMBRSEG(12)
COMMON /COUNTER/ NODESAV,NODETOT,NBNODES,NPLANE
COMMON /HEADER/ ITITLE(20),LINE
COMMON /INITA/ IDIM,MAPTEN,INCHES
COMMON /INITB/ IBOX(4,6),IRAY(4,3),NRAY,IPT1(12),IPT2(12)
COMMON /INPUTA/ EDGE(6,12),POINT(6,8),SIDE(6,6)
COMMON /WRITE/ IWRIC,IWRTI,IWRTN
COMMON /MAPED/ KSEG(12),UAXIS(3,6),IEDGE1(6),IEDGE2(6),GAMMA
COMMON /MAXIMUM/ ETAMAX(10,12),SEGMAX(6,10,12)
COMMON /PARTIAL/ DEDN(3,12),DSDN(3,2),SNORMAL(3,6)
COMMON /SEGMENT/ PT1(6,10,12),PT2(6,10,12),ETA1(10,12),ETA2(10,12)
COMMON /UNITS/ NU5,NU6,NU19,NU20,NU21
COMMON /ZONING/ ISECT,NSECT,IZONE,NMBRnds(3)

INTERMEDIATE PRINT

NODNUM = NODESAV + NODETOT + 1

IF(IWRTI.EQ.0 .OR. MOD(NODNUM,IWRTI).NE.0) GO TO 20

LINE = LINE + 2

IF(LINE.GE.60) THEN

WRITE(NU6,1100) ITITLE,ISECT,NSECT,IZONE

LINE = 3

END IF

CALCULATE EDGE COORDINATES AND DERIVATIVE

20 DO 200 I=1,NRAY
C---DETERMINE WHICH EDGE
C
IEDGE = IRAY(I,IDIR)
C
C---EDGE INITIALIZATION
C
IF(INIT.EQ.1) CALL EDGMAP1(IEDGE,IDIR,I)
C
C---DETERMINE WHICH SEGMENT
C
DO 100 JSEG=1,NMBRSEG(IEDGE)
C
ISEG = JSEG
C
100 IF(ETA.GE.ETA1(JSEG,IEDGE) .AND. ETA.LE.ETA2(JSEG,IEDGE)) GOTO 110
C
KSEG(IEDGE) = ISEG
C
C---DETERMINE WHERE ALONG THE SEGMENT
C
DENOM = ETA2(ISEG,IEDGE) - ETA1(ISEG,IEDGE)
C
IF(DENOM.EQ.0.0) DENOM = 1.0
C
RATIO = (ETA - ETA1(ISEG,IEDGE))/DENOM
C
C---CALCULATE THE COORDINATES AND DERIVATIVE
C
CALL EDGMAP2(IEDGE,ISEG,RATIO,EDGE(1,IEDGE),DEDN(1,IEDGE))
C
C INTERMEDIATE PRINT
C
IF(IWRTI.EQ.0 .OR. MOD(NODNUM,IWRTI).NE.0) GO TO 200
C
LINE = LINE + 1
C
IF(LINE.GE.60) THEN
C
WRITE(NU6,1100) ITITLE,ISECT,NSECT,IZONE
C
LINE = 2
C
END IF
C
WRITE(NU6,1120) NODNUM,IEDGE,
1 (EDGE(J,IEDGE),J=1,3),THETA,PHI,
2 ETA,RATIO,ETAMAX(ISEG,IEDGE)
C
200 CONTINUE
C
RETURN
C---FORMAT STATEMENTS
C
1100 FORMAT(1H1,10X,2A4,13X,8H SECTION,I2,3H OF,I3,9H FOR ZONE,I3)
1120 FORMAT(1X,I6,4X,I2,3X,3(X,F13.7),11X,2(2X,F7.2),3(2X,F7.5))
C
END
SUBROUTINE EDGMAP(IEDGE, IDIR, NMBREDG)

EDGE MAPPING INITIALIZATION

COMMON /COEFF/
COMMON /COUNTER/
COMMON /EDGE0/
& COMMON /EDGE3/
& COMMON /EDGES/
COMMON /HEADER/
COMMON /INITA/
COMMON /INITB/
COMMON /INITC/
COMMON /INPUTA/
COMMON /MAPED/
COMMON /MARCH/
COMMON /MAXIMUM/
COMMON /PARTIAL/
COMMON /SEGMENT/
DIMENSION ETAMX(10,12), VECTER(3)

C---INITIALIZE

KSEG(IEDGE) = 1
DO 10 N=1,NMBRSEG(IEDGE)
10 ETAMX(N,IEDGE) = ETAMAX(N,IEDGE)

ETAMX(NMBRSEG(IEDGE),IEDGE) = 1.0
ETA1(1,IEDGE) = 0.0
ETAMAX(NMBRSEG(IEDGE),IEDGE) = 1.0

C---INITIALIZE SEGMENTS

DO 1000 ISEG=1,NMBRSEG(IEDGE)

C---DETERMINE THE COORDINATES AT THE END POINTS

IF(ISEG.EQ.1) THEN

DO 50 J=1,6

PT1(J, 1,IEDGE) = POINT(J, IPT1(IEDGE))
SEGMAX(J,NMBRSEG(IEDGE),IEDGE) = POINT(J, IPT2(IEDGE))
50

ELSE

DO 60 J=1,6

PT1(J,ISEG,IEDGE) = SEGMAX(J,(ISEG-1),IEDGE)
PT2(J,ISEG,IEDGE) = SEGMAX(J,ISEG,IEDGE)
60

C-21

LOCKHEED-HUNTSVILLE ENGINEERING CENTER
END IF
C---ETA VALUE AT THE END OF EACH SEGMENT
C
   IF(ISEG.GT.1) ETA1(ISEG,IEDGE) = ETA2(ISEG-1,IEDGE)
   ETA2(ISEG,IEDGE) = ETAMX(ISEG,IEDGE)
C
C---VECTORS FROM ORIGIN TO END POINTS
C
   DO 80 J=1,3
      R1(J,ISEG,IEDGE) = PT1(J,ISEG,IEDGE) - COEFE(J,ISEG,IEDGE)
   80   R2(J,ISEG,IEDGE) = PT2(J,ISEG,IEDGE) - COEFE(J,ISEG,IEDGE)
C
   CALL VMAG(R1(1,ISEG,IEDGE),RMAG1)
C
   CALL VMAG(R2(1,ISEG,IEDGE),RMAG2)
C
C---CHOOSE MAPPING FUNCTION
C
   MAP = MAPSEG(ISEG,IEDGE)

GO TO (1000,200,300,400,500) MAP
C
C---CIRCULAR ARC
C
   MAP = 2
C
C---Sweep Angle
C
   200 CALL VDOT(R1(1,ISEG,IEDGE),R2(1,ISEG,IEDGE),R1DOTR2)
C
   ARG = R1DOTR2/(RMAG1*RMAG2)
C
   IF(ABS(ARG).GT.1.0) ARG = ARG/ABS(ARG)
C
   THETA(ISEG,IEDGE) = ACOS(ARG)
C
   GO TO 1000

C---EDGES OF REVOLUTION
C
   MAP = 4

300 CONTINUE
C
C---UK: NORMALIZED VECTOR ALONG AXIS OF REVOLUTION
C
   CALL VMAG(COEFE(4,ISEG,IEDGE),UKM)
C
   DO 410 I=1,3
C
   410 UK(I,ISEG,IEDGE) = COEFE(I + 3,ISEG,IEDGE)/UKM
C
C---UJ: NORMALIZED VECTOR FROM AXIS TOWARD EDGE
C
   CALL CROSS(UK(1,ISEG,IEDGE),R1(1,ISEG,IEDGE),UJ(1,ISEG,IEDGE),11)
C
C---UI: NORMALIZED VECTOR FROM AXIS TO FIRST POINT
C
   CALL CROSS(UJ(1,ISEG,IEDGE),UK(1,ISEG,IEDGE),UI(1,ISEG,IEDGE),12)
C
C---R1: VECTOR FROM AXIS TO FIRST POINT
C
   CALL VDOT(R1(1,ISEG,IEDGE),UK(1,ISEG,IEDGE),RP1(ISEG,IEDGE))
CALL VADD(1.0, R1(1, ISEG, IEDGE), -RPAl(ISEG, IEDGE), UK(1, ISEG, IEDGE),
& R1(1, ISEG, IEDGE), VECTER)
CALL VMAG(R1(1, ISEG, IEDGE), RM1(ISEG, IEDGE))
C---R2: VECTOR FROM AXIS TO SECOND POINT-----------------------------
CALL VDOT(R2(1, ISEG, IEDGE), UK(1, ISEG, IEDGE), RPA2(ISEG, IEDGE))
CALL VADD(1.0, R2(1, ISEG, IEDGE), -RPA2(ISEG, IEDGE), UK(1, ISEG, IEDGE),
& R2(1, ISEG, IEDGE), VECTER)
CALL VMAG(R2(1, ISEG, IEDGE), RM2(ISEG, IEDGE))
C---SWEEP ANGLE-----------------------------------------------------
CALL VDOT(R1(1, ISEG, IEDGE), R2(1, ISEG, IEDGE), R1DOTR2)
ARG = R1DOTR2/(RM1(ISEG, IEDGE)*RM2(ISEG, IEDGE))
IF(ABS(ARG).GT.1.) ARG = ARG/ABS(ARG)
THETA(ISEG, IEDGE) = ACOS(ARG)
GO TO 1000
400 CONTINUE
C USER SUPPLIED SPECIAL EDGE INITIALIZATION MAP = 8
C GO TO 1000
C USER SUPPLIED SPECIAL EDGE INITIALIZATION MAP = 9
C 500 CONTINUE
C 1000 CONTINUE
C RETURN
END
C********************************************
C*******************MAPPING*****************************
C********************************************
SUBROUTINE EDGMAP2(IEDGE, ISEG, RATIO, POINT, TANGENT)
C THIS ROUTINE CALCULATES THE COORDINATES OF A POINT ON AN EDGE.
COMMON /COEFF/ COEFE(8,10,12), COEPS(8,6), NMBRSEG(12)
COMMON /EDGE0/ UI(3,10,12), UJ(3,10,12), UK(3,10,12),
& R1(3,10,12), R2(3,10,12), THETA(10,12)
& COMMON /EDGE3/ ARC(10,12), ARC1(10,12), XLENGTH(10,12),
& RA(10,12), RC(10,12), RE(10,12), THETA1(10,12)
C-23
LOCKHEED-HUNTSVILLE ENGINEERING CENTER
COMMON /EDGE8/   RM1(10,12),RM2(10,12),RPA1(10,12),RPA2(10,12)
COMMON /MAPING/ MAPSIDE(6),MAPSEG(10,12)
COMMON /SEGMENT/ PT1(6,10,12),PT2(6,10,12),ETA1(10,12),ETA2(10,12)
COMMON /ZONING/ ISECT,NSECT,IZONE,NMBRND5(3)

DIMENSION POINT(6),TANGENT(3)
DIMENSION FLOWI(3),FLOWJ(3),FLOWK(3)
DIMENSION VECTOR(3),C(3)

MAP = MAPSEG(ISEG,IEDGE)

GO TO (100,200,300,400,500) MAP

C LINEAR EDGE

100 DO 110 I=1,3
   POINT(I) = (1. - RATIO)*R1(I,ISEG,IEDGE) + RATIO*R2(I,ISEG,IEDGE)
110 VECTOR(I) = R2(I,ISEG,IEDGE) - R1(I,ISEG,IEDGE)

CALL VMAG(VECTOR,RMAG)

IF(RMAG.EQ.0.0) RMAG = 1.0

DO 120 I=1,3
120 TANGENT(I) = (R2(I,ISEG,IEDGE) - R1(I,ISEG,IEDGE))/RMAG

GO TO 1000

C CIRCULAR ARC

MAP = 2

200 PHI1 = (1.0 - RATIO)*THETA(ISEG,IEDGE)
PHII = RATIO*THETA(ISEG,IEDGE)

DO 210 I=1,3
   POINT(I) = COEFE(I,ISEG,IEDGE) +
   1   (SIN(PHI1)*R1(I,ISEG,IEDGE) + SIN(PHI2)*R2(I,ISEG,IEDGE))
   2 /SIN(THETA(ISEG,IEDGE))
210 TANGENT(I) = THETA(ISEG,IEDGE)*
   1   (COS(PHI2)*R2(I,ISEG,IEDGE) - COS(PHI1)*R1(I,ISEG,IEDGE))
   2 /SIN(THETA(ISEG,IEDGE))

GO TO 1000

C EDGE OF REVOLUTION

MAP = 4

C PROJECTION ALONG AXIS AND RADIUS

300 RP = (RPA2(ISEG,IEDGE)-RPA1(ISEG,IEDGE))*RATIO + RPA1(ISEG,IEDGE)
RM = ( RM2(ISEG,IEDGE) - RM1(ISEG,IEDGE))*RATIO + RM1(ISEG,IEDGE)

C ANGLE

GAMMA = THETA(ISEG,IEDGE)*RATIO

C CALCULATE THE POSITION AND TANGENT

DO 410 I=1,3
UR = COS(GAMMA)*UI(I,ISEG,IEDGE) 
& + SIN(GAMMA)*UJ(I,ISEG,IEDGE)

POINT(I) = COEFE(I,ISEG,IEDGE) + RP*UK(I,ISEG,IEDGE) + RM*UR

TANGENT(I) = COS(GAMMA)*UJ(I,ISEG,IEDGE) 
& - SIN(GAMMA)*UI(I,ISEG,IEDGE)

GO TO 1000

HGM HOLE MAP = 8

400 IDUCT = IZONE - 3

IF(IDUCT.EQ.1) THEN

IF(IEDGE.EQ.5) THEN

ICLOCK = 0
ANGLE = (1.0 - RATIO)*130.0 + RATIO*225.0

END IF

IF(IEDGE.EQ.6) THEN

ICLOCK = 1
ANGLE = (1.0 - RATIO)*50.0 - RATIO*50.0

END IF

IF(IEDGE.EQ.1) THEN

ICLOCK = 1
ANGLE = (1.0 - RATIO)*130.0 + RATIO*50.0

END IF

IF(IEDGE.EQ.9) THEN

ICLOCK = 0
ANGLE = (1.0 - RATIO)*225.0 + RATIO*310.0

END IF

ELSE

IF(IEDGE.EQ.1) THEN

ICLOCK = 1
ANGLE = (1.0 - RATIO)*130.0 + RATIO*50.0

END IF

IF(IEDGE.EQ.5) THEN

END IF
ICLOCK = 0
ANGLE = (1.0 - RATIO)*130.0 + RATIO*180.0
END IF

IF (IEDGE.EQ.6) THEN
ICLOCK = 1
ANGLE = (1.0 - RATIO)*50.0
END IF

CALL HOLES(ANGLE,POINT,TANGENT,ICLOCK,IDOCT)
GO TO 1000

HGM FAIRING
CALL FAIRING(PT1(1,ISEG,IERGE),PT2(1,ISEG,IERGE),POINT,TANGENT,C)
GO TO 1000

1000 CONTINUE
RETURN
END.

SUBROUTINE SURFACE(INIT,ISIDE)

IMPLICIT DOUBLE PRECISION (A-H,O-Z)

COMMON /COEFF/ COEFS(8,10,12),COEFS(8,6),NMBRSEG(12)
COMMON /COORD/ UA(3,6),UE(3,6),UN(3,6)
COMMON /COUNTER/ NODESAV,MATESAV,NODETOT,NBNODES,NPLANE
COMMON /HEADER/ ITITLE(20),LINE
COMMON /HGM/ IHOLE
COMMON /HOTGAS/ IHGM,ETA11,ETA12,ETA13,ETA31,ETA32,ETA33,
                   STR11,STR12,STR13,STR31,STR32,STR33,STR34
COMMON /INITA/ IDIM,IFORMAT,MATRIX,MATING,MAPTEN,INCHES
COMMON /INITB/ IBOX(4,6),IRAY(4,3),NRAY,IPT1(12),IPT2(12)
COMMON /INITC/ PI,RADDEG
COMMON /INPUTA/ EDGE(6,12),POINT(6,8),SIDE(6,6),INPUTBC(6)
COMMON /INUTB/ IBOX(4,6),IRAY(4,3),NRAY,IPT1(12),IPT2(12)
COMMON /INUTC/ PI,RADDEG
COMMON /MAPED/ KSEG(12),UAXIS(3,6),IEDGE1(6),IEDGE2(6),GAMMA
COMMON /MAPING/ MAPSEG(10,12)
COMMON /MAXIMUM/ ETAMAX(10,12),SEGMAX(6,10,12)
COMMON /PARTIAL/ DEDN(3,12),DSMN(3,2),SNORMA(3,6)
COMMON /SPACING/ ISTRCH(3),STRETCH(3),ETAS(3,200),ETA(3),DETA(3)
COMMON /UNITS/ NU5,NU6,NU19,NU20
COMMON /ZONING/ ISECT,NSECTION,IZONE,NMBRNGS(3)

DIMENSION EDG(7,4),EDGX(4),EDGA(4),PT(7,4),PTX(4),PTA(4)
DIMENSION ET1(4),ET3(4),SWEEP(3),VDUM(3)
DIMENSION F(8),E1(3),E2(3),R1(3),R2(3)
DIMENSION ZERO(3),VECTER(3)
DIMENSION IETA1(6),IETA2(6),SIGNS(6)
DIMENSION UNITRAD(3)

DATA IETA1 /1,1,1,1,2,2/
DATA IETA2 /2,3,2,3,3,3/
DATA SIGNS /1.0,-1.0,-1.0,1.0,1.0,-1.0/
DATA ZERO /0.0,0.,0./

NODNUM = NODESAV + NODETOT + 1

C --- ETA COEFFICIENTS FOR BI-LINEAR INTERPOLATION

F(1) = 1.0 - ETA(IETA1(ISIDE))
F(2) = ETA(IETA1(ISIDE))
F(3) = 1.0 - ETA(IETA2(ISIDE))
F(4) = ETA(IETA2(ISIDE))
F(5) = F(1)*F(3)
F(6) = F(1)*F(4)
F(7) = F(2)*F(3)
F(8) = F(2)*F(4)

C --- DETERMINE THE EDGES OF THE SURFACE

LINE1 = IBOX(1,ISIDE)
LINE2 = IBOX(2,ISIDE)
LINE3 = IBOX(3,ISIDE)
LINE4 = IBOX(4,ISIDE)

C --- DETERMINE THE CORNER POINTS OF THE SURFACE

LPT1 = IPT1(LINE1)
LPT2 = IPT1(LINE3)
LPT3 = IPT2(LINE3)
LPT4 = IPT2(LINE1)

MAP = MAPSIDE(ISIDE)

GO TO (100,100,300,400,500,600,700,800) MAP

C --- FLAT OR CYLINDRICAL SURFACE

MAP = 1 OR 2

C --- CALCULATE POSITION

100 DO 110 J=1,3

SIDE(J,ISIDE) = F(1)*EDGE(J,LINE1) + F(5)*POINT(J,LPT1)
  + F(2)*EDGE(J,LINE3) + F(6)*POINT(J,LPT4)
  + F(3)*EDGE(J,LINE2) + F(7)*POINT(J,LPT2)
  + F(4)*EDGE(J,LINE4) + F(8)*POINT(J,LPT3)

DSDN(J,1) = EDGE(J,LINE3) + F(3)*POINT(J,LPT1)
  - EDGE(J,LINE1) + F(4)*POINT(J,LPT4)
  + F(3)*DEDN(J,LINE2) - F(3)*POINT(J,LPT2)
  + F(4)*DEDN(J,LINE4) - F(4)*POINT(J,LPT3)

C-27

LOCKHEED-HUNTSVILLE ENGINEERING CENTER
C

110  DSDN(J,2) =   EDGE(J,LINE4) + F(1)*POINT{J,LPT1)

1   - EDGE(J,LINE2) + F(2)*POINT(J,LPT2)

2   + F(1)*DEDN(J,LINE1) - F(1)*POINT(J,LPT4)

3   + F(2)*DEDN(J,LINE3) - F(2)*POINT(J,LPT3)

C

CALL CROSS(DSDN(1,1),DSDN(1,2),SNORMAL(1,ISIDE),40)
C

GO TO 1000
C

C SPECIAL SURFACE MAP = 3
C

300 GO TO 1000
C

C EDGE OF REVOLUTION MAP = 4
C

400 IF(INIT.EQ.1) THEN
C
C-----ETA DIRECTION OF REVOLUTION
C

IF(COEFS(7,ISIDE).EQ.IETA2(ISIDE)) IONETWO = 1
C

C-----EDGE 1
C

IEDGE1(ISIDE) = IBOX(IONETWO,ISIDE)
C

C-----EDGE 2
C

IEDGE2(ISIDE) = IBOX(IONETWO + 2,ISIDE)
C

C-----UA: NORMALIZED VECTOR ALONG AXIS OF REVOLUTION
C

CALL VADD(1.0,COEFS(4,ISIDE),1.0,ZERO,VECTER,UA(1,ISIDE))
C

C-----UN: NORMALIZED VECTOR FROM AXIS TOWARD SURFACE
C

CALL VADD(1.0,EDGE(1,IEDGE1(ISIDE)),-1.0,COEFS(1,ISIDE),E1,VECTER)
C

CALL CROSS(UA(1,ISIDE),E1,UN(1,ISIDE),41)
C

C-----UE: NORMALIZED VECTOR FROM AXIS TO FIRST EDGE
C

CALL CROSS(UN(1,ISIDE),UA(1,ISIDE),UE(1,ISIDE),42)
C

END IF
C

C-----R1: VECTOR FROM AXIS TO FIRST EDGE
C

CALL VADD(1.0,EDGE(1,IEDGE1(ISIDE)),-1.0,COEFS(1,ISIDE),E1,VECTER)
C

CALL VDOT(E1,UA(1,ISIDE),EPA1)
C

CALL VADD(1.0,E1,-EPA1,UA(1,ISIDE),R1,VECTER)
C

CALL VMAG(R1,RM1)
C

C-----DERIVATIVE
C

CALL VDOT(DEDN(1,IEDGE1(ISIDE)),UA(1,ISIDE),DA1)
CALL VDOT(DEDN(1,LEDGE1(ISIDE)),VECTER,DR1)
CALL VMAG(DEDN(1,LEDGE1(ISIDE)),DM1)

--R2: VECTOR FROM AXIS TO SECOND EDGE-----------------------------------
CALL VADD(1.0,EDGE(1,LEDGE2(ISIDE)),-1.0,COEFS(1,ISIDE),E2,VECTER)
CALL VDOT(E2,UA(1,ISIDE),EPA2)
CALL VADD(1.0,E2,-EPA2,UA(1,ISIDE),R2,VECTER)
CALL VMAG(R2,RM2)

--DERIVATIVE
CALL VDOT(DEDN(1,LEDGE2(ISIDE)),UA(1,ISIDE),DA2)
CALL VDOT(DEDN(1,LEDGE2(ISIDE)),VECTER,DR2)
CALL VMAG(DEDN(1,LEDGE2(ISIDE)),DM2)

--SWEEP ANGLE-------------------------------------------------------------
CALL VDOT(R1,R2,R1DOTR2)
ARG = R1DOTR2/(RM1*RM2)
IF(ABS(ARG).GT.1.0) ARG = ARG/ABS(ARG)
THETA = ACOS(ARG)

--CALCULATE RADIUS--------------------------------------------------------
N = COEFS(7,ISIDE)

--POSITION
EP = (EPA2 - EPA1)*ETA(N) + EPA1
RM = (RM2 - RM1 )*ETA(N) + RM1

--DERIVATIVE
DA = (DA2 - DA1)*ETA(N) + DA1
DR = (DR2 - DR1)*ETA(N) + DR1
DM = (DM2 - DM1)*ETA(N) + DM1

--ANGLE
GAMMA = THETA*ETA(N)

--CALCULATE THE POSITION AND SURFACE NORMAL-------------------------------
DO 410 I=1,3
UR = COS(GAMMA)*UE(I,ISIDE) + SIN(GAMMA)*UN(I,ISIDE)
SIDE(I,ISIDE) = COEFS(I,ISIDE) + EP*UA(I,ISIDE) + RM*UR

C-29

LOCKHEED-HUNTSVILLE ENGINEERING CENTER
DSDN(I,1) = (DR*UR + DA*UA(I,ISIDE))/DM

DSDN(I,2) = COS(GAMMA)*UN(I,ISIDE) - SIN(GAMMA)*UE(I,ISIDE)

C----ORIENT SURFACE NORMAL DEPENDING ON DIRECTION OF EDGE REVOLUTION
C
C IF(COEFS(7,ISIDE).EQ.IETA2(ISIDE)) THEN
C
CALL CROSS(DSDN(1,1),DSDN(1,2),SNORMAL(1,ISIDE),43)
C
ELSE
C
CALL CROSS(DSDN(1,2),DSDN(1,1),SNORMAL(1,ISIDE),44)
C
END IF
C
GO TO 1000
C
C BOWL SURFACE WITH HOLES
C
500 ETA1 = F(2)
ET1(1) = (ETA11 - 1.0) / REAL(NMBRNS(1) - 1)
ET1(2) = (ETA12 - 1.0) / REAL(NMBRNS(1) - 1)
ET1(3) = (ETA13 - 1.0) / REAL(NMBRNS(1) - 1)
ET1(4) = 1.0
C
IET1 = 1
IF(ETA1.GE.ET1(1)) IET1 = 2
IF(ETA1.GE.ET1(2)) IET1 = 3
IF(ETA1.GE.ET1(3)) IET1 = 4
C
ETA3 = F(4)
ET3(1) = (ETA31 - 1.0) / REAL(NMBRNS(3) - 1)
ET3(2) = (ETA32 - 1.0) / REAL(NMBRNS(3) - 1)
ET3(3) = (ETA33 - 1.0) / REAL(NMBRNS(3) - 1)
ET3(4) = 1.0
C
IET3 = 1
IF(ETA3.GE.ET3(1)) IET3 = 2
IF(ETA3.GE.ET3(2)) IET3 = 3
IF(ETA3.GE.ET3(3)) IET3 = 4
C
IHOLE = 0
ICLOCK = 0
C
GO TO (510,520,540,560) IET1
C
IET1 = 1
C
510 CALL CONVERT(EDGE(1,3),EDGX(1),EDGA(1))
C
CALL CONVERT(EDGE(1,11),EDGX(3),EDGA(3))
C
RANGE = EDGX(1)
C
ANGLE = (1.0 - ETA3)*EDGA(1) + ETA3*EDGA(3)
C
GO TO 580
C
IET1 = 2
C
C-30

LOCKHEED-HUNTSVILLE ENGINEERING CENTER
C------------------------------------------------------

520  RATIO1 = (ETA1 - ET1(1)) / (ET1(2) - ET1(1))
   
   X1 = RATIO1*STR11/2.0
   ETAX1 = (EXP(X1) - EXP(-X1))/(EXP(X1) + EXP(-X1))
   
   X2 = STR11/2.0
   ETAX2 = (EXP(X2) - EXP(-X2))/(EXP(X2) + EXP(-X2))
   
   EPS1 = ETAX1/ETAX2
   
   GO TO (522,524,526,528) IET3

C---IET3 = 1------------------------------------------------------

522  RATIO3 = ETA3/ET3(1)
   
   X1 = RATIO3*STR31/2.0
   ETAX1 = (EXP(X1) - EXP(-X1))/(EXP(X1) + EXP(-X1))
   
   X2 = STR31/2.0
   ETAX2 = (EXP(X2) - EXP(-X2))/(EXP(X2) + EXP(-X2))
   
   EPS3 = ETAX1/ETAX2
   
   CALL CONVERT(EDGE(1,3),EDGX(1),EDGA(1))
   
   CALL CONVERT(EDGE(1,8),EDGX(4),EDGA(4))
   
   EDGX(4) = SEGMAX(1,1,3) - 6.07
   IDUCT = 1
   PT(7,3) = 130.0
   
   CALL HOLES(PT(7,3),PT(1,3),VDUM,ICLOCK,IDUCT)
   
   CALL CONVERT(PT(1,3),PTX(3),PTA(3))
   
   EDGX(3) = (1.0 - EPS1)*EDGX(4) + EPS1*PTX(3)
   
   RANGE = (1.0 - EPS3)*EDGX(1) + EPS3*EDGX(3)
   
   EDGA(2) = EPS3*PTA(3)
   
   ANGLE = (1.0 - EPS1)*EDGA(4) + EPS1*EDGA(2)
   
   GO TO 580

C---IET3 = 2------------------------------------------------------

524  RATIO3 = (ETA3 - ET3(1))/(ET3(2) - ET3(1))
   
   X1 = (STR32/2.0)/2.0
   ETAMID = (EXP(X1) - EXP(-X1))/(EXP(X1) + EXP(-X1))
   
   X2 = (RATIO3 - 0.5)*STR32/2.0
   ETAMX = (EXP(X2) - EXP(-X2))/(EXP(X2) + EXP(-X2))
   
   EPS3 = (ETAMX + ETAMID)/(2.0*ETAMID)

C-31

LOCKHEED-HUNTSVILLE ENGINEERING CENTER
CALL CONVERT(EDGE(1,8),EDGX(4),EDGA(4))
    EDGX(4) = SEGMAX(1,1,3) - 6.07
    IDUCT = 1
    EDG(7,2) = (1.0 - EPS3)*130.0 + EPS3*225.0
CALL HOLES(EDG(7,2),EDG(1,2),VDUM,ICLOCK,IDUCT)
CALL CONVERT(EDG(1,2),EDGX(2),EDGA(2))
    RANGE = (1.0 - EPS1)*EDGX(4) + EPS1*EDGX(2)
    ANGLE = (1.0 - EPS1)*EDGA(4) + EPS1*EDGA(2)
GO TO 580
---IET3 = 3-----------------------------------------

526 RATIO3 = (ETA3 - ET3(2))/(ET3(3) - ET3(2))
    XI = (STR33/2.0)/2.0
    ETAMID = (EXP(XI) - EXP(-XI))/(EXP(XI) + EXP(-XI))
    X2 = (RATIO3 - 0.5)*STR33/2.0
    ETAMX = (EXP(X2) - EXP(-X2))/(EXP(X2) + EXP(-X2))
    EPS3 = (ETAMX + ETAMID)/(2.0*ETAMID)
CALL CONVERT(EDGE(1,8),EDGX(4),EDGA(4))
    EDGX(4) = SEGMAX(1,1,3) - 6.07
    IDUCT = 1
    PT(7,2) = 225.0
CALL HOLES(PT(7,2),PT(1,2),VDUM,ICLOCK,IDUCT)
CALL CONVERT(PT(1,2),PTX(2),PTA(2))
    IDUCT = 2
    PT(7,3) = 130.0
CALL HOLES(PT(7,3),PT(1,3),VDUM,ICLOCK,IDUCT)
CALL CONVERT(PT(1,3),PTX(3),PTA(3))
    EDGX(2) = (1.0 - EPS3)*PTX(2) + EPS3*PTX(3)
    RANGE = (1.0 - EPS1)*EDGX(4) + EPS1*EDGX(2)
    EDGA(2) = (1.0 - EPS3)*PTA(2) + EPS3*PTA(3)
    ANGLE = (1.0 - EPS1)*EDGA(4) + EPS1*EDGA(2)
GO TO 580
---IET3 = 4-----------------------------------------

528 RATIO3 = 1.0 - (ETA3 - ET3(3))/(ET3(4) - ET3(3))
C

XI = RATIO3*STR34/2.0
ETAX1 = (EXP(X1) - EXP(-X1))/(EXP(X1) + EXP(-X1))

X2 = STR34/2.0
ETAX2 = (EXP(X2) - EXP(-X2))/(EXP(X2) + EXP(-X2))

EPS3 = 1.0 - ETAX1/ETAX2
CALL CONVERT(EDGE(1,8), EDGX(4), EDGA(4))

EDGX(4) = SEGMAX(1,1,3) - 6.07
IDUCT = 2
EDG(7,2) = (1.0 - EPS3)*130.0 + EPS3*180.0
CALL HOLES(EDG(7,2), EDG(1,2), VDUM, ICLOCK, IDUCT)
CALL CONVERT(EDG(1,2), EDG(2,2), EDG(2,2))

RANGE = (1.0 - EPS1)*EDGX(4) + EPS1*EDGX(2)
ANGLE = (1.0 - EPS1)*EDGA(4) + EPS1*EDGA(2)
GO TO 580

C---IET3 = 3

542 RATIO3 = ETA3/ET3(1)

XI = RATIO3*STR31/2.0
ETAX1 = (EXP(X1) - EXP(-X1))/(EXP(X1) + EXP(-X1))

X2 = STR31/2.0
ETAX2 = (EXP(X2) - EXP(-X2))/(EXP(X2) + EXP(-X2))

EPS3 = ETAX1/ETAX2
CALL CONVERT(EDGE(1,3), EDGX(1), EDGA(1))
IDUCT = 1
EDG(7,3) = (1.0 - EPS1)*130.0 + EPS1*50.0
CALL HOLES(EDG(7,3), EDG(1,3), VDUM, ICLOCK, IDUCT)
CALL CONVERT(EDG(1,3), EDG(3,3), EDG(3,3))
RANGE = (1.0 - EPS3)*EDGX(1) + EPS3*EDGX(3)

ANGLE = (1.0 - EPS3)*EDGA(1) + EPS3*EDGA(3)

GO TO 580

---IET3 = 2-----------------------------------------------

544  RATIO3 = (ETA3 - ET3(1))/(ET3(2) - ET3(1))

    X1 = (STR32/2.0)/2.0
    ETAMID = (EXP(X1) - EXP(-X1))/(EXP(X1) + EXP(-X1))

    X2 = (RATIO3 - 0.5)*STR32/2.0
    ETAMX = (EXP(X2) - EXP(-X2))/(EXP(X2) + EXP(-X2))

    EPS3 = (ETAMX + ETAMID)/(2.0*ETAMID)

    IHOLE = 1
    IDUCT = 1

EDG(7,1) = (1.0 - EPS1)*130.0 + EPS1*50.0
EDG(7,2) = (1.0 - EPS3)*50.0 + EPS3*(-50.0)
EDG(7,3) = (1.0 - EPS1)*225.0 + EPS1*310.0
EDG(7,4) = (1.0 - EPS3)*130.0 + EPS3*225.0

PT(7,1) = 130.0
PT(7,2) = 50.0
PT(7,3) = 310.0
PT(7,4) = 225.0

DO 545 N=1,4

CALL HOLES(EDG(7,N),EDG(1,N),VDUM,ICLOCK,IDUCT)
CALL CONVERT(EDG(1,N),EDGX(N),EDGA(N))
CALL HOLES(PT(7,N),PT(1,N),VDUM,ICLOCK,IDUCT)
CALL CONVERT(PT(1,N),PTX(N),PTA(N))

RANGE = (1.0 - EPS3)*EDGX(1) - (1.0 - EPS1)*(1.0 - EPS3)*PTX(1)
        + EPS1*EDGX(2) - (1.0 - EPS3)*EPS1*PTX(2)
        + EPS3*EDGX(3) - EPS1*EPS3*PTX(3)
        + (1.0 - EPS1)*EDGX(4) - (1.0 - EPS1)*EPS1*PTX(4)

ANGLE = (1.0 - EPS3)*EDGA(1) - (1.0 - EPS1)*(1.0 - EPS3)*PTA(1)
        + EPS1*EDGA(2) - (1.0 - EPS3)*EPS1*PTA(2)
        + EPS3*EDGA(3) - EPS1*EPS3*PTA(3)
        + (1.0 - EPS1)*EDGA(4) - (1.0 - EPS1)*EPS1*PTA(4)

GO TO 580

---IET3 = 3-----------------------------------------------

546  RATIO3 = (ETA3 - ET3(2))/(ET3(3) - ET3(2))

    X1 = (STR33/2.0)/2.0
    ETAMID = (EXP(X1) - EXP(-X1))/(EXP(X1) + EXP(-X1))
X2 = (RATIO3 - 0.5)*STR3/2.0
ETAMX = (EXP(X2) - EXP(-X2))/(EXP(X2) + EXP(-X2))
EPS3 = (ETAMX + ETAMID)/(2.0*ETAMID)

IDUCT = 1
EDG(7,1) = (1.0 - EPS1)*225.0 + EPS1*310.0
CALL HOLES(EDG(7,1),EDG(1,1),VDUM,ICLOCK,IDUCT)
CALL CONVERT(EDG(1,1),EDGX(1),EDGA(1))

IDUCT = 2
EDG(7,3) = (1.0 - EPS1)*130.0 + EPS1*50.0
CALL HOLES(EDG(7,3),EDG(1,3),VDUM,ICLOCK,IDUCT)
CALL CONVERT(EDG(1,3),EDGX(3),EDGA(3))
RANGE = (1.0 - EPS3)*EDGX(1) + EPS3*EDGX(3)
ANGLE = (1.0 - EPS3)*EDGA(1) + EPS3*EDGA(3)
GO TO 580

C---IET3 = 4-----------------------------------------------

RATIO3 = 1.0 - (ETA3 - ET3(3))/(ET3(4) - ET3(3))
X1 = RATIO3*STR3/2.0
ETAX1 = (EXP(X1) - EXP(-X1))/(EXP(X1) + EXP(-X1))
X2 = STR3/2.0
ETAX2 = (EXP(X2) - EXP(-X2))/(EXP(X2) + EXP(-X2))
EPS3 = 1.0 - ETAX1/ETAX2
IHOLE = 1
IDUCT = 2
EDG(7,1) = (1.0 - EPS1)*130.0 + EPS1*50.0
EDG(7,2) = (1.0 - EPS3)*50.0
EDG(7,4) = (1.0 - EPS3)*130.0 + EPS3*180.0
PT(7,1) = 130.0
PT(7,2) = 50.0
PT(7,3) = 0.0
PT(7,4) = 180.0
DO 549 N=1,4
CALL HOLES(EDG(7,N),EDG(1,N),VDUM,ICLOCK,IDUCT)
CALL CONVERT(EDG(1,N),EDGX(N),EDGA(N))
CALL HOLES(PT(7,N),PT(1,N),VDUM,ICLOCK,IDUCT)
549 CALL CONVERT(PT(1,N),PTX(N),PTA(N))

C-35
LOCKHED-HUNTSVILLE ENGINEERING CENTER
EDGX(3) = (1.0 - EPS1)*PTX(4) + EPS1*PTX(3)

EDGA(3) = (1.0 - EPS1)*PTA(4) + EPS1*PTA(3)

RANGE = (1.0 - EPS3)*EDGX(1) - (1.0 - EPS1)*(1.0 - EPS3)*PTX(1)
1 + EPS1*EDGX(2) - (1.0 - EPS3)*EPS1*PTX(2)
2 + EPS3*EDGX(3) - EPS1*EPS3*PTX(3)
3 + (1.0 - EPS1)*EDGX(4) - (1.0 - EPS1)*EPS3*PTX(4)

ANGLE = (1.0 - EPS3)*EDGA(1) - (1.0 - EPS1)*(1.0 - EPS3)*PTA(1)
1 + EPS1*EDGA(2) - (1.0 - EPS3)*EPS1*PTA(2)
2 + EPS3*EDGA(3) - EPS1*EPS3*PTA(3)
3 + (1.0 - EPS1)*EDGA(4) - (1.0 - EPS1)*EPS3*PTA(4)

GO TO 580

IET1 = 4

RATIO1 = (ETA1 - ET1(3))/(ET1(4) - ET1(3))

XI = (STR13/2.0)/2.0
ETAMID = (EXP(X1) - EXP(-X1))/(EXP(X1) + EXP(-X1))

X2 = (RATIO1 - 0.5)*STR13/2.0
ETAMX = (EXP(X2) - EXP(-X2))/(EXP(X2) + EXP(-X2))

EPS1 = (ETAMX + ETAMID)/(2.0*ETAMID)

GO TO (562,564,566,568) IET3

IET3 = 1

RATIO3 = ETA3/ET3(1)

X1 = RATIO3*STR31/2.0
ETAX1 = (EXP(X1) - EXP(-X1))/(EXP(X1) + EXP(-X1))

X2 = STR31/2.0
ETAX2 = (EXP(X2) - EXP(-X2))/(EXP(X2) + EXP(-X2))

EPS3 = ETAX1/ETAX2

CALL CONVERT(EDGE(1,3),EDGX(1),EDGA(1))

CALL CONVERT(EDGE(1,7),EDGX(2),EDGA(2))

IDUCT = 1
PT(7,4) = 50.0

CALL HOLES(PT(7,4),PT(1,4),VDUM,ICLOCK,IDUCT)

CALL CONVERT(PT(1,4),PTX(4),PTA(4))

EDGX(3) = (1.0 - EPS1)*PTX(4) + EPS1*EDGX(2)

RANGE = (1.0 - EPS3)*EDGX(1) + EPS3*EDGX(3)

EDGA(4) = EPS3*PTA(4)

ANGLE = (1.0 - EPS1)*EDGA(4) + EPS1*EDGA(2)
C
GO TO 580
C
---IET3 = 2---------------------------------------------
C
564  RATIO3 = (ETA3 - ET3(1))/(ET3(2) - ET3(1))
      X1 = (STR32/2.0)/2.0
      ETAMID = (EXP(X1) - EXP(-X1))/(EXP(X1) + EXP(-X1))
      X2 = (RATIO3 - 0.5)*STR32/2.0
      ETAMX = (EXP(X2) - EXP(-X2))/(EXP(X2) + EXP(-X2))
      EPS3 = (ETAMX + ETAMID)/(2.0*ETAMID)
      CALL CONVERT(EDGE(1,7),EDGX(2),EDGA(2))
      IDUCT = 1
      EDG(7,4) = (1.0 - EPS3)*50.0 + EPS3*(-50.0)
      CALL HOLES(EDG(7,4),EDG(1,4),VDUM,ICLOCK,IDUCT)
      CALL CONVERT(EDGE(1,4),EDGX(4),EDGA(4))
      RANGE = (1.0 - EPS1)*EDGX(4) + EPS1*EDGX(2)
      ANGLE = (1.0 - EPS1)*EDGA(4) + EPS1*EDGA(2)
      GO TO 580
C
---IET3 = 3---------------------------------------------
C
566  RATIO3 = (ETA3 - ET3(2))/(ET3(3) - ET3(2))
      X1 = (STR33/2.0)/2.0
      ETAMID = (EXP(X1) - EXP(-X1))/(EXP(X1) + EXP(-X1))
      X2 = (RATIO3 - 0.5)*STR33/2.0
      ETAMX = (EXP(X2) - EXP(-X2))/(EXP(X2) + EXP(-X2))
      EPS3 = (ETAMX + ETAMID)/(2.0*ETAMID)
      CALL CONVERT(EDGE(1,7),EDGX(2),EDGA(2))
      IDUCT = 1
      PT(7,1) = 310.0
      CALL HOLES(PT(7,1),PT(1,1),VDUM,ICLOCK,IDUCT)
      CALL CONVERT(PT(1,1),PTX(1),PTA(1))
      IDUCT = 2
      PT(7,4) = 50.0
      CALL HOLES(PT(7,4),PT(1,4),VDUM,ICLOCK,IDUCT)
      CALL CONVERT(PT(1,4),PTX(4),PTA(4))
      EDGX(4) = (1.0 - EPS3)*PTX(1) + EPS3*PTX(4)
RANGE = (1.0 - EPS1)*EDGX(4) + EPS1*EDGX(2)

EDGA(4) = (1.0 - EPS3)*PTA(1) + EPS3*PTA(4)

ANGLE = (1.0 - EPS1)*EDGA(4) + EPS1*EDGA(2)

GO TO 580

--- IET3 = 4 ---------------------------------------------------------------

568 RATIO3 = 1.0 - (ETA3 - ET3(3))/(ET3(4) - ET3(3))

XI = RATIO3*STR34/2.0
ETAX1 = (EXP(X1) - EXP(-X1))/(EXP(X1) + EXP(-X1))

X2 = STR34/2.0
ETAX2 = (EXP(X2) - EXP(-X2))/(EXP(X2) + EXP(-X2))

EPS3 = 1.0 - ETAX1/ETAX2

CALL CONVERT(EDGE(1,7), EDGX(2), EDGA(2))

IDUCT = 2
EDG(7,4) = (1.0 - EPS3)*50.0

CALL HOLES(EDG(7,4), EDG(1,4), VDUM, ICLOCK, IDUCT)

CALL CONVERT(EDG(1,4), EDGX(4), EDGA(4))

RANGE = (1.0 - EPS1)*EDGX(4) + EPS1*EDGX(2)

ANGLE = (1.0 - EPS1)*EDGA(4) + EPS1*EDGA(2)

GO TO 580

580 CONTINUE

--- RADIUS AND TANGENT ----------------------------------------------------

XX = RANGE + 6.07

--- FIRST SEGMENT

IF(XX.LE.SEGMAX(1,1,3)) THEN

RAD = 7.434

DSDN(1,1) = 1.0
DSDN(2,1) = 0.0
DSDN(3,1) = 0.0

GO TO 590

END IF

--- SECOND SEGMENT

IF(XX.LT.SEGMAX(1,2,3)) THEN

RAD = SQRT(7.890**2 - (3.375 - RANGE)**2)

C-38

LOCKHEED-HUNTSVILLE ENGINEERING CENTER
TH = ACOS(RAD/7.890)

IF(XX.GT.9.445) TH = -TH

DSDN(1,1) = COS(TH)
DSDN(2,1) = SIN(TH)
DSDN(3,1) = 0.0

GO TO 590

END IF

---LAST SEGMENT---

RAD = 14.5636605 - 0.57735027*XX

TH = -30.0*RADDEG

DSDN(1,1) = COS(TH)
DSDN(2,1) = SIN(TH)
DSDN(3,1) = 0.0

---POSITION---

590 SWEEP(1) = 0.0
SWEEP(2) = COS(ANGLE)
SWEEP(3) = SIN(ANGLE)

UA(1,ISIDE) = 1.0
UA(2,ISIDE) = 0.0
UA(3,ISIDE) = 0.0

CALL VADD(XX,UA(1,ISIDE),RAD,SWEEP,SIDE(1,ISIDE),VDUM)

---NORMAL---

---TANGENT 1---

DA = DSDN(1,1)
DN = DSDN(2,1)

CALL VADD(DN,SWEEP,DA,UA(1,ISIDE),VDUM,DSDN(1,1))

---TANGENT 2---

DSDN(1,2) = 0.0
DSDN(2,2) = -SIN(ANGLE)
DSDN(3,2) = COS(ANGLE)

---SURFACE NORMAL---

CALL CROSS(DSDN(1,1),DSDN(1,2),SNORMAL(1,ISIDE),45)

GO TO 1030

---MATED SURFACE OF DUCT---

MAP = 6

600 EPS1 = F(2)
EPS3 = F(4)
IDUCT = IZONE - 3

IF(IDUCT.EQ.1) THEN

C----UPPER DUCT---------------------------------------------

EDG(7,1) = (1.0 - EPS1)*130.0 + EPS1*50.0
EDG(7,2) = (1.0 - EPS3)*50.0 + EPS3*(-50.0)
EDG(7,3) = (1.0 - EPS1)*225.0 + EPS1*310.0
EDG(7,4) = (1.0 - EPS3)*130.0 + EPS3*225.0

PT(7,1) = 130.0
PT(7,2) = 50.0
PT(7,3) = 310.0
PT(7,4) = 225.0

ELSE

C----LOWER DUCT---------------------------------------------

EDG(7,1) = (1.0 - EPS1)*130.0 + EPS1*50.0
EDG(7,2) = (1.0 - EPS3)*50.0
EDG(7,4) = (1.0 - EPS3)*130.0 + EPS3*180.0

PT(7,1) = 130.0
PT(7,2) = 50.0
PT(7,3) = 0.0
PT(7,4) = 180.0

END IF

C----CALCULATE POINTS AND EDGES--------------------------------

DO 610 N=1,4

CALL HOLES(EDG(7,N),EDG(1,N),VDUM,ICLOCK,IDUCT)
CALL CONVERT(EDG(1,N),EDGX(N),EDGA(N))
CALL HOLES(PT(7,N),PT(1,N),VDUM,ICLOCK,IDUCT)

610 CALL CONVERT(PT(1,N),PTX(N),PTA(N))

IF(IDUCT.EQ.2) THEN

EDGX(3) = (1.0 - EPS1)*PTX(4) + EPS1*PTX(3)
EDGA(3) = (1.0 - EPS1)*PTA(4) + EPS1*PTA(3)

END IF

C----DISTANCE ALONG AXIS AND ANGLE FROM VERTICAL

RANGE = (1.0 - EPS3)*EDGX(1) - (1.0 - EPS1)*(1.0 - EPS3)*PTX(1)
2 + EPS1*EDGX(2) - (1.0 - EPS3)*EPS1*PTX(2)
3 + EPS3*EDGX(3) - EPS1*EPS3*PTX(3)

ANGLE = (1.0 - EPS3)*EDGA(1) - (1.0 - EPS1)*(1.0 - EPS3)*PTA(1)
2 + EPS1*EDGA(2) - (1.0 - EPS3)*EPS1*PTA(2)
3

C-40

LOCKHED-HUNTSVILLE ENGINEERING CENTER
\[ 2 + \text{EPS3} \cdot \text{EDGA}(3) \quad - \quad \text{EPS1} \cdot \text{EPS3} \cdot \text{PTA}(3) \]
\[ 3 + (1.0 - \text{EPS1}) \cdot \text{EDGA}(4) \quad - \quad (1.0 - \text{EPS1}) \cdot \text{EPS3} \cdot \text{PTA}(4) \]

C-POSITION-----------------------------------------------

C
RAD = \sqrt{(7.890^2 - (3.375 - \text{RANGE})^2)}

C
SIDE(1,ISIDE) = \text{RANGE} + 6.07
SIDE(2,ISIDE) = \text{RAD} \cdot \cos(\text{ANGLE})
SIDE(3,ISIDE) = \text{RAD} \cdot \sin(\text{ANGLE})

C---NORMAL-----------------------------------------------

C
VDUM(1) = SIDE(1,ISIDE) - 9.445
VDUM(2) = SIDE(2,ISIDE)
VDUM(3) = SIDE(3,ISIDE)

C
CALL VMAG(VDUM,SMAG)

C
SNORMAL(1,ISIDE) = VDUM(1)/SMAG
SNORMAL(2,ISIDE) = VDUM(2)/SMAG
SNORMAL(3,ISIDE) = VDUM(3)/SMAG

C
GO TO 1030

C
OUTER SURFACE OF DUCT MAP = 7

C
700 IF(ISIDE.LT.4) THEN

C
SURFACE 1 AND 3
M = LINE2
N = LINE4

C
ELSE
C
SURFACE 5 AND 6
M = LINE1
N = LINE3
C
END IF
C
CALL FAIRING(EDGE(1,M),EDGE(1,N),SIDE(1,ISIDE),
VDUM,SNORMAL(1,ISIDE))
C
GO TO 1030

C
SPECIAL SURFACE MAP = 8

C
800 GO TO 1000

C
DIRECT SURFACE NORMAL INTO FLOW DOMAIN

C
1000 DO 1010 I=1,3
1010 SNORMAL(I,ISIDE) = SNORMAL(I,ISIDE)*SIGNS(ISIDE)
C
C
ERROR MESSAGE IF SURFACE NORMAL = 0
C
1030 CALL VMAG(SNORMAL(1,ISIDE),SMAG)
C

C-41

LOCKHEED-HUNTSVILLE ENGINEERING CENTER
IF(SNMAG.EQ.0.0) THEN
   LINE = LINE + 5
   IF (LINE.GE.60) THEN
      WRITE(NU6,1100) ITITLE,ISECT,NSECT,IZONE
      LINE = 6
   END IF
   WRITE(NU6,1110)
RETURN
END IF

INTERMEDIATE PRINT

IF(IWRTI.EQ.0 .OR. MOD(NODNUM,IWRTI).NE.0) RETURN
   LINE = LINE + 3
   IF(LINE.GE.60) THEN
      WRITE(NU6,1100) ITITLE,ISECT,NSECT,I2ONE
      LINE = 4
   END IF
RETURN

FORMAT STATEMENTS

1100 FORMAT(1H1,10X,20A4,13X,8H SECTION,12,3H OF,I3,9H FOR ZONE,I3)
1110 FORMAT(/ 49H SURFACE NORMAL EQUALS ZERO IN SUBROUTINE SURFACE)

END

SUBROUTINE CONVERT(POINT,RANGE,ANGLE)
CALCULATES PROJECTION AND SWEEP ANGLE OF POINT ON BOWL SURFACE

POINT = POSITION ON BOWL SURFACE
RANGE = PROJECTION ONTO X AXIS FROM EDGE OF BOWL
ANGLE = SWEEP ANGLE FROM VERTICAL

IMPLICIT DOUBLE PRECISION (A-H,O-Z)
DIMENSION POINT(3),BOWL(3),AXIS(3),VECTOR(3),SWEEP(3),VDUM(3)

C-42

LOCKHED-HUNTSVILLE ENGINEERING CENTER
C

DATA BOWL /6.07,0.0,0.0/
DATA AXIS /1.0,0.0,0.0/

C---DISTANCE ALONG AXIS FROM EDGE OF BOWL
C
ONE = 1.0
C
CALL VADD(ONE,POINT,-ONE,BOWL,VECTOR,VDUM)
C
CALL VDOT(VECTOR,AXIS,RANGE)

C---ANGLE FROM VERTICAL
C
CALL VADD(ONE,VECTOR,-RANGE,AXIS,VDUM,SWEEP)
C
ANGLE = ACOS(SWEEP(2))
C
RETURN

END

***************

SUBROUTINE BLKOUT(NUNIT,NODSTOR)

C WRITES THE FORMATTED BLOCKEDGEOMETRY FILE (NUNIT)

COMMON /INITA/ IDIM,MAPTEN,INCHES
COMMON /IOCOUN/ IREWIND(40),NREAD(40),NWRITE(40)
COMMON /OUT/ NODE(5,10000)
COMMON /ZONING/ ISECT,NSECT,IZONE,NMBRNDS(3)
COMMON /MARCHS/ MARCH,INDEX(3)

IPLANE = NWRITE(NUNIT) + 1

WRITE(NUNIT,1000) NODSTOR,IPLANE,(NMBRNDS(I),1=1,3),MARCH
WRITE(NUNIT,1010) (NODE(1,I),1=1,NODSTOR)
WRITE(NUNIT,1010) (NODE(2,I),I=1,NODSTOR)
WRITE(NUNIT,1010) (NODE(3,I),I=1,NODSTOR)

RETURN

C FORMAT STATEMENTS

1000 FORMAT(24I5)
1010 FORMAT(10E13.7)

END

C

SUBROUTINE REWRITE

C

C-43

LOCKHEED-HUNTSVILLE ENGINEERING CENTER
This program makes the HGM geometry output file compatible with PLOT3D geometry files.

```
COMMON /ZONING/ ISECT,NSECT,IZONE,NMBRND5S(3)
COMMON /UNITS/ NU5,NU6,NU19,NU20,NU21
COMMON /MARCHS/ MARCH,INDEX(3)

DIMENSION ID(10),JD(10),KD(10)
DIMENSION XBUF(155000),YBUF(155000),ZBUF(155000)

REWIND (NU20)

12 = 0
NGRID = 0
READ(NU20,9000,END=200)NSTORE,IPLN,INOD2,JNOD2,KNOD2,
* MARCH2

9000 FORMAT(8I5)
WRITE(NU6,9000)NSTORE,IPLN,INOD2,JNOD2,KNOD2,MARCH2

II = I2 + 1
I2 = I2 + NSTORE

READ(NU20,9010,ERR=400)(XBUF(I),I=I1,I2)
9010 FORMAT(10E13.7)
READ(NU20,9010,ERR=400)(YBUF(I),I=I1,I2)
READ(NU20,9010,ERR=400)(ZBUF(I),I=I1,I2)

100 READ(NU20,9000,END=200)NSTORE,IPLN,INOD,JNOD,KNOD,MARCH
WRITE(NU6,9000)NSTORE,IPLN,INOD,JNOD,KNOD,MARCH

IF(INOD.NE.INOD2.OR.JNOD.NE.JNOD2.OR.KNOD.NE.KNOD2)THEN
NGRID = NGRID + 1

IF(MARCH2.EQ.1)THEN
ID(NGRID) = KNOD2
JD(NGRID) = JNOD2
KD(NGRID) = INOD2
END IF

IF(MARCH2.EQ.2)THEN
ID(NGRID) = INOD2
JD(NGRID) = KNOD2
KD(NGRID) = JNOD2
END IF

IF(MARCH2.EQ.3)THEN
ID(NGRID) = JNOD2
JD(NGRID) = KNOD2
KD(NGRID) = INOD2
END IF
```

This program makes the HGM geometry output file compatible with PLOT3D geometry files.

**COMMON /ZONING/ ISECT,NSECT,IZONE,NMBRND5S(3)**

**COMMON /UNITS/ NU5,NU6,NU19,NU20,NU21**

**COMMON /MARCHS/ MARCH,INDEX(3)**

**DIMENSION ID(10),JD(10),KD(10)**

**DIMENSION XBUF(155000),YBUF(155000),ZBUF(155000)**

**REWIND (NU20)**

```
12 = 0
NGRID = 0
READ(NU20,9000,END=200)NSTORE,IPLN,INOD2,JNOD2,KNOD2,
* MARCH2

9000 FORMAT(8I5)
WRITE(NU6,9000)NSTORE,IPLN,INOD2,JNOD2,KNOD2,MARCH2

II = I2 + 1
I2 = I2 + NSTORE

READ(NU20,9010,ERR=400)(XBUF(I),I=I1,I2)
9010 FORMAT(10E13.7)
READ(NU20,9010,ERR=400)(YBUF(I),I=I1,I2)
READ(NU20,9010,ERR=400)(ZBUF(I),I=I1,I2)

100 READ(NU20,9000,END=200)NSTORE,IPLN,INOD,JNOD,KNOD,MARCH
WRITE(NU6,9000)NSTORE,IPLN,INOD,JNOD,KNOD,MARCH

IF(INOD.NE.INOD2.OR.JNOD.NE.JNOD2.OR.KNOD.NE.KNOD2)THEN
NGRID = NGRID + 1

IF(MARCH2.EQ.1)THEN
ID(NGRID) = KNOD2
JD(NGRID) = JNOD2
KD(NGRID) = INOD2
END IF

IF(MARCH2.EQ.2)THEN
ID(NGRID) = INOD2
JD(NGRID) = KNOD2
KD(NGRID) = JNOD2
END IF

IF(MARCH2.EQ.3)THEN
ID(NGRID) = JNOD2
JD(NGRID) = KNOD2
KD(NGRID) = INOD2
END IF
```

This program makes the HGM geometry output file compatible with PLOT3D geometry files.

**COMMON /ZONING/ ISECT,NSECT,IZONE,NMBRND5S(3)**

**COMMON /UNITS/ NU5,NU6,NU19,NU20,NU21**

**COMMON /MARCHS/ MARCH,INDEX(3)**

**DIMENSION ID(10),JD(10),KD(10)**

**DIMENSION XBUF(155000),YBUF(155000),ZBUF(155000)**

**REWIND (NU20)**

```
12 = 0
NGRID = 0
READ(NU20,9000,END=200)NSTORE,IPLN,INOD2,JNOD2,KNOD2,
* MARCH2

9000 FORMAT(8I5)
WRITE(NU6,9000)NSTORE,IPLN,INOD2,JNOD2,KNOD2,MARCH2

II = I2 + 1
I2 = I2 + NSTORE

READ(NU20,9010,ERR=400)(XBUF(I),I=I1,I2)
9010 FORMAT(10E13.7)
READ(NU20,9010,ERR=400)(YBUF(I),I=I1,I2)
READ(NU20,9010,ERR=400)(ZBUF(I),I=I1,I2)

100 READ(NU20,9000,END=200)NSTORE,IPLN,INOD,JNOD,KNOD,MARCH
WRITE(NU6,9000)NSTORE,IPLN,INOD,JNOD,KNOD,MARCH

IF(INOD.NE.INOD2.OR.JNOD.NE.JNOD2.OR.KNOD.NE.KNOD2)THEN
NGRID = NGRID + 1

IF(MARCH2.EQ.1)THEN
ID(NGRID) = KNOD2
JD(NGRID) = JNOD2
KD(NGRID) = INOD2
END IF

IF(MARCH2.EQ.2)THEN
ID(NGRID) = INOD2
JD(NGRID) = KNOD2
KD(NGRID) = JNOD2
END IF

IF(MARCH2.EQ.3)THEN
ID(NGRID) = JNOD2
JD(NGRID) = KNOD2
KD(NGRID) = INOD2
END IF
```
WRITE (NU6, 9120) NGRID, ID(NGRID), JD(NGRID), KD(NGRID), I2
9120 FORMAT (' NGRID, IDIM, JDIM, KDIM, I2 ', 515)
WRITE (19, 9010, ERR=450)
   1 (XBUF(I), I=1, I2),
   2 (YBUF(I), I=1, I2),
   3 (ZBUF(I), I=1, I2)
INOD2 = INOD
JNOD2 = JNOD
KNOD2 = KNOD
MARCH2 = MARCH
I2 = 0
END IF
II = I2 + 1
I2 = I2 + NSTORE
READ (NU20, 9010, ERR=400) (XBUF(I), I=II, I2)
READ (NU20, 9010, ERR=400) (YBUF(I), I=II, I2)
READ (NU20, 9010, ERR=400) (ZBUF(I), I=II, I2)
GOTO 100
200 CONTINUE
WRITE (NU6, 9115)
9115 FORMAT (' END OF FILE REACHED ON FILE 20')
NGRID = NGRID + 1
IF (MARCH2.EQ.1) THEN
   ID(NGRID) = KNOD2
   JD(NGRID) = JNOD2
   KD(NGRID) = INOD2
END IF
IF (MARCH2.EQ.2) THEN
   ID(NGRID) = INOD2
   JD(NGRID) = KNOD2
   KD(NGRID) = JNOD2
END IF
IF (MARCH2.EQ.3) THEN
   ID(NGRID) = JNOD2
   JD(NGRID) = INOD2
   KD(NGRID) = KNOD2
END IF
ID(NGRID) = INOD2
JD(NGRID) = JNOD2
KD(NGRID) = KNOD2
WRITE(NU6,9120) NGRID, ID(NGRID), JD(NGRID), KD(NGRID), I2
WRITE(NU6,8000) INOD2, JNOD2, KNOD2
8000 FORMAT(' INOD2, JNOD2, KNOD2 ',/S15)

WRITE(19,9010,ERR=450)
1 (XBUF(I), I=1,I2),
2 (YBUF(I), I=1,I2),
3 (ZBUF(I), I=1,I2)

WRITE(NU6,8100)
8100 FORMAT(' GRID WRITTEN TO 19 FILE')

REWIND(19)

IF(NGRID.GT.1) THEN
WRITE(NU6,8300) NGRID
8300 FORMAT(' NGRID ',/I5)
WRITE(NU21,ERR=475) NGRID
END IF

WRITE(NU6,8350)(ID(N), JD(N), KD(N), N=1, NGRID)
8350 FORMAT(' IDIM,JDIM,KDIM ',/S15)
WRITE(NU21,ERR=475)(ID(N), JD(N), KD(N), N=1, NGRID)

DO 300 N=1, NGRID
  I2 = ID(N)*JD(N)*KD(N)
READ(19,9010,ERR=450)
1 (XBUF(I), I=1,I2),
2 (YBUF(I), I=1,I2),
3 (ZBUF(I), I=1,I2)
WRITE(NU21,ERR=475)
1 (XBUF(I), I=1,I2),
2 (YBUF(I), I=1,I2),
3 (ZBUF(I), I=1,I2)
WRITE(NU6,9130) N
9130 FORMAT(' GRID ',/I5,' WRITTEN TO XYZHGM.DAT')

300 CONTINUE

WRITE(NU6,9150) NGRID
9150 FORMAT(' THERE ARE ',/I5,' GRIDS WRITTEN TO XYZHGM.DAT')
GOTO 500

400 WRITE(NU6,9140)
9140 FORMAT('ERROR TRYING TO READ THE 20 FILE')
GOTO 500

450 WRITE(NU6,9160)
9160 FORMAT('ERROR TRYING TO READ FILE 19')
GOTO 500

475 WRITE(NU6,9170)
9170 FORMAT('ERROR TRYING TO WRITE TO 20 FILE')
SUBROUTINE PROJECT(POINT,RANGE,ANGLE)
C
C
C
C
C
C
C

POINT = POSITION ON BOWL SURFACE
RANGE = PROJECTION ONTO X AXIS FROM EDGE OF BOWL
ANGLE = SWEEP ANGLE FROM VERTICAL

C

DIMENSION POINT(3),BOWL(3),AXIS(3),VECTOR(3),SWEEP(3),VDUM(3)

DATA BOWL /6.07,0.0,0.0/
DATA AXIS /1.0,0.0,0.0/

C
C---DISTANCE ALONG AXIS FROM EDGE OF BOWL
C
ONE = 1.0
CALL VADD(ONE,POINT,-ONE,BOWL,VECTOR,VDUM)
CALL VDOT(VECTOR,AXIS,RANGE)

C
C---ANGLE FROM VERTICAL
C
CALL VADD(ONE,VECTOR,-RANGE,AXIS,VDUM,SWEEP)
ANGLE = ACOS(SWEEP(2))

RETURN
END

SUBROUTINE CROSS(A,B,C,N)
C
C
COMMON /COUNTER/ NODESAV,NODETOT,NBNODES,NPLANE
COMMON /HEADER/ ITITLE(20),LINE
COMMON /INITA/ IDIM,MAPTEN,INCHES
COMMON /IWRITE/ IWRITC,IWRITI,IWRITN
COMMON /UNITS/ NU5,NU6,NU19,NU20,NU21
COMMON /ZONING/ ISECT,NSECT,IZONE,NMBRND(3)

DIMENSION A(3),B(3),C(3)

C-47

LOCKHEED-HUNTSVILLE ENGINEERING CENTER
C---CROSS PRODUCT
C
C(1) = A(2)*B(3) - A(3)*B(2)
C(2) = A(3)*B(1) - A(1)*B(3)
C(3) = A(1)*B(2) - A(2)*B(1)
C
C---MAGNITUDE
C
CALL VMAG(C,CMAG)
C
IF(CMAG.GT.0.0) THEN
C
C---NORMALIZE
C
C(1) = C(1)/CMAG
C(2) = C(2)/CMAG
C(3) = C(3)/CMAG
C
ELSE
C(1) = 0.0
C(2) = 0.0
C(3) = 0.0
C
ENDIF
C
C IF(IWRTI.EQ.0) RETURN
C
C-------------------------------------------------------------------------
C ERROR PRINT (CROSS PRODUCT EQUALS ZERO)
C
C N = 10 -> 21 SUBROUTINE EDGMAP1
C 30 -> 32 " EDGMAP2
C 40 -> 45 " SURFACE
C 50 -> 56 " BC
C 60 -> 63 " HOLES
C 70 -> 73 " FAIRING
C
C-------------------------------------------------------------------------
C NODNUM = NODESAV + NODETOT
C
IF(N.LT.50 .OR. N.GT.59) NODNUM = NODNUM + 1
C
LINE = LINE + 1
C
IF(LINE.GE.60) THEN
C
WRITE(NU6,1000) ITITLE,ISECT,NSECT,IZONE
C
LINE = 2
C
END IF
C
WRITE(NU6,1010) N,NODNUM
C
END IF
C
RETURN
C
C---FORMAT STATEMENTS
C
1000 FORMAT(1H1,10X,20A4,13X,8H SECTION,I2,3H OF,I3,9H FOR ZONE,I3)
1010 FORMAT(9H LOCATION,I3,36H: CROSS PRODUCT EQUALS ZERO FOR NODE,I6)
C
END
C
C**************************************************************************
SUBROUTINE ETABC(MARCH, INDEX, NODE)

C** GRID SPACING

C**-----------------------------------------------

C** SUBROUTINE ETABC(MARCH, INDEX, NODE)

C** THIS ROUTINE CALCULATES THE VALUE OF ETA.

C**-----------------------------------------------

COMMON /INITA/ IDIM, MAPPED, INCHES
COMMON /INIC/ PI, RADDG
COMMON /INPUTA/ EDGE(6, 12), POINT(6, 8), SIDE(6, 6)
COMMON /INPUTBC/ ISIDE(3)
COMMON /SPACING/ ISTRITCH(3), STRETCH(3), ETAS(3, 200), ETA(3), DETA(3)
COMMON /ZONING/ ISECT, NSECT, IZONE, NMBRND(3)

C** FIRST NODE

C** IF(NODE.EQ.1) THEN

C ETA(INDEX) = 0.0

C** DETERMINE SIDE

C ISIDE(INDEX) = 10 + (INDEX - 6)*INDEX

C** STORE SPACING

C ETAS(INDEX,1) = 0.0

C RETURN

C END IF

C** LAST NODE

C** IF(NODE.EQ.NMBRND(INDEX)) THEN

C ETA(INDEX) = 1.0

C** DETERMINE SIDE

C ISIDE(INDEX) = 9 + (INDEX - 7)*INDEX/2

C** STORE SPACING

C ETAS(INDEX,NODE) = 1.0

C RETURN

C END IF

C** INTERIOR NODES

C** IF(ISECT.GT.1 .AND. INDEX.NE.MARCH) GO TO 310

C** CALCULATE ETA

C-49

LOCKHEED-HUNTSVILLE ENGINEERING CENTER
ISTR = ISTRCH(INDEX) + 1
C
GO TO (100,110,120,130,140,150,160,170,180,190,200) ISTR
C
---EQUAL SPACING---------------------------------------------(3)
C
100 ETA(INDEX) = ETA(INDEX) + DETA(INDEX)
C
GO TO 300
C
---INPUT ETA SPACING------------------------------------------(1)
C
110 ETA(INDEX) = ETAS(INDEX,NODE)
C
GO TO 300
C
---DECREASING SPACING; INPUT STRETCHING FACTOR-----------------(2)
C
120 RATIO = REAL(NODE - 1)/REAL(NMBRNDs(INDEX) - 1)
C
X1 = RATIO*STRETCH(INDEX)/2.0
ETA1 = (EXP(X1) - EXP(-X1))/(EXP(X1) + EXP(-X1))
C
X2 = STRETCH(INDEX)/2.0
ETA2 = (EXP(X2) - EXP(-X2))/(EXP(X2) + EXP(-X2))
C
ETA(INDEX) = ETA1/ETA2
GO TO 300
C
---INCREASING SPACING; INPUT STRETCHING FACTOR-----------------(3)
C
130 RATIO = REAL(NMBRNDs(INDEX) - NODE)/REAL(NMBRNDs(INDEX) - 1)
C
X1 = RATIO*STRETCH(INDEX)/2.0
ETA1 = (EXP(X1) - EXP(-X1))/(EXP(X1) + EXP(-X1))
C
X2 = STRETCH(INDEX)/2.0
ETA2 = (EXP(X2) - EXP(-X2))/(EXP(X2) + EXP(-X2))
C
ETA(INDEX) = 1.0 - ETA1/ETA2
GO TO 300
C
---DOUBLE STRETCHING; INPUT STRETCHING FACTOR-----------------(4)
C
140 X1 = (STRETCH(INDEX)/2.0)/2.0
ETAMID = (EXP(X1) - EXP(-X1))/(EXP(X1) + EXP(-X1))
C
RATIO = REAL(NODE - 1)/REAL(NMBRNDs(INDEX) - 1)
C
X2 = (RATIO - 0.5)*STRETCH(INDEX)/2.0
ETAMX = (EXP(X2) - EXP(-X2))/(EXP(X2) + EXP(-X2))
C
ETA(INDEX) = (ETAMX + ETAMID)/(2.0*ETAMID)
GO TO 300
C
---DECREASING SPACING; INPUT MINIMUM SPACING-------------------(5)
150  ARGI = STRETCH(INDEX)*REAL(NODE - 1)
    EXPI = EXP(ARGI)
    EXPII = 1.0/EXPI
    TANHI = (EXPI - EXPII)/(EXPI + EXPII)

C

    ARGN = STRETCH(INDEX)*REAL(NMBRNDS(INDEX) - 1)
    EXPN = EXP(ARGN)
    EXPNI = 1.0/EXPN
    TANHN = (EXPN - EXPNI)/(EXPN + EXPNI)

C

    ETA(INDEX) = TANHI/TANHN
    GO TO 300

C---INCREASING SPACING; INPUT MINIMUM SPACING--------------------------(6)

C

160  ARGI = STRETCH(INDEX)*REAL(NMBRNDS(INDEX) - NODE)
    EXPI = EXP(ARGI)
    EXPII = 1.0/EXPI
    TANHI = (EXPI - EXPII)/(EXPI + EXPII)

C

    ARGN = STRETCH(INDEX)*REAL(NMBRNDS(INDEX) - 1)
    EXPN = EXP(ARGN)
    EXPNI = 1.0/EXPN
    TANHN = (EXPN - EXPNI)/(EXPN + EXPNI)

C

    ETA(INDEX) = 1.0 - TANHI/TANHN
    GO TO 300

C---DOUBLE STRETCHING; INPUT MINIMUM SPACING--------------------------(7)

C

170  ARGI = STRETCH(INDEX)*REAL(2*NODE - NMBRNDS(INDEX) - 1)
    EXPI = EXP(ARGI)
    EXPII = 1.0/EXPI
    TANHI = (EXPI - EXPII)/(EXPI + EXPII)

C

    ARGN = STRETCH(INDEX)*REAL(NMBRNDS(INDEX) - 1)
    EXPN = EXP(ARGN)
    EXPNI = 1.0/EXPN
    TANHN = (EXPN - EXPNI)/(EXPN + EXPNI)

C

    ETA(INDEX) = 0.5*(1.0 + TANHI/TANHN)
    GO TO 300

C---DECREASING SPACING; INPUT STRETCHING FACTOR-----------------------(8)

C

180  PIDN = PI/(STRETCH(INDEX)*NMBRNDS(INDEX))

C

    ETA(INDEX) = 1.0 - TAN(PIDN*(NMBRNDS(INDEX) - NODE))
    &/TAN(PIDN*(NMBRNDS(INDEX) - 1))

C

    GO TO 300

C---INCREASING SPACING; INPUT STRETCHING FACTOR-----------------------(9)

C

190  PIDN = PI/(STRETCH(INDEX)*NMBRNDS(INDEX))

C

    ETA(INDEX) = TAN(PIDN*( NODE - 1))

C-51

LOCKHEED-HUNTSVILLE ENGINEERING CENTER
& /TAN(PIDN*(NMBRNDN(INDEX) - 1))
C
GO TO 300
C
C USER INPUT STRETCHING FUNCTION--------------------------(10)
C
200 CONTINUE
C
C STORE SPACING------------------------------------------
C
300 ETA(INDEX,NODE) = ETA(INDEX)
C
310 ETA(INDEX) = ETA(INDEX,NODE)
C
RETURN
END
C
******************************************************************************
C
******************************************************************************
C
SUBROUTINE HOLES(ANGLE,POINT,TANGENT,ICLOCK,IDUCT)

C
CALCULATE POINT AND TANGENT ON HOLE
C
ANGLE = ANGULAR LOCATION OF POINT
POINT = POSITION AND FLOW VECTOR
TANGENT = TANGENT
ICLOCK = DIRECTION OF TANGENT
C
COMMON /INITC/ PI,RADDEG
C
DIMENSION POINT(6),TANGENT(3),US(3),BC(3),REF(3)
C
ANG = ANGLE/RADDEG
C
SANG = SIN(ANG)
CANG = COS(ANG)
C
IF(IDUCT.EQ.1) THEN
C
UPPER DUCT
C
UNIT VECTOR ALONG AXIS
C
A1 = 0.25506
A3 = 0.092215
A5 = 0.96252
C
INTERMEDIATE VECTOR
C
A2 = 2.07828*CANG - 0.050785*SANG
A4 = 5.564 + 2.14084*SANG
A6 = -0.55072*CANG - 0.19165*SANG
C
REFERENCE POINT ON AXIS
C
REF(1) = 9.445
REF(2) = 5.564
REF(3) = 0.0
ELSE
C--LOWER DUCT
C--UNIT VECTOR ALONG AXIS
C   A1 = 0.40139
   A3 = 0.0
   A5 = 0.91591
C--INTERMEDIATE VECTOR
C   A2 = 1.96921*CANG - 2.11734
   A4 = 2.150*SANG
   A6 = 0.37334 - 0.86299*CANG
C--REFERENCE POINT ON AXIS
C   REF(1) = 7.32766
   REF(2) = 0.0
   REF(3) = 0.37334
C END IF
C--POSITION
C   RAD = 7.890
C--QUADRATIC EQUATION
C   AA = A1*A1 + A3*A3 + A5*A5
C   BB = 2.0*(A1*A2 + A3*A4 + A5*A6)
C   CC = A2*A2 + A4*A4 + A6*A6 - RAD**2
C   ARC = BB*BB - 4.0*AA*CC
C--DISTANCE ALONG AXIS
C   C = 0.5*(-BB + SQRT(ARG))/AA
C--VECTOR FROM SPHERE ORIGIN TO POINT ON HOLE
C   US(1) = C*A1 + A2
   US(2) = C*A3 + A4
   US(3) = C*A5 + A6
C--VECTOR FROM ORIGIN TO POINT ON HOLE
C   POINT(1) = US(1) + 9.445
   POINT(2) = US(2)
   POINT(3) = US(3)
C--TANGENT
C--VECTOR FROM AXIS TO POINT ON HOLE
C   BC(1) = POINT(1) - C*A1 - REF(1)
\[
BC(2) = POINT(2) - C*A3 - REF(2)
\]
\[
BC(3) = POINT(3) - C*A5 - REF(3)
\]

\begin{verbatim}
C-----TANGENT AND FLOW VECTOR
C IF(ICLOCK.EQ.0) THEN
C CALL CROSS(US,BC,TANGENT,60)
C CALL CROSS(US,TANGENT,POINT(4),61)
C ELSE
C CALL CROSS(BC,US,TANGENT,62)
C CALL CROSS(TANGENT,US,POINT(4),63)
C END IF

RETURN
END
C*******************************************************************************
C*****OUTPUT*****************************************************************************
C SUBROUTINE OUTPUT(NUNIT,NODSTOR)
C PRINTOUT AND STORE DATA
C DOUBLE PRECISION NODE
REAL NODE
C COMMON /COUNTER/ NODESAV,NODETOT,NBNODES,NPLANE
COMMON /HEADER/ ITITLE(20),LINE
COMMON /INITA/ IDIM,MAPTEN,INCHES
COMMON /IWRITE/ IWRTC,IWRTI,IWRTN
COMMON /OUT/
COMMON /UNITS/
COMMON /ZONING/

NODE(5,10000)
NU5,NU6,NU19,NU20,NU21
ISECT,NSECT,IZONE,NMBRNDS(3)

C-----TOTAL NUMBER OF PLANES
C NPLANE = NPLANE + 1
C
C----PRINT OUTPUT
C----INTERMEDIATE PRINT
C 100 IF(IWRTI.GT.0 .AND. LINE.LE.56) THEN
C WRITE(NU6,1030)
C LINE = LINE + 3
C END IF

C-----PRINT NODAL INFORMATION
C  IPRINT = 1
\end{verbatim}
DO 120 I=1,NODSTOR

IF(IWRTN.EQ.0 .OR. I.LT.IPRINT) GO TO 120

110 IPRINT = I + IWRTN
NODNUM = I + NODETOT

LINE = LINE + 1

IF(LINE.GE.60) THEN

WRITE(NU6,1000) ITITLE,ISECT,NSECT,IZONE
WRITE(NU6,1030)

LINE = 5
END IF

WRITE(NU6,1040) NODNUM,(NODE(J,I),J=1,5)

120 CONTINUE

C STORE OUTPUT

C

C 200 CALL BLKOUT(NUNIT,NODSTOR)

C

C--NODE COUNTERS

C

NODETOT = NODSTOR + NODETOT
NODESAV = NODESAV - NODSTOR

C

C--PRINT TOTAL NUMBER OF POINTS STORED

C

IF(IWRTI.EQ.0) RETURN

C

LINE = LINE + 1

C

WRITE(NU6,1060) NODSTOR,NPLANE,NUNIT,NODETOT

C RETURN

C--FORMAT STATEMENTS

C

1000 FORMAT(1H1,10X,20A4,13X,8H SECTION,I2,3H OF,I3,9H FOR ZONE,I3)
1020 FORMAT(1X,I6,2(1X,F13.7),5X,F7.2,6X,F7.2,5X,I2)
1030 FORMAT( / 44H NODE X Y Z )
1040 FORMAT(1X,I6,3(1X,F13.7),3X,2(2X,F7.2),3X,2(2X,F7.2),5X,I2)
1050 FORMAT( 5X,I5,18H POINTS FROM PLANE,I4,15H STORED ON UNIT,I3,
& 23H: TOTAL POINTS STORED =,I6 )
1060 FORMAT(10X,I5,18H POINTS FROM PLANE,I4,15H STORED ON UNIT,I3,
& 23H: TOTAL POINTS STORED =,I6 )

C END

C******************************************************************************

C******************************************************************************

C-54

LOCKHEED-HUNTSVILLE ENGINEERING CENTER
SUBROUTINE PICTURE(IDRAW)

THIS ROUTINE DESCRIBES THE NOMENCLATURE

COMMON /UNITS/ NU5,NU6,NU19,NU20,NU21

WRITE(NU6,300)
WRITE(NU6,310)
WRITE(NU6,320)
WRITE(NU6,330)
RETURN

C FORMAT STATEMENTS

300 FORMAT( 40X,37H IE 3D NOMENCLATURE 37H)
  1 /40X,19H IE 7
  2 /40X,19H IT 1
  3 /40X,34H IA SURFACE 4
  4 /40X,32H I2 (TOP)
  5 /40X,50H 8/I EDGE 3
  6 /40X,42H 7/I O
  7 /40X,42H 8/I EDGE 3
  8 /40X,50H E/ I S)

310 FORMAT( 40X,50H G/ I SURFACE 1 50H)
  0-----DI-------------------O POINT 7 G A
  1 /40X,50H G/ I (BACK) 50H
  2 /40X,50H D/ I E R
  3 /40X,50H E/ I D F
  4 /40X,50H E/ I S)
  5 /40X,50H I GI EDGE 11
  6 /40X,50H I IE I E C
  7 /40X,50H I ID I 2
  8 /40X,50H I IG I 6
  9 /40X,50H I IE I O ETAL1)

320 FORMAT( 40X,51H C POINT 1 /5 EDGE 1 51H)
  1 /40X,42H E 1 I /E 51H
  2 /40X,41H 2 I /G I /E
  3 /40X,40H 5 I /D I /G
  4 /40X,39H 4 I /E I /D
  5 /40X,38H 0/ 51H
  6 /40X,44H POINT 5 EDGE 9 44H
  7 /40X,10H /
  8 /40X,26H /
  9 /40X,25H /A (BOTTOM)

330 FORMAT( 40X,8H /T 330)
  6 /40X,7H /E 8)

END
VADD computes the sum of vectors $c = a \cdot c_a + b \cdot c_b$, where $c_a$ and $c_b$ are scalars. $u_c$ is a unit vector directed along $c$.

```c
DIMENSION A(3),B(3),C(3),UC(3)

SUM = 0.0
DO 10 I=1,3
  C(I) = C A(I) + C B(I)
10  SUM = SUM + C(I)*C(I)
CMAG = SQRT(SUM)
RMAG = 0.0
IF(CMAG.GT.0.0) RMAG = 1.0/CMAG
UC(1) = C(1)*RMAG
UC(2) = C(2)*RMAG
UC(3) = C(3)*RMAG
RETURN
END
```

VDOT computes the dot product of vectors $a$ and $b$.

```c
DIMENSION A(3),B(3)

C = 0.0
DO 10 I=1,3
  C = C + A(I)*B(I)
10  RETURN
END
```

VMAG determines the magnitude of a vector.

```c
DIMENSION VECTOR(3)

VECMAG = SQRT(VECTOR(1)**2 + VECTOR(2)**2 + VECTOR(3)**2)
IF(VECMAG.LT.0.0000001) VECMAG = 0.0
RETURN
```
SUBROUTINE FAIRING(PT1,PT2,POINT,TANGENT,SNORMAL)

TRANSITION FROM BOWL TO DUCT

PT1 = POINT ON HOLE (ON SPHERE)
PT2 = POINT ON DUCT (END OF FAIRING)
POINT = POSITION
TANGENT = TANGENT
SNORMAL = SURFACE NORMAL

COMMON /COUNTER/ NODESAV,NODETOT,NBNODES,NPLANE
COMMON /INITC/ PI,RADDEG
COMMON /SPACING/ ISTRTCH(3),STRETCH(3),ETAS(3,200),ETA(3),DETA(3)
COMMON /ZONING/ ISECT,NSECT,IZONE,NMBRNDS(3)

COMMON /20NING/ ISECT,NSECT,IZONE,NMBRNDS(3)
DIMENSION PT1(3),PT2(3),POINT(6),TANGENT(3),SNORMAL(3)
DIMENSION PTW(3),UC(3),CP(3),REF(3),VDUM(3)
DIMENSION R1(3),R2(3),UN(3),UP(3)

NODNUM = NODESAV + NODETOT + 1

VECTOR FROM SPHERE ORIGIN TO EDGE OF HOLE

PI = PT1(1) - 9.445
P2 = PT1(2)
P3 = PT1(3)

PMAG = SQRT(P1*P1 + P2*P2 + P3*P3)

---NORMALIZED

S1 = P1/PMAG
S2 = P2/PMAG
S3 = P3/PMAG

IDUCT = IZONE-3

WRITE(6,3030) IAXIS,JAXIS,KAXIS
3030 FORMAT(' IAXIS(2),JAXIS(3),KAXIS(1) = ',3110)

IF(IDUCT.EQ.1) THEN

UPPER DUCT

---AXIS

U1 = 0.25506
U2 = 0.092215
U3 = 0.96252

---INTERMEDIATE VALUES

CADD = 5.564*U2
C
A1 = S1*(1.0 - U1**2) - S2*U1*U2 - S3*U1*U3
A2 = S2*(1.0 - U2**2) - S1*U1*U2 - S3*U2*U3
A3 = S3*(1.0 - U3**2) - S1*U1*U3 - S2*U2*U3
C
B1 = 5.564*U1*U2
B2 = 5.564*(U2*U2 - 1.0)
B3 = 5.564*U2*U3
C
---REFERENCE POINT ON AXIS
C
REF(1) = 9.445
REF(2) = 5.564
REF(3) = 0.0
C
WRITE(6,3010)
C 3010 FORMAT(' UPPER DUCT')
C
ELSE

---LOWER DUCT
-------AXIS
C
U1 = 0.40139
U2 = 0.0
U3 = 0.91591
C
---INTERMEDIATE VALUES
C
CADD = -2.11734*U1 + 0.37334*U3
C
A1 = S1*(1.0 - U1*U1) - S3*U1*U3
A2 = S2
A3 = S3*(1.0 - U3*U3) - S1*U1*U3
C
B1 = -2.11734*(U1*U1 - 1.0) + 0.37334*U1*U3
B2 = 0.0
B3 = -2.11734*U1*U3 + 0.37334*(U3*U3 - 1.0)
C
---REFERENCE POINT ON AXIS
C
REF(1) = 7.32766
REF(2) = 0.0
REF(3) = 0.37334
C
WRITE(6,3020)
C 3020 FORMAT(' LOWER DUCT')
C
END IF

---CALCULATE END POINT OF FAIRING
C
-------QUADRATIC EQUATION
C
AA = A1*A1 + A2*A2 + A3*A3 - 1.0
C
BB = 2.0*(A1*B1 + A2*B2 + A3*B3 + 5.930)
C
C
DD = BB*BB - 4.0*AA*CC
C
SDD = SQRT(DD)
C
RH01 = 0.5*(-BB + SDD)/AA
RH02 = 0.5*(-BB - SDD)/AA
C
---RADIUS FROM SPHERE ORIGIN TO ARC CENTER
C
RHO = RH01
IF(RHO2.GT.RH01) RHO = RH02
C
---ARC RADIUS
C
RAD = RHO - 7.890
C
---RADIUS FROM DUCT CENTERLINE TO ARC CENTER
C
CN = 1.96 + RAD
C
---UNIT VECTOR FROM DUCT CENTERLINE TO ARC CENTER
C
C1 = (RHO*A1 + B1)/CN
C2 = (RHO*A2 + B2)/CN
C3 = (RHO*A3 + B3)/CN
C
---SWEEP ANGLE OF ARC
C
SN = C1*S1 + C2*S2 + C3*S3
C
ANG = ACOS(SN)
C
---VECTOR FROM ORIGIN TO ARC CENTER
C
DC1 = RHO*S1 + 9.445
DC2 = RHO*S2
DC3 = RHO*S3
C
---DISTANCE ALONG DUCT CENTERLINE
C
CA = RHO*(S1*U1 + S2*U2 + S3*U3) - CADD
C
---VECTOR FROM ORIGIN TO END POINT OF FAIRING
C
PTW(1) = CA*U1 + 1.96*C1 + REF(1)
PTW(2) = CA*U2 + 1.96*C2 + REF(2)
PTW(3) = CA*U3 + 1.96*C3 + REF(3)
C
ANGL = ANG*RADDEG
C
WRITE(6,3000) NODNUM,RHO,RAD,CA,ANGL,C1,C2,C3,(PTW(I),I=1,3)
C 3000 FORMAT(' NODE =',110 / ' RHO,RAD,CA,ANGL =',4(2X,F12.5)
C 1 / ' C1,C2,C3 =',3(2X,F12.5)
C 2 / ' PTW =',3(2X,F12.5))
C
JUNCTION = 7
C
IF(UAXIS.LT.JUNCTION) THEN

FAIRING SURFACE

C-59

LOCKHEED-HUNTSVILLE ENGINEERING CENTER
VECTOR FROM ARC CENTER TO HOLE

\[ R_1(1) = PT1(1) - DC1 \]
\[ R_1(2) = PT1(2) - DC2 \]
\[ R_1(3) = PT1(3) - DC3 \]

VECTOR FROM ARC CENTER TO END OF FAIRING

\[ R_2(1) = PTW(1) - DC1 \]
\[ R_2(2) = PTW(2) - DC2 \]
\[ R_2(3) = PTW(3) - DC3 \]

LOCAL COORDINATE SYSTEM

CALL CROSS(\( R_1 \), \( R_2 \), \( UN \), 70)

CALL CROSS(\( R_1 \), \( UN \), \( UP \), 71)

CALL CROSS(\( UN \), \( UP \), \( R_1 \), 72)

CALL VDOT(\( UP \), \( R_2 \), \( UPE \))

IF(\( UPE \).LT.0.0) CALL CROSS(\( UN \), \( R_1 \), \( UP \), 73)

ARC ANGLE ALONG FAIRING

\[ \text{THET} = \text{ANG} \times (\text{FLOAT(IAXIS)} - 1.0)/3.0 \]

\[ \text{CANG} = \cos(\text{THET}) \]
\[ \text{SANG} = \sin(\text{THET}) \]

UNIT VECTOR FROM ARC CENTER TO POINT ON FAIRING

CALL VADD(\( \text{CANG} \), \( R_1 \), \( \text{SANG} \), \( UP \), \( VDUM \), \( UC \))

POSITION

\[ \text{POINT}(1) = DC1 + \text{RAD} \times \text{UC}(1) \]
\[ \text{POINT}(2) = DC2 + \text{RAD} \times \text{UC}(2) \]
\[ \text{POINT}(3) = DC3 + \text{RAD} \times \text{UC}(3) \]

WRITE(6,3050) \( \text{PT1,PT2,RHO1,RHO2,POINT} \)

3050 FORMAT(1X,6F12.5)

TANGENT

CALL VADD(\( \text{CANG} \), \( UP \), \( \text{-SANG} \), \( R_1 \), \( \text{VDUM} \), \( \text{TANGENT} \))

\[ \text{CE} = \text{TANGENT}(1) \times \text{U1} + \text{TANGENT}(2) \times \text{U2} + \text{TANGENT}(3) \times \text{U3} \]

IF(\( \text{CE} \).LT.0.0) THEN

\[ \text{TANGENT}(1) = -\text{TANGENT}(1) \]
\[ \text{TANGENT}(2) = -\text{TANGENT}(2) \]
\[ \text{TANGENT}(3) = -\text{TANGENT}(3) \]

END IF

C-60

LOCKHEED-HUNTSVILLE ENGINEERING CENTER
SNORMAL(1) = -UC(1)
SNORMAL(2) = -UC(2)
SNORMAL(3) = -UC(3)

C----ELSE

C DUCT SURFACE

C----RATIO BETWEEN END OF FAIRING AND END OF SECTION

C DUCT SURFACE

C----RATIO BETWEEN END OF FAIRING AND END OF SECTION

TOTND = FLOAT(NMBRND(2))
RATIO = (TOTND - IAXIS) / (TOTND - JUNCION)

STR21 = 3.0

XI = RATIO*STR21/2.0
ETA1 = (EXP(X1) - EXP(-X1))/(EXP(X1) + EXP(-X1))

X2 = STR21/2.0
ETA2 = (EXP(X2) - EXP(-X2))/(EXP(X2) + EXP(-X2))

EPS = 1.0 - ETA1/ETA2

EPS1 = 1.0 - EPS

C----VECTOR FROM ORIGIN TO POINT ON DUCT

CALL VADD(EPS1,PTW,EPS,PT2,CP,VDUM)

C----POSITION

POINT(1) = CP(1)
POINT(2) = CP(2)
POINT(3) = CP(3)

C----TANGENT

TANGENT(1) = U1
TANGENT(2) = U2
TANGENT(3) = U3

C----NORMAL

SNORMAL(1) = C1
SNORMAL(2) = C2
SNORMAL(3) = C3

END IF

C----FLOW VECTOR

POINT(4) = TANGENT(1)
POINT(5) = TANGENT(2)
POINT(6) = TANGENT(3)
RETURN
END
C
C**BOR**
C**EOI**