FINAL REPORT

DUCT FLOW NONUNIFORMITIES FOR SPACE SHUTTLE MAIN ENGINE (SSME)

30 June 1988

Contract NAS8-34507

(NASA-CR-183595) DUCT FLOW NONUNIFORMITIES FOR SPACE SHUTTLE MAIN ENGINE (SSME) Final Report (Lockheed Missiles and Space Co.)

Prepared for:

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 NAS8-34507 for NASA-Marshall Space Flight Center.

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

Section                          Page

1 INTRODUCTION                   1-1

2 TECHNICAL APPROACH            2-1

   2.1 General                    2-1
   2.2 Elliptical Two-Duct Manifold Code  2-2
   2.3 Strut in Annulus          2-7

3 RESULTS                       3-1

   3.1 Manifold Grid Code        3-1
      3.1.1 Geometry and Grid    3-1
      3.1.2 Computer Code       3-10
      3.1.3 Code Implementation 3-13
   3.2 Strut Laminar Computation 3-14

4 CONCLUDING REMARKS            4-1

5 REFERENCES                    5-1

Appendixes

A HGM Grid Code Input Guide     A-1
B HGM Grid Code Input Listing   B-1
C HGM Grid Code Listing         C-1
D HGM Output Rewrite Listing    D-1

LIST OF FIGURES

Figure                          Page

2-1 Hexahedral Element Showing Local Intrinsic Coordinates 2-2
3-1 Schematic Showing Outer Wall of HGM Geometry that has been Modeled 3-1
3-2 Schematic Showing First Two Zones (Zone 1 Left; Zone 2, Right) into Which the Two-Duct HGM was Subdivided 3-3
3-3 Schematic Showing Zones 3, Top, and 4, Middle, and 5 of the Two-Duct HGM Model 3-4

iii

LOCKHEED-HUNTSVILLE ENGINEERING CENTER
<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>3-4</td>
<td>Cartesian Coordinate System Relative to Which Positions of All Nodes are Referred</td>
<td>3-5</td>
</tr>
<tr>
<td>3-5</td>
<td>Grid Plots for Inner and Outer Surfaces of Zones 1 and 2 as Well as Internal Grid at the Common Place of Intersection</td>
<td>3-6</td>
</tr>
<tr>
<td>3-6</td>
<td>Grid Plots of TAD Internal Grid (Bottom) and Grid Distribution of Inner and Outer Surfaces</td>
<td>3-7</td>
</tr>
<tr>
<td>3-7</td>
<td>Surface and Cross-Section Node Distribution for Transfer Duct</td>
<td>3-8</td>
</tr>
<tr>
<td>3-8</td>
<td>Grid Plot Showing Node Distribution on Outer Wall of Zones 1, 2, and 3 Displaying Continuity of Grid Lines from One Zone to Another</td>
<td>3-9</td>
</tr>
<tr>
<td>3-9</td>
<td>Primary Calling Sequence for Two-Duct HGM Grid Code</td>
<td>3-11</td>
</tr>
<tr>
<td>3-10</td>
<td>Secondary Calling Sequences for Two-Duct HGM Grid Code</td>
<td>3-12</td>
</tr>
<tr>
<td>3-11</td>
<td>Grid Plots Showing C-Grid Used in each of the 31 Circumferential Planes (Top) and Partial Surface Grids of Strut and Inner Outer Wall</td>
<td>3-15</td>
</tr>
<tr>
<td>3-12</td>
<td>Static Pressure Contours (Top) and Velocity Magnitude Contours for Strut in Annular Channel near Inner Wall</td>
<td>3-17</td>
</tr>
<tr>
<td>3-13</td>
<td>Static Pressure Contours (Top) and Velocity Magnitude Contours for Strut in Annular Channel at Mid-Channel</td>
<td>3-18</td>
</tr>
<tr>
<td>3-14</td>
<td>Static Pressure Contours (Top) and Velocity Magnitude Contours for Strut in Annular Channel near Upper Wall</td>
<td>3-19</td>
</tr>
<tr>
<td>3-15</td>
<td>Static Pressure Contours (Top) and Velocity Magnitude Contours for Entire Length of Computational Circumferential Plane at Mid-Channel</td>
<td>3-20</td>
</tr>
<tr>
<td>3-16</td>
<td>Velocity Vectors in near Wake Region in a Central Plane for Case of Strut in Annular Channel (Mean Flow is from Left to Right)</td>
<td>3-21</td>
</tr>
<tr>
<td>3-17</td>
<td>Velocity Vectors (Top) and Velocity Magnitude Contours in Central Radial Plane in near Wake Region Behind Strut in Annular Channel (Mean Flow is from Left to Right)</td>
<td>3-22</td>
</tr>
<tr>
<td>3-18</td>
<td>Particle Traces for Flow Past Strut in Annular Channel Showing Paths of Particles Released near Front of Strut in a Plane near the Inner Wall (Top), one Quarter Channel Height Above Inner Wall (Bottom Left) and Mid-Channel</td>
<td>3-23</td>
</tr>
<tr>
<td>3-19</td>
<td>Particle Traces of Particles Released near Rear Surface of Strut</td>
<td>3-24</td>
</tr>
<tr>
<td>3-20</td>
<td>Particle Traces of Particles Released near Central Radial Plane but just Downstream of Strut Rear Surface</td>
<td>3-25</td>
</tr>
<tr>
<td>3-21</td>
<td>Surface Pressure Contours on Strut Front Surface (Left, Flow into Page) and Side Surface (Right, Flow Left to Right)</td>
<td>3-26</td>
</tr>
</tbody>
</table>
1. INTRODUCTION

This report describes results of efforts by personnel of the Computational Mechanics Section 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 report specifically details work completed subsequent to the interim technical report submitted in August 1985 (Ref. 1). Computational and experimental results reported in Ref. 1 will not be reiterated here, and the reader is referred to that earlier document for those details.

Emphasis in this final report is placed on construction and documentation of a computational grid code for modeling an elliptical two-duct version of the Space Shuttle Main Engine (SSME) fuel side hot gas manifold (HGM). Also computational results for flow past a support strut in an annular channel. These three-dimensional results constitute the initial phase of a more detailed study of flow through the SSME/HGU strut region being completed by Lockheed under contract NAS8-37359 (Ref. 2).

The approach for both of the aforementioned tasks is presented in Section 2. Sample results are contained in Section 3. Included in Appendixes A, B, and C are a brief input guide for the two-duct HGM code, a listing of the code input file, and a source listing of the grid code itself.
2. TECHNICAL APPROACH

2.1 GENERAL

The two-duct HGM grid generation uses as its fundamental building block the flexible algebraic techniques coded by the Lockheed Computational Mechanics Section into its own in-house geometry code. These techniques employ vector algebra, analytical geometry and transfinite interpolation to perform the computations in local curvilinear coordinates and generate the grid of discrete points in Cartesian coordinates. With this as a foundation, special subroutines were constructed for describing particular surface shapes for this complicated structure such as the hold in the bowl and the fairing for the transfer duct. The combination of these codings was assembled, debugged, streamlined, and well commented for use by NASA-MSFC computational engineers.

Early computational fluid dynamics results, reported in the interim report for this contract (Ref. 1) were performed with a finite difference code employing an explicit solution algorithm. Steady state was obtained from a trial initialization by performing successive iterations in time until all transients have involved away. The size of the time step used in this procedure is a strong function of the density of points in the grid. The higher the density, the smaller the allowable time step. For this reason, relatively coarse grids were used, even for regions near the solid walls.

Experience has dictated that much larger nodal densities near the walls are desirable for more accurate computational predictions. This precludes using an explicit code because of the unnecessarily large number of time steps required to obtain a steady state solution. The implicit code INS3D, developed at NASA-Ames (Ref. 3), was then chosen for additional SSME related computations. The implicit solution algorithm incorporated into this code allows for
much larger time steps even for grids of large nodal densities. The results reported in Section 3.2 were obtained with INS3D. A brief description of this code, and how it was applied, is contained in Section 2.3.

2.2 ELLIPTICAL TWO-DUCT MANIFOLD 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, \((\eta_1, \eta_2, \eta_3)\). 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](image-url)
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 \((\eta_1, \eta_2, \eta_3)\). 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 \((\eta_1, \eta_2, \eta_3)\) 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 of multivariate blending function interpolation (Ref. 1). A brief description of this method follows.

Let \(F(\eta_1, \eta_2, \eta_3)\) be a vector-valued functional representing the region \(R\) of interest in curvilinear space. Then as \((\eta_1, \eta_2, \eta_3)\) range over \(R\), \(F\) traces out the region in Euclidean space \((x, y, z)\). Also let \(\phi, \psi, \) and \(\lambda\) be blending functions which obey the cardinality conditions:

\[
\phi_i(\eta_1=1) = \begin{cases} 1 & \text{if } i = 1 \\ 0 & \text{if } i \neq 1 \end{cases} \\
\psi_j(\eta_2=m) = \begin{cases} 1 & \text{if } j = m \\ 0 & \text{if } j \neq m \end{cases} \\
\lambda_k(\eta_3=n) = \begin{cases} 1 & \text{if } k = n \\ 0 & \text{if } k \neq n \end{cases}
\]

The simplest form of blending functions meeting these conditions are

\[
\phi_0(\eta_1) = 1 - \eta_1 \\
\psi_0(\eta_2) = 1 - \eta_2 \\
\lambda_0(\eta_3) = 1 - \eta_3
\]

Then a trilinearly blended interpolant of \(F\), which will map a unit cube onto the region \(R\) is given by

\[
\begin{align*}
\phi_1(\eta_1) &= \eta_1 \\
\phi_2(\eta_1) &= \eta_1 \\
\phi_3(\eta_1) &= \eta_1 \\
\psi_0(\eta_2) &= \eta_2 \\
\psi_1(\eta_2) &= \eta_2 \\
\psi_2(\eta_2) &= \eta_2 \\
\psi_3(\eta_2) &= \eta_2 \\
\lambda_0(\eta_3) &= \eta_3 \\
\lambda_1(\eta_3) &= \eta_3 \\
\lambda_2(\eta_3) &= \eta_3 \\
\lambda_3(\eta_3) &= \eta_3
\end{align*}
\]
\[ U(n_1, n_2, n_3) = \begin{bmatrix} x(n_1, n_2, n_3) \\ y(n_1, n_2, n_3) \\ z(n_1, n_2, n_3) \end{bmatrix} \]

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

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

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

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

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

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

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

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

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
and which can be rewritten as

\[ F(0, n_2, n_3) = \{ \frac{X}{2} \} \text{ on } \text{SIDE}_5, \text{ etc.} \]

\[ F(0, 0, n_3) = \{ \frac{Y}{2} \} \text{ along } \text{EDGE}_5, \text{ etc.} \]

\[ F(0, 0, 0) = \{ \frac{Z}{2} \} \text{ at } \text{POINT}_1, \text{ etc.} \]

\[
\begin{align*}
    U &= \{ \frac{X}{2} \} \\
    &= (1-n_1)\text{SIDE}_5 + n_1\text{SIDE}_6 + (1-n_2)\text{SIDE}_2 + n_2\text{SIDE}_4 + (1-n_3)\text{SIDE}_1 + n_3\text{SIDE}_3 \\
    &\quad - (1-n_1)(1-n_2)\text{EDGE}_5 - (1-n_1)n_2\text{EDGE}_8 - n_1(1-n_2)\text{EDGE}_6 - n_1n_2\text{EDGE}_7 \\
    &\quad - (1-n_1)(1-n_3)\text{EDGE}_4 - (1-n_1)n_3\text{EDGE}_{12} - n_1(1-n_3)\text{EDGE}_2 - n_1n_3\text{EDGE}_{10} \\
    &\quad - (1-n_1)(1-n_3)\text{EDGE}_1 - (1-n_2)n_3\text{EDGE}_9 - n_2(1-n_3)\text{EDGE}_3 - n_2n_3\text{EDGE}_{11} \\
    &\quad + (1-n_1)(1-n_2)(1-n_3)\text{POINT}_1 + (1-n_1)(1-n_2)\text{POINT}_5 + (1-n_1)n_2(1-n_3)\text{POINT}_4 \\
    &\quad + (1-n_1)n_2n_3\text{POINT}_8 + n_1(1-n_2)(1-n_3)\text{POINT}_2 + n_1(1-n_2)n_3\text{POINT}_6 \\
    &\quad + n_1n_2(1-n_3)\text{POINT}_3 + n_1n_2n_3\text{POINT}_7
\end{align*}
\]

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.

2.3 STRUT IN ANNULUS

Previous SSME/HGM computations reported earlier by Lockheed (Refs. 1 and 4) and other investigations (Ref. 5) have either not modeled the support strut region in the manifold or poorly approximated its influence on the flowfield environment. It was for this reason that NASA requested additional computations be made which accurately included these obstacles in the flow path at the entrance to the fuel bowl of the manifold. Initial work on this task was performed under this contract and is being reported here. Results of follow-on work are presented in the final report for NASA-37359 (Ref. 2).

The approach was to generate a computational grid consisting of a single strut in an annular channel with the dimensions of the strut size, channel width, and channel curvature being approximately the same as those in the actual SSME/HGM. Numerical experiments were then to be performed using a three-dimensional incompressible Navier-Stokes code which employed an implicit solution algorithm. The Lockheed in-house algebraic grid code was used to model the geometry and MNS3D was used to obtain flowfield solutions.
The INS3D code solves the three-dimensional incompressible Navier-Stokes equations in primitive variables. An implicit finite difference operator is used in a general curvilinear coordinate system. The solution procedure uses the standard approximate factorization scheme. The pressure field solution is based on the concept of adding a time-like pressure term into the continuity equation via an artificial compressibility factor. This approach was first introduced by Chorin (Ref. 6) and later adopted by Steger and Kutler (Ref. 7) using an implicit approximate factorization scheme by Beam and Warming (Ref. 8). It is from these earlier developments that INS3D evolved (Ref. 3).

Values of the artificial compressibility factor are bounded in order not to influence the steady state mass conservation. In the INS3D methodology mass conservation is of crucial importance if a stable solution is to result. Since the continuity equation is modified to obtain a hyperbolic-type equation, pressure waves of finite speed will be introduced. The speed of propagation of these pressure waves depends on the magnitude of the compressibility parameter. When the pressure waves travel through a given location a pressure gradient is created there. Near boundaries, the viscous boundary layer must respond to this pressure fluctuation. To accelerate convergence and avoid slow fluctuations it is desirable that the time required for pressure waves to propagate through the region of interest be much less than the time needed for the boundary layer to fully adjust itself. This condition provides for a lower bound on the artificial compressibility factor. The upper bound on this factor comes not from the physics but from the effects of the approximate factorization of the governing equations. When the finite difference form of the equation is factored, higher order cross-differencing terms are added to the left-hand side of the equation. These added terms must be made smaller than the original terms everywhere in the computational domain. This condition results in an upper bound on the compressibility factor.
It is well known in the computational fluid dynamics community that the approximate factorization schemes which employ alternating direction type implicit methods have stability problems in three dimensions (Refs. 9 through 12). The INS3D code satisfactorily overcomes this difficulty by providing second and fourth order smoothing terms to the algorithm to ensure stability without adversely affecting mass conservation.

Currently, the Computational Mechanics Section at Lockheed-Huntsville has the INS3D code operational on the Cray-XMP at NASA-MSFC and NASA-Ames as well as on its own VAX 11/785.
3. RESULTS

3.1 MANIFOLD GRID CODE

3.1.1 Geometry and Grid

A schematic representation showing the construction of the geometry which has been modeled is presented in Fig. 3-1. Only the outer wall is displayed and two perspectives are shown. These solid surface plots were generated with the actual output from the grid code. Only half of the manifold is represented since, due to the plane of symmetry which divides the two transfer ducts, only half need be computationally modeled.

Fig. 3-1 Schematic Showing Outer Wall of HGM Geometry that has been Modeled
The geometry is generated in five separate pieces or zones. The five zones are shown in Figs. 3-2 and 3-3. Each zone has a general hexahedral shape, and in these figures all of the eight corner points of each are clearly indicated. Figure 3-4 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 2 and the fairing at the entrances to Zone 3, the edges of each zone can be described as piecewise continuous segments composed of either straight lines, circular arcs, or elliptical arcs. In addition, excluding the two special regions previously mentioned, the surfaces of each zone can be generated by rotating an edge about an appropriate axis. For Zones 1, 2, 4, and 5, the X axis is the axis of revolution.

Zones 1 and 2 make up the bowl section of the manifold and contain 58 nodes in the x-direction, from bowl entrance to rear of the bowl, 109 nodes in the circumferential direction and 21 nodes between the inner and outer surfaces. Distribution of nodes in the bowl is presented in Fig. 3-5.

Zones 4 and 5, the turnaround duct (TAD), are composed of 71 nodes in the streamwise direction, 109 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. 3-6.

Zone 3 is the elliptical transfer duct portion. This duct has been generated with 59 nodes along the duct axis and 44 x 30 nodes in a cross section. Surface and cross-section grids for this zone are given in Fig. 3-7. Figure 3-8 displays a view of the manifold outer wall grid to show grid continuity from one zone to another. To summarize, each zone contains the following number of nodal points:

- Zones 1 and 2 (Bowl): 132,762
- Zones 3 (Transfer Duct): 77,880
- Zones 4 and 5 (TAD): 162,519
- Manifold total Nodes: 373,161
Fig. 3-2 Schematic Showing First Two Zones (Zone 1 Left; Zone 2, Right) into Which the Two-Duct HGM was Subdivided
Fig. 3-3 Schematic Showing Zones 3, Top, and 4, Middle, and 5 of the Two-Duct HGM Model
Fig. 3-4 Cartesian Coordinate System Relative to Which Positions of All Nodes are Referred
Fig. 3-5 Grid Plots for Inner and Outer Surfaces of Zones 1 and 2 as Well as Internal Grid at the Common Place of Intersection
Fig. 3-6  Grid Plots of TAD Internal Grid (Bottom) and Grid Distribution on Inner and Outer Surfaces
Fig. 3-7 Surface and Cross-Section Node Distribution for Transfer Duct
Fig. 3-8 Grid Plot Showing Node Distribution on Outer Wall of Zones 1, 2, and 3 Displaying Continuity of Grid Lines from One Zone to Another
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.

3.1.2 Computer Code

A concise input guide and an input listing for the grids displayed in Figs. 3-4 through 3-8 is also included in Appendixes A and B, respectively. A source listing of the two-duct HGM geometry code is provided in Appendix C.

Figure 3-9 shows the primary calling sequence. For completeness, secondary calling sequences are given in Fig. 3-10. A brief explanation of the function of the major subroutines in the primary calling sequence 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.
- **MESH** 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.
- **EDGES** 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 distributions.
- **OUTPUT** Provides printed output and stores geometry in File 20 for use in plotting or as input to an integration code.

The output to File 20 is in the format to be input as a multi-grid geometry file to the PLOT3D plotting code.
Fig. 3-9 Primary Calling Sequence for Two-Duct HGM Grid Code
Fig. 3-10 Secondary Calling Sequences for Two-Duct HGM Grid Code
The grid 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 a different machine.

3.1.3 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.

The input file is labeled on the card image which begins each zone input to indicate the zone being described by the succeeding card images. These zone labels correspond to those shown in Figs. 3-2 and 3-3. The input file, as listed in Appendix B, will create a multi-grid file containing all five zones. Using this input is, of course, possible only if run on a computer with sufficient CPU and/or disk storage. Alternatively the bowl, transfer duct, and TAD can be run separately by changing the second parameter on card image two, of the input, from a 6 to a 1, 3, and 4, respectively.

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 file and the detailed grid pictures presented in Figs. 3-3 through 3-8. 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 the last line of the input for that zone. The 6.0 appearing in the position could be changed to 10.0 (see page A-10).

Dimensioning in the program has been kept to a minimum. The largest dimensioned array in the bulk of the code is NODENUM(5000). At the very end of all computations the PLOT3D file can be generated by employing the program in Appendix D.
In this subroutine the x, y, z coordinate arrays are each dimensioned to 200,000. The 5,000 corresponds to twice the maximum number of nodes in a plane perpendicular to the marching direction (n direction input on card 7) for creating the geometry. The 200,000 must be equal to or greater than the number of nodes in the largest zone, which is zone 5, the second half of the TAD. In Zones 1, 2, 4, and 5, the direction of \( n_1 \) is from TAD entrance to bowl back wall; the \( n_2 \) direction is from inner to outer wall; and \( n_3 \) is directed from side opposite transfer duct circumferentially. In zone 3, \( n_2 \) increases in the streamwise direction from bowl outward, and \( n_1 \times n_3 \) form the cross planes in the duct.

Note that the code is designed to output each zone of the manifold 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 1 and 2, and 4 and 5 can easily be combined into two larger zones since at all common planes the nodal positions match exactly. It is not recommended that zones 1, 2, 4, and 5 be combined into one zone. This could easily be accomplished, since continuity in all three \( n \) directions would be maintained, but the total number of nodes would be untenable on all but a Cray II or an ETA 10 machine.

3.2 STRUT LAMINAR COMPUTATION

The Lockheed-Huntsville algebraic computational grid generation code was employed to generate a model for a single support strut in an annular channel. A C-type grid was selected for nodal distribution in parallel annular surfaces and is shown in Fig. 3-11. This is a 26 x 201 node structure with stretching toward the strut surface. The three-dimensional model consisted of 31 such surfaces in concentric cylindrical fashion with stretching toward inner and outer surfaces. Figure 3-11 also shows part of the solid surfaces in the model. The total number of grid points in the computational domain was 162,006.
Fig. 3-11 Grid Plots Showing C-Grid Used in Each of the 31 Circumferential Planes (Top) and Partial Surface Grids of Strut and Inner Outer Wall
The basic INS3D code was modified for treating the sides of the computational domain as periodic boundaries and for special treatment of the common grid plane at the center of the grid behind the strut. At the entrance plane, the velocity components were initialized for fully developed laminar channel flow at zero degree incidence to the strut. Velocity and pressure were then held fixed throughout the computation. A nondimensional time step of 0.025 was used and a steady solution for Reynolds number of 500 was obtained in 800 iterations. Figures 3-12 to 3-21 present various characteristics of this steady laminar solution. Velocity magnitude contours and static pressure contours in the region surrounding the strut in concentric cylindrical surfaces near the inner wall, center of annulus, and near outer wall are shown in Figs. 3-12, 3-13, and 3-14, respectively. An expanded view of the same information for the central annular surface appears in Fig. 3-15. The velocity contours in the figure clearly shows the extension of the strut wake region several strut lengths downstream. In the SSME/HGM the support struts are at the entrance to the bowl (exit of the TAD). Even though these current results are laminar and steady, the influence of the strut wake is shown to be significant and could cause a significant difference in the predicted flow through the transfer ducts.

More details of the strut near wake region for a central circumferential surface and central radial surface are given in Figs. 3-16 and 3.17. The influence of the wake flow pattern on flow particles originating upstream of the wake is indicated by the particle traces shown in Fig. 3-18. To further illustrate the three-dimensional character of this region of the wake Fig. 3-19 and 3-20 trace particle paths beginning at positions in the wake itself. Clearly, if the wake were unsteady, as it is in the actual HGM, these complicated flow patterns will move into the bowl, interact, and exit through the transfer ducts.

For completeness, surface pressure distribution on the strut itself is presented in Fig. 3-21.
Fig. 3-12  Static Pressure Contours (Top) and Velocity Magnitude Contours for Strut in Annular Channel near Inner Wall
Fig. 3-13 Static Pressure Contours (Top) and Velocity Magnitude Contours for Strut in Annular Channel at Mid-Channel
Fig. 3-14 Static Pressure Contours (Top) and Velocity Magnitude Contours for Strut in Annular Channel near Upper Wall
Fig. 3-15  Static Pressure Contours (Top) and Velocity Magnitude Contours for Entire Length of Computational Circumferential Plane at Mid-Channel
Fig. 3-16 Velocity Vectors in near Wake Region in a Central Plane for Case of Strut in Annular Channel (Mean Flow is from Left to Right)
Fig. 3-17 Velocity Vectors (Top) and Velocity Magnitude Contours in Central Radial Plane in near Wake Region Behind Strut in Annular Channel (Mean Flow is from Left to Right)
Fig. 3-18 Particle Traces for Flow Past Strut in Annular Channel Showing Paths of Particles Released near Front of Strut in a Plane near the Inner Wall (Top), one Quarter Channel Height Above Inner Wall (Bottom Left) and Mid-Channel
Fig. 3-19 Particle Traces of Particles Released near Rear Surface of Strut
Fig. 3-20  Particle Traces of Particles Released near Central Radial Plane but just Downstream of Strut Rear Surface
Fig. 3-21 Surface Pressure Contours on Strut Front Surface (Left, Flow into Page) and Side Surface (Right, Flow Left to Right)
4. CONCLUDING REMARKS

An elliptical two-duct SSME fuel side hot gas manifold geometry code has been developed for use by the Computational Fluid Dynamics Staff of NASA-MSFC. This report describes the methodology of the program, makes recommendations on the implementation, and provides an input guide, input deck listing, and source code listing for the code. The listing is well commented in order to assist the user in following its development and logic. A magnetic tape containing the source deck will be provided, upon request, to NASA-MSFC for use on its EADS network.

The NASA-Ames three-dimensional incompressible Navier-Stokes computational fluid dynamics code, INS3D, was obtained and implemented on the MSFC IBM/Cray computer facility. A low Reynolds number laminar calculation was performed with this code on a 162,006 node model of a strut in an annular channel. The dimensions were approximately those of the SSME/HGN fuel bowl entrance region which contains 12 such support struts circumferentially distributed in hot gas flow path. The computation was made as an initial step in a thorough numerical investigation of the influence of these obstacles on the flow exiting the manifold and impinging on the main injector LOX posts.

Results of this steady laminar computation indicate that a complete three-dimensional analysis of the whole manifold would require a "strut zone" for reliable predictions of the transfer duct exit plane flow structure. This is especially evident since flow visualization results have shown that the duct flow is largely unsteady (Ref. 13).
5. REFERENCES


Appendix A

HGM GRID CODE INPUT GUIDE
Appendix A

INTRODUCTION

This 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 internal coordinate system, and is described using points, edges, and surfaces. An edge may consist of from one to ten segments. A segment or a surface may require special input depending on its type. This is the case for zones 2 and 3 of this two-duct HGM model.

The second section of this appendix 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, but may be set in specific subroutines in the program itself.

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. Twelve edges are used to describe a 3D zone.

Map: The geometry maps a point from η space into real space. When describing a surface mapping one could say that setting map = 2 refers to a planar η space surface being mapped onto a cylindrical surface in real space.

Node: At each intersection of η 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.

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 η coordinate system.

η₁, η₂, η₃: Localized coordinate directions within a zone. These coordinates describe a cube in η space. 0 ≤ ηᵢ ≤ 1
### SUMMARY OF CARDS

<table>
<thead>
<tr>
<th>Card Type</th>
<th>Parameter List/Format</th>
</tr>
</thead>
<tbody>
<tr>
<td>Flowfield Parameters</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>ITITLE(I), I=1,80 (20A4)</td>
</tr>
<tr>
<td>2</td>
<td>NZONE, IZINDEX, MAPTEN, INCHES (8I5)</td>
</tr>
<tr>
<td>3</td>
<td>DCT1(I), DCT2(I), DCT3(I), I=1,37 (8F10.3)</td>
</tr>
<tr>
<td>Zone Parameters</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>NSECT (I5)</td>
</tr>
<tr>
<td>Section Parameters</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>(NMBRNDS(I), I=1,3), (ISTRCH(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), I=1,3), J=1, 8 (8E10.4)</td>
</tr>
<tr>
<td>13</td>
<td>[(SEGMAX(I,K,J), I=1,3), 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 Format(20A4)

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

CARD TYPE 2  Problem Option Controls Flags Format(5I5)

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

IZINDEX  Zone index for selecting individual components of the manifold to be computed.

= 1 zones 1 and 2 of Bowl
= 3, Transfer Duct
= 4 zones 4 and 5 of Turnaround Duct
= 6 all five zones

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
Bowl Hole Parameters

CARDS TYPE 3a Angles, in degrees, about the axis of the transfer duct, measured counterclockwise from the x, z-plane as viewed down the z-axis.

CARDS TYPE 3b Radial distance difference (in inches) between transfer duct ellipse and hole perimeter along angular directions specified on cards 3a.

CARDS TYPE 3c Radius of curvature (in inches) at each angular distance, specified on cards 3a, for describing the transfer duct weld or fairing.

Zone Parameter

CARD TYPE 4 Index dividing zone data Format(I5)

NSECT Integer must be 1 and is used for separating data for each zone on the input file.

Section Parameters

CARD TYPE 5 Edge Shape Function Indicators Format(12I5) or 3(I10)

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 T1 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 functions 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 Conics (input COEFE(I))
- 4 Edge of revolution (input COEFE(I))
- 5 Special segment (input COEFE(I))
- 6 Special segment (input COEFE(I))
- 7 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 Hole in bowl surface
- 6 Duct surface at bowl
- 7 Duct fairing 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(S(I),I=1 3

NMBRND(S(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(S(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 ISTRTCH(I) ≥ 2)

STRETCH(I), I=1, 3

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

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

<table>
<thead>
<tr>
<th>ISTRTCH</th>
<th>STRETCH</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>2.0</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>STRETCH</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>4.0</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>STRETCH</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>6.0</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>STRETCH</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>8.0</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>STRETCH</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>10.0</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>ISTRTCH</th>
<th>STRETCH</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>2.0</td>
</tr>
</tbody>
</table>

A-9

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

COEF(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 | \( \text{COEFE}(I), I=1,3 \) are the \( x, y, \) and \( z \) coordinates of the center of the arc.

3 | Edge of Revolution | 
\( \text{COEFE}(I), I=1,3 \) are coordinates of the center of the arc.
\( \text{COEFE}(I), I=4,6 \) 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 \( \text{MAPSIDE}(I) = 4 \) and \( \text{IDIM} = 3 \))

\( \text{COEFS}(I), I=1,8 \)

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 \( \text{MAPSIDE}(I) = 4 \) on card type 8 is input on a separate card in the same order as they occur on card type 8.

#### CARD TYPE 16

Coordinates of Points Format(5E10.4)

\( \text{POINT}(I,J), J = \text{Point number} \)

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

**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(I,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 $k^{th}$ segment of edge J.
SEGMAX(2,K,J) - The extremal y coordinate for the $k^{th}$ segment of edge J.
SEGMAX(3,K,J) - The extremal z coordinate for the $k^{th}$ segment of edge J.

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

The maximum value of the $\eta_I$ coordinate on the $k^{th}$ segment of edge J. Input a negative value when defining a fold line.
Appendix B

HGM GRID CODE INPUT LISTING
### Two Duct HGM II+

<table>
<thead>
<tr>
<th>Zone</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>Angle</td>
<td>0.0</td>
<td>0.30</td>
<td>0.06</td>
<td>0.55</td>
<td>0.75</td>
</tr>
<tr>
<td>Difference</td>
<td>0.28</td>
<td>0.25</td>
<td>0.17</td>
<td>0.90</td>
<td>0.43</td>
</tr>
<tr>
<td>Curvature</td>
<td>2.0</td>
<td>2.0</td>
<td>2.0</td>
<td>2.0</td>
<td>2.0</td>
</tr>
</tbody>
</table>

**Bowl Zone 1**

<table>
<thead>
<tr>
<th>Zone</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>Angle</td>
<td>0.20</td>
<td>0.05</td>
<td>0.20</td>
<td>0.55</td>
<td>0.55</td>
</tr>
<tr>
<td>Difference</td>
<td>0.23</td>
<td>0.12</td>
<td>0.20</td>
<td>0.90</td>
<td>0.35</td>
</tr>
<tr>
<td>Curvature</td>
<td>2.0</td>
<td>2.0</td>
<td>2.0</td>
<td>2.0</td>
<td>2.0</td>
</tr>
</tbody>
</table>

**Bowl with Hole Zone 2**

<table>
<thead>
<tr>
<th>Zone</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>Angle</td>
<td>0.20</td>
<td>0.66</td>
<td>0.55</td>
<td>0.43</td>
<td>0.35</td>
</tr>
<tr>
<td>Difference</td>
<td>0.23</td>
<td>0.17</td>
<td>0.20</td>
<td>0.90</td>
<td>0.35</td>
</tr>
<tr>
<td>Curvature</td>
<td>2.0</td>
<td>2.0</td>
<td>2.0</td>
<td>2.0</td>
<td>2.0</td>
</tr>
</tbody>
</table>

**Duct Zone 3**

<table>
<thead>
<tr>
<th>Zone</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>Angle</td>
<td>0.20</td>
<td>0.66</td>
<td>0.55</td>
<td>0.43</td>
<td>0.35</td>
</tr>
<tr>
<td>Difference</td>
<td>0.23</td>
<td>0.17</td>
<td>0.20</td>
<td>0.90</td>
<td>0.35</td>
</tr>
<tr>
<td>Curvature</td>
<td>2.0</td>
<td>2.0</td>
<td>2.0</td>
<td>2.0</td>
<td>2.0</td>
</tr>
</tbody>
</table>

**TAD # 1 Zone 4**

<table>
<thead>
<tr>
<th>Zone</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>Angle</td>
<td>0.20</td>
<td>0.66</td>
<td>0.55</td>
<td>0.43</td>
<td>0.35</td>
</tr>
<tr>
<td>Difference</td>
<td>0.23</td>
<td>0.17</td>
<td>0.20</td>
<td>0.90</td>
<td>0.35</td>
</tr>
<tr>
<td>Curvature</td>
<td>2.0</td>
<td>2.0</td>
<td>2.0</td>
<td>2.0</td>
<td>2.0</td>
</tr>
</tbody>
</table>

**TAD # 2 Zone 5**

<table>
<thead>
<tr>
<th>Zone</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>Angle</td>
<td>0.20</td>
<td>0.66</td>
<td>0.55</td>
<td>0.43</td>
<td>0.35</td>
</tr>
<tr>
<td>Difference</td>
<td>0.23</td>
<td>0.17</td>
<td>0.20</td>
<td>0.90</td>
<td>0.35</td>
</tr>
<tr>
<td>Curvature</td>
<td>2.0</td>
<td>2.0</td>
<td>2.0</td>
<td>2.0</td>
<td>2.0</td>
</tr>
</tbody>
</table>

/EOR
Appendix C
HGM GRID CODE LISTING
**MAIN**

```
PROGRAM HGM2DUCT(INPUT,OUTPUT,FILE20,TAPE5,TAPE6=OUTPUT,
                    &
                    TAPE20=FILE20)

PROGRAM HGM2DUCT

ELLIPITCAL TWO-DUCT HOT GAS MANIFOLD GRID CODE
DEVELOPED BY THE COMPUTATIONAL MECHANICS SECTION
LOCKHEED ENGINEERING CENTER, HUNTSVILLE, ALABAMA.
TAPE20 GEOMETRY DATA

COMMON /COUNTER/ NODESAV,NODETOT,NBNODES,NPLANE
COMMON /INITA/ IZINDEX,MAPten,INCHES
COMMON /ZONING/ IZONE,IINDEX,NINDEX,NMBRNSD(3)
COMMON /UNITS/ NU5,NU6,NU20

---INITIALIZE PROGRAM
CALL INITIAL(NZONE)

---READ INPUT FOR EACH ZONE
DO 300 IZONE=1,NZONE
    READ(NU5,1000) NZINDEX
    CALL INPUT
    CALL MESH
    300 CONTINUE

---PRINT FILE STATUS
CALL STATUS(0)
STOP

---FORMAT STATEMENTS
1000 FORMAT(I5)
END
```

**INPUT**

```
SUBROUTINE INITIAL(NZONE)

COMMON /COEFS/ COEFS(8,10,12),COEFS(8,6),NMBSSEG(12)
COMMON /COUNTER/ NODESAV,NODETOT,NBNODES,NPLANE

C-1
LOCKHEED-HUNTSVILLE ENGINEERING CENTER
```
COMMON /HEADER/ ITITLE(20), LINE
COMMON /INITA/ IZINDEX, MAPIEN, INCHES
COMMON /INITB/ IBOX(4,6), IRAY(4,3), NRAY, IPT1(12), IPT2(12)
COMMON /INITC/ PI, RADDEG
COMMON /NODNUMB/ NODENUM(4000), TOL(3)
COMMON /UNITS/ NU5, NU6, NU20

C
DIMENSION NAME(20)
C
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/

C

C
INITIALIZE
C

C
NU5 = 5
NU6 = 6
NU20 = 20
MUNIT = 20
C
NPLANE = 0
NB NODES = 0
NIMODES = 0
NODETOT = 0
NODESNAV = 0
C

C
COEFFICIENTS
C

C
DO 20 I=1,8
C

C
DO 10 J=1,6
10 COEFS(I,J) = 0.0
C

C
DO 20 J=1,10
DO 20 K=1,12
20 COEFS(I,J,K) = 0.0
C

C
NZONE = NUMBER OF ZONES TO BE USED TO GENERATE MESH
IZINDEX = NO OF THE ZONE TO BE COMPUTED
I.
.
C
1 DO ZONES 1 & 2
3 DO ZONE 3
4 DO ZONES 4 & 5
6 DO ALL FIVE ZONES
C
MAPIEN = 0 FIVE SEGMENTS PER EDGE
C
1 TEN SEGMENTS PER EDGE
C
INCHES = 0 COORDINATES INPUT IN FEET
C
1 COORDINATES INPUT IN INCHES
C

C

C
ECHO INPUT
C

C
WRITE(NU6,1100)
C

C
DO 90 I=1,500

C-2

LOCKHEED-HUNTSVILLE ENGINEERING CENTER
READ(NU5,1000,END=100) NAME
90 WRITE(NU6,1110) NAME
C
100 REWIND NU5
C
---PRINT HEADER
C
WRITE(NU6,1120)
C
---READ PROBLEM DEFINITION
C
READ(NU5,1000) ITITLE
READ(NU5,1010) NZONE,IZINDEX, &
MAPTEN,INCHES
C
---WRITE PROBLEM DEFINITION
C
NRAY = 4
C
WRITE(NU6,1130) ITITLE
WRITE(NU6,1140) NZONE,IZINDEX, &
MAPTEN,INCHES
C
CALL RWIND(NU20)
C
---DRAW PICTURE
C
IDIM = 3
CALL PICTURE(IDIM)
C
---HGM TWO DUCT DATA
C
CALL DCTDAT
C
RETURN
C
1000 FORMAT(20A4)
1010 FORMAT(1615)
1100 FORMAT(1H1,25X,17H INPUT DATA IMAGE // )
1110 FORMAT(5X,20A4)
1120 FORMAT(1H1
3 / 40X,34H LOCKHEED-HUNTSVILLE
4 / 40X,34H VAX 11/785 VERSION
5 / 40X,38H 2-DUCT HGM GEOMETRY MODULE )
1130 FORMAT(/ 12H CASE TITLE: // 5X,20A4 )
1140 FORMAT(/ 21H GEOMETRY PARAMETERS:
1 / 23H NZONE IZINDEX,
2 22H MAPTEN INCHES // 6I10 )
C
END
C
*************************************************************************
C INPUT*************************************************************************
C
SUBROUTINE INPUT
C
COMMON /COEFF/ COEFE(8,10,12),COEFS(8,6),NMBRSEG(12)
COMMON /COUNTER/ NODESAV,NODETOT,NBNODES,NPLANE
COMMON /HEADER/ ITITLE(20),LINE

C-3

LOCKHEED-HUNTSVILLE ENGINEERING CENTER
COMMON /INITA/ IIINDEX, MAPTN, INCHES
COMMON /INITB/ IBOX(4,6), IRAY(4,3), NRAY, IPT1(12), IPT2(12)
COMMON /INPUT/ EDGE(3,12), POINT(3,8), SIDE(3,6)
COMMON /MAPING/ MAPSIDE(6), MAPSEG(10,12)
COMMON /MAXIMUM/ ETAMAX(10,12), SEGMAX(3,10,12)
COMMON /NODNMBR/ NODENUM(4000), TOL(3)
COMMON /SPACING/ ISTRTCH(3), STRETCH(3), ETAS(3,200), ETA(3), DETA(3)
COMMON /UNITS/ NU5, NU6, NU20
COMMON /ZONING/ IZONE, ISECT, NZINDEX, NMBRNDS(3)

C

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

C

C MAPEDGE(I) INDICATES TYPE OF GEOMETRY FOR EDGE I
= 1 LINEAR
  = 2 CIRCULAR ARC
  = 3 CONIC(PARABOLA, ELLIPSE, HYPERBOLA)
  = 4 HELICAL ARC
  = 5 TRIG FUNCTION OF X
  = 6 TRIG FUNCTION OF ANGLE
  = 7 CUBIC POLYNOMIAL

C MAPSIDE(I) TYPE OF GEOMETRY FOR SURFACE I
  = 1 FLAT SURFACE
  = 2 CYLINDRICAL SURFACE
  = 4 EDGE OF REVOLUTION

C MARCH ETA DIRECTION IN WHICH COMPUTATION IS TO ADVANCE
  = 0 NO STRETCHING IN ETA(I) DIRECTION
  = 1 INPUT VALUES OF ETA(I) (N)
  = 2 DECREASE SPACING IN ETA(I) DIRECTION (INPUT FACTOR)
  = 3 INCREASE SPACING IN ETA(I) DIRECTION (INPUT FACTOR)
  = 4 DOUBLE STRETCHING (INPUT FACTOR)
  = 5 DECREASING SPACING IN ETA(I) DIRECTION (MINIMUM)
  = 6 INCREASING SPACING IN ETA(I) DIRECTION (MINIMUM)
  = 7 DOUBLE STRETCHING (MINIMUM SPACING)
  = 8 ORIGINAL DECREASING STRETCHING FUNCTION
  = 9 ORIGINAL INCREASING STRETCHING FUNCTION
  =10 USER INPUT STRETCHING FUNCTION

C WRITE(NU6,1100) ITITLE, IZONE
C READ INPUT PARAMETERS
C
100 IF(MAPTN.EQ.0) THEN
  C READ(NU5,1000) MAPEDGE
  C END IF
  C READ(NU5,1000) MAPSIDE
  C READ(NU5,1000) MARCH
  C IF(MARCH.LT.1 .OR. MARCH.GT.3) MARCH = 1

C-4

LOCKHEED-HUNTSVILLE ENGINEERING CENTER
READ(NU5,1000) NMBRND5,ISTRTCH

C---PRINT OUT INPUT VARIABLES
C
WRITE(NU6,3000) MAPEDGE
WRITE(NU6,3010) MAPSIDE
WRITE(NU6,3020) MARCH
C
WRITE(NU6,3040) NMBRND5,ISTRTCH
C
C---INITIALIZE FLAGS TO CHECK IF POINT, EDGE, OR SURFACE HAS BEEN INPUT
C
DO 130 I=1,8
  130 IFLAGP(I) = 0
C
DO 140 I=1,12
  140 IFLAGE(I) = 0
C
DO 150 I=1,6
  150 IFLAGS(I) = 0
C
C---SET ORDER OF EXECUTION
C
INDEX(1) = MARCH
INDEX(3) = 3
C
DO 160 I=2,3
  160 INDEX(I) = INDEX(I-1) + 1
C
IF(INDEX(I).GT.3) INDEX(I) = 1
C
C---INITIALIZE ETA
C
ETA(1) = 0.0
ETA(2) = 0.0
ETA(3) = 0.0
C

C-----------------------------------------------
C READ STRETCHING PARAMETERS
C-----------------------------------------------
C
300 DO 310 I=1,3
C
STRETCH(I) = 0.0
C
IF(ISTRTCH(I).LE.1) GO TO 310
C
READ(NU5,1020) STRETCH
C
GO TO 320
C
310 CONTINUE
C
C-----------------------------------------------
C COMPUTE STRETCHING FUNCTION PARAMETER B USING NEWTON-RAPHSON
C-----------------------------------------------
C
320 DO 370 I=1,3
C
B = STRETCH(I)
DS = B
TNODE = REAL(NMBRND5(I))
C---DECREASING OR INCREASING SPACING (INPUT MINIMUM SPACING)
C
IF(ISTRCH(I).EQ.5 .OR. ISTRCH(I).EQ.6) THEN
B = SQRT(1.0 - (TNODE - 1.0)*DS)/(TNODE - 1.0)
DO 330 ITER=1,10
ARG1 = B*(TNode - 1.)
ARG2 = B*(TNode - 2.)
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)
TANHP1 = 1.0 - TANH1**2
TANHP2 = 1.0 - TANH2**2
PSIP = (1.0/TANH1)*(TANHP2*(TNode - 2.) - (TANH2/TANH1) * TANHP1*(TNode - 1.))
IF(PSIP.EQ.0.) THEN
WRITE(NU6,5100)
STOP
END IF
B0 = B
B = B0 - PSI/PSIP
DBF = (B - B0)/B0
WRITE(NU6,5200) I, PSI, PSIP, B, DBF
IF(ABS(DBF).LE.0.001) GO TO 340
END IF
C---DOUBLE STRETCHING (INPUT MINIMUM SPACING)
C
IF(ISTRCH(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)
C
C-6
LOCKHEED-HUNTSVILLE ENGINEERING CENTER
PSI = DS - 0.5*(1.0 - TANH3/TANH1)

TANHP1 = 1.0 - TANH1**2
TANHP3 = 1.0 - TANH3**2

PSIP = (0.5/TANH1)*(TANHP3*(TNODE - 3.0) - (TANH3/TANH1)* TANHP1*(TNODE - 1.0))

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

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

WRITE(NU6,5200) I,PSI,PSIP,B,DBF

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

360 END IF

370 STRETCH(I) = B

C---WRITE STRETCHING PARAMETERS
C  DO 390 I=1,3
C    IF(I(MRTCH(I)).LE.1) GO TO 390
C    WRITE(NU6,3050) STRETCH
C    GO TO 400

390 CONTINUE

C---READ INPUT PARAMETERS FOR ARBITRARY GRID SPACING (ETAS)
C  400 DO 410 I=1,3
C    IF(I(MRTCH(I)).NE.1) GO TO 410
C    IF(I(SECT.GT.1 .AND. I.NE.MARCH) GO TO 410
C    READ(NU5,1020) (ETAS(I,J),J=1,NMRNDS(I))
C    WRITE(NU6,1110) I,(ETAS(I,J),J=1,NMRNDS(I))

410 CONTINUE

C---HGS TWO DUCT INPUT DATA
C  CALL HGMIN

C---INPUT PARAMETERS FOR EDGE COEFFICIENTS
C  WRITE(NU6,1100) ITITLE,IZONE
C  WRITE(NU6,2040)
C
LINE = 6
II = 0
C
C---TOTAL NUMBER OF EDGES
C NEDGES = 8*3 - 12
C DO 540 I = 1, NEDGES
C IF(IFLAGE(I).EQ.1) GO TO 540
C ITOTAL = 1
MAP = MAPEDGE(I)
C
C---DETERMINE THE NUMBER OF SEGMENTS ON AN EDGE
C DO 500 J = 1, 10
C 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
C---DETERMINE THE MAPPING FOR EACH SEGMENT
C MAPSEG(J, I) = MAP/ITOTAL
MAP = MAP - MAPSEG(J, I)*ITOTAL
C
C---EDGE COEFFICIENTS FOR EACH SEGMENT
C IF(MAPSEG(J, I).LE.1) GO TO 530
C IF(I.NE.II) THEN
C LINE = LINE + 2
C IF(LINE.GE.60) THEN
C WRITE(NU6, 1100) ITITLE, 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 C-8
LOCKHEED-HUNTSVILLE ENGINEERING CENTER
WRITE(NU6,1100) ITITLE,IZONE
WRITE(NU6,2040)

LINE = 8

END IF

WRITE(NU6,2060) J,(COEFE(K,J,I),K=1,8)

END IF

II = I

530   ITOTAL = ITOTAL/10

540 CONTINUE

-----------------------------------
INPUT PARAMETERS FOR SURFACE COEFFICIENTS
-----------------------------------

LINE = LINE + 5

IF(LINE.GE.60) THEN

WRITE(NU6,1100) ITITLE,IZONE

LINE = 6

END IF

WRITE(NU6,3060)

DO 610 I=1,6

IF(IFLAGS(I).EQ.1) GO TO 610

IF(MAPSIDE(I).LE.2) GO TO 610

LINE = LINE + 2

IF(LINE.GE.60) THEN

WRITE(NU6,1100) ITITLE,IZONE

WRITE(NU6,3060)

LINE = 8

END IF

WRITE(NU6,3065) I,(COEFS(J,I),J=1,8)

610 CONTINUE

-----------------------------------
INPUT DATA FOR CORNER POINTS AND SEGMENT END POINTS
-----------------------------------

LINE = LINE + 5

IF(LINE.GE.60) THEN

WRITE(NU6,1100) ITITLE,IZONE

C-9

LOCKHEED-HUNTSVILLE ENGINEERING CENTER
C
C LINE = 6
END IF
C
WRITE(NU6,3070)
DO 750 I=1,NEDGES
IF(I.GT.8) GO TO 700
C--- CORNER POINTS
LINE = LINE + 2
IF(LINE.GE.60) THEN
WRITE(NU6,1100) ITITLE,IZONE
WRITE(NU6,3070)
LINE = 8
END IF
WRITE(NU6,3080) I,(POINT(J,I),J=1,3)
700 IF(NMBRSEG(I).EQ.1) GO TO 750
C--- SEGMENT MAXIMUMS
DO 740 J=1,NMBRSEG(I) - 1
LINE = LINE + 2
IF(LINE.GE.60) THEN
WRITE(NU6,1100) ITITLE,IZONE
WRITE(NU6,3070)
LINE = 8
END IF
IF(FLAGE(I).EQ.1) ETAMAX(J,I) = ETAMAX(J,I)*(NMBRND(N) - 1.0) + 1.0
C
C 720 DO 720 N=1,3
DO 720 L=1,4
IF(IRAY(L,N).EQ.1) GO TO 730
720 CONTINUE
C--- CONVERT NODE NUMBER TO ETA VALUE
ETAMAX(J,I) = (ETAMAX(J,I) - 1.0)/(NMBRND(N) - 1.0)
C-10

LOCKHEED-HUNTSVILLE ENGINEERING CENTER
740 CONTINUE
750 CONTINUE
C
DO 930 I=1,3
  DETA(I) = 1.0/(NMBRNS(I) - 1.0)
  FACTOR = NMBRNS(I)*10.0
  IF(IISTRCH(I).GT.0) FACTOR = NMBRNS(I)*20.0
C
DO 920 J=1,NRAY
  L = IRAY(J,I)
C---COMPARE X, Y, AND Z BETWEEN END POINTS OF AN EDGE
C
DO 920 K=1,3
  DELTA = ABS(POINT(K,IPT1(L)) - POINT(K,IPT2(L)))/FACTOR
  IF(DELTA.LE.0.0) GO TO 920
  IF(DELTA.LT.TOL(K)) TOL(K) = DELTA
C
920 CONTINUE
930 CONTINUE
C----------------------------------"""-------~------------------------
C PRINT TITLE
C-----------------------------------------------------------------------
C---FORMAT STATEMENTS
C
WRITE(NU6,1100) ITITLE,IZONE
LINE = 1
WRITE(NU6,1140)
LINE = 3
C
RETURN
C---FORMAT STATEMENTS
C
1000 FORMAT(16I5)
1010 FORMAT(6I10)
1020 FORMAT(8E10.0)
1030 FORMAT((4I10))
C
1100 FORMAT(1H1,10X,20A4,13X,6H ZONE,I3)
1110 FORMAT(// 11H FIXED ETA_,I1,8H VALUES: // (10(3X,F10*7)) )
1140 FORMAT(// 44H NODE X Y Z )
C
2000 FORMAT(// 32H EDGE SHAPE FUNCTION INDICATORS:
  6 // 40H EDGE_1 EDGE_2 EDGE_3 EDGE_4 / 4I10 )
2010 FORMAT(// 26H BOUNDARY CONDITION FLAGS:
  1 // 40H EDGE_1 EDGE_2 EDGE_3 EDGE_4 / 4I10
  2 // 26H MARCHING DIRECTION = ETA_,IT)
2020 FORMAT(// 17H NUMBER OF NODES:
  1 // 20H ETA_1 ETA_2 / 2I10
  2 // 22H STRETCHING FUNCTIONs:
  3 // 20H ETA_1 ETA_2 / 2I10 )
2030 FORMAT(// 32H STRETCHING FUNCTION PARAMETERS:

C-11

LOCKHEED-HUNTSVILLE ENGINEERING CENTER
&  // 30H  ETA_1  ETA_2  /  2(2X,F13.7)  )
2040 FORMAT(// 25H EDGE SHAPE COEFFICIENTS:  
   1  // 42H EDGE SEGMENT  COEFF_1  COEFF_2,
   2  // 42H EDGE_3  COEFF_4  COEFF_5,
   3  // 42H EDGE_6  COEFF_7  COEFF_8  )
2050 FORMAT(/  2X,I2,6X,I1,3X,8(I1,F13.7))
2060 FORMAT(/  10X,I1,3X,8(I1,F13.7))
2080 FORMAT(/  7H POINT , I2,1H:12X,2(7X,F13.7),9X,F7.2)
2090 FORMAT(/  7H EDGE , I2,1H:7X,13,2X,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_I1  EDGE_I2  /  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  // 22H STRETCHING FUNCTIONS:  
   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(I1,F13.7))
3070 FORMAT(// 29H COORDINATES AND ETAMAXES:  
   1  // 10X,39H SEGMENT  X  
   2  // 41H  0  ETAMAX
3080 FORMAT(/  7H POINT , I2,1H:8X,3(6X,F13.7),2(8X,F7.2))
3090 FORMAT(/  7H EDGE , I2,1H:3X,13,2X,3(6X,F13.7),2(8X,F7.2),
               &  7X,F9.2)
C
5000 FORMAT(1H0,43H SINGULARITY IN TWO-END STRETCHING FUNCTION,
   &  10H PARAMETER)
5100 FORMAT(1H0,43H SINGULARITY IN ONE-END STRETCHING FUNCTION,
   &  10H PARAMETER)
5200 FORMAT(1H0,4H I =,I2,7H PSI =,E12.4,8H PSIP =,E12.4,
   &  5H B =,E12.4,7H DBF =,E12.4)
C
END
C
***MAPPING***
C
SUBROUTINE MESH
C
MESH CONTROLS THE GENERATION OF THE SPATIAL COORDINATES OF EACH NODE
C IN THE MESH.
C
COMMON /COUNTER/ NODESAV, NODETOT, NBNODES, NPLANE
COMMON /HEADER/ ITITLE(20), LINE
COMMON /INITA/ IZINDEX, MAPTEN, INCHES

C-12

LOCKEED-HUNTSVILLE ENGINEERING CENTER
COMMON /INITB/ IBOX(4,6),IRAY(4,3),NRAY,IPT1(12),IPT2(12)
COMMON /INPUTA/ EDGE(3,12),POINT(3,8),SIDE(3,6)
COMMON /INPUTBC/ INODEBC(3),ISIDE(3)
COMMON /MAPING/ MAPSIDE(6),MAPSEG(10,12)
COMMON /MARCHS/ MARCH,INDEX(3)
COMMON /NODNMBR/ NODENUM(4000),TOL(3)
COMMON /OUT/ NODE(3,4000)
COMMON /PARTIAL/ DEDN(3,12),DSDN(3,2),SNORMAL(3,6)
COMMON /SPACING/ ISTRCH(3),STRETCH(3),ETA(3,200),ETA(3),DETA(3)
COMMON /UNITS/ NU5,NU6,NU20
COMMON /ZONING/ IZONE,ISECT,NZINDEX,NMBRnds(3)

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

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/

INIT = 1
FOLD = 0

C---Determine Surfaces
C
ISIDE1 = IFACE1(MARCH,INDEX(2))
ISIDE2 = IFACE2(MARCH,INDEX(2))
ISIDE3 = IFACE1(MARCH,INDEX(3))
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))

INDEX(1) = MARCH

C---Axist
C
DO 700 IAXIS=1,NMBRnds(INDEX(1))
C
IF(IAXIS.EQ.1 .AND. ISECT.GT.1) GO TO 700
C
C---Separate 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)),FOLD)
C
C---Number of Nodes to Be Stored
C
NODSTOR = NODESAV
C
C---Row
C
DO 500 JAXIS=1,NMBRNDS(INDEX(2))

C---SEPERATE BOUNDARY CONDITIONS AND DETERMINE ETA
CALL ETABC(MARCH,INDEX(2),JAXIS)

C---CALCULATE COORDINATES AND DERIVATIVES FOR POINTS ON EDGES
CALL EDGES(INIT,INDEX(2),ETA(INDEX(2)),FOLD)

C---CALCULATE COORDINATES AND SURFACE NORMAL FOR POINTS ON SURFACES-----
CALL SURFACE(INIT,ISIDE1)

CALL SURFACE(INIT,ISIDE2)

C-14

CALL SURFACE(INIT,ISIDE3)

CALL SURFACE(INIT,ISIDE4)

CALL SURFACE(INIT,ISIDE5)

CALL SURFACE(INIT,ISIDE6)

C---ETA COEFFICIENTS FOR TRI-LINEAR INTERPOLATION

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---INCREMENT NODE COUNTERS
C
INIT = 0
C
NODSAV = NODSAV + 1
NODNUM = NODSAV + NODETOT
C
C---CALCULATE THE COORDINATES
C
DO 340 L=1,3

350 CONTINUE
400 CONTINUE
500 CONTINUE

C-------------------------------------------
C PLANE OUTPUT
C-------------------------------------------

IF(IAXIS.EQ.1) GO TO 700
C
C---PRINT AND STORE DATA---------------------
C
IF(NODSTOR.NE.0) THEN
CALL OUTPUT(NU20,NODSTOR)
C
C---TRANSFER DATA SECOND PLANE TO FIRST PLANE
C
DO 630 I=1,NODESAV
630 NODE(J,I) = NODE(J,I + NODSTOR)
C
600 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  OUTPUT  
C-------------------------------  
C---PRINT AND STORE DATA  
NODSTOR = NODESAV  
C  IF(NODESAV NE 0) CALL OUTPUT(NU20,NODSTOR)  
C  
C  RETURN  
C---FORMAT STATEMENTS  
C  1100 FORMAT(1H1,10X,20A4,13X,8H SECTION,12,3H OF,13,9H FOR ZONE,13)  
C  
C  1110 FORMAT(8H NODE =,I10  
1 / 41X,2H X,13X,2H Y,12X,2H Z  
2 / 8H E( 1) =,F13.7,15H SIDE(1) =,6(1X,F13.7)  
3 / 8H E( 3) =,F13.7,15H SIDE(3) =,6(1X,F13.7)  
5 / 8H E( 6) =,F13.7,15H SIDE(6) =,6(1X,F13.7)  
6 / 8H E( 5) =,F13.7,15H SIDE(5) =,6(1X,F13.7)  
7 / 8H E( 2) =,F13.7,15H SIDE(2) =,6(1X,F13.7)  
C  
C  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(14) =,F13.7,15H EDGE( 4) =,6(1X,F13.7)  
4 / 8H E( 9) =,F13.7,15H EDGE( 5) =,6(1X,F13.7)  
5 / 8H E( 7) =,F13.7,15H EDGE( 6) =,6(1X,F13.7)  
C  
C  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)  
3 / 8H E(12) =,F13.7,15H EDGE(10) =,6(1X,F13.7)  
4 / 8H E(18) =,F13.7,15H EDGE(11) =,6(1X,F13.7)  
5 / 8H E(11) =,F13.7,15H EDGE(12) =,6(1X,F13.7)  
C  
C  1140 FORMAT(8H E(19) =,F13.7,15H POINT(1) =,6(1X,F13.7)  
1 / 8H E(23) =,F13.7,15H POINT(2) =,6(1X,F13.7)  
2 / 8H E(25) =,F13.7,15H POINT(3) =,6(1X,F13.7)  
3 / 8H E(21) =,F13.7,15H POINT(4) =,6(1X,F13.7)  
4 / 8H E(20) =,F13.7,15H POINT(5) =,6(1X,F13.7)  
5 / 8H E(24) =,F13.7,15H POINT(6) =,6(1X,F13.7)  
6 / 8H E(26) =,F13.7,15H POINT(7) =,6(1X,F13.7)  
7 / 8H E(22) =,F13.7,15H POINT(8) =,6(1X,F13.7)  
C  
C END  
C"""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""
COMMON /COEFF/, COEFE(8, 10, 12), COEFS(8, 6), NMBRSEG(12)
COMMON /HEADER/, NODESAV, NODETOT, NBNODES, NPLANE
COMMON /INITA/, ITITLE(20), LINE
COMMON /INITB/, IZINDEX, MAPLEN, INCHES
COMMON /INPUTA/, IBOX(4, 6), IRAY(4, 3), NRAY, IPT1(12), IPT2(12)
COMMON /MAPED/, EDGE(3, 12), POINT(3, 8), SIDE(3, 6)
COMMON /MAXIMUM/, KSEG(12), UIAXIS(3, 6), IEDGE1(6), IEDGE2(6), GAMMA
COMMON /PARTIAL/, ETAMAX(10, 12), SEGMAX(3, 10, 12)
COMMON /SEGMENT/, PT1(6, 10, 12), PT2(6, 10, 12), ETA1(10, 12), ETA2(10, 12)
COMMON /UNITS/, NU5, NU6, NU20
COMMON /ZONING/, IZONE, ISECT, NZINDEX, NMBRnds(3)

C---------------------------------------------
C  INTERMEDIATE PRINT
C---------------------------------------------

NODNUM = NODESAV + NODETOT + 1

C---------------------------------------------
C  CALCULATE EDGE COORDINATES AND DERIVATIVE
C---------------------------------------------

20 DO 200 I=1, NRAY

C----DETERMINE WHICH EDGE

  IEDGE = IRAY(I, IDIR)

C----EDGE INITIALIZATION

  IF(INIT.EQ.1) CALL EMAPI(IEDGE, FOLD, IDIR, I)

C----DETERMINE WHICH SEGMENT

  DO 100 JSEG=1, NMBRSEG(IEDGE)

  ISEG = JSEG

  100 IF(ETA.GE.ETA1(JSEG, IEDGE) .AND. ETA.LE.ETA2(JSEG, IEDGE)) GOTO 110

  110 KSEG(IEDGE) = ISEG

C----DETERMINE WHERE ALONG THE SEGMENT

  DENOM = ETA2(ISEG, IEDGE) - ETA1(ISEG, IEDGE)

  IF(DENOM.EQ.0.0) DENOM = 1.0

  RATIO = (ETA - ETA1(ISEG, IEDGE))/DENOM

C----CALCULATE THE COORDINATES AND DERIVATIVE

  CALL EMAP(IEDGE, ISEG, RATIO, EDGE(1, IEDGE), DEDN(1, IEDGE))

C---------------------------------------------
C  INTERMEDIATE PRINT
C---------------------------------------------

200 CONTINUE

RETURN

C----FORMAT STATEMENTS
C
1100 FORMAT(1H1,10X,20A4,13X,8H SECTION,12,3H OF,13,9H FOR ZONE,13)
1110 FORMAT(/ 41H NODE EDIT X Z TANGENT: THETA,
2 36H PHI ETA RATIO ETAMAX)
1120 FORMAT(1X,I6,4X,I2,3X,3(3X,Fl3~7)~llX~2(2X~F7~2)~3(2X~F7~5))
C
END
C**********************************************************************
C*******************************~**************************************
C
SUBROUTINE EMAPI(EDGE,POLD,DIR,NMBEDG)
C
C EDGE MAPPING INITIALIZATION
C
COMMON /COEFF/ COEFE(8,10,12),COEFS(8,6),NMRSEG(12)
COMMON /COUNTER/ NODES,NODETOT,NNBnodes,NPLANE
COMMON /EDGE0/ UI(3,10,12),UJ(3,10,12),UK(3,10,12),
& RI(3,10,12),R2(3,10,12),THETA(10,12),
& RA(10,12),RC(10,12),RE(10,12),THETA1(10,12)
COMMON /EDGE1/ RM1(10,12),RM2(10,12),RPA1(10,12),RPA2(10,12)
COMMON /HEADER/ ITITLE(20),LINE
COMMON /INITA/ IZINDEX,MAPTEN,INCHES
COMMON /INITB/...
COMMON /INPUTA/ EDGE(3,12),POINT(3,8),SIDE(3,6)
COMMON /MAPED/ KSEG(12),UAXIS(3,6),EDGE1(6),EDGE2(6),ETANTMAX
COMMON /MAPING/ MAPSIDE(6),MAPSEG(10,12)
COMMON /MARCHS/ MARCHEINDEX(3)
COMMON /MAXIMUM/ ETAMAX(10,12),SEGMAX(3,10,12)
COMMON /PARTIAL/ DEDN(3,12),DSIDN(3,2),SNORMAL(3,6)
COMMON /SEGMENT/ PT1(6,10,12),PT2(6,10,12),EPA1(10,12),EPA2(10,12)
C
DIMENSION ETANTMAX(10,12),VECTER(3)
C
---INITIALIZE
C
KSEG(EDGE) = 1
C
DO 10 N=1,NMRSEG(EDGE)
10 ETAMX(N,EDGE) = ETAMAX(N,EDGE)
C
40 ETAMX(NMRSEG(EDGE),EDGE) = 1.0
C
ETAL1(1,EDGE) = 0.0
ETAMAX(NMRSEG(EDGE),EDGE) = 1.0
C
---INITIALIZE SEGMENTS
C
DO 1000 ISEG=1,NMRSEG(EDGE)
C
---DETERMINE THE COORDINATES AT THE END POINTS
C
IF(ISEG.EQ.1) THEN
DO 50 J=1,3
PT1(J, 1,EDGE) = POINT(J, IPT1(EDGE))
C
C
C-18
LOCKHEED-HUNTSVILLE ENGINEERING CENTER
SEGMAX(J,NMBRSEG(IEDGE),IEDGE) = POINT(J,IPT2(IEDGE))
50 PT2(J,ISEG,IEDGE) = SEGMAX(J,ISEG,IEDGE)
C
ELSE C
DO 60 J=1,3
60 PT1(J,ISEG,IEDGE) = SEGMAX(J,(ISEG-1),IEDGE)
PT2(J,ISEG,IEDGE) = SEGMAX(J,ISEG,IEDGE)
C
END IF C
C---ETA VALUE AT THE END OF EACH SEGMENT C
IF(ISEG.GT.1) ETA1(ISEG,IEDGE) = ETA2(ISEG-1,IEDGE)
ETA2(ISEG,IEDGE) = ETA MAX(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)
R2(J,ISEG,IEDGE) = PT2(J,ISEG,IEDGE) - COEFE(J,ISEG,IEDGE)
80 CALL VMAG(R1(J,ISEG,IEDGE),RMAG1)
CALL VMAG(R2(J,ISEG,IEDGE),RMAG2)
C
C---CIRCULAR ARC C
MAP = MAPSEG(ISEG,IEDGE)
C
GO TO (1000,200,300,400,500,600,700) MAP
C
C---Sweep Angle C
200 CALL VDOT(R1(J,ISEG,IEDGE),R2(J,ISEG,IEDGE),R1DOTR2)
ARG = R1DOTR2/(RMAG1*RMAG2)
C
C---Projection onto Conic Axis C
300 IF(COEFE(7,ISEG,IEDGE).GT.0.0) THEN
C
DO 305 I=1,3
305 TEMP = R1(I,ISEG,IEDGE)
R1(I,ISEG,IEDGE) = R2(I,ISEG,IEDGE)
R2(I,ISEG,IEDGE) = TEMP
C
END IF

CALL VDOT(R1(1,ISEG,IEDEGE),COEFE(4,ISEG,IEDEGE),A1)
CALL VDOT(R2(1,ISEG,IEDEGE),COEFE(4,ISEG,IEDEGE),A2)

C---RECIPIROCAL OF ECCENTRICITY (RE = 1/E)
ECT = ABS((A1-A2)/(RMAG1-RMAG2))

IF(ECT.GT.0.9999.AND.ECT.LT.1.0001) ECT = 1.0
RE(ISEG,IEDEGE) = ECT

C---DISTANCE FROM DIRECTRIX TO FOCUS
RC(ISEG,IEDEGE) = RMAG1*RE(ISEG,IEDEGE) - A1

C---THETA1: ANGLE BETWEEN NEGATIVE OF CONIC AXIS AND R1
ARG = A1/RMAG1
IF(ABS(ARG).GT.1.0) ARG = ARG/ABS(ARG)
THETA1(ISEG,IEDEGE) = ACOS(-ARG)

C---THETA2: ANGLE BETWEEN NEGATIVE OF CONIC AXIS AND R2
ARG = A2/RMAG2
IF(ABS(ARG).GT.1.0) ARG = ARG/ABS(ARG)
THETA2 = ACOS(-ARG)

C---SWEEP ANGLE: THETA2 - THETA1
THETA(ISEG,IEDEGE) = THETA2 - THETA1(ISEG,IEDEGE)

C---LOCAL COORDINATE SYSTEM
DO 310 J=1,3
   UJ(J,ISEG,IEDEGE) = R1(J,ISEG,IEDEGE)/RMAG1
   UI(J,ISEG,IEDEGE) = R2(J,ISEG,IEDEGE)/RMAG2
310
CALL CROSS(UJ(1,ISEG,IEDEGE),UI(1,ISEG,IEDEGE),UK(1,ISEG,IEDEGE),10)

C---INTEGRATION CONSTANTS
RA(ISEG,IEDEGE) = SQRT(RE(ISEG,IEDEGE)**2 - 1.0))
T = TAN(THETA1(ISEG,IEDEGE)**0.5)

C---ELLIPSE-----------------------------------
IF(RE(ISEG,IEDEGE).LT.1.0001) GO TO 320

C---ARC LENGTH INTEGRAL (CRC EQN. 341) EVALUATED AT THETA1
ARG = RA(ISEG,IEDEGE)*T/(RE(ISEG,IEDEGE) + 1.0)
ARC1(ISEG,IEDEGE) = 2.*RC(ISEG,IEDEGE)*ATAN(ARG)/RA(ISEG,IEDEGE)

C---ARC LENGTH INTEGRAL (CRC EQN. 341) EVALUATED AT THETA2

C-20

LOCKHEED-HUNTSVILLE ENGINEERING CENTER
T = TAN(THETA2*0.5)
ARG = RA(ISEG, IEDGE)*T/(RE(ISEG, IEDGE) + 1.0)
ARC2 = 2.0*RC(ISEG, IEDGE)*ATAN(ARG)/RA(ISEG, IEDGE)

---TOTAL ARC LENGTH
ARC(ISEG, IEDGE) = ARC2 - ARC1(ISEG, IEDGE)

---TANGENT CONSTANT
XLENGTH(ISEG, IEDGE) = ARC(ISEG, IEDGE)*(RE(ISEG, IEDGE) + 1.0)
&
/((THETA(ISEG, IEDGE)*RC(ISEG, IEDGE))

---HYPERBOLA
320 IF(RE(ISEG, IEDGE).GT.0.9999) GO TO 330

---ARC LENGTH INTEGRAL EVALUATED AT THETA1
ARG = (RA(ISEG, IEDGE)*T + RE(ISEG, IEDGE) + 1.)
&
/(RA(ISEG, IEDGE)*T - RE(ISEG, IEDGE) - 1.)
ARG = ABS(ARG)
ARC1(ISEG, IEDGE) = RC(ISEG, IEDGE)*LOG(ARG)/RA(ISEG, IEDGE)

---ARC LENGTH INTEGRAL EVALUATED AT THETA2
T = TAN(THETA2*0.5)
ARG = (RA(ISEG, IEDGE)*T + RE(ISEG, IEDGE) + 1.)
&
/(RA(ISEG, IEDGE)*T - RE(ISEG, IEDGE) - 1.)
ARG = ABS(ARG)
ARC2 = RC(ISEG, IEDGE)*LOG(ARG)/RA(ISEG, IEDGE)

---TOTAL ARC LENGTH
ARC(ISEG, IEDGE) = ARC2 - ARC1(ISEG, IEDGE)

---TANGENT CONSTANT
XLENGTH(ISEG, IEDGE) = -4.0*(1.0 + RE(ISEG, IEDGE))
&
*ARC(ISEG, IEDGE)/(THETA(ISEG, IEDGE)*RC(ISEG, M))

---PARABOLA

---ARC LENGTH INTEGRAL EVALUATED AT THETA1
330 ARC1(ISEG, IEDGE) = RC(ISEG, IEDGE)*T

---ARC LENGTH INTEGRAL EVALUATED AT THETA2
T = TAN(THETA2*0.5)

---TOTAL ARC LENGTH
ARC(ISEG, IEDGE) = RC(ISEG, IEDGE)*T - ARC1(ISEG, IEDGE)
C---TANGENT CONSTANT
C     XLENGTH(ISEG, IEDGE) = 2.0*ARC(ISEG, IEDGE)
     & /(THETA(ISEG, IEDGE)*RC(ISEG, IEDGE))
C
GO TO 1000
C
EDGE OF REVOLUTION                           MAP = 4
C
400 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), 20)
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), 21)
C
C---R1: VECTOR FROM AXIS TO FIRST POINT
C     CALL VDOT(R1(1, ISEG, IEDGE), UK(1, ISEG, IEDGE), RPA1(ISEG, IEDGE))
C     CALL VADD(1.0, R1(1, ISEG, IEDGE), -RPA1(ISEG, IEDGE), UK(1, ISEG, IEDGE),
     & R1(1, ISEG, IEDGE), VECTER)
C     CALL VMAG(R1(1, ISEG, IEDGE), RM1(ISEG, IEDGE))
C
C---R2: VECTOR FROM AXIS TO SECOND POINT
C     CALL VDOT(R2(1, ISEG, IEDGE), UK(1, ISEG, IEDGE), RPA2(ISEG, IEDGE))
C     CALL VADD(1.0, R2(1, ISEG, IEDGE), -RPA2(ISEG, IEDGE), UK(1, ISEG, IEDGE),
     & R2(1, ISEG, IEDGE), VECTER)
C     CALL VMAG(R2(1, ISEG, IEDGE), RM2(ISEG, IEDGE))
C
C---SWEEP ANGLE
C     CALL VDOT(R1(1, ISEG, IEDGE), R2(1, ISEG, IEDGE), R1DOTR2)
C     ARG = R1DOTR2/(RM1(ISEG, IEDGE)*RM2(ISEG, IEDGE))
C     IF(ABS(ARG).GT.1.) ARG = ARG/ABS(ARG)
C     THETA(ISEG, IEDGE) = ACOS(ARG)
C
GO TO 1000
C
USER SUPPLIED SPECIAL EDGE INITIALIZATION      MAP = 5
C
500 CONTINUE
C-----------------------------------------------
C USER SUPPLIED SPECIAL EDGE INITIALIZATION      MAP = 6
C-----------------------------------------------
600 CONTINUE
C-----------------------------------------------
C USER SUPPLIED SPECIAL EDGE INITIALIZATION      MAP = 7
C-----------------------------------------------
700 CONTINUE
C 1000 CONTINUE
C RETURN
END
C***********************************************
C***********************************************
C SUBROUTINE EMAP(EDGE,ISEG,RATIO,POINT,TANGENT)
C***********************************************
C THIS ROUTINE CALCULATES THE COORDINATES, AND TANGENT
C OF A POINT ON AN EDGE.
C-----------------------------------------------
COMMON /COEFF/  COEFE(8,10,12), COEFS(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)
COMMON /EDGE8/  RM1(10,12), RM2(10,12), RPA1(10,12), RPA2(10,12)
COMMON /MAPING/  MAPSIDE(6), MAPSEG(10,12)
COMMON /SEGMENT/  PT(6,10,12), PT2(6,10,12), ETA1(10,12), ETA2(10,12)
C
COMMON /DFN1/  D1U(3), D2U(3), D3U(3), DFNB, DFND, DFNF
COMMON /DFN2/  AE, BE, DFNR, ZSTAR, AGL, BETA1, BETA2, BETA3, BETA4
COMMON /DFN3/  AD, BD, BETA(37), DELRHO(37), RADOC(37)
COMMON /DFN5/  CE, CQ, ABOT, ATOP
C
DIMENSION XN(3), XR(3), XP(3), VDUM(3)
C
DIMENSION POINT(6), TANGENT(3)
DIMENSION VECTOR(3)
C
MAP = MAPSEG(ISEG,EDGE)
C
GO TO (100, 200, 300, 400, 500, 600, 700) MAP
C
C LINEAR EDGE
C-----------------------------------------------
100 DO 110 I=1,3
   POINT(I) = (1. - RATIO)*R1(I,ISEG,EDGE) + RATIO*R2(I,ISEG,EDGE)
C 110 VECOR(I) = R2(I,ISEG,EDGE) - R1(I,ISEG,EDGE)
C CALL VMAG(VECTOR,RMAG)
C IF(RMAG.EQ.0.0) RMAG = 1.0
C DO 120 I=1,3
C 120 TANGENT(I) = (R2(I,ISEG,EDGE) - R1(I,ISEG,EDGE))/RMAG

C-23

LOCKHEED-HUNTSVILLE ENGINEERING CENTER
GO TO 1000

C CIRCULAR ARC
MAP = 2

200 PHI1 = (1.0 - RATIO)*THETA(ISEG, IEDGE)
PHI2 = RATIO*THETA(ISEG, IEDGE)

C DO 210 I=1,3
1 POINT(I) = COEFE(I, ISEG, IEDGE) +
2 (SIN(PHI1)*R1(I, ISEG, IEDGE) + SIN(PHI2)*R2(I, 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))

C GO TO 1000

C CONICS
MAP = 3

300 DARC = RATIO*ARC(ISEG, IEDGE)
IF(COEFE(7, ISEG, IEDGE).GT.0.0) DARC = (1. - RATIO)*ARC(ISEG, IEDGE)
C
C RE = 1/E
C
IF(RE(ISEG, IEDGE).LT.0.9999) GO TO 310
IF(RE(ISEG, IEDGE).GT.1.0001) GO TO 320
GO TO 330

C HYPERBOLA

C CALCULATE ANGLE CORRESPONDING TO ARC LENGTH
C
310 ARG1 = (DARC + ARC1(ISEG, IEDGE))*RA(ISEG, IEDGE)
& /RC(ISEG, IEDGE)
ARG2 = (EXP(ARG1) - 1.0)*(RE(ISEG, IEDGE) + 1.0)
& /RA(ISEG, IEDGE)*(EXP(ARG1) + 1.0))
ALPHA = 2.0*ATAN(ARG2)
C
C DERIVATIVE
C
DEDN = XLENGTH(ISEG, IEDGE)*EXP(ARG1)
&/(1.0 + ARG2**2)*((1.0 - EXP(ARG1))**2)
GO TO 340
C

C ELLIPSE

C CALCULATE ANGLE CORRESPONDING TO ARC LENGTH
C
320 ARG1 = (DARC + ARC1(ISEG, IEDGE))*RA(ISEG, IEDGE)**0.5
& /RC(ISEG, IEDGE)
ARG2 = (RE(ISEG, IEDGE) + 1.0)*TAN(ARG1)/RA(ISEG, IEDGE)
ALPHA = 2.0*ATAN(ARG2)
C
C DERIVATIVE
C
DEDN = XLENGTH(ISEG, IEDGE)*(1.0 + TAN(ARG1)**2)/(1.0 + ARG2**2)
C
IF(COEFE(7, ISEG, IEDGE).GT.0.0) DEDN = -DEDN
C

C-24

LOCKHEED-HUNTSVILLE ENGINEERING CENTER
GO TO 340

C --- PARABOLA ---------------------------------------------------------------
C
C --- CALCUlate ANGLE CORRESPONDING TO ARC LENGTH
C
330 ARG = -(DARC + ARC(ISEG, IEDGE))/RC(ISEG, IEDGE)
ALPHA = 2.0*ATAN(ARG)

C --- DERIVATIVE
C
DEDN = XLENGTH(ISEG, IEDGE)/(1.0 + ARG**2)

C --- MAGNITUDE OF POSITION VECTOR
C
340 R = RC(ISEG, IEDGE)/(RE(ISEG, IEDGE) + COS(ALPHA))

C --- RATIO USING ANGLES
C
ESP = (ALPHA - THETA(ISEG, IEDGE))/THETA(ISEG, IEDGE)

PHI1 = (1.0 - ESP)*THETA(ISEG, IEDGE)
PHI2 = ESP*THETA(ISEG, IEDGE)

C --- POSITION AND TANGENT
C
DO 350 I=1,3
POINT(I) = COEFE(I, ISEG, IEDGE) +
1 R*(SIN(PHI1)*UI(I, ISEG, IEDGE) + SIN(PHI2)*UJ(I, ISEG, IEDGE))
2 /SIN(THETA(ISEG, IEDGE))

TANGENT(I) = (POINT(I) - COEFE(I, ISEG, IEDGE))
1 *THETA(ISEG, IEDGE)*R*SIN(ALPHA)/RC(ISEG, IEDGE)
2 + THETA(ISEG, IEDGE)*R
3*(-COS(PHI1)*UI(I, ISEG, IEDGE) + COS(PHI2)*UJ(I, ISEG, IEDGE))
4 /SIN(THETA(ISEG, IEDGE))

350 TANGENT(I) = TANGENT(I)*DEDN

GO TO 1000

C --- EDGE OF REVOLUTION
C
C --- PROJECTION ALONG AXIS AND RADIUS
C
400 RP = (RPA2(ISEG, IEDGE) - RPA1(ISEG, IEDGE))*RATIO + RPA1(ISEG, IEDGE)
RM = (RM2(ISEG, IEDGE) - RM1(ISEG, IEDGE))*RATIO + RM1(ISEG, IEDGE)

C --- ANGLE
C
GAMMA = THETA(ISEG, IEDGE)*RATIO

C --- CALCULATE THE POSITION AND TANGENT---------------------------------
C
DO 410 I=1,3

UR = COS(GAMMA)*UI(I, ISEG, IEDGE)
6 + SIN(GAMMA)*UJ(I, ISEG, IEDGE)

C-25

LOCKHEED–HUNTSVILLE ENGINEERING CENTER
C

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

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

C

GO TO 1000
C

HGM HOLE (EDGES 1,6,9,5)  MAP = 5

---ANGULAR LOCATION (B)
C

500  
BETAI = COEFE(1,1,1,EDGE)  
BETAF = COEFE(2,1,1,EDGE)
C

DBTA = BETAF - BETAI  
B = BETAI + RATIO*DBTA
C

---CALCULATE HOLE RADIUS (RHO)
C

CALL DELRAD(B,RHO,DRHO,RHOD,RDOC)
C

---CALCULATE AXIAL DISTANCE (CP)
C

CALL CAXIS(B,RHO,CP)
C

---CALCULATE COORDINATES OF POINT ON HOLE
C

CALL HOLE(B,POINT,VDUM)
C

---TANGENT--------------------------------------------
C

CB = COS(B)  
SB = SIN(B)
C

---RADIAL UNIT VECTOR
C

XR(1) = CB*DU2(1) + SB*DU3(1)  
XR(2) = CB*DU2(2) + SB*DU3(2)  
XR(3) = CB*DU2(3) + SB*DU3(3)
C

CALL DERIV(B,RHO,CP,DRDB,DCDB)
C

CALL VADD(DCDB,DU1,DRDB,XR,TANGENT,VDUM)
C

---TANGENT
C

TANGENT(1) = (RHO*(-SB*DU2(1) + CB*DU3(1)) + TANGENT(1))*DBTA  
TANGENT(2) = (RHO*(-SB*DU2(2) + CB*DU3(2)) + TANGENT(2))*DBTA  
TANGENT(3) = (RHO*(-SB*DU2(3) + CB*DU3(3)) + TANGENT(3))*DBTA
C

GO TO 1000
C

---DUCT EXIT PLANE (EDGES 3,7,8,11)  MAP = 6
C

---ANGULAR LOCATION (B)
C

600  
BETAI = COEFE(1,1,1,EDGE)  
BETAF = COEFE(2,1,1,EDGE)
C

C-26

LOCKHEED-HUNTSVILLE ENGINEERING CENTER
DBTA = BETAF - BETAI
B = BETAI + RATIO*DBTA

C---CALCULATE COORDINATES OF POINTS ON THE EXIT PLANE
C
CALL DEXIT(B,POINT,VDUM)
C---TANGENT------------------------------------------
C
CB = COS(B)
SB = SIN(B)
C---DUCT RADIUS (RHOD)
C
EX1 = (AD*SB)**2 + (BD*CB)**2
EX2 = BD**2 - AD**2
C
RHOD = AD*BD/SQRT(EX1)
C
DRDDB = RHOD*SB*CB*EX2/EX1
DCDB = (RHOD*CB + SB*DRDDB)*DFNB/(CQ - CE)
C
CALL VADD(CB,DU2,SB,DU3,XR,VDUM)
CALL VADD(DCDB,DU1,DRDDB,XR,XP,VDUM)
CALL VADD(-SB,DU2,CB,DU3,XN,VDUM)
C---TANGENT
C
TANGENT(1) = (XP(1) + RHOD*XN(1))*DBTA
TANGENT(2) = (XP(2) + RHOD*XN(2))*DBTA
TANGENT(3) = (XP(3) + RHOD*XN(3))*DBTA
C
GO TO 1000

C---WELD (EDGES 2,4,10,12)
C---ANGULAR LOCATION (BB)
C
700 PI2 = 1.57079631
C
IF(RATIO.LT.0.0) BB = COEFE(1,1,IEDGE)
RATIO = ABS(RATIO)
C---WELD RADIUS (RDOC) & DIFFERENCE BETWEEN HOLE AND DUCT RADIUS (DRHO)
C
CALL DELRAD(BB,RHO,DRHO,RHOD,RDOC)
C
C---CALCULATE AXIAL DISTANCE (CP)
C
CALL CAXIS(BB,RHO,CP)
C
Y = RDOC - DRHO
C
AAD = 0.5*((ABOT + ATOP) + (ATOP - ABOT)*COS(BB - PI2))
X = SQRT(DRHO*(2.0*RDOC - DRHO))
C---WELD ANGLE
C
TH2 = ATAN(X/Y)
TH1 = PI2 - TH2
C
XX = AAD - X
C
TH3 = ATAN(XX/RDOC)
C
ANGL = RATIO*(TH2 + TH3) + TH1
C
C---POINT & TANGENT----------------------------------------------
C
IF(ANGL.GE.PI2) THEN
C
C---DUCT---------------------------------------------------------
C
PSI = (ANGL - PI2)/TH3
C
C---AXIAL DISTANCE
CS = CP + X + PSI*XX
C
C---RADIUS
RS = RHOD
C
C---COORDINATES
CALL DUCT(BB,RS,CS,POINT)
C
C---TANGENT
TANGENT(1) = DU1(1)
TANGENT(2) = DU1(2)
TANGENT(3) = DU1(3)
C
RETURN
C

C---WELD----------------------------------------------------------
C
ELSE
C
XS = RDOC*COS(ANGL)
YS = RDOC*SIN(ANGL)
C
C---AXIAL DISTANCE
CS = CP + X - XS
C
C---RADIUS
RS = RHOD + RDOC - YS
C
C---COORDINATES
CALL DUCT(BB,RS,CS,POINT)
C
C---TANGENT
DXDN = TAN(ANGL)
C
CB = COS(BB)
SB = SIN(BB)
C
CALL VADD(CB,DU2,SB,DU3,VDUM,TANGENT)
C
CALL VADD(DXDN,DU1,-1.0,VDUM,C,TANGENT)
C
RETURN
C

END IF
C

C
1000 CONTINUE
C
RETURN
C

C-28

LOCKHEED-HUNTSVILLE ENGINEERING CENTER
1100 DO 1110 I=4,6
1110 POINT(I) = PT1(I,ISEG,EDGE)
C
      RETURN
      END
C
**************************************************************************************
C**** MAPPING **************************************************************************************
C**************************************************************************************
C
SUBROUTINE SURFACE(INIT,ISIDE)
C
      THIS ROUTINE DETERMINES THE COORDINATES AND NORMAL OF A
C POINT ON A SURFACE.
C
      COMMON / COEFF/     COEFE(8,10,12),COEPS(8,6),NMBRSEG(12)
      COMMON /COORD/     UA(3,6),UE(3,6),UN(3,6)
      COMMON /COUNTER/   NODESAV,NODETOT,NNODES,NPLANE
      COMMON /HEADER/     ITITLE(20),LINE
      COMMON /INITA/      IZINDEX,MAPTN,INCHES
      COMMON /INITB/      IBOX(4,6),IRAY(4,3),NRAY,IP1(12),IP2(12)
      COMMON /INICT/      PI,RADDEG
      COMMON /INPUTA/     EDGE(3,12),POINT(3,8),SIDE(3,6)
      COMMON /MARCHS/     MARCH,INDEX(3)
      COMMON /MAPED/      KSEG(12),UAXIS(3,6),IEDGE1(6),IEDGE2(6),GAMMA
      COMMON /MAPPING/    MAPSIDE(6),MAPSEG(10,12)
      COMMON /MAXIMUM/    ETAMAX(10,12),SEGMAX(3,10,12)
      COMMON /PARTIAL/    DEDN(3,12),DSN(3,2),SNORMAL(3,6)
      COMMON /SPACING/    ISTRITCH(3),STRETCH(3),ETAS(3,200),ETA(3),DETA(3)
      COMMON /UNITS/      NUS,NU6,NU20
      COMMON /ZONING/     IZONE,ISELECT,NINDEX,NMBRNSD(3)
C
      COMMON /DFN1/      DU1(3),DU2(3),DU3(3),DFNB,DFND,DFNF
      COMMON /DFN2/      AE,BE,DFNR,ISTAR,AGL,BETA1,BETA2,BETA3,BETA4
      COMMON /DFN3/      AD,BD,BETA(37),DELROH(37),RAMOC(37)
      COMMON /DFN4/      IHOLE,PL(6,6,4),PA(2,6,4)
      COMMON /DFN5/      CE,CQ,ABOT,ATOP
C
DIMENSION RAM(3),SAC(3)
DIMENSION R1(6),R2(6),R3(6),R4(6)
DIMENSION E1(6),E2(6),E3(6),E4(6)
C
DIMENSION F(8)
DIMENSION NMBRPT(4,6),SPOINT(3,4,6),SEDGE(3,4),SWEEP(3)
DIMENSION UNORMAL(3,6),URADIAL(3,6)
DIMENSION ZERO(3),VECTOR(3)
DIMENSION IETA1(6),IETA2(6),SIGNS(6)
DIMENSION UNITRAC(3)
C
      DATA IETA1 /1,1,1,1,1,2,2/
      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/
C
      NODNUM = NODESAV + NODETOT + 1
C
--- ETA COEFFICIENTS FOR BI-LINEAR INTERPOLATION
C
      F(1) = 1.0 - ETA(IETA1(ISIDE))
      F(2) = ETA(IETA2(ISIDE))
C
C-29
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
C
LINE1 = IBOX(1,ISIDE)
LINE2 = IBOX(2,ISIDE)
LINE3 = IBOX(3,ISIDE)
LINE4 = IBOX(4,ISIDE)

C --- DETERMINE THE CORNER POINTS OF THE SURFACE
C
LPT1 = IPT1(LINE1)
LPT2 = IPT1(LINE3)
LPT3 = IPT2(LINE3)
LPT4 = IPT2(LINE1)

MAP = MAPSIDE(ISIDE)

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

C --- FLAT OR CYLINDRICAL SURFACE
C MAP = 1 OR 2

C --- CALCULATE POSITION
C
100 DO 110 J=1,3
C
SIDE(J,ISIDE) = F(1)*EDGE(J,LINE1) - F(5)*POINT(J,LPT1)
1 + F(2)*EDGE(J,LINE3) - F(6)*POINT(J,LPT4)
2 + F(3)*EDGE(J,LINE2) - F(7)*POINT(J,LPT2)
3 + F(4)*EDGE(J,LINE4) - F(8)*POINT(J,LPT3)

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

C 110 DSDN(J,2) = EDGE(J,LINE4) + F(1)*POINT(J,LPT1)
1 - EDGE(J,LINE2) + F(2)*POINT(J,LPT4)
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),41)
C

GO TO 1000

C --- SPECIAL SURFACE
C MAP = 3

300 GO TO 1000

C --- EDGE OF REVOLUTION
C MAP = 4

400 IF(INIT.EQ.1) THEN
C
C --- ETA DIRECTION OF REVOLUTION
C
IONETWO = 1
IF(COEFS(7,ISIDE).EQ.IETA2(ISIDE)) IONETWO = 2
C
C---EDGE 1
C
IEDGEL(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,VECTOR,UA(1,ISIDE))
C
C---UN: NORMALIZED VECTOR FROM AXIS TOWARD SURFACE
C
CALL VADD(1.0,EDGE(1,IEDGEL(ISIDE)),-1.0,COEFS(1,ISIDE),E1,VECTOR)
C
CALL CROSS(UA(1,ISIDE),E1,UN(1,ISIDE),42)
C
C---UE: NORMALIZED VECTOR FROM AXIS TO FIRST EDGE
C
CALL CROSS(UN(1,ISIDE),UA(1,ISIDE),UE(1,ISIDE),43)
C
END IF
C
C---R1: VECTOR FROM AXIS TO FIRST EDGE-----------------------------
C
CALL VADD(1.0,EDGE(1,IEDGEL(ISIDE)),-1.0,COEFS(1,ISIDE),E1,VECTOR)
C
CALL VDOT(E1,UA(1,ISIDE),EPA1)
C
CALL VADD(1.0,E1,-EPA1,UA(1,ISIDE),R1,VECTOR)
C
CALL VMAG(R1,RM1)
C
C---DERIVATIVE
C
CALL VDOT(DEDN(1,IEDGEL(ISIDE)),UA(1,ISIDE),DA1)
C
CALL VDOT(DEDN(1,IEDGEL(ISIDE)),VECTOR,DR1)
C
CALL VMAG(DEDN(1,IEDGEL(ISIDE)),DM1)
C
C---R2: VECTOR FROM AXIS TO SECOND EDGE-----------------------------
C
CALL VADD(1.0,EDGE(1,IEDGE2(ISIDE)),-1.0,COEFS(1,ISIDE),E2,VECTOR)
C
CALL VDOT(E2,UA(1,ISIDE),EPA2)
C
CALL VADD(1.0,E2,-EPA2,UA(1,ISIDE),R2,VECTOR)
C
CALL VMAG(R2,RM2)
C
C---DERIVATIVE
C
CALL VDOT(DEDN(1,IEDGE2(ISIDE)),UA(1,ISIDE),DA2)
C
CALL VDOT(DEDN(1,IEDGE2(ISIDE)),VECTOR,DR2)
CALL VMAG(DEDN(1, IEDGE2(ISIDE)), DM2)

C---SWEEP ANGLE-----------------------------------------------

CALL VDOT(R1, R2, R1DOTT2)

ARG = R1DOTT2/(RM1*RM2)

IF(ABS(ARG).GT.1.0) ARG = ARG/ABS(ARG)

THETA = ACOS(ARG)

N = COEFS(7, ISIDE)

C---POSITION

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

C---DERIVATIVE

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

C---ANGLE

GAMMA = THETA*ETA(N)

C---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
DSDN(I, 1) = (DR*UR + DA*UA(I, ISIDE))/DM

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

C---ORIENT SURFACE NORMAL DEPENDING ON DIRECTION OF EDGE REVOLUTION

IF(COEFS(7, ISIDE).EQ.IETA2(ISIDE)) THEN

CALL CROSS(DSDN(1, 1), DSDN(1, 2), SNORMAL(1, ISIDE), 44)

ELSE

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

END IF

GO TO 1000

C---BOWL SURFACE WITH HOLE

MAP = 5

500 IHOLE = 0
C IF(IET1.EQ.1) THEN
  IET1 = 1
  IF(ETA(1).GE.ETAMAX(1,3)) IET1 = 2
  IF(ETA(1).GE.ETAMAX(2,3)) IET1 = 3
  IF(ETA(1).GE.ETAMAX(3,3)) IET1 = 4
  IF(ETA(1).GE.ETAMAX(4,3)) IET1 = 5
C
C STR11 = 2.0
C STR12 = 6.0
C STR13 = 2.0
C
C EPS1 = ETA(1)/ETAMAX(1,3)
C IF(IET1.EQ.2 .OR. IET1.EQ.3) THEN
  RATIO1 = (ETA(1) - ETAMAX(1,3)) / (ETAMAX(3,3) - ETAMAX(1,3))
  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
  END IF
C
C IF(IET1.EQ.4) THEN
  RATIO1 = (ETA(1) - ETAMAX(3,3)) / (ETAMAX(4,3) - ETAMAX(3,3))
  X1 = STR12/2.0/2.0
  ETAMID = (EXP(X1) - EXP(-X1))/(EXP(X1) + EXP(-X1))
  X2 = (RATIO1 - 0.5)*STR12/2.0
  ETAMX = (EXP(X2) - EXP(-X2))/(EXP(X2) + EXP(-X2))
  EPS1 = (ETAMX + ETAMID)/(2.0*ETAMID)
  END IF
C
C IF(IET1.EQ.5) THEN
  RATIO1 = (ETA(1) - ETAMAX(4,3)) / (1.0 - ETAMAX(4,3))
  X1 = 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)
  END IF
C
C IET3 = 1
  IF(ETA(3).GE.ETAMAX(1,8)) IET3 = 2
  IF(ETA(3).GT.ETAMAX(2,8)) IET3 = 3
C STR31 = 2.0
C
C-33

LOCKHEED-HUNTSVILLE ENGINEERING CENTER
STR32 = 4.0
STR33 = 2.0

IF(IET3.EQ.1) THEN
  RATIO3 = ETA(3)/ETAMAX(1,8)
  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
END IF

IF(IET3.EQ.2) THEN
  RATIO3 = (ETA(3) - ETAMAX(1,8))/(ETAMAX(2,8) - ETAMAX(1,8))
  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)
END IF

IF(IET3.EQ.3) THEN
  RATIO3 = 1.0 - (ETA(3) - ETAMAX(1,8))/(1.0 - ETAMAX(2,8))
  X1 = RATIO3*STR33/2.0
  ETAX1 = (EXP(X1) - EXP(-X1))/(EXP(X1) + EXP(-X1))
  X2 = STR33/2.0
  ETAX2 = (EXP(X2) - EXP(-X2))/(EXP(X2) + EXP(-X2))
  EPS3 = 1.0 - ETAX1/ETAX2
END IF

C--BOWL AXIS
UAXIS(1,ISIDE) = 1.0
UAXIS(2,ISIDE) = 0.0
UAXIS(3,ISIDE) = 0.0

C--ARC CENTER ON AXIS
SAC(2) = 0.0
SAC(3) = 0.0

DO 510 J=1,6

C--SUB-SURFACE CORNER POINTS

C-34

LOCKHEED-HUNTSVILLE ENGINEERING CENTER
R1(J) = PL(J, IET1, IET3)
R2(J) = PL(J, IET1+1, IET3)
R3(J) = PL(J, IET1+1, IET3+1)
R4(J) = PL(J, IET1, IET3+1)

C---SUB-SURFACE EDGE POINTS
C
E1(J) = R1(J) + EPS1*(R2(J) - R1(J))
E2(J) = R2(J) + EPS3*(R3(J) - R2(J))
E3(J) = R4(J) + EPS1*(R3(J) - R4(J))
E4(J) = R1(J) + EPS3*(R4(J) - R1(J))

GO TO (520, 540, 560) IET3

IET3 = 1

GO TO (522, 524, 528, 530) IET1

IET1 = 1

DO 522 J=1,6
   E1(J) = EDGE(J, 3)
   E4(J) = EDGE(J, 8)

   SAC(1) = SEGMAX(1, 1, 3)

   CALL SPEDGE(R2, R3, EPS3, SAC, E2)

GO TO 580

IET1 = 2

DO 524
R2(J) = PL(J, 1, IET3)
R3(J) = PL(J, 1, IET3+1)

E2(J) = R2(J) + EPS3*(R3(J) - R2(J))

EXM = SEGMAX(1, 1, 3) + EPX1*(E2(J) - SEGMAX(1, 1, 3))

IF(EXM.GT.SEGMAX(1, 2, 3)) GO TO 526

---INTERIOR
C
EPS1 = (EXM -SEGMAX(1, 1, 3))/(SEGMAX(1, 2, 3) -SEGMAX(1, 1, 3))

C---SUB-SURFACE EDGE POINTS
C
IET1 = 2

DO 525 J=1,6

R1(J) = PL(J, IET1, IET3)
R2(J) = PL(J, IET1+1, IET3)
R3(J) = PL(J, IET1+1, IET3+1)
R4(J) = PL(J, IET1, IET3+1)

E1(J) = EDGE(J, 3)
E3(J) = R4(J) + EPS1*(R3(J) - R4(J))

SAC(1) = SEGMAX(1, 2, 3)

C-35
CALL SPEDGE(R2, R3, EPS3, SAC, E2)

SAC(1) = SEGMAX(1,1,3)

CALL SPEDGE(R1, R4, EPS3, SAC, E4)

GO TO 580

---IET1 = 3-----------------------------------------------

526 EPS1 = (EXM - SEGMAX(1,2,3))/(E2(1) - SEGMAX(1,2,3))

---SUB-SURFACE EDGE POINTS

IET1 = 3

DO 527 J=1,6

R1(J) = PL(J, IET1, IET3)
R2(J) = PL(J, IET1+1, IET3)
R3(J) = PL(J, IET1+1, IET3+1)
R4(J) = PL(J, IET1, IET3+1)

E1(J) = EDGE(J, 3)
E3(J) = R4(J) + EPS1*(R3(J) - R4(J))

SAC(1) = SEGMAX(1,3,3)

CALL SPEDGE(R2, R3, EPS3, SAC, E2)

SAC(1) = SEGMAX(1,2,3)

CALL SPEDGE(R1, R4, EPS3, SAC, E4)

GO TO 580

---IET1 = 4-----------------------------------------------

528 DO 529 J=1,6

529 E1(J) = EDGE(J, 3)

SAC(1) = SEGMAX(1,4,3)

CALL SPEDGE(R2, R3, EPS3, SAC, E2)

SAC(1) = SEGMAX(1,3,3)

CALL SPEDGE(R1, R4, EPS3, SAC, E4)

B = BETA1 + EPS1*(BETA2 - BETA1)

CALL HOLE(B, E3, VECTOR)

GO TO 580

---IET1 = 5-----------------------------------------------

530 DO 531 J=1,6
C E1(J) = EDGE(J,3)
      E2(J) = EDGE(J,7)
C
      SAC(1) = SEGMAX(1,4,3)
C
      CALL SPEDGE(R1,R4,EPS3,SAC,E4)
C
      GO TO 580
C------------------------------------------------------------------------
C IET3 = 2
C------------------------------------------------------------------------
C 540 GO TO (542,544,544,548,550) IET1
C---IET1 = 1-------------------------------------------------------------------
C 542 DO 543 J=1,6
C 543      E4(J) = EDGE(J,8)
C      SAC(1) = SEGMAX(1,1,3)
C
      CALL SPEDGE(R2,R3,EPS3,SAC,E2)
C
      GO TO 580
C---IET1 = 2-------------------------------------------------------------------
C 544      B = BETA1 + EPS3*(BETA4 - BETA1)
C
      CALL HOLE(B,E2,VECTOR)
C
      EXM = SEGMAX(1,1,3) + EPX1*(E2(1) - SEGMAX(1,1,3))
C
      IF(EXM.GT.SEGMAX(1,2,3)) GO TO 546
C---INTERIOR
C
      EPS1 = (EXM -SEGMAX(1,1,3))/(SEGMAX(1,2,3) -SEGMAX(1,1,3))
C---SUB-SURFACE EDGE POINTS
C
      IET1 = 2
C
      DO 545 J=1,6
C
      R1(J) = PL(J,IET1 ,IET3)
      R2(J) = PL(J,IET1+1,IET3)
      R3(J) = PL(J,IET1+1,IET3+1)
      R4(J) = PL(J,IET1 ,IET3+1)
C
      E1(J) = R1(J) + EPS1*(R2(J) - R1(J))
      E3(J) = R4(J) + EPS1*(R3(J) - R4(J))
C 545
C
      SAC(1) = SEGMAX(1,2,3)
C
      CALL SPEDGE(R2,R3,EPS3,SAC,E2)
C
      SAC(1) = SEGMAX(1,1,3)

C-37
LOCKEED-HUNTSVILLE ENGINEERING CENTER
CALL SPEDGE(R1,R4,EP3,SAC,E4)
GO TO 580

---IET1 = 3-----------------------------------------------

546 EPS1 = (EXM - SEGMAX(1,2,3))/(E2(1) - SEGMAX(1,2,3))

---SUB-SURFACE EDGE POINTS

IET1 = 3
DO 547 J=1,6

R1(J) = PL(J,IET1,IET3)
R2(J) = PL(J,IET1+1,IET3)
R3(J) = PL(J,IET1+1,IET3+1)
R4(J) = PL(J,IET1,IET3+1)

E1(J) = R1(J) + EPS1*(R2(J) - R1(J))
E3(J) = R4(J) + EPS1*(R3(J) - R4(J))
SAC(1) = SEGMAX(1,2,3)

CALL SPEDGE(R1,R4,EP3,SAC,E4)
B = BET1 + EPS3*(BETA4 - BET1)

CALL HOLE(B,E2,VECTOR)

GO TO 580

---IET1 = 4-----------------------------------------------

548 IHOLE = 1
B = BET1 + EPS1*(BETA2 - BET1)

CALL HOLE(B,E1,VECTOR)
B = BET2 + EPS3*(BETA2 - BET2 - 2.*PI)

CALL HOLE(B,E2,VECTOR)
B = BET4 + EPS1*(BETA3 - BET4)

CALL HOLE(B,E3,VECTOR)
B = BET1 + EPS3*(BETA4 - BET1)

CALL HOLE(B,E4,VECTOR)

GO TO 580

---IET1 = 5-----------------------------------------------

550 DO 551 J=1,6

551 E2(J) = EDGE(J,7)

C-2

LOCKHEED-HUNTSVILLE ENGINEERING CENTER
B = BETA2 + EPS3*(BETA3 - BETA2 - 2.*PI)

CALL HOLE(B,E4,VECTOR)

GO TO 580

IET3 = 3

560 GO TO (562,564,564,568,570) IET1

IET1 = 1

562 DO 563 J=1,6

E3(J) = EDGE(J,11)
E4(J) = EDGE(J,8)

SAC(1) = SEGMAX(1,1,3)

CALL SPEDGE(R2,R3,EPS3,SAC,E2)

GO TO 580

IET1 = 2

564 R2(1) = PL(1,4,IET3)
R3(1) = PL(1,4,IET3+1)

E2(1) = R2(1) + EPS3*(R3(1) - R2(1))

EXM = SEGMAX(1,1,3) + EPX1*(E2(1) - SEGMAX(1,1,3))

IF(EXM.GT.SEGMAX(1,2,3)) GO TO 566

IET1 = 2

DO 565 J=1,6

R1(J) = PL(J,IET1 ,IET3)
R2(J) = PL(J,IET1+1,IET3)
R3(J) = PL(J,IET1+1,IET3+1)
R4(J) = PL(J,IET1 ,IET3+1)

E1(J) = R1(J) + EPS1*(R2(J) - R1(J))
E3(J) = EDGE(J,11)

565 SAC(1) = SEGMAX(1,2,3)

CALL SPEDGE(R2,R3,EPS3,SAC,E2)

SAC(1) = SEGMAX(1,1,3)

CALL SPEDGE(R1,R4,EPS3,SAC,E4)

GO TO 580
C---IET1 = 3--------------------------------------------------------
C
566  EPS1 = (EXM - SEGMAX(1,2,3))/(E2(1) - SEGMAX(1,2,3))
C
      IET1 = 3
C
DO 567 J=1,6
C
R1(J) = PL(J,IET1,IET3)
R2(J) = PL(J,IET1+1,IET3)
R3(J) = PL(J,IET1+1,IET3+1)
R4(J) = PL(J,IET1,IET3+1)
C
E1(J) = R1(J) + EPS1*(R2(J) - R1(J))
E2(J) = R2(J) + EPS1*(R3(J) - R2(J))
567  E3(J) = EDGE(J,11)
C
SAC(1) = SEGMAX(1,2,3)
C
CALL SPEDGE(R1,R4,EPS3,SAC,E4)
C
GO TO 580
C---IET1 = 4--------------------------------------------------------
C
DO 569 J=1,6
C
569  E3(J) = R4(J) + EPS1*(R3(J) - R4(J))
C
      B = BETA4 + EPS1*(BETA3 - BETA4)
C
CALL HOLE(B,E1,VECTOR)
C
GO TO 580
C---IET1 = 5--------------------------------------------------------
C
DO 571 J=1,6
C
571  E3(J) = EDGE(J,11)
      E2(J) = EDGE(J,7)
C
INTERPOLATION
C
POSITION
C
Do 582 J=1,3
C
582  RAM(J) = (1. - EPS1)*(E4(J) - (1. - EPS3)*R1(J) - EPS3*R4(J))
      + EPS1*(E2(J) - (1. - EPS3)*R2(J) - EPS3*R3(J)) + E1(J)
      + EPS3*(E3(J) - E1(J))
C
RADIUS AND TANGENT
C
AXIAL DISTANCE
C
RAX = RAM(1)
C
C-40

LOCKHEED-HUNTSVILLE ENGINEERING CENTER
C---REGION 1---------------------------------------------
C
IF(RAX.GT.SEGMAX(1,1,3)) GO TO 586
C---RADIUS
C
PSI = RAX/SEGMAX(1,1,3)
C
RAD1 = SQRT(POINT(2,4)**2 + POINT(3,4)**2)
RAD2 = SQRT(SEGMAX(2,1,3)**2 + SEGMAX(3,1,3)**2)
C
RAD = RAD1 + PSI*(RAD2 - RAD1)
C---TANGENT
C
DSDN(1,1) = SEGMAX(1,1,3) - POINT(1,4)
DSDN(2,1) = RAD2 - RAD1
DSDN(3,1) = 0.
C
GO TO 590
C---REGION 2---------------------------------------------
C
586 IF(RAX.GT.SEGMAX(1,2,3)) GO TO 588
C---RADIUS
C
PSI = (RAX - SEGMAX(1,1,3))/(SEGMAX(1,2,3) - SEGMAX(1,1,3))
C
RAD1 = SQRT(SEGMAX(2,1,3)**2 + SEGMAX(3,1,3)**2)
RAD2 = SQRT(SEGMAX(2,2,3)**2 + SEGMAX(3,2,3)**2)
C
RAD = RAD1 + PSI*(RAD2 - RAD1)
C---TANGENT
C
DSDN(1,1) = SEGMAX(1,2,3) - SEGMAX(1,1,3)
DSDN(2,1) = RAD2 - RAD1
DSDN(3,1) = 0.
C
GO TO 590
C---REGION 3---------------------------------------------
C
C---ELLIPITIC REGION OF BOWL
C---RADIUS
C
588 RAD = DFNR + BE*SQRT(1.0 - ((RAX - DFND)/AE)**2)
C---TANGENT
C
DRDX = -((BE/AE)**2)*(RAX - DFND)/(RAD - DFNR)
C
TH = ATAN(DRDX)
C
DSDN(1,1) = COS(TH)
DSDN(2,1) = SIN(TH)
DSDN(3,1) = 0.0
IF(IET3.EQ.2 .AND. IET1.EQ.4) THEN
I = 0
THETA = ATAN( RAM(2) / RAM(3) )
DO 587 J=1,36
CALL HOLE(BETA(J),E1,VECTOR)
THETA1 = ATAN( E1(2) / E1(3) )
CALL HOLE(BETA(J+1),E2,VECTOR)
THETA2 = ATAN( E2(2) / E2(3) )
ENDIF

IF(THETA.GT.THETA1 .AND. THETA.LT.THETA2) MATCH = 1
IF(THETA.GT.THETA2 .AND. THETA.LT.THETA1) MATCH = 2
IF(MATCH.GT.0) THEN
I = I + 1
ENDIF

RATIO = (THETA - THETA1) / (THETA2 - THETA1)
B = RATIO*(BETA(J+1) - BETA(J)) + BETA(J)
ELSE
RATIO = (THETA - THETA2) / (THETA1 - THETA2)
B = RATIO*(BETA(J) - BETA(J+1)) + BETA(J+1)
ENDIF

IF(I.EQ.1) THEN
CALL HOLE(B,E3,VECTOR)
RAD1 = SQRT( E3(2)**2 + E3(3)**2)
GO TO 587
ELSE
CALL HOLE(B,E4,VECTOR)
RAD2 = SQRT( E4(2)**2 + E4(3)**2)
GO TO 589
ENDIF

587 CONTINUE
589 CONTINUE
IF(E3(L).LT.E4(L)) THEN
  RATIO = (RAM(1) - E3(L)) / (E4(L) - E3(L))
  RAD = RATIO*(RAD2 - RAD1) + RAD1
ELSE
  RATIO = (RAM(1) - E4(L)) / (E3(L) - E4(L))
  RAD = RATIO*(RAD1 - RAD2) + RAD2
END IF

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

C---BELOW HOLE
C
IF(IET3.EQ.3 .AND. RAX.LE.SEGMAX(1,4,11)) THEN
  RAD = SQRT(RAM(2)**2 + RAM(3)**2)
  DSDN(1,L) = 1.0
  DSDN(2,L) = 0.0
  DSDN(3,L) = 0.0
END IF

C-------------------------------------------------------
C OUTPUT POSITION AND NORMAL
C-------------------------------------------------------
C---AXIAL COMPONENT OF THE TANGENT
C
590 CALL VDOT(DSDN(1,L),UAXIS(L,ISIDE),DA)
C
C---NORMAL COMPONENT OF THE TANGENT
C
  DN = DSDN(2,L)
C
C---INTERPOLATED RADIUS
C
  RADX = SQRT(RAM(2)**2 + RAM(3)**2)
C
C---ANGULAR LOCATION
C
  ANG = ASIN(RAM(3)/RADX)
C
  CANG = COS(ANG)
  SANG = SIN(ANG)
C
C---TANGENT 1 (AXIAL DIRECTION)
C
  DSDN(1,L) = DA
  DSDN(2,L) = DN*CANG
  DSDN(3,L) = DN*SANG
C
C---TANGENT 2 (CIRCUMFERENTIAL DIRECTION)
C
C-43
C
DSDN(1,2) = 0.0
DSDN(2,2) = -SANG
DSDN(3,2) = CANG
C---NORMAL
C
CALL CROSS(DSDN(1,1),DSDN(1,2),SNORMAL(1,ISIDE),45)
C---POSITION
C
SIDE(1,ISIDE) = RAX
SIDE(2,ISIDE) = RAD*CANG
SIDE(3,ISIDE) = RAD*SANG
C
IF(EPS3.EQ.1.0) THEN
C
SIDE(1,ISIDE) = E3(1)
SIDE(2,ISIDE) = E3(2)
SIDE(3,ISIDE) = E3(3)
C
END IF
C
GO TO 1000
C
MATED DUCT SURFACE
C
600
IHOLE = 1
C
EPS1 = ETA(1)
EPS3 = ETA(3)
C---SUB-SURFACE CORNER POINTS
C
DO 610 J=1,6
C
R1(J) = PL(J,4,2)
R2(J) = PL(J,5,2)
R3(J) = PL(J,5,3)
R4(J) = PL(J,4,3)
610
C---SUB-SURFACE EDGES
C
B = BETA1 + EPS1*(BETA2 - BETA1)
C
CALL HOLE(B,E1,VECTOR)
C
B = BETA2 + EPS3*(BETA3 - BETA2 - 2.*PI)
C
CALL HOLE(B,E2,VECTOR)
C
B = BETA4 + EPS1*(BETA3 - BETA4)
C
CALL HOLE(B,E3,VECTOR)
C
B = BETA1 + EPS3*(BETA4 - BETA1)
C
CALL HOLE(B,E4,VECTOR)
C---INTERPOLATION
C
C-44

LOCKHEED-HUNTSVILLE ENGINEERING CENTER
IF(EPS1.EQ.0. .OR.EPS1.EQ.1. .OR.EPS3.EQ.0. .OR.EPS3.EQ.1.) THEN
RAD  =  DFNR  +  BE*SQRT(1.0 - ((RAX - DFND)/AE)**2)
ELSE
RAD  =  SQRT(RAM(2)**2 + RAM(3)**2)

DO 640 5=1,36
CALL HOLE(BETA(J),EZ,VECTOR)
THETA1 = ATAN( E1(2) / E1(3) )
CALL HOLE(BETA(J+1),EZ,VECTOR)
THETA2 = ATAN( E2(2) / E2(3) )
IF(THETA.GT.THETA1 .AND. THETA.LT.THETA2) MATCH = 1
IF(THETA.GT.THETA2 .AND. THETA.LT.THETA1) MATCH = 2
IF(MATCH.GT.0) THEN
  I = I + 1
IF(MATCH.EQ.1) THEN
  RATIO = (THETA - THETA1) / (THETA2 - THETA1)
  B = RATIO*(BETA(J+1) - BETA(J)) + BETA(J)
ELSE
  RATIO = (THETA - THETA2) / (THETA1 - THETA2)
  B = RATIO*(BETA(J) - BETA(J+1)) + BETA(J+1)
END IF
IF(I.EQ.1) THEN
  CALL HOLE(B,E3,VECTOR)
C
RAD1 = SQRT( E3(2)**2 + E3(3)**2)
GO TO 640
ELSE
CALL HOLE(B3,E4,VECTOR)
RAD2 = SQRT( E4(2)**2 + E4(3)**2)
GO TO 650
END IF
END IF
640 CONTINUE
650 CONTINUE
C
IF(E3(1).LT.E4(1)) THEN
RATIO = (RAM(1) - E3(1)) / (E4(1) - E3(1))
RAD = RATIO*(RAD2 - RAD1) + RAD1
ELSE
RATIO = (RAM(1) - E4(1)) / (E3(1) - E4(1))
RAD = RATIO*(RAD1 - RAD2) + RAD2
END IF
END IF
C
C---TANGENT
C
DRDX = -((BE/AE)**2)*(RAX - DFND)/(RAD - DFNR)
TH = ATAN(DRDX)
C
DSDN(1,1) = COS(TH)
DSDN(2,1) = SIN(TH)
DSDN(3,1) = 0.0
C
C OUTPUT POSITION AND NORMAL
C
C---AXIAL COMPONENT OF TANGENT
C
DA = DSDN(1,1)
C
C---NORMAL COMPONENT OF TANGENT
C
DN = DSDN(2,1)
C
C---ANGULAR LOCATION
C
RADX = SQRT(RAM(2)**2 + RAM(3)**2)
ANG = ASIN(RAM(3)/RADX)
C
CANG = COS(ANG)
SANG = SIN(ANG)
C---TANGENT 1 (AXIAL DIRECTION)
C
DSDN(1,1) = DA
DSDN(2,1) = DN*CANG
DSDN(3,1) = DN*SANG
C
C---TANGENT 2 (CIRCUMFERENTIAL DIRECTION)
C
DSDN(1,2) = 0.0
DSDN(2,2) = -SANG
DSDN(3,2) = CANG
C
C---NORMAL
C
CALL CROSS(DSDN(1,1),DSDN(1,2),SNORMAL(1,ISIDE),46)
C
C---POSITION
C
SIDE(1,ISIDE) = RAX
SIDE(2,ISIDE) = RAD*CANG
SIDE(3,ISIDE) = RAD*SANG
C
IF(EPS3.EQ.1.0) THEN
C
SIDE(1,ISIDE) = E3(1)
SIDE(2,ISIDE) = E3(2)
SIDE(3,ISIDE) = E3(3)
C
END IF
C
C GO TO 1000
C
700 IF(ISIDE.LT.4) THEN
C
IEDG2 = LINE4
EPS1 = ETA(1)
EPS2 = ETA(2)
C
ELSE
C
IEDG2 = LINE3
EPS1 = ETA(3)
EPS2 = ETA(2)
C
END IF
C
C---WELD REGION-------------------
C
PI2 = 1.57079633
C
B = COEFS(1,ISIDE) + EPS1*(COEFS(2,ISIDE) - COEFS(1,ISIDE))
C
EPS3 = (COEFS(3,ISIDE) - 1.0)/(NMBRND(2) - 1.0)
C  RATIO = EPS2/EPS3
     IF(RATIO.GT.1.0) RATIO = 1.0
C
C----WELD RADIUS (RDOC) & DIFFERENCE BETWEEN HOLE AND DUCT RADIUS (DRHO)
C
CALL DELRAD(B,RHO,DRHO,RHOD,RDOC)
C
C----CALCULATE AXIAL DISTANCE (CP)
C
CALL CAXIS(B,RHO,CP)
C
     Y = RDOC - DRHO
C
     AAD = 0.5*((ABOT + ATOP) + (ATOP - ABOT)*COS(B - PI2))
     X = SQRT(DRHO*(2.0*RDOC - DRHO))
C
C----WELD ANGLE
C
     TH2 = ATAN(X/Y)
     TH1 = PI2 - TH2
C
     XX = AAD - X
C
     TH3 = ATAN(XX/RDOC)
C
     ANGL = RATIO*(TH2 + TH3) + TH1
C
C----POSITION AND TANGENT 1 (AXIAL DIRECTION)---------------------------
C
     IF(ANGL.GE.PI2) THEN
C
C----DUCT----------------------------
     PSI = (ANGL - PI2)/TH3
C
C----AXIAL DISTANCE
     CS = CP + X + PSI*XX
C
C----RADIUS
     RS = RHOD
C
C----COORDINATES
     CALL DUCT(B,RS,CS,SIDE(1,ISIDE))
C
C----TANGENT
     DSDN(1,1) = DU1(1)
     DSDN(2,1) = DU1(2)
     DSDN(3,1) = DU1(3)
C
C----WELD-----------------------------
     ELSE
C
     XS = RDOC*COS(ANGL)
     YS = RDOC*SIN(ANGL)
C
C----AXIAL DISTANCE
     CS = CP + X - XS
C
C----RADIUS
     RS = RHOD + RDOC - YS
C
C----COORDINATES
     CALL DUCT(B,RS,CS,SIDE(1,ISIDE))
C
C----TANGENT
     DXDN = TAN(ANGL)
C
     CB = COS(B)
     SB = SIN(B)
CALL VADD(CB, DU2, SB, DU3, VECTOR, DSDN(1,1))

ONE = 1.0

CALL VADD(DXDN, DU1, -ONE, VECTOR, C, DSDN(1,1))

END IF

--DUCT REGION-----------------------------------------------

IF (RATIO.EQ.1.0) THEN

---POSITION

PSI = (EPS2 - EPS3)/(1. - EPS3)

PSII = 1.0 - PSI

CALL VADD(PSII, SIDE(1, ISIDE),
          PSI, EDGE(1, IEDG2), SIDE(1, ISIDE), VECTOR)

---TANGENT 1 (AXIAL DIRECTION) : INPUT

DSDN(1,1) = DU1(1)
DSDN(2,1) = DU1(2)
DSDN(3,1) = DU1(3)

END IF

---TANGENT 2 (CIRCUMFERENTIAL DIRECTION)-------------------

CB = COS(B)
SB = SIN(B)

CALL VADD(CB, DU2, SB, DU3, VECTOR, R1)

CALL CROSS(DSDN(1,1), VECTOR, DSDN(1,2), 47)

---NORMAL

CALL CROSS(DSDN(1,1), DSDN(1,2), SNORMAL(1, ISIDE), 48)

---DIRECT SURFACE NORMAL INTO FLOW DOMAIN

1000 DO 1010 I=1,3
1010 SNORMAL(I, ISIDE) = SNORMAL(I, ISIDE)*SIGNS(ISIDE)

RETURN

---FORMAT STATEMENTS

1100 FORMAT(1H1, 10X, 20A4, 13X, 8H SECTION, I2, 3H OF, I3, 9H FOR ZONE, I3)

END

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

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

SUBROUTINE BLKOUT(NUNIT, NODSTOR)
WRITES THE FORMATTED BLOCKED GEOMETRY FILE (NUNIT)

COMMON /INITA/  IZINDEX,MAPTEN,INCHES
COMMON /IOCOUNT/  IREWIND(40),NREAD(40),NWRITE(40)
COMMON /MARCHS/  MARCH,INDEX(3)
COMMON /OUT/  NODE(3,4000)
COMMON /ZONING/  IZONE,ISECT,NZINDEX,NMBRnds(3)

IPLANE = NWRITE(NUNIT) + 1

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

10 CONTINUE

40 NWRITE(NUNIT) = NWRITE(NUNIT) + 1

RETURN

---FORMAT STATEMENTS

1000 FORMAT(24I5)
1010 FORMAT(6E22.14)
1020 FORMAT(22I6)

END

*****UTILITY*****

---SUBROUTINE CROSS(A,B,C,N)

C = CROSS PRODUCT OF A AND B (UNIT VECTOR)

COMMON /COUNTER/ NODESNAV,NODETOT,NNODES,NPLAN
COMMON /HEADER/ ITITLE(20),LINE
COMMON /INITA/ IZINDEX,MAPTEN,INCHES
COMMON /UNITS/ NU5,NU6,NU20
COMMON /ZONING/ IZONE,ISECT,NZINDEX,NMBRnds(3)

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

C ---CROSS PRODUCT

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 ---MAGNITUDE

CALL VMAG(C,CMAG)

IF(CMAG.GT.0.0) THEN

C ---NORMALIZE

C-50

LOCKHEED-HUNTSVILLE ENGINEERING CENTER
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
END IF
RETURN
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** GRID SPACING**
******************************************************************************
C
SUBROUTINE ETABC(MARCH,INDEX,NODE)
C
THIS ROUTINE SEPERATES THE BOUNDARY CONDITIONS AND
CALCULATES THE VALUE OF ETA.
C
COMMON /INITA/ IZINDEX,MAPTEN,INCHES
COMMON /INIC/ PI,RADDEG
COMMON /INPUTA/ EDGE(3,12),POINT(3,8),SIDE(3,6)
COMMON /INPUTBC/ INODEBC(3),ISIDE(3)
COMMON /SPACING/ ISTRCH(3),STRETCH(3),ETAS(3,200),ETA(3),DETA(3)
COMMON /ZONING/ IZONE,ISECT,NZINDEX,NMBRND(3)
C
FIRST NODE
C
IF(NODE.EQ.1) THEN
C
ETA(INDEX) = 0.0
C
END IF
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(NODE)) THEN
C
ETA(INDEX) = 1.0
C
C-51

LOCKHEED-HUNTSVILLE ENGINEERING CENTER
C---DETERMINE SIDE
C
ISIDE(INDEX) = 9 + (INDEX - 7)*INDEX/2
C
C---STORE SPACING
C
ETAS(INDEX,NODE) = .1.0
C
RETURN
C
END IF
C
C INTERIOR NODES
C
INODEB(INDEX) = 9
C
IF(ISELECT.GT.1 .AND. INDEX.NE.MARCH) GO TO 310
C
C---CALCULATE ETA
C
ISTR = ISTRCH(INDEX) + 1
C
GO TO (100,110,120,130,140,150,160,170,180,190,200) ISTR
C
C---EQUAL SPACING---------------------------------------------(0)
C
100 ETA(INDEX) = ETA(INDEX) + DETA(INDEX)
C
GO TO 300
C
C---INPUT ETA SPACING------------------------------------------(1)
C
110 ETA(INDEX) = ETAS(INDEX,NODE)
C
GO TO 300
C
C---DECREASING SPACING; INPUT STRETCHING FACTOR----------------------(2)
C
120 RATIO = REAL(NODE - 1)/REAL(NMBRNSD(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
C
GO TO 300
C
C---INCREASING SPACING; INPUT STRETCHING FACTOR---------------------(3)
C
130 RATIO = REAL(NMBRNSD(INDEX) - NODE)/REAL(NMBRNSD(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
C
GO TO 300
C
---DOUBLE STRETCHING; INPUT STRETCHING FACTOR-------------------------(4)
C
140   X1 = (STRETCH(INDEX)/2.0)/2.0
ETA1M = (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
ETAMAX = (EXP(X2) - EXP(-X2))/(EXP(X2) + EXP(-X2))
C
ETA(INDEX) = (ETAMAX + ETA1M)/(2.0*ETA1M)
C
GO TO 300
C
---DECREASING SPACING; INPUT MINIMUM SPACING-------------------------(5)
C
150   ARG1 = STRETCH(INDEX)*REAL(NODE - 1)
EXP1 = EXP(ARG1)
EXP1II = 1.0/EXP1
TANHI = (EXP1 - EXP1II)/(EXP1 + EXP1II)
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
C
GO TO 300
C
---INCREASING SPACING; INPUT MINIMUM SPACING-------------------------(6)
C
160   ARG1 = STRETCH(INDEX)*REAL(NMBRNDs(INDEX) - NODE)
EXP1 = EXP(ARG1)
EXP1II = 1.0/EXP1
TANHI = (EXP1 - EXP1II)/(EXP1 + EXP1II)
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
C
GO TO 300
C
---DOUBLE STRETCHING; INPUT MINIMUM SPACING-------------------------(7)
C
170   ARG1 = STRETCH(INDEX)*REAL(2*NODE - NMBRNDs(INDEX) - 1)
EXP1 = EXP(ARG1)
EXP1II = 1.0/EXP1
TANHI = (EXP1 - EXP1II)/(EXP1 + EXP1II)
C
ARGN = STRETCH(INDEX)*REAL(NMBRNDs(INDEX) - 1)
EXPN = EXP(ARGN)
EXPNI = 1.0/EXPN

C-53

LOCKHEED-HUNTSVILLE ENGINEERING CENTER
TANHN = (EXPN - EXPNI)/(EXPN + EXPNI)

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

GO TO 300

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

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

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

GO TO 300

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

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

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

GO TO 300

C---USER INPUT STRETCHING FUNCTION---------------------------------------- (10)

200 CONTINUE

C---STORE SPACING---------------------------------------------------------

300 ETAS(INDEX,NODE) = ETA(INDEX)

310 ETA(INDEX) = ETAS(INDEX,NODE)

RETURN

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

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

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

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

C SUBROUTINE IOPACK(NUNIT)

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

C GENERAL PURPOSE FORTRAN I/O PACKAGE FOR UNITS 1 -> 40

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

COMMON /IOCOUNT/ IREWIND(40),NREAD(40),NWRITE(40)
COMMON /UNITS/ NU5,NU6,NU20

DATA IREWIND,NREAD,NWRITE /120*0/

C ENTRY RWINd: REWIND FILE ON NUNIT

C ENTRY RWINd(NUNIT)

C IREWIND(NUNIT) = 1

C REWIND NUNIT

C RETURN

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

C-54

LOCKHEED-HUNTSVILLE ENGINEERING CENTER
C ENTRY STATUS: PRINT STATUS OF I/O OPERATIONS ON ALL UNITS
C
ENTRY STATUS(NUNIT)
C
WRITE(NU6,1000)
C
DO 10 NU=1,40
10 IF(IREWIND(NU).EQ.1) WRITE(NU6,1010) NU,NREAD(NU),NWRITE(NU)
C
RETURN
C
C FORMAT STATEMENTS
C
1000 FORMAT(/42X,38H STATUS OF I/O OPERATIONS ON ALL UNITS
1     /42X,32H UNIT NO. OF NO. OF .
2     /42X,32H NUMBER READS WRITES / )
1010 FORMAT(42X,3110)
C
END
C
C********************************************************************
C********************************************************************
C
C********************************************************************
C
SUBROUTINE OUTPUT(NUNIT,NODSTOR)
C
C PRINTOUT AND STORE DATA
C
C COMMON /COUNTER/ NODSAV,NODETOT,NBNODES,NPLANE
COMMON /HEADER/ ITITLE(20),LINE
COMMON /INITA/ IZINDEX,MAPTEM,INCHES
COMMON /OUT/ NODE(3,4000)
COMMON /UNITS/ NU5,NU6,NU20
COMMON /ZONING/ IZONE,ISECNT,NIINDEX,NMBRSDS(3)
C
C TOTAL NUMBER OF PLANES
C
NPLANE = NPLANE + 1
C
C PRINT NODAL INFORMATION
C
IPRINT = 1
C
NODETOT = NODSTOR + NODETOT
NODSAV = NODSAV - NODSTOR
C
C TOTAL NUMBER OF POINTS STORED
C
LINE = LINE + 3
C
IF(LINE.GE.60) THEN
C
WRITE(NU6,1000) ITITLE,IZONE
WRITE(NU6,1030)
C
LINE = 7
C
END IF
C
C-55

LOCKEED-HUNTSVILLE ENGINEERING CENTER
WRITE(NU6,1060) NODSTOR,NPLANE,NUNIT,NODETOT

C---------------------------------------------------------------
C STORE OUTPUT
C---------------------------------------------------------------

200 CALL BLKOUT(NUNIT,NODSTOR)
C
RETURN
C
C----FORMAT STATEMENTS
C
1000 FORMAT(1H1,10X,20A4,13X,8H SECTION,I2,3H OF,I3,9H FOR ZONE,I3)
1020 FORMAT(1X,16,21X,F13.7),5X,F7.2,6X,F7.2,5X,I2)
1030 FORMAT( / 44H NODE X Y Z )
1040 FORMAT(1X,16,3(1X,F13.7),2X,2(2X,F7.2),F11.3,3X,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****OUTPUT*****************************************************************
C*****************************************************************
C
SUBROUTINE PICTURE(IDRAW)
C
THIS ROUTINE DESCRIBES THE NOMENCLATURE
*****************************************************************
COMMON /UNITS/ NUS,NUG,NU20

WRITE(NU6,300)
WRITE(NU6,310)
WRITE(NU6,320)
WRITE(NU6,330)
C
RETURN
C
C----FORMAT STATEMENTS
C
300 FORMAT(40X,37H)
1 /40X,19H IE
2 /40X,19H IT
3 /40X,34H IA SURFACE 4 (TOP)
4 /40X,32H IZ
5 /40X,50H POI 4 N--------------------------O
6 /40X,42H 8/I EDGE 3 7/I
7 /40X,42H E/I S)
8 /40X,50H E/I
310 FORMAT( 40X,50H)
1 /40X,50H G/ I SURFACE 1 G/ I U
2 /40X,50H D/ I (BACK) D/ I E R
3 /40X,50H E/ I D/ I I D F
4 /40X,50H G/ I EDGE 11 I I E C
5 /40X,50H E I
6 /40X,46H U E I I ID I 2
7 /40X,50H R D I 4I SURFACE 3 IG I 6
8 /40X,42H F G I (FRONT) IE I
9 /40X,50H A E I O C-56

LOCKHEED-HUNTSVILLE ENGINEERING CENTER
**SUBROUTINE VADD(CA, A, CB, B, C, UC)**

VADD computes C, the sum of vectors CA*A and CB*B, WHERE CA and CB are scalars. UC 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) = CA*A(I) + CB*B(I)
   SUM = SUM + C(I)*C(I)
10 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
```

**SUBROUTINE VDOT(A, B, C)**

VDOT computes C, the dot product of vectors A and B.

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

C = 0.0
```

C-57

LOCKHEED-HUNTSVILLE ENGINEERING CENTER
DO 10 I=1,3
10 C = C + A(I)*B(I)
C RETURN
END
C
C**********************************************************************
C**********************************************************************
C**********************************************************************
C SUBROUTINE VMAG(VECTOR,VECMAG)
C**********************************************************************
C VMAG DETERMINES THE MAGNITUDE OF A VECTOR
C**********************************************************************
C DIMENSION VECTOR(3)
C
C VECMAG = SQRT(VECTOR(1)**2 + VECTOR(2)**2 + VECTOR(3)**2)
C IF(VECMAG.LT.0.0000001) VECMAG = 0.0
C RETURN
END
C
C**********************************************************************
C**********************************************************************
C**********************************************************************
C SUBROUTINE CAXIS(B,RHO,C)
C**********************************************************************
C CALCULATE AXIAL DISTANCE CORRESPONDING TO A POINT ON THE HOLE.
C B ANGULAR LOCATION OF POINT ON THE HOLE
C C AXIAL DISTANCE
C RHO HOLE RADIUS
C
C COMMON /DFN1/ DU1(3),DU2(3),DU3(3),DFNB,DFND,DFNF
C COMMON /DFN2/ AE,BE,DFNR,ZSTAR,AGL,BETA1,BETA2,BETA3,BETA4
C COMMON /INITC/ PI,RADDEG
C
C TOLERANCE FOR NEWTON-RAPHSON ITERATION
C EPS = 1.0E-07
C
C ANGLE OF DUCT AXIS
C SA = SIN(AGL)
C CA = COS(AGL)
C
C ANGLULAR LOCATION OF A POINT ON THE HOLE
C SB = SIN(B)
C CB = COS(B)
C
C AXIAL DISTANCE ALONG LOWER EDGE OF DUCT
C DELTA = 0.0/RADDEG
C
C IF((B.GE.(BETA4 - DELTA)) .AND. (B.LE.BETA3)) THEN

C-58
LOCKHEED-HUNTSVILLE ENGINEERING CENTER
C = (2STAR + RHO*SA*CB)/CA

RETURN

END IF

C---DISTANCE ALONG MAJOR AXIS OF BOWL ELLIPSE
C
XT = DFNF - DFND + RHO*CB
IF(ABS(XT).GT.AE) XT = AE
C
C---VERTICAL DISTANCE TO HOLE
C
YT = DFNB + RHO*SB
C
C---DISTANCE ALONG MINOR AXIS OF BOWL ELLIPSE
C
DR = SQRT(AE**2 - XT**2)*BE/AE
C
C---INITIAL DUCT AXIAL DISTANCE
C
C = SQRT((DFNR + DR)**2 - YT**2)
C
C---NEWTON-RAPHSON ITERATION-----------------------------------------------
C
10 XTT = DFNF - DFND + C*SA + RHO*CA*CB
YTT = YT
ZTT = C*CA - RHO*SA*CB
C
C---FUNCTION
C
DR = SQRT(AE**2 - XTT**2)*BE/AE
FC = (DFNR + DR)**2 - YTT**2 - ZTT**2
C
C---DERIVATIVE
C
DFDC1 = XTT*SA*(DFNR + DR)*BE/AE
DFDC2 = DR*AE/BE
DFDC3 = ZTT*CA
DFDC = -2.*(DFDC1/DFDC2 + DFDC3)
C
C---DUCT AXIAL DISTANCE
C
C = C - FC/DFDC
C
IF(ABS(FC).GT.FEPS) GO TO 10
C
RETURN
END
C

C*******************************************************************************
C***********************************************************
C***********************************************************
C
SUBROUTINE DCTDAT
C
DATA DESCRIBING HOLE AND WELD RADIUS
C
-----------------------------
COMMON /DFN3/ AD, BD, BETA(37), DELRHO(37), RADOC(37)

DIMENSION DCT1(37), DCT2(37), DCT3(37)

DATA RADDEG/57.29577951/

---ANGULAR INCREMENT AROUND HOLE

---DIFFERENCE IN RADIUS BETWEEN DUCT AND HOLE

---RADIUS OF CURVATURE OF FAIRING

---READ DUCT DATA

READ(5,1000) (DCT1(I), I=1, 37)
READ(5,1000) (DCT2(I), I=1, 37)
READ(5,1000) (DCT3(I), I=1, 37)

---CONVERT DEGREES TO RADIANS AND INCHES TO FEET

DO 10 I = 1, 37
   BETA(I) = DCT1(I) / RADDEG
   DELRHO(I) = DCT2(I)
   RADOC(I) = DCT3(I)

---FORMAT STATEMENT

1000 FORMAT((8F10.3))

RETURN
END

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

SUBROUTINE DELRAD(B, RHO, DRHO, RHOD, RDOC)

INTERPOLATES HOLE RADIUS, WELD RADIUS, DUCT RADIUS, AND DIFFERENCE BETWEEN HOLE AND DUCT RADIUS FOR POINTS BETWEEN ANGULAR INCREMENTS.

B       ANGULAR LOCATION OF POINT ON HOLE
RHO     HOLE RADIUS
RHOD    DUCT RADIUS
RDOC    WELD RADIUS
DRHO    DIFFERENCE BETWEEN HOLE AND DUCT RADIUS

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

COMMON /DFN3/ AD, BD, BETA(37), DELRHO(37), RADOC(37)

---ANGULAR LOCATION OF A POINT ON THE DUCT

BB = B
IF(B.LT.0.0) BB = 6.283185308 + B
C---BRACKET ANGULAR INCREMENT
C
DO 10 I = 1,36
C
DBK = BETA(I+1) - BETA(I)
DB = BB - BETA(I)
C
IF(DB.LE.DBK) THEN
C
C---INTERPOLATE-------------------------------------------------
C
C---SEGMENT RATIO
PSI = DB/DBK
C
C---DIFFERENCE IN RADIUS BETWEEN DUCT AND HOLE
C
DRHO = DELRHO(I)*(1. - PSI) + DELRHO(I+1)*PSI
C
C---RADIUS OF WELD
C
RDOC = RADOC(I)*(1. - PSI) + RADOC(I+1)*PSI
C
C---RADIUS OF DUCT ELLIPSE
C
CBSQ = COS(B)**2
SBSQ = SIN(B)**2
C
RADX = SQRT(SBSQ*AD*AD + CBSQ*BD*BD)
RHOD = AD*BD/RADX
C
C---HOLE RADIUS
C
RHO = RHOD + DRHO
C
RETURN
C
END IF
C
10 CONTINUE
C
RETURN
END
C
************************************************************************
C
************************************************************************
C
SUBROUTINE DERIV(B,RHO,C,DRDB,DCDB)
C
************************************************************************
C
************************************************************************
C
COMMON /DFN1/ DU1(3),DU2(3),DU3(3),DFNB,DFND,DFNF
COMMON /DFN2/ AE,BE,DFNR,ZSTAR,AGL,BETA1,BETA2,BETA3,BETA4
COMMON /DFN3/ AD,BD,BETA(37),DELRHO(37),RADOC(37)
C
C---POINT ANGLE
C
CB = COS(B)
SB = SIN(B)

C-61

LOCKHEED-HUNTSVILLE ENGINEERING CENTER
C
EX1 = BD**2 - AD**2
EX2 = ((AD*SB)**2 + (BD*CB)**2)**1.5
DRDDB = EX1*AD*BD*SB*CB/EX2
C---BRACKET BETA
C
DO 10 I = 1,36
C
DBI = BETA(I+1) - BETA(I)
DB = B - BETA(I)
IB = I
C IF(DB.LE.DB) GO TO 20
C
10 CONTINUE
20 CONTINUE
C
DDRO = (DELRHO(IB+1) - DELRHO(IB))/(BETA(IB+1) - BETA(IB))
C
DRDB = DRDDB + DDRO
C---FOR BETA BETWEEN BETA4 AND BETA3
C
IF((B.GE.BETA4).AND.(B.LE.BETA3)) THEN
C
DCDB = (DRDB*CB-RHO*SB)*TAN(AGL)
RETURN
END IF
C
CA = COS(AGL)
SA = SIN(AGL)
C
XMD = DFNF - DFND + C*SA + RHO*CA*CB
EXR1 = BE/AE
EXR2 = SQRT(AE**2 - XMD**2)
EXR = (DFNR + EXR1*EXR2)*XMD*EXR1/EXR2
C
EXC1 = C*CA - RHO*SA*CB
EXC2 = RHO*CB + DRDB*SB
EXC3 = DRDB*CB - RHO*SB
EXC4 = DFNB + RHO*SB
EXCN = (EXC1*SA - EXR*CA)*EXC3 - EXC2*EXC4
EXCD = EXR*SA + EXC1*CA
DCDB = EXC4/EXCD
C
RETURN
END
C
C****************************************************************************
C****************************************************************************
C***************************************************************************
C***************************************************************************
C
SUBROUTINE DEXIT(B,POINT,DE)
C
C COMPUTE POINT COORDINATES ON DUCT EXIT PLANE
C
C-62

LOCKHEED-HUNTSVILLE ENGINEERING CENTER
COMMON /DFN1/ DU1(3),DU2(3),DU3(3),DFNB,DFND,DFNF
COMMON /DFN2/ AE,BE,DFNR,STAR,AGL,BETA1,BETA2,BETA3,BETA4
COMMON /DFN3/ AD,BD,BETA(37),DELH0(37),RADOC(37)
COMMON /DFN5/ CE,CQ,ABOT,ATOP

DIMENSION POINT(6)

COA = COS(AGL)
SIA = SIN(AGL)

COB = COS(B)
SIB = SIN(B)

RADX = SQRT((AD*SIB)**2 + (BD*COB)**2)
RHOD = AD*BD/RADX

C = CE + RHOD*SIB*DFNB/(COA - CE)

RETURN
END

SUBROUTINE HGMIN

INPUT FOR TWO DUCT HGM WITH TURN AROUND DUCT

COMMON /COEFF/ COEFE(8,10,12),COEFS(8,6),NMBRSEG(12)
COMMON /INITA/ IZINDEX,MAPTEN,INCHES
COMMON /INITC/ PI,RADDEG
COMMON /INPUTA/ EDGE(3,12),POINT(3,8),SIDE(3,6)
COMMON /MAPING/ MAPSIDE(6),MAPSEG(10,12)

C-63

LOCKHEED-HUNTSVILLE ENGINEERING CENTER
COMMON /MAXIMUM/ ETAMAX(10,12),SEGMAX(3,10,12)
COMMON /ZONING/ IZONE,ISECT,NZINDEX,NMBRND5(3)
C
COMMON /DFN1/ DU1(3),DU2(3),DU3(3),DFNB,DFND,DFNF
COMMON /DFN2/ AE, BE,DFNR,ZSTAR,AGL,BETA1,BETA2,BETA3,BETA4
COMMON /DFN3/ AD, BD, BETA(37), DELRHO(37), RADOC(37)
COMMON /DFN4/ IHOLE, PL(6,6,4), PA(2,6,4), EDG(6,4)
COMMON /DFN5/ CE, CQ, ABOT, ATOP
COMMON /DFN6/ RTADI, RTADO, RCI, RCO, RDI, RDO, XCI, XCO, XTADI, XTADO
COMMON /DFN7/ RWO21, XWO21, RADII, RADIO
C
DIMENSION CADD(3), VEC(3), VDUM(3), UE(3)
C
C GENERAL CONSTANTS
C---------------------------------------------------------------
C
IHOLE = 0

C----BOWL ELLIPSE-----------------------------------------------
C
AE = 4.5
BE = 2.85776

C----ORIGIN OF BOWL ELLIPSE
C
DFNB = 5.0
DFNF = 5.2

C----BOWL ELLIPSE FOCI
C
FE = SQRT(AE*AE - BE*BE)

C----FIRST BOWL ELLIPSE FOCI AXIAL DISTANCE
C
XFE = DFND - FE

C----ANGLES OF POINTS ON HOLE---------------------------------
C
BETA1 = 130./RADDEG
BETA2 = 50./RADDEG
BETA3 = 298./RADDEG
BETA4 = 220./RADDEG

C----ORIGIN OF DUCT AXIS---------------------------------------
C
DFNB = 5.0
DFNF = 5.2

C----ANGLE OF DUCT
C
AGL = 10.0/RADDEG
C
C----UNIT VECTORS OF DUCT COORDINATE SYSTEM
C
DU1(1) = SIN(AGL)
DU1(2) = 0.
DU1(3) = COS(AGL)

C
DU2(1) = COS(AGL)
DU2(2) = 0.
DU2(3) = -SIN(AGL)

C-64

LOCKHEED-HUNTSVILLE ENGINEERING CENTER
DU3(1) = 0.
DU3(2) = 1.
DU3(3) = 0.

---DUCT ELLIPSE
AD = 2.98
BD = 3.55

---AXIAL LENGTH TO DUCT EXIT PLANE
CE = 16.2

---AXIAL LENGTH TO POINT ON EXIT PLANE NORMAL IN X-Z PLANE
CQ = 25.206

---UNKNOWN
ATOP = 7.
ABOT = 3.

---POINT 1: BOWL ENTRANCE - INNER
RADI1 = 6.6

---POINT 2: END OF BOWL - INNER
XLI = 9.810
RADOI = 6.029

---POINT 3: END OF BOWL - OUTER
XLO = 9.830
CXL = ((XLO - DFND)/AE)**2
RADOO = DFNR + BE*SQR(1.0 - CXL)

---POINT 4: BOWL ENTRANCE - OUTER
RADIO = 7.5

---EDGE 1: SEGMAX
XMAXI1 = 2.450

---EDGE 1: SEGMAX 3
XMAXI3 = 7.912
RMAXI3 = 6.380

---EDGE 3: SEGMAX 1
XMAXI3 = 1.0
RMAXI3 = 7.815
ZSTAR = RMAXI3

C-65

LOCKHEED--HUNTSVILLE ENGINEERING CENTER
C----EDGE 3: SEGMAX 2
C
CR23 = ((RMAX13 - DFNR)/BE)**2
C
XMAX23 = DFND - AE*SQRT(1.0 - CR23)
C----EDGE 3: SEGMAX 3
C
XMAX33 = 3.0
XMAX33 = 2.35
C----EDGE 3: SEGMAX 4
C
XMAX43 = DFND + AE*SQRT(1.0 - CR23)
C----EDGE 11: SEGMAX 3
C
XMAX311 = 3.4
C
CX311 = ((XMAX311 - DFND)/AE)**2
C
RMAX311 = DFNR + BE*SQRT(1.0 - CX311)
C----EDGE 11: SEGMAX 4
C
XMAX411 = 8.5
C
CX411 = ((XMAX411 - DFND)/AE)**2
C
RMAX411 = DFNR + BE*SQRT(1.0 - CX411)
C----ANGLE OF INTERSECTION BETWEEN SECTION 1 AND SECTION 2----------
C
THETAD = 75.0
C
TH = THETAD/RADDEG
CTH = COS(TH)
STH = SIN(TH)
C----ANGLE OF INTERSECTION FOR SEGMENT 1 & 2 AND SEGMENT 2 & 3--------
C
THETAD2 = 18.0
C
THE2 = THETAD2/RADDEG
CTHE2 = COS(THE2)
STHE2 = SIN(THE2)
C
THETAD3 = 82.0
C
THE3 = THETAD3/RADDEG
CTHE3 = COS(THE3)
STHE3 = SIN(THE3)
C
IF(IZONE.EQ.1.AND.IZINDEX.GT.1)THEN
  GO TO 40
ELSE
  GO TO 50
END IF
40 GO TO (50, 50, 200, 300, 300, 50) IZINDEX

C 50 IF (IZONE.GT.1) GO TO 100

C******************************************************************************
C ZONE 1 - (SIDE OF BOWL WITHOUT HOLE)
C******************************************************************************
C---EDGE COEFFICIENTS--------------------------------------------------------

C---EDGE 3: ELLIPSE
C
COEFE(1, 2, 3) = XFE
COEFE(3, 3, 3) = -DFNR
COEFE(4, 3, 3) = 1.0

C---EDGE 5: CIRCULAR ARC
C
COEFE(1, 1, 5) = 0.0
COEFE(4, 1, 5) = 1.0

C---EDGE 6: CIRCULAR ARC
C
COEFE(1, 1, 6) = XLI
COEFE(4, 1, 6) = 1.0

C---EDGE 7: CIRCULAR ARC
C
COEFE(1, 1, 7) = XLO
COEFE(4, 1, 7) = 1.0

C---EDGE 8: CIRCULAR ARC
C
COEFE(1, 1, 8) = 0.0
COEFE(4, 1, 8) = 1.0

C---EDGE 11: ELLIPSE
C
DO 10 K = 3, 5
C
COEFE(1, K, 11) = XFE
COEFE(2, K, 11) = DFNR*STH
COEFE(3, K, 11) = -DFNR*CTh
10  COEFE(4, K, 11) = 1.0

C---SURFACE COEFFICIENTS-----------------------------------------------------

COEFS(4, 2) = 1.0
COEFS(7, 2) = 3.0

COEFS(4, 4) = 1.0
COEFS(7, 4) = 3.0

COEFS(4, 6) = 1.0
COEFS(7, 6) = 3.0

C---CORNER POINT COORDINATES-------------------------------------------------
C
POINT(1, 1) = 0.0
POINT(2, 1) = 0.0
POINT(3, 1) = -RADI

C-67

LOCKHEED–HUNTSVILLE ENGINEERING CENTER
C
POINT(1,2) = XLI
POINT(2,2) = 0.0
POINT(3,2) = -RADOI

C
POINT(1,3) = XLO
POINT(2,3) = 0.0
POINT(3,3) = -RADOO

C
POINT(1,4) = 0.0
POINT(2,4) = 0.0
POINT(3,4) = -RADIO

C
POINT(1,5) = 0.0
POINT(2,5) = RADOI*STH
POINT(3,5) = -RADOI*CTH

C
POINT(1,6) = XLI
POINT(2,6) = RADOI*STH
POINT(3,6) = -RADOI*CTH

C
POINT(1,7) = XLO
POINT(2,7) = RADOO*STH
POINT(3,7) = -RADOO*CTH

C
POINT(1,8) = 0.0
POINT(2,8) = RADIO*STH
POINT(3,8) = -RADIO*CTH

C
---EDGE SEGMENT COORDINATES -----------------------------------------------------

C
SEGMAX(1,1, 1) = XMAX11
SEGMAX(3,1, 1) = -RADI1

C
SEGMAX(1,2, 1) = XMAX31
SEGMAX(3,2, 1) = -RMAX31

C
SEGMAX(1,1, 3) = XMAX13
SEGMAX(3,1, 3) = -RMAX13

C
SEGMAX(1,2, 3) = XMAX23
SEGMAX(3,2, 3) = -RMAX13

C
SEGMAX(1,1, 9) = XMAX11
SEGMAX(2,1, 9) = RADI1*STH
SEGMAX(3,1, 9) = -RADI1*CTH

C
SEGMAX(1,2, 9) = XMAX31
SEGMAX(2,2, 9) = RMAX31*STH
SEGMAX(3,2, 9) = -RMAX31*CTH

C
SEGMAX(1,1,11) = XMAX13
SEGMAX(2,1,11) = RMAX13*STH
SEGMAX(3,1,11) = -RMAX13*CTH

C
SEGMAX(1,2,11) = XMAX23
SEGMAX(2,2,11) = RMAX13*STH
SEGMAX(3,2,11) = -RMAX13*CTH

C
SEGMAX(1,3,11) = XMAX311


\[
\begin{align*}
\text{SEGMAX}(2, 3, 11) &= \text{RMAX311} \cdot \text{STH} \\
\text{SEGMAX}(3, 3, 11) &= -\text{RMAX311} \cdot \text{CTH} \\
\text{SEGMAX}(1, 4, 11) &= \text{XMAX411} \\
\text{SEGMAX}(2, 4, 11) &= \text{RMAX411} \cdot \text{STH} \\
\text{SEGMAX}(3, 4, 11) &= -\text{RMAX411} \cdot \text{CTH}
\end{align*}
\]

---EDGE NODE DISTRIBUTION---------------------------------------

\[
\begin{align*}
\text{ETAMAX}(1, 1) &= 12.0 \\
\text{ETAMAX}(2, 1) &= 44.0 \\
\text{ETAMAX}(1, 3) &= 5.0 \\
\text{ETAMAX}(2, 3) &= 10.0 \\
\text{ETAMAX}(1, 9) &= 12.0 \\
\text{ETAMAX}(2, 9) &= 44.0 \\
\text{ETAMAX}(1, 11) &= 5.0 \\
\text{ETAMAX}(2, 11) &= 10.0 \\
\text{ETAMAX}(3, 11) &= 19.0 \\
\text{ETAMAX}(4, 11) &= 48.0
\end{align*}
\]

RETURN

ZONE 2 - (SIDE OF BOWL WITH HOLE)

100 IF(IZONE.GT.2) GO TO 200

---EDGE COEFFICIENTS---------------------------------------------

---EDGE 3: ELLIPSE

\[
\begin{align*}
\text{DO 110 } K &= 3, 5 \\
\text{COEFE}(1, K, 3) &= XPE \\
\text{COEFE}(2, K, 3) &= \text{DFNR} \cdot \text{STH} \\
\text{COEFE}(3, K, 3) &= -\text{DFNR} \cdot \text{CTH} \\
110 \text{ COEFE}(4, K, 3) &= 1.0
\end{align*}
\]

---EDGE 5: CIRCULAR ARC

\[
\begin{align*}
\text{COEFE}(1, 1, 5) &= 0.0 \\
\text{COEFE}(4, 1, 5) &= 1.0
\end{align*}
\]

\[
\begin{align*}
\text{COEFE}(1, 2, 5) &= 0.0 \\
\text{COEFE}(4, 2, 5) &= 1.0 \\
\text{COEFE}(1, 3, 5) &= 0.0 \\
\text{COEFE}(4, 3, 5) &= 1.0
\end{align*}
\]

---EDGE 6: CIRCULAR ARC

\[
\begin{align*}
\text{COEFE}(1, 1, 6) &= XLI \\
\text{COEFE}(4, 1, 6) &= 1.0
\end{align*}
\]

\[
\begin{align*}
\text{COEFE}(1, 2, 6) &= XLI \\
\text{COEFE}(4, 2, 6) &= 1.0
\end{align*}
\]
C
COEFE(1, 3, 6) = XLI
COEFE(4, 3, 6) = 1.0
C
---EDGE 7: CIRCULAR ARC
C
COEFE(1, 1, 7) = XLO
COEFE(4, 1, 7) = 1.0
C
COEFE(1, 2, 7) = XLO
COEFE(4, 2, 7) = 1.0
C
COEFE(1, 3, 7) = XLO
COEFE(4, 3, 7) = 1.0
C
---EDGE 8: CIRCULAR ARC
C
COEFE(1, 1, 8) = 0.0
COEFE(4, 1, 8) = 1.0
C
COEFE(4, 2, 8) = 1.0
COEFE(4, 3, 8) = 1.0
C
---EDGE 11: ELLIPSE
C
COEFE(1, 3, 11) = 0.0
COEFE(2, 3, 11) = 0.0
COEFE(3, 3, 11) = 0.0
COEFE(4, 3, 11) = 0.0
C
COEFE(1, 4, 11) = 0.0
COEFE(2, 4, 11) = 0.0
COEFE(3, 4, 11) = 0.0
COEFE(4, 4, 11) = 0.0
C
COEFE(1, 5, 11) = XFE
COEFE(2, 5, 11) = 0.0
COEFE(3, 5, 11) = DFNR
COEFE(4, 5, 11) = 1.0
C
---SURFACE COEFFICIENTS
C
COEFS(4, 2) = 1.0
COEFS(7, 2) = 3.0
C
COEFS(4, 6) = 1.0
COEFS(7, 6) = 3.0
C
---CORNER POINT COORDINATES
C
POINT(1, 1) = 0.0
POINT(2, 1) = RADI*STH
POINT(3, 1) = -RADI*CTH
POINT(4, 1) = 0.0
POINT(5, 1) = 0.0
C
POINT(1, 2) = XLI
POINT(2, 2) = RADOI*STH
POINT(3, 2) = -RADOI*CTH
POINT(4, 2) = THETAF
POINT(5, 2) = 90.0
C
POINT(1,3) = XLO
POINT(2,3) = RADO0*STH
POINT(3,3) = -RADO0*CTH
POINT(4,3) = THETAF
POINT(5,3) = 90.0
C
POINT(1,4) = 0.0
POINT(2,4) = RADIO*STH
POINT(3,4) = -RADIO*CTH
POINT(4,4) = 12.831
POINT(5,4) = -7.491
C
POINT(1,5) = 0.0
POINT(2,5) = 0.0
POINT(3,5) = RADII
POINT(4,5) = 0.0
POINT(5,5) = 0.0
C
POINT(1,6) = XLI
POINT(2,6) = 0.0
POINT(3,6) = RAD01
POINT(4,6) = 0.0
POINT(5,6) = 0.0
C
POINT(1,7) = XLO
POINT(2,7) = 0.0
POINT(3,7) = RADO0
POINT(4,7) = 0.0
POINT(5,7) = -50.0
C
POINT(1,8) = 0.0
POINT(2,8) = 0.0
POINT(3,8) = RADIO
POINT(4,8) = 0.0
POINT(5,8) = 14.7
C
C---UPPER LEFT CORNER ON HOLE----------------------------------------
C
CALL HOLE(BETA1,EDG(1,1),VDUM)
C
RAD1 = SQRT(EDG(2,1)**2 + EDG(3,1)**2)
ANG1 = ACOS(EDG(2,1)/RAD1)
C
C---UPPER RIGHT CORNER ON HOLE
C
CALL HOLE(BETA2,EDG(1,2),VDUM)
C
RAD2 = SQRT(EDG(2,2)**2 + EDG(3,2)**2)
ANG2 = ACOS(EDG(2,2)/RAD2)
C
C---LOWER RIGHT CORNER ON HOLE
C
CALL RHOS(BETA(31),DELRHO(31))
C
CALL HOLE(BETA3,EDG(1,3),VDUM)
C
RAD3 = SQRT(EDG(2,3)**2 + EDG(3,3)**2)
ANG3 = ACOS(EDG(2,3)/RAD3)
C
C-71
LOCKHEED-HUNTSVILLE ENGINEERING CENTER
C---LOWER LEFT CORNER ON HOLE
C
C    CALL RHOS(BETA(23),DELRH0(23))
C    CALL HOLE(BETA4,EDG(1,4),VDUM)
C
RAD4 = SQRT(EDG(2,4)**2 + EDG(3,4)**2)
ANG4 = ACOS(EDG(2,4)/RAD4)
C
C---EDGE SEGMENT COORDINATES -----------------------------
C
SEGMAX(1,1, 1) = XMAX11
SEGMAX(2,1, 1) = RADII*STH
SEGMAX(3,1, 1) = -RADII*CTH
C
SEGMAX(1,2, 1) = XMAX31
SEGMAX(2,2, 1) = RMAX31*STH
SEGMAX(3,2, 1) = -RMAX31*CTH
C
SEGMAX(1,1, 3) = XMAX13
SEGMAX(2,1, 3) = RMAX13*STH
SEGMAX(3,1, 3) = -RMAX13*CTH
C
SEGMAX(1,2, 3) = XMAX23
SEGMAX(2,2, 3) = RMAX23*STH
SEGMAX(3,2, 3) = -RMAX23*CTH
C
SEGMAX(1,3, 3) = XMAX311
SEGMAX(2,3, 3) = RMAX311*STH
SEGMAX(3,3, 3) = -RMAX311*CTH
C
SEGMAX(1,4, 3) = XMAX411
SEGMAX(2,4, 3) = RMAX411*STH
SEGMAX(3,4, 3) = -RMAX411*CTH
C
SEGMAX(1,1, 7) = XLO
SEGMAX(2,1, 7) = RADOO*CTHE2
SEGMAX(3,1, 7) = RADOO*STHE2
C
SEGMAX(1,2, 7) = XLO
SEGMAX(2,2, 7) = RADOO*COS(ANG3)
SEGMAX(3,2, 7) = RADOO*SIN(ANG3)
C
SEGMAX(1,1, 8) = 0.0
SEGMAX(2,1, 8) = RADIO*CTHE2
SEGMAX(3,1, 8) = RADIO*STHE2
C
SEGMAX(1,2, 8) = 0.0
SEGMAX(2,2, 8) = RADIO*CTHE3
SEGMAX(3,2, 8) = RADIO*STHE3
C
SEGMAX(1,1, 9) = XMAX11
SEGMAX(2,1, 9) = 0.0
SEGMAX(3,1, 9) = RADII
C
SEGMAX(1,2, 9) = XMAX31
SEGMAX(2,2, 9) = 0.0
SEGMAX(3,2, 9) = RMAX31
C
C-72

LOCKHEED-HUNTSVILLE ENGINEERING CENTER
SEGMAX(1,1,11) = XMAX13
SEGMAX(2,1,11) = 0.0
SEGMAX(3,1,11) = RMAX13

SEGMAX(1,2,11) = XMAX23
SEGMAX(2,2,11) = 0.0
SEGMAX(3,2,11) = RMAX13

SEGMAX(1,3,11) = XMAX33
SEGMAX(2,3,11) = 0.0
SEGMAX(3,3,11) = RMAX13

SEGMAX(1,4,11) = XMAX43
SEGMAX(2,4,11) = 0.0
SEGMAX(3,4,11) = RMAX13

C-----DESCRIPTION OF COORDINATES ON SURFACE WITH HOLE (SIDE 4)-----

DO 130 J = 1,3

PL(1,J,1) = POINT(J,4)
PL(1,J,2) = SEGMAX(J,1,8)
PL(1,J,3) = EDG(J,1)
PL(1,J,4) = EDG(J,2)
PL(1,J,5) = SEGMAX(J,1,7)

PL(2,J,1) = SEGMAX(J,1,3)
PL(2,J,2) = EDG(J,4)
PL(2,J,3) = EDG(J,3)
PL(2,J,4) = SEGMAX(J,2,7)

PL(3,J,1) = POINT(J,8)
PL(3,J,2) = SEGMAX(J,1,11)
PL(3,J,3) = SEGMAX(J,2,11)
PL(3,J,4) = SEGMAX(J,3,11)
PL(3,J,5) = SEGMAX(J,4,11)
PL(3,J,6) = POINT(J,7)

PL(1,J,2) = XMAX13
PL(2,J,2) = RMAX13*CTHE2
PL(3,J,2) = RMAX13*STHE2

PL(1,J,3) = XMAX23
PL(2,J,3) = RMAX13*CTHE2
PL(3,J,3) = RMAX13*STHE2

PL(1,J,4) = XMAX13
PL(2,J,4) = RMAX13*CTHE3
PL(3,J,4) = RMAX13*STHE3

PL(1,J,5) = XMAX23
PL(2,J,5) = RMAX13*CTHE3
PL(3,J,5) = RMAX13*STHE3

PL(1,J,6) = XMAX23
PL(2,J,6) = RMAX13*CTHE3
PL(3,J,6) = RMAX13*STHE3

C-73

LOCKHEED-HUNTSVILLE ENGINEERING CENTER
C----EDGE NODE DISTRIBUTION----------------------------------------
C
ETAMAX(1, 1) = 12.0
ETAMAX(2, 1) = 44.0
C
ETAMAX(1, 3) = 5.0
ETAMAX(2, 3) = 10.0
ETAMAX(3, 3) = 19.0
ETAMAX(4, 3) = 48.0
C
ETAMAX(1, 9) = 12.0
ETAMAX(2, 9) = 44.0
C
ETAMAX(1,11) = 5.0
ETAMAX(2,11) = 13.0
ETAMAX(3,11) = 19.0
ETAMAX(4,11) = 48.0
C
DO 140 K = 7,8
C
140 ETAMAX(1, K) = 25.0
140 ETAMAX(2, K) = 68.0
C
C
RETURN
C**************************************************************************
C**************************************************************************
C-ZONE 3 (DUCT)
C**************************************************************************
C
200 CONTINUE
GO TO (500,500,201,300,300,201) IZINDEX
C
201 CONTINUE
C
IF(IZONE.GT.3) GO TO 300
C
DO 205 I=1,8
DO 205 J=1,6
COEFS(I,J) = 0.0
C
DO 205 K=1,12
205 COEFS(I,J,K) = 0.0
C
C---CORNER POINT COORDINATES ON HOLE------------------------------------
C
DO 210 J=1,3
C
POINT(J,1) = PL(J,4,2)
POINT(J,2) = PL(J,5,2)
POINT(J,6) = PL(J,5,3)
210 POINT(J,5) = PL(J,4,3)
C
C---CORNER POINT COORDINATES ON EXIT PLANE-----------------------------
C
CALL DEXIT(BETA1,POINT(1,4),VDUM)
C
CALL DEXIT(BETA2,POINT(1,3),VDUM)
C
C-74

LOCKHEED-HUNTSVILLE ENGINEERING CENTER
CALL DEXIT(BETA3,POINT(1,7),VDUM)
C
CALL DEXIT(BETA4,POINT(1,8),VDUM)
C
C---EDGE COEFFICIENTS ALONG HOLE (MAP = 7)----------------------
C
COEFE(1,1, 1) = BETA1
COEFE(2,1, 1) = BETA2
C
COEFE(1,1, 5) = BETA1
COEFE(2,1, 5) = BETA4
C
COEFE(1,1, 6) = BETA2
COEFE(2,1, 6) = BETA3 - 2.0*PI
C
COEFE(1,1, 9) = BETA4
COEFE(2,1, 9) = BETA3
C
C---EDGE COEFFICIENTS ALONG DUCT (MAP = 9)----------------------
C
COEFE(1,1, 2) = BETA2
C
COEFE(1,1, 4) = BETA1
C
COEFE(1,1,10) = BETA3
C
COEFE(1,1,12) = BETA4
C
C---EDGE COEFFICIENTS ALONG EXIT ELLIPSE (MAP = 8)----------------------
C
COEFE(1,1, 3) = BETA1
COEFE(2,1, 3) = BETA2
C
COEFE(1,1, 7) = BETA2
COEFE(2,1, 7) = BETA3 - 2.0*PI
C
COEFE(1,1, 8) = BETA1
COEFE(2,1, 8) = BETA4
C
COEFE(1,1,11) = BETA4
COEFE(2,1,11) = BETA3
C
C---SEGMENT MAXIMUMS ALONG DUCT (CIRCULAR ARCS)----------------------
C
ISEG = 1
RATIO = -1.0
C
DO 250 J=1,4
   IEDGE = 4
   IF(J.EQ.2) IEDGE = 2
   IF(J.EQ.3) IEDGE = 10
   IF(J.EQ.4) IEDGE = 12
C
   EDG(1,J) = COEFE(1,ISEG,IEDGE)
   MAPSEG(ISEG,IEDGE) = 9
C
   CALL EMAP(IEDGE,ISEG,RATIO,EDG(1,J),VDUM)
C
DO 240 I=1,3
240 SEGMAX(I,ISEG,IEDGE) = EDG(I,J)

C-75

LOCKHEED-HUNTSVILLE ENGINEERING CENTER
C
MAXETA = 0.65 * NMBRNDS(2)
ETAMAX(ISEG, IEDGE) = MAXETA

C
SEGMAX(4, ISEG, IEDGE) = TXY
250 SEGMAX(5, ISEG, IEDGE) = TXZ

C
S----SURFACE COEFFICIENTS-------------------------
C
COEFS(1,1) = BETA1
COEFS(2,1) = BETA2
COEFS(3,1) = ETAMAX(1,2)

C
COEFS(1,3) = BETA4
COEFS(2,3) = BETA3
COEFS(3,3) = ETAMAX(1,12)

C
COEFS(1,5) = BETA1
COEFS(2,5) = BETA4
COEFS(3,5) = ETAMAX(1,4)

C
COEFS(1,6) = BETA2
COEFS(2,6) = BETA3 - 2 * PI
COEFS(3,6) = ETAMAX(1,10)

C
RETURN

C*******************************************************************************
C
C ZONE 4 & 5 (TURN AROUND DUCT)
C*******************************************************************************

C 300 CONTINUE
C
GO TO (500, 500, 500, 301, 301, 301) IZINDEX

C 301 CONTINUE
C
IF (IZINDEX .GE. 4 .AND. IZONE .EQ. 2) GO TO 350
C
IF (IZONE .GT. 4) GO TO 350

C--INITIALIZE INPUT FOR ZONE 3 TO ZERO-------------------------------
C
DO 325 N = 1, 8
C
DO 310 M = 1, 5
310 POINT(M, N) = 0.0
C
DO 325 M = 1, 6
325 COEFS(N, M) = 0.0
C
DO 340 N = 1, 12
DO 340 M = 1, 5
C
DO 330 J = 1, 5
330 SEGMAX(J, M, N) = 0.0
C
DO 335 J = 1, 8
335 COEFE(J, M, N) = 0.0
C
340 ETAMAX(M, N) = 0.0

C-76

LOCKHEED-HUNTSVILLE ENGINEERING CENTER
---INITIALIZE CONSTANTS-----------------------------------------------

---POINT 1: ENTRANCE

350 XTADI = -2.76
RDO = 5.6

---POINT 4: ENTRANCE

RDI = 4.56

---INNER CIRCULAR ARC

RTADI = 0.4570
XCI = -4.331
RCI = RDO + RTADI
RWI21 = RDO + 2.0*RTADI

---OUTER CIRCULAR ARC

RTADO = 1.36
XCO = -4.35
RCO = RDI + RTADO

---TAD EXIT

XTADO = 0.0

---TANGENCY POINT ON OUTER SURFACE

CALL TADWALL

---ANGLE 1

THETAX = 15.0
THX = THETAX/RADDEG
CTH = COS(THX)
STH = SIN(THX)

---ANGLE 2

CTH1 = COS(THE2)
STH1 = SIN(THE2)

---ANGLE 3

CTH4 = COS(THE3)
STH4 = SIN(THE3)

THETAC = 90.0 - THETAX
ANG1C = 90.0 - THETAD2
ANG4C = 90.0 - THETA3

ZONE 4 (TURN AROUND DUCT)

IF(IZINDEX.GE.4.AND.IZONE.EQ.2) GO TO 400
IF(IZONE.GT.4) GO TO 400

---EDGE COEFFICIENTS---

---EDGE 1: CIRCULAR ARC
COEFE(1,2, 1) = XCI
COEFE(3,2, 1) = -RCI

---EDGE 3: CIRCULAR ARC
COEFE(1,2, 3) = XCO
COEFE(3,2, 3) = -RCO

---EDGE 9: CIRCULAR ARC
COEFE(1,2, 9) = XCI
COEFE(3,2, 9) = RCI

---EDGE 11: CIRCULAR ARC
COEFE(1,2,11) = XCO
COEFE(3,2,11) = RCO

---EDGE : CIRCULAR ARC
DO 360 J = 1,4
COEFE(1,J, 5) = XTADI
COEFE(1,J, 8) = XTADI
COEFE(1,J, 6) = XCI - RTADI
360 COEFE(1,J, 7) = XCO - RTADO

---SURFACE COEFFICIENTS---
COEFS(4,2) = 1.0
COEFS(7,2) = 3.0

COEFS(4,4) = 1.0
COEFS(7,4) = 3.0

COEFS(4,6) = 1.0
COEFS(7,6) = 3.0

---CORNER POINT COORDINATES---
POINT(1,1) = XTADI
POINT(2,1) = 0.0
POINT(3,1) = -RDO

POINT(1,2) = XCI - RTADI
POINT(2,2) = 0.0
POINT(3,2) = -RCI
SEGMAX(2, 2, 7) = RCO*CTH1
SEGMAX(3, 2, 7) = RCO*STH1

SEGMAX(2, 3, 7) = RCO*CTH4
SEGMAX(3, 3, 7) = RCO*STH4

SEGMAX(2, 1, 8) = RDI*CTH
SEGMAX(3, 1, 8) = -RDI*STH

SEGMAX(2, 2, 8) = RDI*CTH1
SEGMAX(3, 2, 8) = RDI*STH1

SEGMAX(2, 3, 8) = RDI*CTH4
SEGMAX(3, 3, 8) = RDI*STH4

SEGMAX(1, 1, 9) = XCI
SEGMAX(3, 1, 9) = RDO

SEGMAX(1, 1, 11) = XCO
SEGMAX(3, 1, 11) = RDI

---EDGE NODE DISTRIBUTION---

ETAMAX(1, 1) = 13.0
ETAMAX(1, 3) = 13.0
ETAMAX(1, 9) = 13.0
ETAMAX(1, 11) = 13.0

DO 380 J = 7, 8

ETAMAX(1, J) = 36.0
ETAMAX(2, J) = 60.0
ETAMAX(3, J) = 103.0

ETAMAX(1, 5) = 36
ETAMAX(1, 6) = 36

RETURN

---INITIALIZE EDGE COEFFICIENTS---

400  COEFE(1, 2, 1) = 0.0
     COEFE(3, 2, 1) = 0.0

     COEFE(1, 2, 3) = 0.0
     COEFE(3, 2, 3) = 0.0

     COEFE(1, 2, 9) = 0.0
     COEFE(2, 2, 9) = 0.0
     COEFE(3, 2, 9) = 0.0

     COEFE(1, 2, 11) = 0.0
     COEFE(2, 2, 11) = 0.0
     COEFE(3, 2, 11) = 0.0

---EDGE COEFFICIENTS---
C---EDGE 1: CIRCULAR ARC
C
    COEFE(1,1, 1) = XCI
    COEFE(3,1, 1) = -RCI
C
C---EDGE 3: CIRCULAR ARC
C
    COEFE(1,1, 3) = XCO
    COEFE(3,1, 3) = -RCO
C
C---EDGE 9: CIRCULAR ARC
C
    COEFE(1,1, 9) = XCI
    COEFE(3,1, 9) = RCI
C
C---EDGE 11: CIRCULAR ARC
C
    COEFE(1,1,11) = XCO
    COEFE(3,1,11) = RCO
C
C---EDGE : CIRCULAR ARC
C
    DO 410 J = 1,4
C
    COEFE(1,J,5) = XCI - RTADI
    COEFE(1,J,8) = XCO - RTADO
    COEFE(1,J,6) = XTADO
     410  COEFE(1,J,7) = XTADO
C
C---SURFACE COEFFICIENTS-----------------------------------------------
C
    COEFS(1,2) = 0.0
    COEFS(4,2) = 1.0
    COEFS(7,2) = 3.0
C
    COEFS(1,4) = 0.0
    COEFS(4,4) = 1.0
    COEFS(7,4) = 3.0
C
    COEFS(1,5) = 0.0
    COEFS(4,5) = 1.0
    COEFS(7,5) = 3.0
C
    COEFS(1,6) = 0.0
    COEFS(4,6) = 0.0
    COEFS(7,6) = 0.0
C
C---CORNER NODE COORDINATES-------------------------------------------
C
    POINT(1,1) = XCI - RTADI
    POINT(3,1) = -RCI
C
    POINT(1,2) = XTADO
    POINT(3,2) = -RADI
C
    POINT(1,3) = XTADO
    POINT(3,3) = -RADIO
C
    POINT(1,4) = XCO - RTADO
    POINT(3,4) = -RCO

C-81

LOCKHEED-HUNTSVILLE ENGINEERING CENTER
C
POINT(1,5) = XCI - RTADI
POINT(3,5) = RCI

C
POINT(1,6) = XTADO
POINT(3,6) = RADII

C
POINT(1,7) = XTADO
POINT(3,7) = RADIO

C
POINT(1,8) = XCO - RTADO
POINT(3,8) = RCO

C---EDGE SEGMENT COORDINATES-----------------------------

C
SEGMAX(1,1, 1) = XCI
SEGMAX(3,1, 1) = -RWI21

C
SEGMAX(1,2, 1) = -2.128
SEGMAX(3,2, 1) = -6.514

C
SEGMAX(1,1, 3) = WWO21
SEGMAX(3,1, 3) = -RWO21

C
SEGMAX(1,2, 3) = -2.2
SEGMAX(3,2, 3) = -7.5

C
DO 420 J = 1,3
420 SEGMAX(1,J, 5) = XCI - RTADI
SEGMAX(1,J, 8) = XCO - RTADO

C
SEGMAX(2,1, 5) = RCI*CTH
SEGMAX(3,1, 5) = -RCI*STH

C
SEGMAX(2,2, 5) = RCI*CTH1
SEGMAX(3,2, 5) = RCI*STH1

C
SEGMAX(2,3, 5) = RCI*CTH4
SEGMAX(3,3, 5) = RCI*STH4

C
DO 440 J = 1,3
440 SEGMAX(1,J, 6) = 0.0
SEGMAX(1,J, 7) = 0.0

C
SEGMAX(2,1, 6) = RADII*CTH
SEGMAX(3,1, 6) = -RADII*STH

C
SEGMAX(2,2, 6) = RADII*CTH1
SEGMAX(3,2, 6) = RADII*STH1

C
SEGMAX(2,3, 6) = RADII*CTH4
SEGMAX(3,3, 6) = RADII*STH4

C
SEGMAX(2,1, 7) = RADIO*CTH
SEGMAX(3,1, 7) = -RADIO*STH

C
SEGMAX(2,2, 7) = RADIO*CTH1
SEGMAX(3,2, 7) = RADIO*STH1

C-82

LOCKHEED-HUNTSVILLE ENGINEERING CENTER
SEGMAX(2,3, 7) = RADIO*CTH4
SEGMAX(3,3, 7) = RADIO*STH4
SEGMAX(2,1, 8) = RCO*CTH
SEGMAX(3,1, 8) = -RCO*STH
SEGMAX(2,2, 8) = RCO*CTH1
SEGMAX(3,2, 8) = RCO*STH1
SEGMAX(2,3, 8) = RCO*CTH4
SEGMAX(3,3, 8) = RCO*STH4
SEGMAX(1,1, 9) = XCI
SEGMAX(3,1, 9) = RWI21
SEGMAX(1,2, 9) = -2.128
SEGMAX(3,2, 9) = 6.514
SEGMAX(1,1,11) = XWO21
SEGMAX(3,1,11) = RWO21
SEGMAX(1,2,11) = -2.2
SEGMAX(3,2,11) = 7.5

---EDGE NODE DISTRIBUTION---

ETAX(1, 1) = 13.0
ETAX(1, 3) = 13.0
ETAX(1, 9) = 13.0
ETAX(1,11) = 13.0
ETAX(2, 1) = 31.0
ETAX(2, 9) = 31.0
ETAX(2, 3) = 31.0
ETAX(2,11) = 31.0

DO 388 J = 7,8
388 ETAX(1, J) = 36.0
ETAX(2, J) = 60.0
ETAX(3, J) = 103.0

ETAX(1,5) = 36
ETAX(1,6) = 36

500 CONTINUE
RETURN
END

******************************************************************************
HGM 2******************************************************************************

SUBROUTINE HOLE(B, POINT, TANGENT)

COMPUTE COORDINATES AND DERIVATIVE FOR A POINT ON THE HOLE.
B  ANGULAR LOCATION OF A POINT ON THE HOLE
POINT  COORDINATES AT THE POINT
TANGENT  DERIVATIVE AT THE POINT

COMMON /DFN1/ DU1(3),DU2(3),DU3(3),DFNB,DFND,DFNF
DIMENSION POINT(6),TANGENT(3),BC(3),US(3)

CALL DELRAD(B,RHO,DRHO,RHOD,RDOC)
CALL CAXIS(B,RHO,C)

SB = SIN(B)
CB = COS(B)

BC(1) = CB*DU2(1) + SB*DU3(1)
BC(2) = CB*DU2(2) + SB*DU3(2)
BC(3) = CB*DU2(3) + SB*DU3(3)

XD = DFNF - DFND + C*DU1(1) + RHO*BC(1)
YD = DFNB + C*DU1(2) + RHO*BC(2)
ZD = C*DU1(3) + RHO*BC(3)

POINT(1) = XD + DFND
POINT(2) = YD
POINT(3) = ZD

PMAG = SQRT(XD*XD + YD*YD + ZD*ZD)
US(1) = XD/PMAG
US(2) = YD/PMAG
US(3) = ZD/PMAG

RETURN
END

SUBROUTINE DUCT(B,RHO,CAXIS,POINT)

COMPUTE DUCT POINT = X,Y,Z FOR GIVEN B,RHO,C

C-84

LOCKHEED-HUNTSVILLE ENGINEERING CENTER
C

COMMON /DFN1/ DU1(3), DU2(3), DU3(3), DFNB, DFND, DFNF
C
DIMENSION POINT(6), BC(3)
C
--- POINT ANGLE
C
CB = COS(B)
SB = SIN(B)
C
--- RADIAL UNIT VECTOR
C
BC(1) = CB*DU2(1) + SB*DU3(1)
BC(2) = CB*DU2(2) + SB*DU3(2)
BC(3) = CB*DU2(3) + SB*DU3(3)
C
--- COORDINATES
C
POINT(1) = DFNF + CAXIS*DU1(1) + RHO*BC(1)
POINT(2) = DFNB + CAXIS*DU1(2) + RHO*BC(2)
POINT(3) = CAXIS*DU1(3) + RHO*BC(3)
C
RETURN
END
C

--- Compute the difference in radius between duct and hole for the lower corner points on the hole.
--- B = angular location of a point on the hole
--- DRHO = difference between hole and duct radius
C

SUBROUTINE RHOS(B, DRHO)
C
COMMON /DFN1/ DU1(3), DU2(3), DU3(3), DFNB, DFND, DFNF
COMMON /DFN2/ AE, BE, DFNR, ZSTAR, AGL, BETA1, BETA2, BETA3, BETA4
COMMON /DFN3/ AD, BD, BETA(37), DELRHO(37), RADOC(37)
C
--- TOLERANCE FOR NEWTON-RAPHSON ITERATION
C
FEPS = 1.0E-07
C
--- ANGLE OF DUCT AXIS
C
SA = SIN(AGL)
CA = COS(AGL)
C
--- ANGULAR LOCATION OF A POINT ON THE HOLE
C
SB = SIN(B)
CB = COS(B)
C
--- RADIUS OF ELLIPTIC DUCT
C
FD = SQRT((AD*SB)**2 + (BD*CB)**2)
RHOD = AD*BD/FD

C-85
LOCKHEED-HUNTSVILLE ENGINEERING CENTER
C---INITIAL HOLE RADIUS
C  RHO = RHOD + DRHO
C
C---DISTANCE ALONG MAJOR AXIS OF BOWL ELLIPSE------------------------
C  10  XMD = DFNF - DFND + ZSTAR*SA/CA + RHO*CB/CA
C
C---ELLIPSE RATIO OF BOWL
C  BOA = BE/AE
C
C---DISTANCE ALONG MINOR AXIS OF BOWL ELLIPSE
C  F1 = SQRT(AE**2 - XMD**2)
C
C---RADIAL DISTANCE TO HOLE
C  F2 = DFNR + BOA*F1
C
C---VERTICAL DISTANCE TO HOLE
C  F3 = DFNB + RHO*SB
C
C---HORIZONTAL DISTANCE TO HOLE - ZSTAR**2
C  FR = F2**2 - F3**2 - ZSTAR**2
C
C---DERIVATIVE
C  DF1 = BOA*XMD*CB*F2/(F1*CA)
C  DF2 = F3*SB
C  DFDR = -2.*(DF1 + DF2)
C
C---HOLE RADIUS
C  RHO = RHO - FR/DFDR
C  IF(ABS(FR).GT.FEPS) GO TO 10
C
C---DIFFERENCE IN RADIUS BETWEEN HOLE AND DUCT------------------------
C  DRHO = RHO - RHOD
C
C  RETURN
C
END
C*****************************************************************************
C*****************************************************************************
C*****************************************************************************
C*****************************************************************************
C*****************************************************************************
C*****************************************************************************
C*****************************************************************************
}

C-86

LOCKHEED-HUNTSVILLE ENGINEERING CENTER
COMMON /INITA/ MATPEN, INCHES
COMMON /INIC/ PI, RADDEG

DIMENSION VECTOR(3)

ONE = 1.0

DO 10 I = 1, 3
  10 IF(ABS(VECTOR(I)).LT.0.00001) VECTOR(I) = 0.0

THETA = 0.0
PHI = 0.0

CALL VMAG(VECTOR, VECMAG)

IF(VECMAG.EQ.0.0) RETURN

20

IF(ABS(VNORM2).GT.1.0) VNORM2 = SIGN(ONE, VNORM2)

THETA = ASIN(VNORM2)*RADDEG
XZPLANE = SQRT(VECTOR(1)**2 + VECTOR(3)**2)

IF(XZPLANE.LT.ABS(VECTOR(2))/1000.) RETURN

IF(ABS(VNORM3).GT.1.0) VNORM3 = SIGN(ONE, VNORM3)

PHI = ACOS(VNORM3)*RADDEG
PHI = -PHI

RETURN

END

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

SUBROUTINE SPEDGE(SRI, SRF, EPS, SAC, SEDGE)

C CIRCULAR SUB-EDGE COORD., SIDE 4 OF BOWL

DIMENSION SRI(6), SRF(6), SEDGE(6), SAC(3), VDUM(3)
DIMENSION PC1(6), PC2(6), PVEC1(3), PVEC2(3)
DIMENSION UN(3), UP(3), UR(3), XN(3), XP(3), XR(3)

DATA PI /3.141592654/

DO 10 J = 1, 6
  10 PC1(J) = SRI(J)

DO 20 J = 1, 3
  20 PVEC1(J) = PC1(J) - SAC(J)

CALL VMAG(PVEC1, RM1)
CALL VMAG(PVEC2,RM2)
CALL CROSS(PVEC2,PVEC1,UN,71)
CALL CROSS(PVEC1,UN,UP,72)
CALL CROSS(UN,UP,UR,73)
RCC = RM1*RM2
CALL VDOT(PVEC1,PVEC2,RR)

THETA = PI
IF(ABS(RR).LE.RCC) THETA = ACOS(RR/RCC)
RC = 1.0/SIN(THETA)

THETA1 = THETA
ANG1 = (1.0 - EPS)*THETA
ANG2 = EPS*THETA
CANG1 = COS(ANG1)
SANG1 = SIN(ANG1)
CANG2 = COS(ANG2)
SANG2 = SIN(ANG2)

DO 40 J=1,3
40 SEDGE(J) = SAC(J) + RC*(SANG1*PVEC1(J) + SANG2*PVEC2(J))
CALL VDOT(PC1(4),PC2(4),RR)
IF(RR.GT.0.9999) GO TO 70
CALL CROSS(PC2(4),PC1(4),XN,74)
CALL CROSS(PC1(4),XN,XP,75)
CALL CROSS(XN,XP,XR,76)
THET = ACOS(RR)
ALPH = EPS*THET
CANG = COS(ALPH)
SANG = SIN(ALPH)
CALL VADD(CANG,XR,SANG,XP,VSUM,SEDGE(4))
RETURN
70 CONTINUE
DO 80 J=1,3
80 SEDGE(J+3) = PC1(J+3)
RETURN
END
SUBROUTINE TADWALL

COMPUTE TANGENT POINT ON OUTER TAD WALL

TASQ = (XTADO - XCO)**2 + (RADIO - RCO)**2 - RTADO**2
TAL = SQRT(TASQ)
ETA = TAL/RTADO
ETASQ = ETA**2
A = 1.0 + ETASQ
B = RADIO + ETASQ*RCO
BSQ = B**2
C = TASQ - RADIO**2 - ETASQ*RCO**2
DSQ = 1.0 + (A*C/BSQ)
D = SQRT(DSQ)
RWO21 = B*(1.0 + D)/A
XWO21 = XTADO - (RWO21 - RCO)*ETA

RETURN
END
Appendix D

HGM OUTPUT REWRITE LISTING
SUBROUTINE REWRITE

THIS PROGRAM CONVERTS HGM2DUCT CODE GEOMETRY FILES TO
PLOT3D GEOMETRY FILES

DIMENSION IDIM(110), JDIM(110), KDIM(110)
DIMENSION XBUF(200000), YBUF(200000), ZBUF(200000)

MATE = 0

---ASSIGN UNITS---

DATA I20, INPUT, ISCR, IP3D /1, 2, 3, 4/

OPEN(UNIT=I20, FILE='HGM2DUCT.DAT', STATUS='OLD')
OPEN(UNIT=INPUT, FILE='REWRITE.OUT', STATUS='NEW')
OPEN(UNIT=ISCR, STATUS='SCRATCH', FORM='UNFORMATTED')

---COUNTERS---

I2 = 0
NGRID = 0

---READ PARAMETERS FROM GEOMETRY---

IERR = 1
READ(I20, 1000, END=200) NSTORE, IPLN,
& INOD2, JNOD2, KNOD2, MARCH2

---WRITE DATA RANGE---

I1 = I2 + 1
I2 = I2 + NSTORE
WRITE(INPUT, 1210)
WRITE(INPUT, 1220) NSTORE, IDYN, IPLN, INOD2, JNOD2, KNOD2,
& MARCH2, I1, I2

---READ X, Y & Z FROM GEOMETRY---

IERR = 2
READ(I20, 1010, ERR=400) (XBUF(I), I=I1, I2)
IERR = 3
READ(I20, 1010, ERR=400) (YBUF(I), I=I1, I2)
IERR = 4
READ(I20, 1010, ERR=400) (ZBUF(I), I=I1, I2)
IERR = 5

IPLANE = 1

D-1

LOCKHEED-HUNTSVILLE ENGINEERING CENTER
C---READ NEXT SET OF PARAMETERS FROM GEOMETRY
C 100 READ(I20,1000,END=200) NSTORE,IPLN,INOD,JNOD,KNOD,MATE,MARCH
C---CHECK FOR NEW ZONE-------------------------------------------------------------
C IF(INOD.NE.INOD2 .OR. JNOD.NE.JNOD2 .OR. KNOD.NE.KNOD2) THEN
C IF(MARCHZ.EQ.1 .AND. JNOD.EQ.JNOD2 .AND. KNOD.EQ.KNOD2) GO TO 110
IF(MARCHZ.EQ.2 .AND. INOD.EQ.INOD2 .AND. KNOD.EQ.KNOD2) GO TO 110
IF(MARCHZ.EQ.3 .AND. JNOD.EQ.JNOD2 .AND. INOD.EQ.INOD2) GO TO 110
C NGRID = NGRID + 1
C IF(MARCHZ.EQ.1) THEN
C      INOD2 = IPLANE
C      IDIM(NGRID) = KNOD2
C      JDIM(NGRID) = JNOD2
C      KDIM(NGRID) = INOD2
C END IF
C IF(MARCHZ.EQ.2) THEN
C      JNOD2 = IPLANE
C      IDIM(NGRID) = INOD2
C      JDIM(NGRID) = KNOD2
C      KDIM(NGRID) = JNOD2
C END IF
C IF(MARCHZ.EQ.3) THEN
C      KNOD2 = IPLANE
C      IDIM(NGRID) = JNOD2
C      JDIM(NGRID) = INOD2
C      KDIM(NGRID) = KNOD2
C END IF
C WRITE(INPUT,1230)
WRITE(INPUT,1220) NGRID,IDIM(NGRID),JDIM(NGRID),KDIM(NGRID)
C IERR = 6
C WRITE(ISCR,ERR=411)
1          (XBUF(I),I=1,I2),
2          (YBUF(I),I=1,I2),
3          (ZBUF(I),I=1,I2)
C WRITE(INPUT,1250)
C INOD2 = INOD
JNOD2 = JNOD
KNOD2 = KNOD
C MARCH2 = MARCH
C WRITE(INPUT,1210)
I2 = 0
IPLANE = 0

END IF

WRITE DATA RANGE--------------------------------------------

110 CONTINUE

I1 = I2 + 1
I2 = I2 + NSTORE

WRITE(INPUT,1220) NSTORE,IDYN,IPLN,INOD,JNOD,KNOD,MARCH,I1,I2

READ X,Y & Z FROM GEOMETRY

IERR = 7
READ(I20,1010,ERR=400) (XBUF(I),I=I1,I2)

IERR = 8
READ(I20,1010,ERR=400) (YBUF(I),I=I1,I2)

IERR = 1
READ(I20,1010,ERR=400) (ZBUF(I),I=I1,I2)

IPLANE = IPLANE + 1
GO TO 100

END OF GEOMETRY DATA---------------------------------

200 CONTINUE

NGRID = NGRID + 1

IF(MARCH2.EQ.1) THEN

INOD2 = IPLANE
IDIM(NGRID) = KNOD2
JDIM(NGRID) = JNOD2
KDIM(NGRID) = INOD2

END IF

IF(MARCH2.EQ.2) THEN

JNOD2 = IPLANE
IDIM(NGRID) = INOD2
JDIM(NGRID) = KNOD2
KDIM(NGRID) = JNOD2

END IF

IF(MARCH2.EQ.3) THEN

KNOD2 = IPLANE
IDIM(NGRID) = JNOD2

END IF
JDIM(NGRID) = INOD2
KDIM(NGRID) = KNOD2

END IF

WRITE(INPUT,1230)
WRITE(INPUT,1220) NGRID,JDIM(NGRID),KDIM(NGRID)
WRITE(INPUT,1240)

---WRITE X,Y & Z ON SCRATCH FILE

C IERR = 10
C WRITE(ISCR,ERR=411)
  1 (XBUF(I),I=1,I2),
  2 (YBUF(I),I=1,I2),
  3 (ZBUF(I),I=1,I2)
C WRITE(INPUT,1250)
C CLOSE(UNIT=120)
C WRITE(INPUT,1260)
C REWIND(UNIT=ISCR)
C WRITE(INPUT,1270)

---WRITE BINARY FILE

OPEN(UNIT=IP3D,FILE='PLT3D.BIN',STATUS='NEW',FORM='UNFORMATTED')
C WRITE(INPUT,1300)

IF(NGRID.GT.1) THEN
  IERR = 11
  WRITE(INPUT,1310) NGRID
  WRITE(IP3D,ERR=420) NGRID
END IF

IERR = 11
WRITE(INPUT,1320)
WRITE(INPUT,1330) (IDIM(N),JDIM(N),KDIM(N),N=1,NGRID)
WRITE(IP3D,ERR=420) (IDIM(N),JDIM(N),KDIM(N),N=1,NGRID)

---WRITE BINARY PLT3D FILE

DO 300 N=1,NGRID
  IERR = 12
  I2 = IDIM(N)*JDIM(N)*KDIM(N)
  C

300

D-4

LOCKHEED-HUNTSVILLE ENGINEERING CENTER
READ(ISCR,ERR=410)
  1            (XBUF(I),I=1,12),
  2            (YBUF(I),I=1,12),
  3            (ZBUF(I),I=1,12)
C
        IERR = 13
C
        WRITE(IP3D,ERR=420)
  1            (XBUF(I),I=1,12),
  2            (YBUF(I),I=1,12),
  3            (ZBUF(I),I=1,12)
C
        WRITE(INPUT,1340) N
C
  300 CONTINUE
C
        WRITE GRID NUMBER
C
        WRITE(INPUT,1350) NGRID
C
        GO TO 500
C
        ERRORS
C
        ERROR READING GEOMETRY DATA FROM FILE20
C
        400 WRITE(INPUT,1400) IERR
C
        GO TO 500
C
        ERROR READING SCRATCH FILE
C
        410 WRITE(INPUT,1410) IERR
        411 WRITE(INPUT,1411) IERR
C
        GO TO 500
C
        ERROR WRITING TO PLOT3D BINARY FILE
C
        420 WRITE(INPUT,1420) IERR
C
        END OF PROGRAM
C
  500 CONTINUE
C
        CLOSE(UNIT=ISCR)
        CLOSE(UNIT=IP3D)
        CLOSE(UNIT=INPUT)
C
        FORMAT STATEMENTS
C
  1000 FORMAT(915)
  1010 FORMAT(6E22.14)
C
  1200 FORMAT(/' *** ERROR: RERUN GEOMN WITH MATE SET TO 0 ***')
  1210 FORMAT(/'STORE IDYN IPLN INOD JNOD KIOD MARCH II 12')
  1220 FORMAT(1X,715,216)
  1230 FORMAT(/' NGRID IDIM JDIM KDIM')
  1240 FORMAT(/' END OF FILE REACHED ON 120')
  1250 FORMAT(/' GRID WRITTEN TO ISCR')

D-5

LOCKHEED-HUNTSVILLE ENGINEERING CENTER
1260 FORMAT(//' I20 CLOSED')
1270 FORMAT(//' ISCR REWIND')

C
1300 FORMAT(//' IP3D OPENED')
1310 FORMAT(//' NGRID =',I3)
1320 FORMAT(//' IDIM JDIM KDIM ')
1330 FORMAT(//' 315')
1340 FORMAT(//' GRID ',I2, ' WRITTEN TO PLT3D.BIN')
1350 FORMAT(//' THERE ARE ',I2, ' GRIDS WRITTEN TO PLT3D.BIN')

C
1400 FORMAT(//' ERROR READING GEOMETRY DATA FROM FILE20 ON UNIT I20',I2)
1410 FORMAT(//' ERROR READING SCRATCH FILE ON UNIT ISCR',I2)
1411 FORMAT(//' ERROR WRITING SCRATCH FILE TO UNIT ISCR',I2)
1420 FORMAT(//' ERROR WRITING TO PLOT3D BINARY FILE ON UNIT IP3D',I2)

C
STOP
END

C