Deformation Analysis of Rotary Combustion Engine Housings

Final Report on NASA Grant NAG 3-456

by
Carl Vilmann
Michigan Technological University

(NASA-CR-188187) DEFORMATION ANALYSIS OF ROTARY COMBUSTION ENGINE HOUSINGS Final Report (Michigan Technological Univ.) 361 p N91-27158
CSCL 21E Unclas
G3/07 0012362

NASA Grant NAG 3-456, "Deformation Analysis of Rotary Combustion Engine Housings," was initiated with three objectives in mind. The first of these objectives was the generation of a detailed finite element model of a rotary engine's center (trochoid) housing. Once this model had been generated, the second objective of the project, the prediction of the stress and deformation fields within the trochoid housing during engine operation, could be attacked. Finally, the third objective of this work was the development of a preprocessor which would simplify the generation of subsequent finite element models for alternate center housing designs. The purpose of this preprocessor was to greatly shorten the development time for modified trochoid housings. While the first and third objectives were fully met, the second objective proved difficult to achieve. The following report details the work done at MTU on NASA Grant NAG 3-456.

Objective 1

Development and Verification of a Finite Element Model the Trochoid Housing

Initially the engine type under investigation was manufactured by Mazda. A picture of this engine's trochoid housing is included as Figure 1. The final finite element model for this housing is

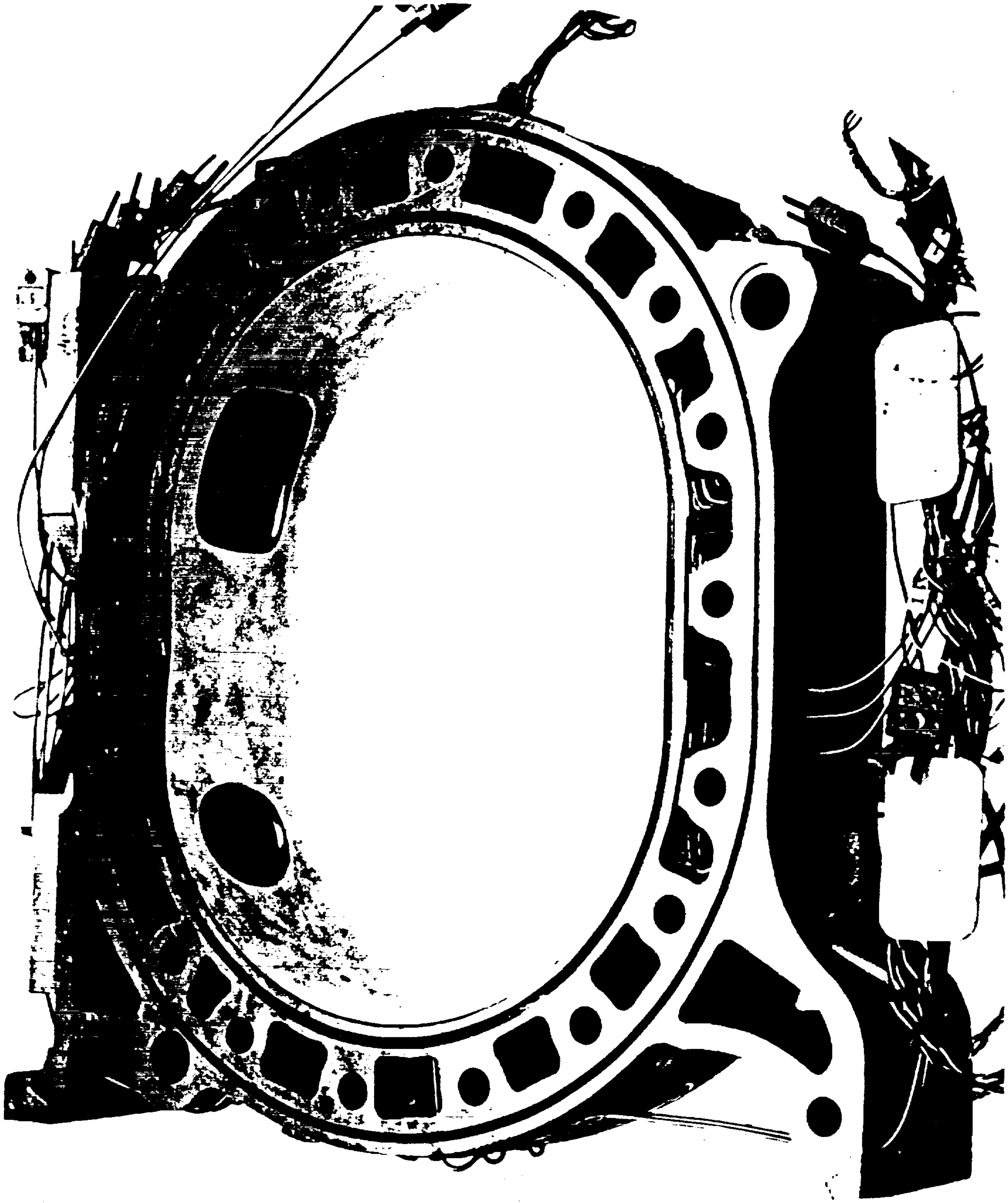


Figure 1

Mazda Type Trochoid Housing

shown in Figure 2. This model was developed and tested as the M.S. thesis of Scott Bradley. During Scott's development process various element types were examined for their suitability and different mesh densities were explored to determine the coarsest mesh which would model the housing's response accurately. In order to verify the response of the finite element model, a comparison was made with experimentally measured stresses reduced from strain gage measurements. The discrepancy between the experimental results and the finite element predictions were found to be acceptable, and the model shown in Figure 2 was accepted for use in the dynamic simulation phase of the project. A more detailed description of the development and testing of this model is contained in Scott's M.S. thesis which is attached as Appendix A.

Objective 2

Prediction of the Stress and Deformation Fields Present within the Trochoid Housing During Operating Conditions

At this point in the project NASA changed the engine under investigation from the Mazda engine to an engine produced by Outboard Marine Corporation (The OMC engine). The geometric similarities between the Mazda engine and the OMC engine were few. While the basic trochoidal shape of the bore remained the same, the cooling channel pattern was radically different. This change required the complete regeneration of a finite element model. The information on mesh refinement gained from the development of a FEM model for the Mazda engine was useful, but the input of the geometrical characteristics needed to be completely redone. An M.S. student was enlisted to generate and exercise a model of the OMC engine. Unfortunately, this student left after one year without making much progress. A second student (A M.S. student who remained at MTU after finishing his degree.) with FEM experience was employed to help with the modelling. A finite element model was generated and delivered after nine months. When an attempt was made to exercise this model, it was discovered that elements with unacceptable geometric shapes were present within the model. In order to eliminate these unacceptable elements, the project director revised the model. The final finite element model of the OMC center housing is shown in Figure 3. In viewing this model it should be noted that since the trochoid housing is symmetric about its midplane, only 1/2 of the housing needed to be modeled.

With the model complete, the next step in the prediction of the stress and deformation fields present during operating conditions was the calculation of the model shapes and natural frequencies of the housing. The boundary conditions necessary were generated, and the job was submitted to the MTU computational facility. Unfortunately, the operating system present on the MTU computer did

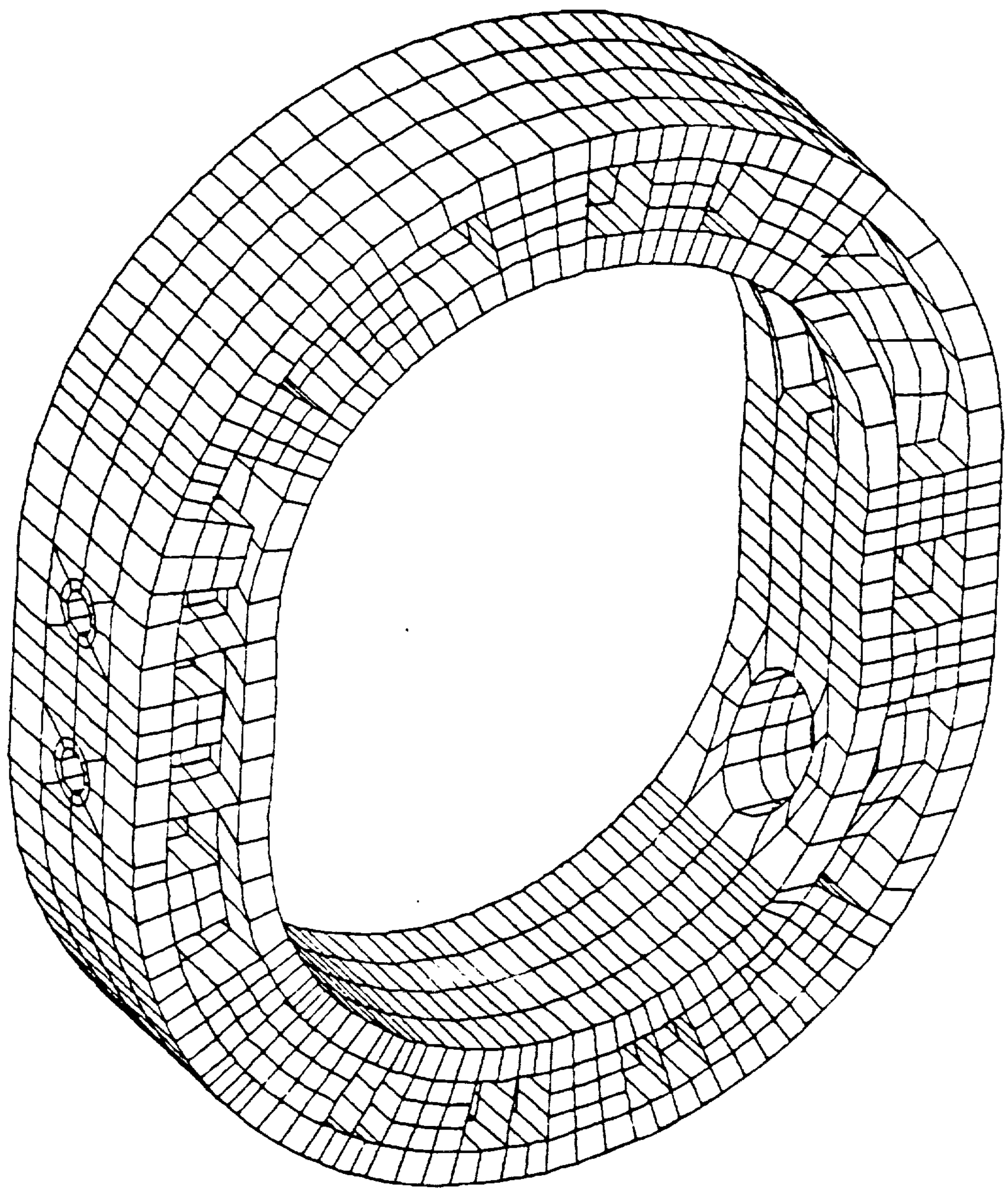


Figure 2

**Finite Element Mesh
for the
Mazda Type Trochoid Housing**

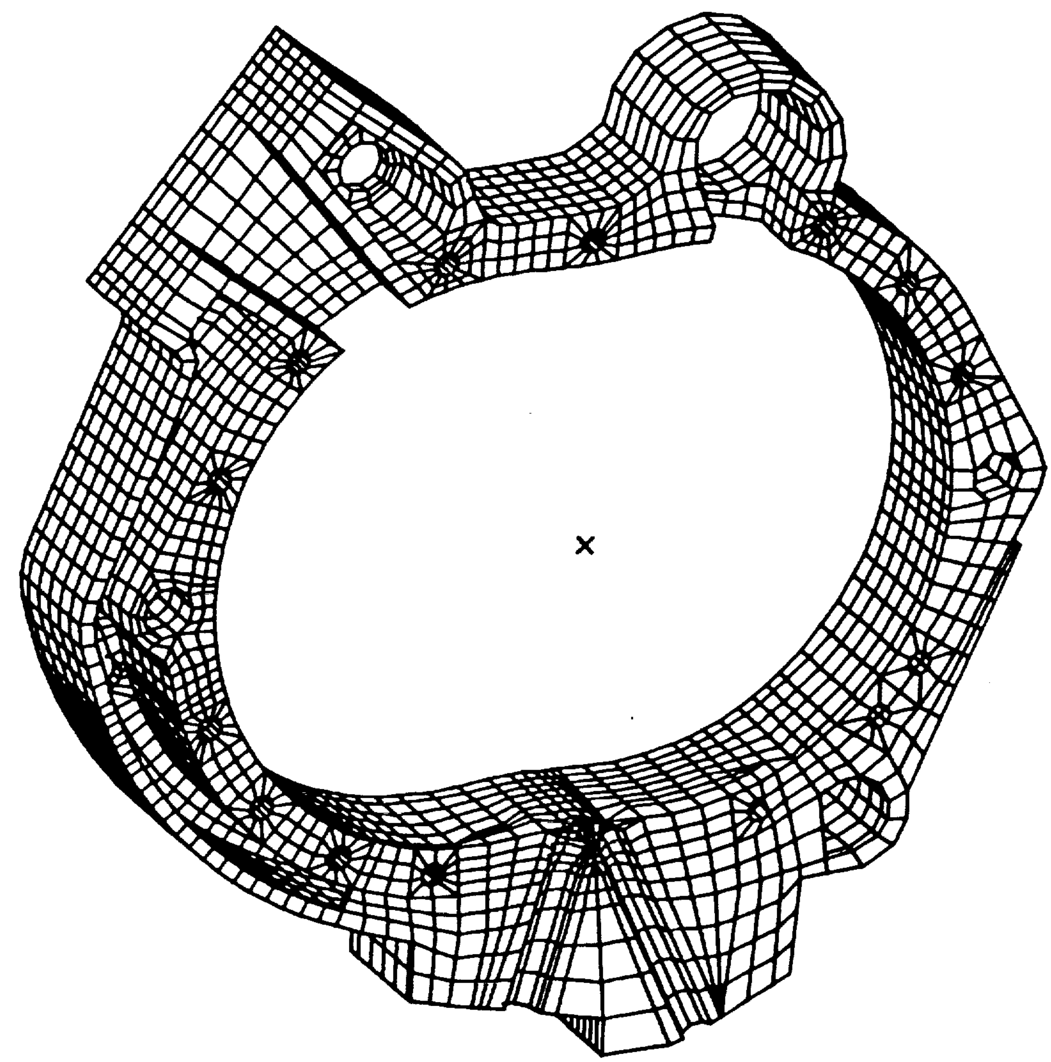


Figure 3

Finite Element Mesh
for the
OMC Type Trochoid Housing

not allow file lengths of sufficient magnitude to allow calculation of the dynamic characteristics of this model. After conferring with NASTRAN consultants, a fix to this problem was discovered. The model needed to be divided into substructures. These subsets of the original model would then be treated separately and finally computationally recombined, thereby eliminating the file length problem. This substructuring was performed by the project director, and the revised model was once again submitted. Time estimates for completion of the job calculated by NASTRAN were on the order of 150 hours. Since it is virtually impossible to complete a job of that length without the MTU system going down, it was decided that competing this work with the computational facility at MTU was unfeasible.

At this point in the project a NASA Cray account was requested and received. The FEM model of the OMC housing was loaded on tape and forwarded to Lewis Research Center. The eigenvalue problem associated with determining the mode shapes and natural frequencies was completed and the lowest natural frequencies calculated are listed are shown in Table 1.

The modal results were then ready for recombination in order to determine the stress history at the placed where the experimental data had been extracted. This required the transfer of the eigenvalue results back to MTU. At this point another obstacle was encountered. The file transfer routines required for the transfer

Table 1

Lowest Natural Frequencies for the Trochoid Housing

Mode	Natural Frequency (Cycles/Sec)
1	1621
2	1908
3	1981
4	2458

of large amounts of data between MTU and NASA were not working correctly. The project director spent approximately one month trying to find a solution to this problem and finally obtained a band-aid, temporary, fix to the data transfer problem.

This is where the dynamic analysis is at the present time. The delays have been unfortunate. The work remaining on the dynamic analysis includes the recombination of the modal response to the pressure loading on the housing, and the superposition of these results with the thermal stresses arising from the temperature distribution present within the engine during operating conditions. Since the final report is now being requested, these tasks remain unfinished.

Objective 3

Development of a Specialized Preprocessor for the Trochoid Housing

The objective of this portion of the project was to develop a software product which would simplify the preparation of finite element models for a rotary engine's trochoid housing. Since a Mazda engine was initially under consideration, and since there were no indications early in the project that a switch to an alternate engine type would be desired or necessary, the preprocessor developed generates the complete finite element mesh for a rotary engine center housing of the Mazda type. This preprocessor shortens the development time necessary for mesh generation of a trochoid housing's FEM model from roughly one man month to approximately two man hours. The creation of the is preprocessor was the M.S. thesis of W. Lychuk.

In developing this preprocessor it was decided that duplication of the commercially available software for generating a finite element mesh within a defined geometric volume was unnecessary. Therefore, the preprocessor developed was written to prepare data for the commercial preprocessor SUPERTAB. SUPERTAB, now part of the IDEAS package available from Structural Dynamics Research Corporation was in place both at NASA and at Michigan

Tech. The Michigan Tech preprocessor generates a program file for SUPERTAB. After the input parameters detailed in Figure 4 are supplied to the MTU preprocessing software, an output file results which can be used to drive SUPERTAB to complete a finite element model of the housing. The housing parameters which can be altered include:

1) The maximum and minimum diameters of the trochoidal bore.
2) The thickness and axial depth of the center housing.
3) The thickness of both the inner and outer shells.
4) The thickness of the bore insert.
5) The size, type, and location of each rib.
6) The type of each channel.
7) The size and location of the intake port.
8) The size and location of the exhaust port.
9) The size and location of the spark plug ports.

The details of the development of this preprocessor are contained in the M.S. thesis of W.M. Lychuk. A copy of this thesis is attached as Appendix B. A listing of the code developed as the aforementioned preprocessor is included as Appendix C.

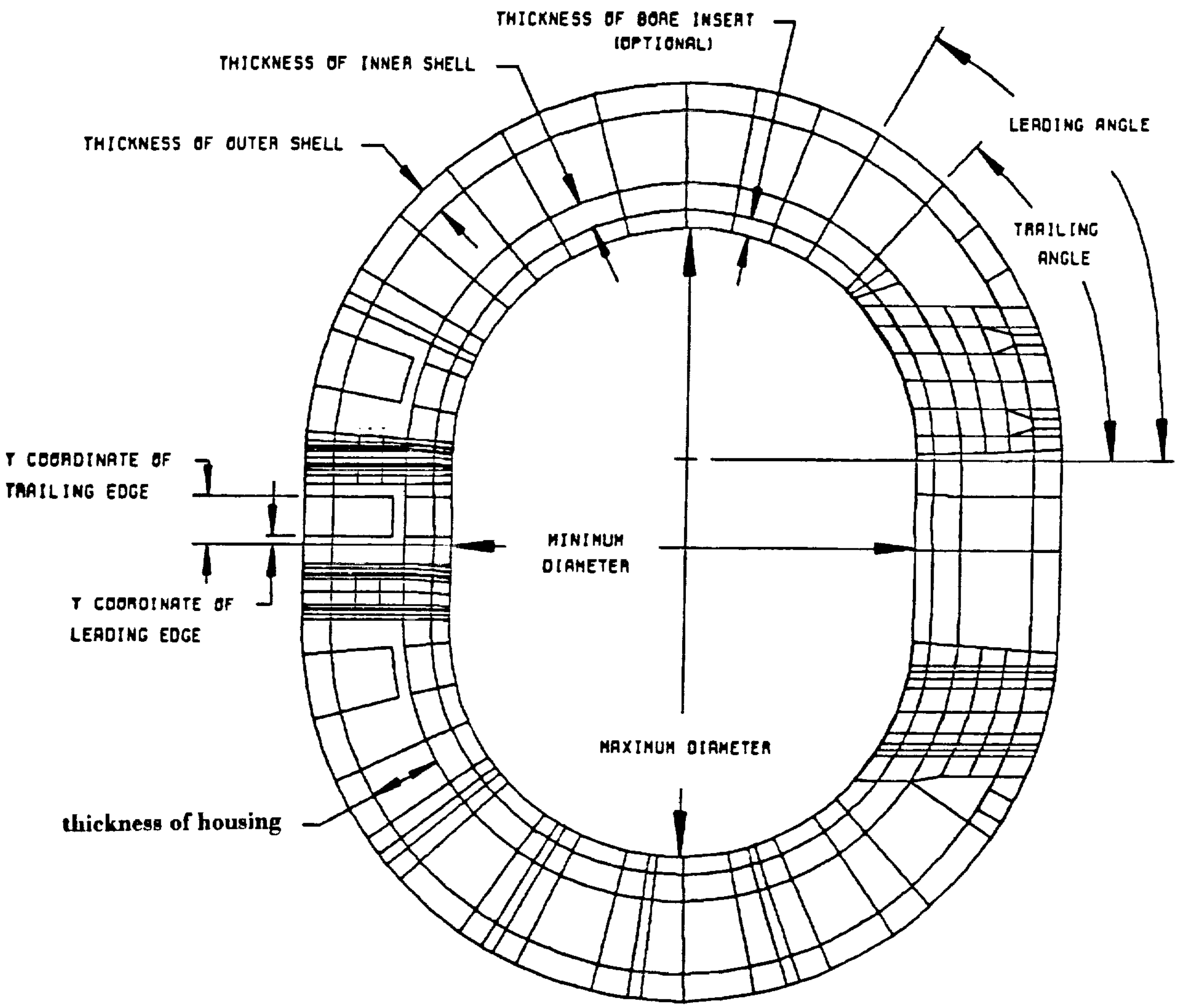


Figure 4

Input Parameters for the MTU Preprocessor

Conclusion

Problems with personal and computational facilities were responsible for this project falling short of all of its originally stated objectives. In spite of these problems, detailed, executable, finite element models were developed for both the Mazda and the OMC trochoid housings. In addition, it was demonstrated that a preprocessor which would hasten the generation of finite element models of a rotary engine could be developed. Two publications resulted from this grant and summarize the work detailed in Appendices A & B. These publications are:

1) "Finite Element Model Development of Multiple Rotary Combustion Engine Housing Configurations," by C.E. Passerello, C.R.Vilmann, S.A. Bradley, and W.M. Lychuk, published in the Proceedings of the CADAM User Exchange, May 1985 meeting in New Orleans, LA.

2) "Stress and Deformation Modeling of Multiple Rotary Combustion Engine Trochoid Housings," by W.M. Lychuk, S.A. Bradley, C.R. Vilmann, C.E. Passerello, and C. Lee, SAE paper number 860614, published in the

proceedings of the SAE International Congress and Exposition, Feb. 1986 meeting in Detroit MI.

While all of the initial objectives were not met, they were, possibly, too ambitious given the constraints that were present. As a final note, the project director would like to express his disappointment in not being able to completely fulfill the original goals, and thank NASA for their support of this project.

Appendix

A

FINITE ELEMENT MODELING OF A WANKEL ENGINE CENTER HOUSING

by

Scott Bradley

A THESIS

Submitted in partial fulfillment of the requirements

for the degree of

MASTER OF SCIENCE IN ENGINEERING MECHANICS

MICHIGAN TECHNOLOGICAL UNIVERSITY

1985

ABSTRACT

In this study, construction of a finite element model of a Wankel rotary combustion engine's center housing is presented. Analysis of the model is purely static, however, thermal and dynamic analysis considerations are taken into account. Verification of the model is accomplished through experimental tests on the actual housing. A convergence study is also completed by varying element types and interpolation orders. From the analysis and subsequent testing, it is believed that constuction of a valid FEM of a Wankel engine's center housing has been completed.

ACKNOWLEDGEMENTS

The author would like to thank his advisor, Dr. Chris E. Passerello, for his invaluable assistance in preparing this thesis.

Thanks is also expressed to Dr. Carl R. Vilmann, Dr. L. Bouge Sandberg, and Dr. Madhukar Vable for taking time out from their busy schedules to review and examine the paper.

Further graditude is extended to the employees in the Michigan Technological University's graphics lab. Their expertise provided the necessary software to execute this study.

In addition, the author sincerely extends thanks to his parents. Without their confidence and support this study could not have been completed.

Most of all, to his wife, who never lost sight of their goals, the author expresses his undying love.

TABLE OF CONTENTS

LIST OF FIGURES

1.00 BACKGROUND

The Wankel rotary combustion engine (RCE) was initially introduced to the automotive industry in the early 1960's. Praised as the engine of the future, it is basically an internal combustion engine which operates on the otto cycle: intake, compression, combustion, and exhaust. It executes the cycle by utilizing a triangular shaped rotor within a trochoidal shaped working chamber, both of which are shown in figure 1.1. These working chambers, commonly known as the center housings, are located along the mainshaft and are separated by endcovers. The full engine configuration is shown in figure 1.2. Utilized in a variety of enviroments, the Wankel is a lightweight, compact, smooth running engine which has multifuel capability and can be stratified charged very easily (1).

1.10 BASIC OPERATION

In the RCE, each face of the rotor can be considered a combustion chamber sliding along the inner bore. By rotating the rotor within the bore, the otto cycle can be simultaneously completed three times for every revolution of the rotor. This process is illustrated in figure 1.3. The combustion phase exerts pressure on one face of the rotor which produces a rotary motion and brings the next chamber into firing position. In order to produce the rotary motion, the pressure from combustion must be directed on a point away from the mainshaft centerline. For this reason, the rotor is mounted on eccentric bearings which provide the leverage required. Phasing between the rotor rotation and the eccentric bearing rotation is accomplished through the use of phasing gears. A part of this gearing is a

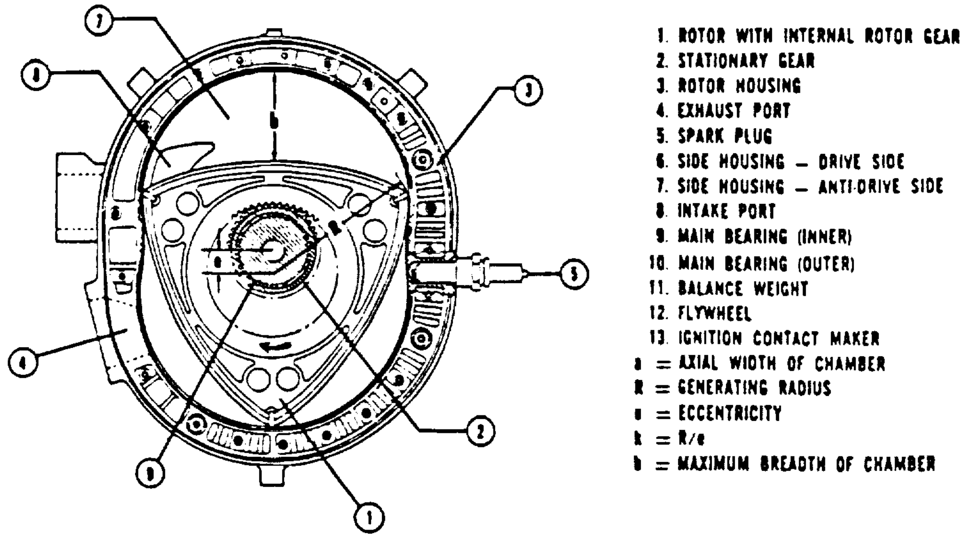


Figure 1.1: View of rotor and center housing.

(Courtesy of Chilton Book Co.)

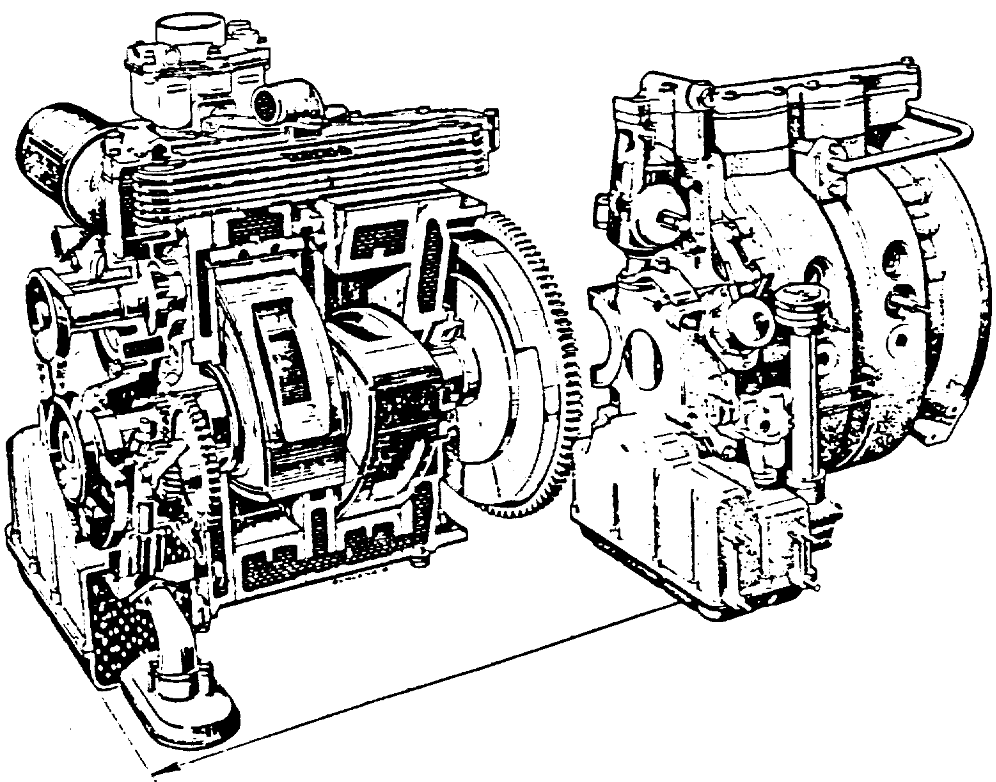


Figure 1.2: Full engine configuration.

(Courtesy of Chilton Book Co.)

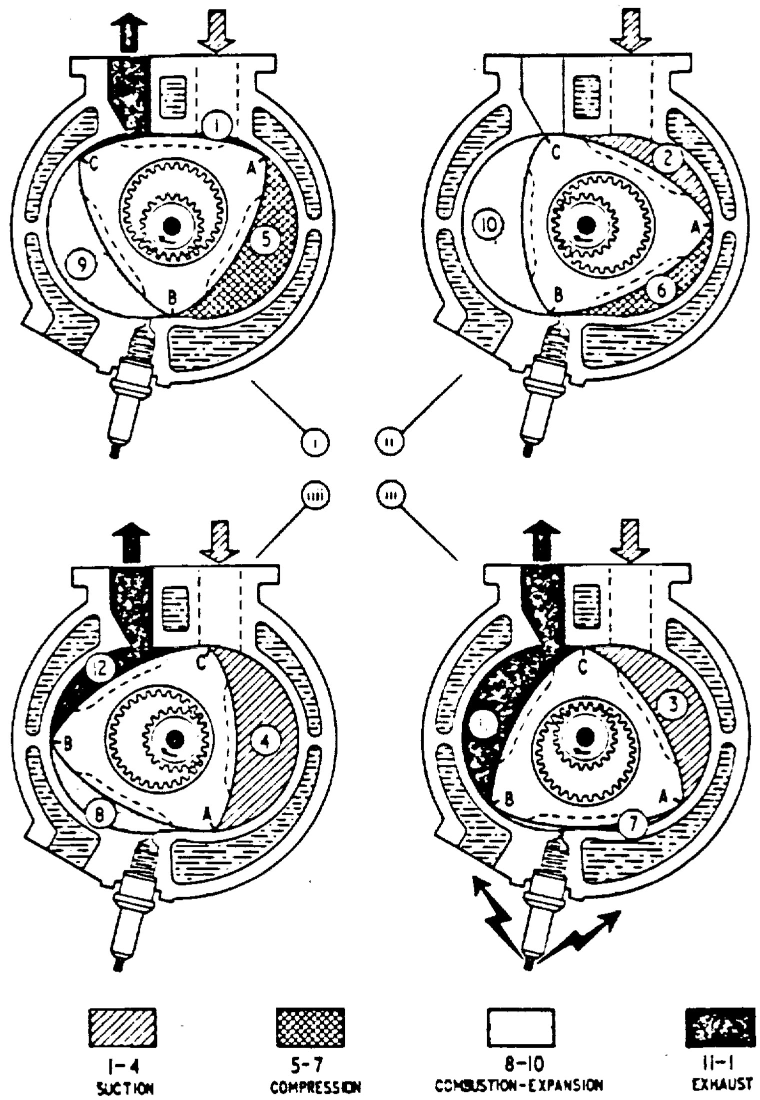


Figure 1.3: Combustion cycle cf Wankel engine.

(Courtesy of A. S. Barnes & Co.)

stationary reaction gear mounted to the end cover and concentric to the mainshaft. This reaction gear meshes with an inner ring gear on the rotor. A gear ratio of 3:2 must exist. This gear ratio assures that for every rotation of the rotor, the mainshaft completes three rotations. Thus, a positive torque is applied for every one-third rotation of the rotor which gives one complete power phase for each rotation of the mainshaft.

1.20 COMPARISON TO A PISTON ENGINE

When comparing the rotary combustion engine (RCE) to a reciprocating engine (RPE), certain distinctions are clear. Instead of utilizing a cylindrical piston and its associated connecting rod, the RCE uses a rotor centered around a mainshaft. Also, ports, which are opened and closed by the motion of the rotor, replace the complicated valves and valve trains of the RPE. With the connecting rods, camshafts, pushrods, and the other assorted parts eliminated, the RCE's size and weight is reduced to about half that of a RPE with comparable power output. In addition to fewer parts, there are fewer moving parts, indicating that the RCE operates at a lower manufacturing and repair cost. Fewer moving parts also produce less wear and power loss so reliability is high.

Furthermore, the use of a rotor eliminates the large inertia forces created by reciprocating masses. This, in addition to the application of positive torque over a full mainshaft rotation, leads to a smoother operating engine. Gas pressure loads on the bearings are higher in the RCE than in a piston engine. However, centrifugal loads are lower, thus the risk of bearing failure is far smaller which again makes the rotary engine highly reliable. Overall, the rotary combustion engine is theoretically far superior to the reciprocating piston engine.

1.30 PROBLEMS PLAGUING THE ROTARY ENGINE

Although the rotary engine's performance should be superior to a piston engine, actual operation and development have been plagued with problems. A 12% higher specific fuel consumption, higher hydrocarbon emissions, and excessive wear of the center housing bore have hampered the success of the engine. The cause of these problems seem to be centered around the sealing system, specifically the apex seal. The apex seals which are located on each tip of the rotor, tend to leave the inner trochoidal bore's surface just after ignition (2). When the seal retains contact with the surface it skids before complete semifrictionless contact is acquired. This liftoff and the subsequent skidding is thought to produce the effects discussed earlier. Many theories have been formulated on why apex seal liftoff occurs. Breakdown of the lubricating agents, vibration of the seal within its setting, and actual housing deformation are suspected causes. Changes in material and geometry can reduce effects but further study is required.

A few studies have been done on the effects of modifying the lubricating agents and seal properties: Behling and Weise (4) and Rogers, et. al. (5). There also have been studies performed on apex seal vibration : Prasse, McCormick, and Anderson (6), Matsura, et. al. (2) (7). Yet no study has been found in which a finite element deformation analysis of the center housing was performed. The need for a user friendly finite element model of the housing, which can be modified to encompass geometry and loading changes is therefore needed. The scope of such a project, though, is too large to be completed by this thesis. Thus, it is the purpose of this thesis to lay down a foundation on which future study can be done. This thesis will present the construction and validation of a finite element model of the housing alone. The model will be verified using static loading conditions with experimentally collected data.

1.40 CENTER HOUSING GEOMETRY

The housing used in this study, shown in figure 1.4, was provided by the NASA Lewis Research Center. It was obtained from a custom built, 573 cc (35 cubic inch), high performance single rotor engine. The housing is centered about the mainshaft and is sandwiched between two endplates. It is made of an aluminum alloy and has a steel liner with a thickness of approximately 1 mm within the trochoidal shaped bore. Cooling channels pass

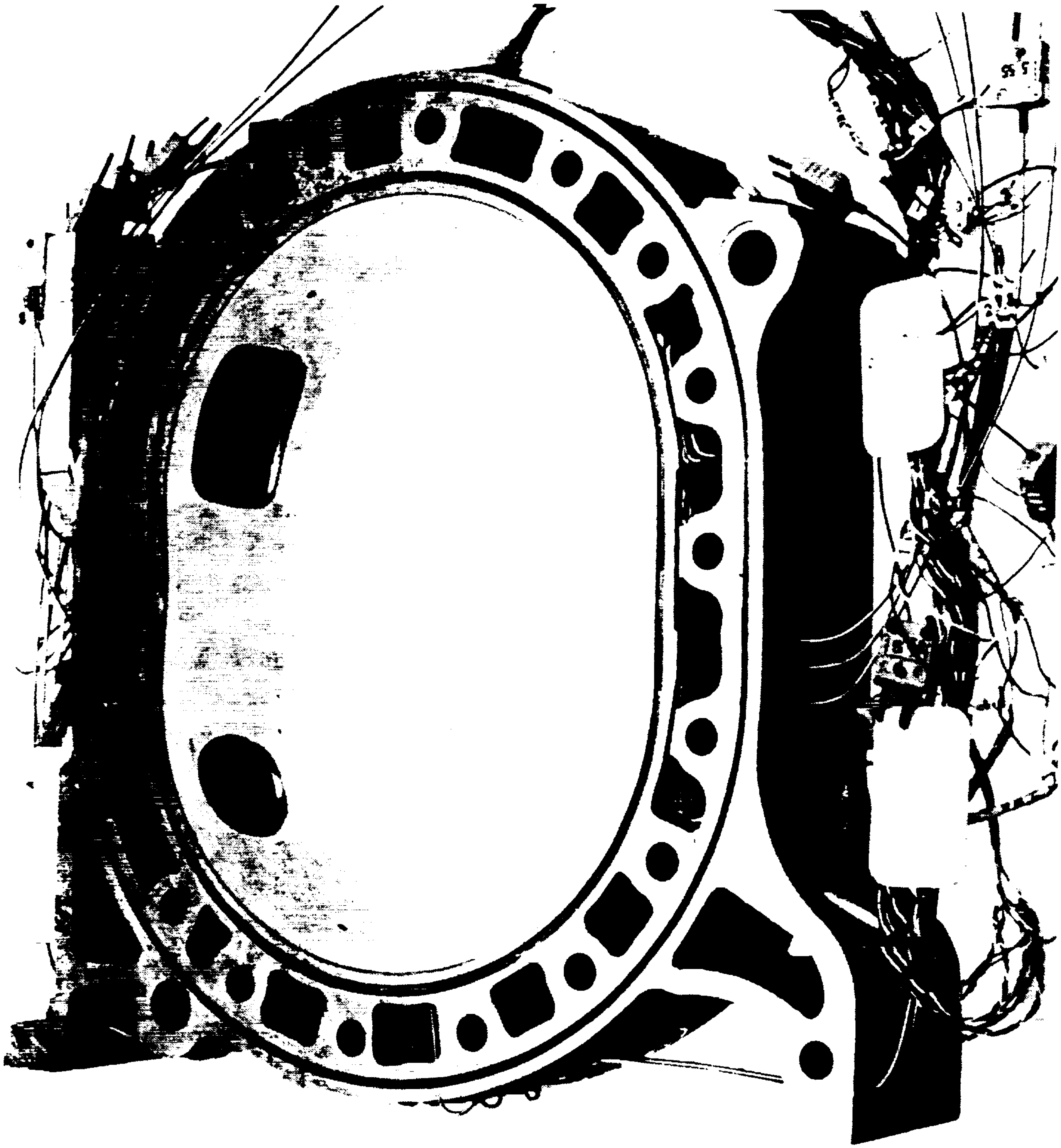


Figure 1.4: Actual housing supplied by NASA.

ORIGINAL PAGE IS
OF POOR QUALITY

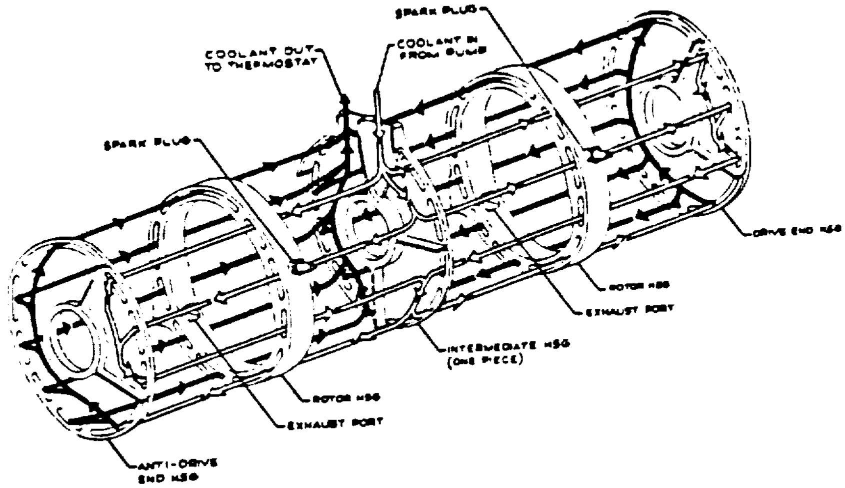


Figure 1.5: Coolant flow diagram.
(Courtesy of Chilton Book Co.)

through the housing to allow axial flow of coolant as shown in figure 1.5. Noticeable features of the housing include the intake and exhaust ports, spark plug locations, bolt holes, and engine mounts. An added feature is the inner surface around the spark plug region. In order to aid in thermal diffusion, the surface is thinned. Terminology that will be used to describe the housing is illustrated in figure 1.6.

1.50 TROCHODIAL BORE

A prominent feature of the Wankel RCE is the shape of the inner bore. Although it is of trochoidal shape, the generation of the trochoid is first accomplished by constructing a true epitrochoid. Since the definition of an epitrochoid is the loci of a point on the radius of a circle which rolls without slip around a fixed base circle, see figure 1.7, an infinite number of epitrochoids are available. Only by making the base circle's radius twice that of the rolling circle's radius, is the familiar two lobed epitrochoid obtained. Thus, the basic dimensions needed to describe the epitrochoid are the base circle radius (R1), the rolling circle radius (R2=R1/2), and the distance of the generating point from the center of the rolling circle, or the eccentricity (e).

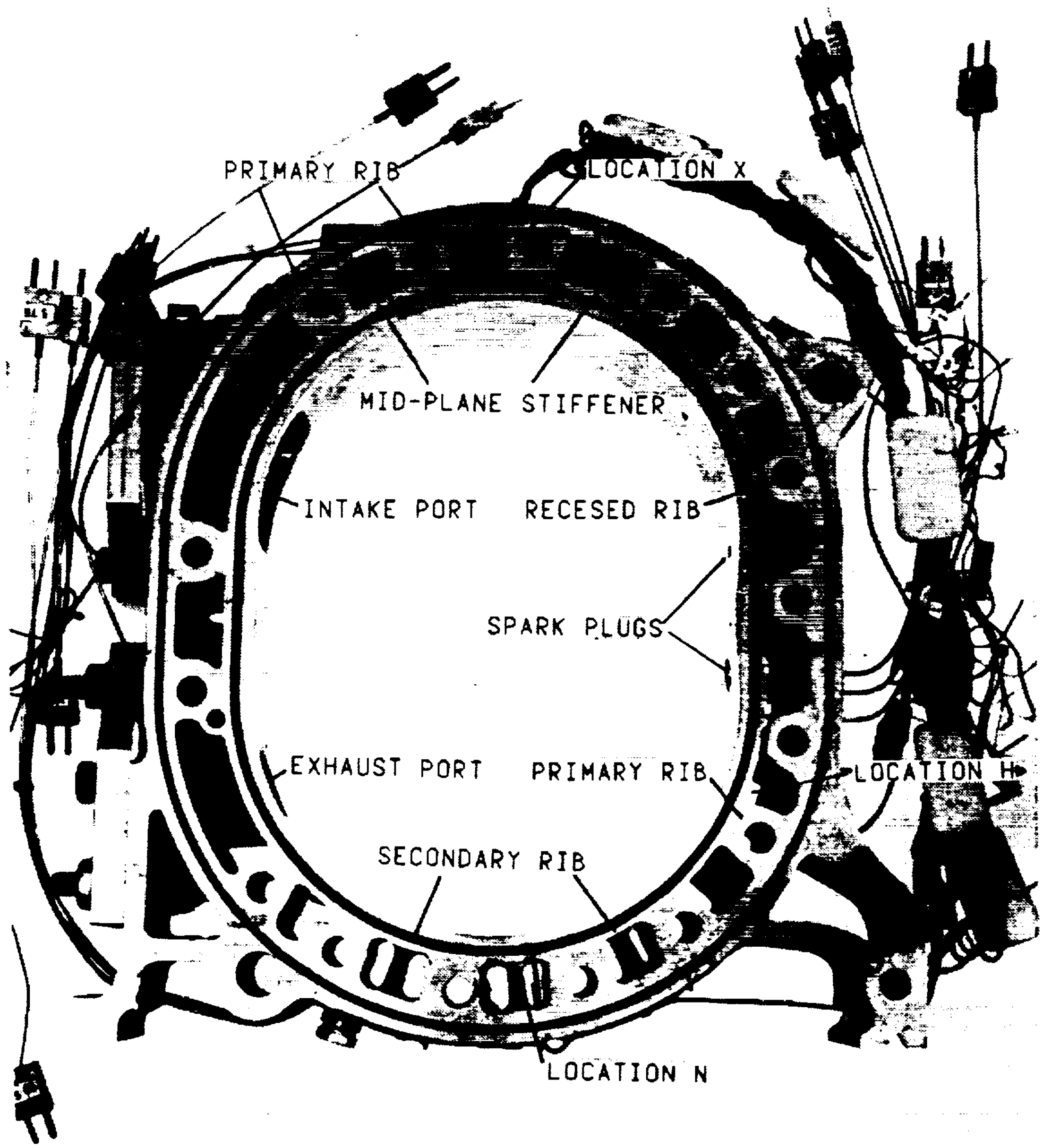


Figure 1.6: Illustration of the various regions of the housing.

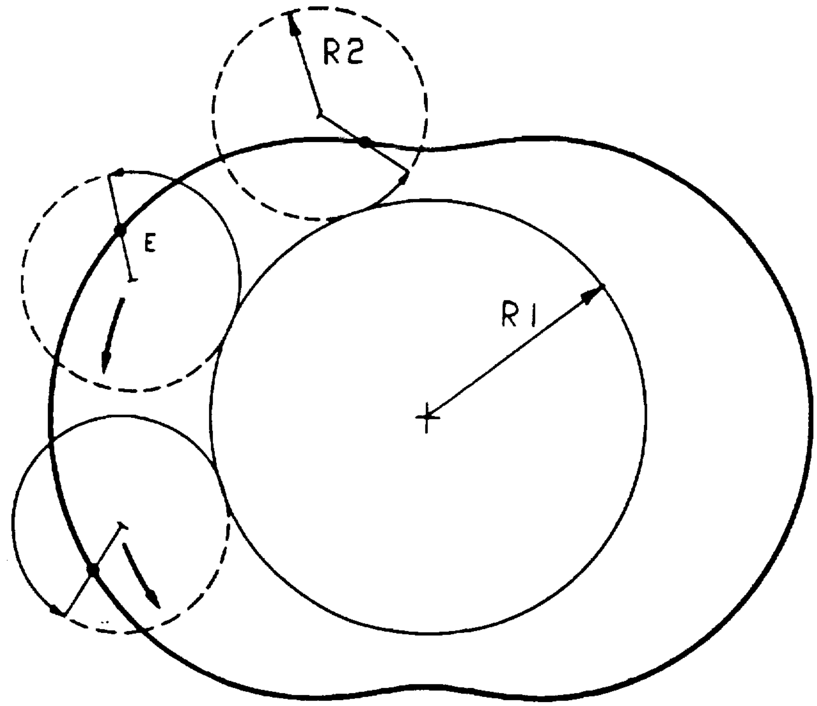


Figure 1.7: Illustration of epitrochoid generation.

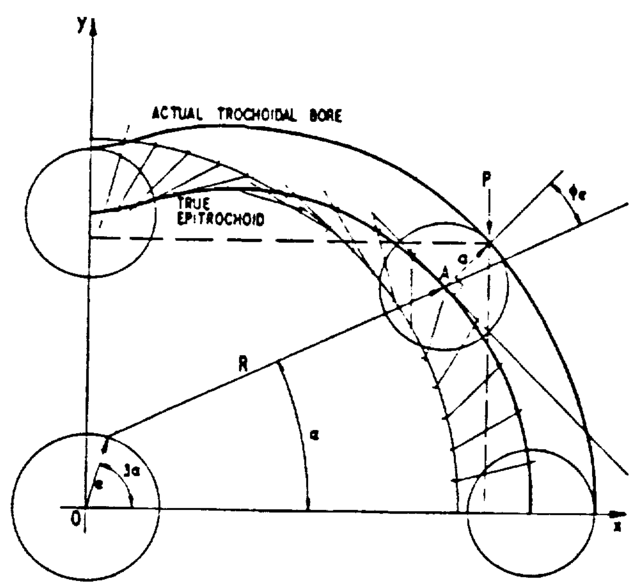


Figure 1.8: Illustration of trochoid generation..

If the true epitrochoid shape were utilized in the Wankel RCE, rotation of the rotor would force the seals to move in and out of their respective slots. Therefore, the trochoidal shape, shown in figure 1.8, is employed. By introducing the parameter (a), the translation between the apex seal and its containing slot is eliminated.

2.00 MODEL EXPECTATIONS

In determining how the finite element model (FEM) should be constructed, certain parameters were considered. Parameters such as model expectations, housing response, modeling accuracy, and result interpretation were defined and evaluated. With the optimum mix of these parameters, the FEM was constructed and tested to see if it accurately predicted the actual response to specified inputs in an efficient manner. The methods used in constructing the FEM and the manner in which they were verified are presented in this thesis.

Looking at the housing's geometry (see figure 1.4) and assuming the loading due to running conditions will be some type of fluctuating thermal and pressure loads, several conditions prevail. First, the inner and outer surfaces experience a complex dynamic loading which includes bending, membrane, and thermal stresses. Second, the primary ribs experience a simple fluctuating compressive load. Third, the spark plug and exhaust port regions are high thermal stress areas. Fourth and finally, no symmetry considerations are available since the endplates do not exhibit similar contact forces on either side of the housing.

From reviewing these conditions and the engine configuration, it is obvious that construction of a FEM that is capable of performing a dynamic analysis of the center housing under running conditions is far beyond the scope of a single thesis. Therefore, as an initial thrust toward a FEM fully capable of performing a dynamic analysis of the housing, a FEM was constructed and a static deformation analysis was performed. Furthermore, verification of the FEM was completed by utilizing the results from experimental testing of the actual housing. However, since the FEM of the housing is to be a base on which further study will be done it was built with the capability of completing all the required studies: thermal, dynamic, and endplate interaction, with only minor changes. So, though only a static analysis was done, the construction of the FEM was completed with the running conditions previously mentioned in mind.

2.10 MODELING REQUIREMENTS

With no previous studies to draw from, the initial thoughts on modeling the housing were to construct it as two elongated cylinders with solid connecting ribs. Presumably, the cylinders could be modeled with basic shell elements and the ribs with basic 3-D solid "brick" elements. The additional geometry, exhaust and intake ports, spark plug, secondary ribs, and mid-plane stiffeners could be constructed using various types of shell elements. Some added qualities, though, would be required for the elements in certain regions. For example, thermal stresses would be high in the exhaust port and spark plug areas so the elements modeling those areas would need the additional capability of modeling temperature distributions. The inner and outer surfaces would also require this capability. The structural characteristics of the housing would require the elements modeling the inner and outer surfaces to be able to model bending, membrane, and shearing actions. In the primary ribs and port regions, though, elements would undergo only very simple structural loading. Furthermore, the elements modeling the stiffening agents would be required to stiffen the area involved in the most accurate manner possible. These considerations, being just an initial estimate of what is required to model the housing, were built and expanded upon as the study proceded.

2.20 MODELING CAPABILITIES

During the study, several finite element codes were available: SAP6, SUPERB, and NASTRAN. But due to the complicated geometry, a graphical pre- and post-processor was required. SUPERTAB was, for most of the study, the only pre-processor available and was only interactive with SUPERB. Since the need for a more general FEM program such as NASTRAN or ANSYS wa not great and the need for a graphical pre-processor was, SUPE was chosen to solve the model and its supporting tests.

SUPERTAB creates the specific geometry using points, li edges, surfaces, and volumes. Meshes are placed on the defin geometry then nodes and elements are created according to the mesh size and configuration. The SUPERTAB file is then adjusted into SUPERB format. Using the nodal coordinates and element connectivity defined in this file, SUPERB solves the set of simultaneous equations obtained with an elemental wavefront technique. The wavefront of a finite element model is the number of degrees of freedom needed to be held by the computer in order to solve for a specific displacement. In general, an elemental wavefront solution differs from a nodal bandwidth solution in that a relatively small main frame computer is needed but an extreme amount of storage space is required.

When constructing geometry with SUPERTAB, a prescribed order must be followed (see figure 2.1). Points must be defined before lines, arcs, and splines can be created. Edges need to be defined before surfaces or volumes can be created. Also, the order in which construction occurs determines a number of key parameters. Nodal and elemental numbering, coordinate system orientation, and in the case of thick shell elements, actual element configurations are determined by the sequence of detecting existing pieces of geometry.

When defining the elements to be used, the specific element type must be selected. SUPERB has a limited finite element library but does support the basic types capable of modeling the housing. Interestingly enough, SUPERB supports two variations of shell elements: a 2-D thin shell and a 3-D thick shell element. Both of these shell elements have bending, membrane, and shearing

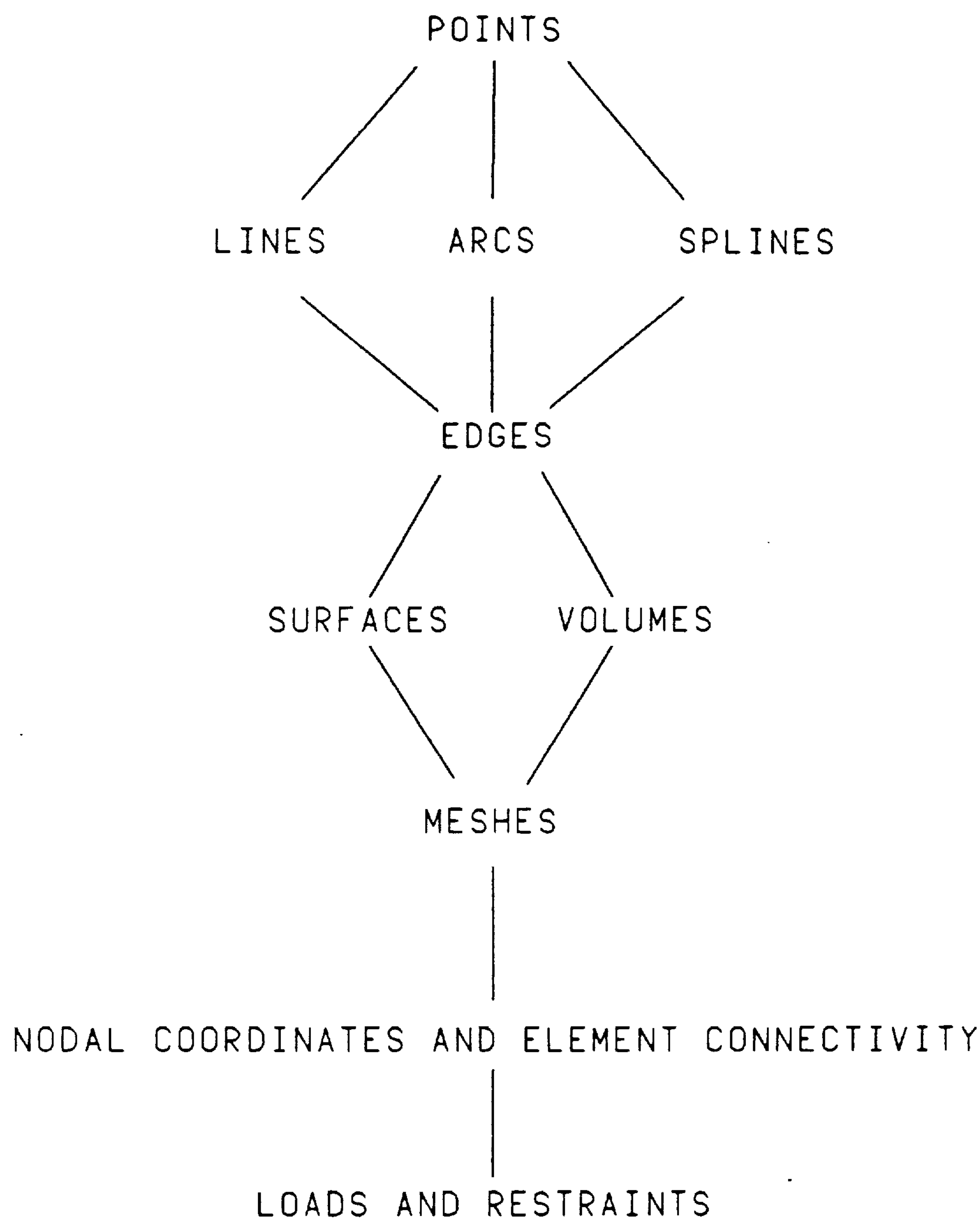


FIGURE 2.1: SUPERTAB GENERATION LEVELS.

capabilities. Also, both elements are capable of predicting in-plane temperature distributions, but only the thick shell element is capable of modeling a temperature distribution through the shell's thickness. Furthermore, each shell element is available in various nodal configurations. A more indepth discussion on each element type is listed in appendix 1.

Since two types of shell elements were available to model the complex and varied loading conditions, a benchmark study was completed. The study was designed to determine the qualities and performance of each shell element. Mesh sizes, aspect ratios, and nodal configurations were varied and the results analyzed. The actual tests and results are given in appendix 2. A basic summary of this study, though, reveals four important shell element characteristics:

1) Linear (4-node) thin shell elements are not capable of modeling shearing actions and therefore should not be used in areas where shearing actions are evident.

2) Linear (8-node) thick shell elements perform very well but converge slowly so a smaller mesh size is required.

3) Parabolic (8-node) thin and (16-node) thick shell elements perform extremely well and exactly alike.

4) The smallest recommended mesh size for modeling the housing would require at least six elements in the axial direction. A reduction to four elements would produce an estimated 10% error

These attributes, along with the considerations mentioned in appendix 1 provided the nescessary knowledge to begin modeling the housing.

2.30 MODEL CONSTRUCTION

With the necessary information on hand, actual model construction began. Proceding with the assumption that the housing could be constructed using a mix of solid and shell elements, ideas on how and where these elements would be used were discussed. Since a complex stress field was not expected, solid elements would be used exclusively in modeling the primary ribs. Furthermore, due to the high temperatures expected, thick shell elements would be required in the port and spark plug regions. For the remaining geometry: the inner and outer surfaces, secondary ribs and mid-plane stiffeners, the pros and cons of utilizing either thick or thin shell element were weighed. The thin shell element was simple to construct, contained fewer nodes than the thick shell element, and performed extremely well. However, there were difficulties in implementing the thin shell elements. For one thing, a compatibility problem was introduced whenever thin shell and solid elements would be joined. Also, the thin shell element assumed a constant thermal distribution through its thickness and, due to its construction, only the displacement of its middle surface was available. On the other hand, the thick shell element had excellent thermal as well as structural capabilities, portrayed the actual geomtry shape, and had no compatibility problems with solid elements. It does contain twice as many nodes as the thin shell element but, since each node supports half the degrees of freedom, the model's size is not affected. Taking this into account, the thick shell element was determined to be an excellent choice to model the inner and outer surfaces. The secondary ribs and mid-plane stiffeners, however, would be modeled with thin shell elements. This was because the adverse qualities of the thin shell element did not affect its performance in these regions.

The final aspect of the initial construction phase was that the model had to be constructed utilizing parabolic nodal configurations. The performance of the shell elements were greatly affected by this parameter. Furthermore, even though the solid elements performance was not affected by its nodal configuration, they also would utilize parabolic elements. This would ease construction since the use of transition elements would not be required.

Given the blueprint for the final model, outlined above, convergence requirements as well as computer limitations were considered. In order to satisfy both, a modeling sequence was established. Modeling construction methods as well as the model's capabilities were improved and enhanced as determined by previous models. Basically, the sequence consisted of four models: the linear thin shell model, a parabolic thin shell model, a parabolic thick shell model, and the final model which utilized all of the proven construction features. The actual construction and the subsequent analysis of each model is presented later on in this thesis.

3.00 CENTER HOUSE TESTING

After the construction sequence was determined and implemented, methods for verification of each model were considered. Undoubtedly, experimental tests of the actual housing were necessary. But, it was important that the testing be indicative of the dynamic stress fields the housing would encounter during operation. Thus, a test which applied a stress field that encompassed bending, membrane, shearing, and uniaxial effects in varying degrees was required. To accomplish this, two tests were performed. The initial test consisted of merely subjecting the housing to an internal pressure. However, due to complications in simulating this test, a second test was performed. This second test, a simple tensile test, was easily simulated and provided a loading that exercised each model's capability to predict a complex stress field.

3.10 UNIFCRM INTERNAL PRESSURE TEST (IPT)

The first test was performed by NASA. The housing was sealed and pressurized to 500 psi in increments of 100 psi. At each increment the strains in three locations were recorded. This was accomplished through the use of strain gage rosettes mounted by NASA. A graphical representation of the test and the location of the rosettes is given in figure 3.1. These strains are plotted in figures 3.2-3.4.

Several tests were performed and consistent data was obtained. Yet, in analyzing the data, certain discrepancies were noticed. First, the strains were scattered. Second, the maximum principal stress occurred in the axial direction instead of the

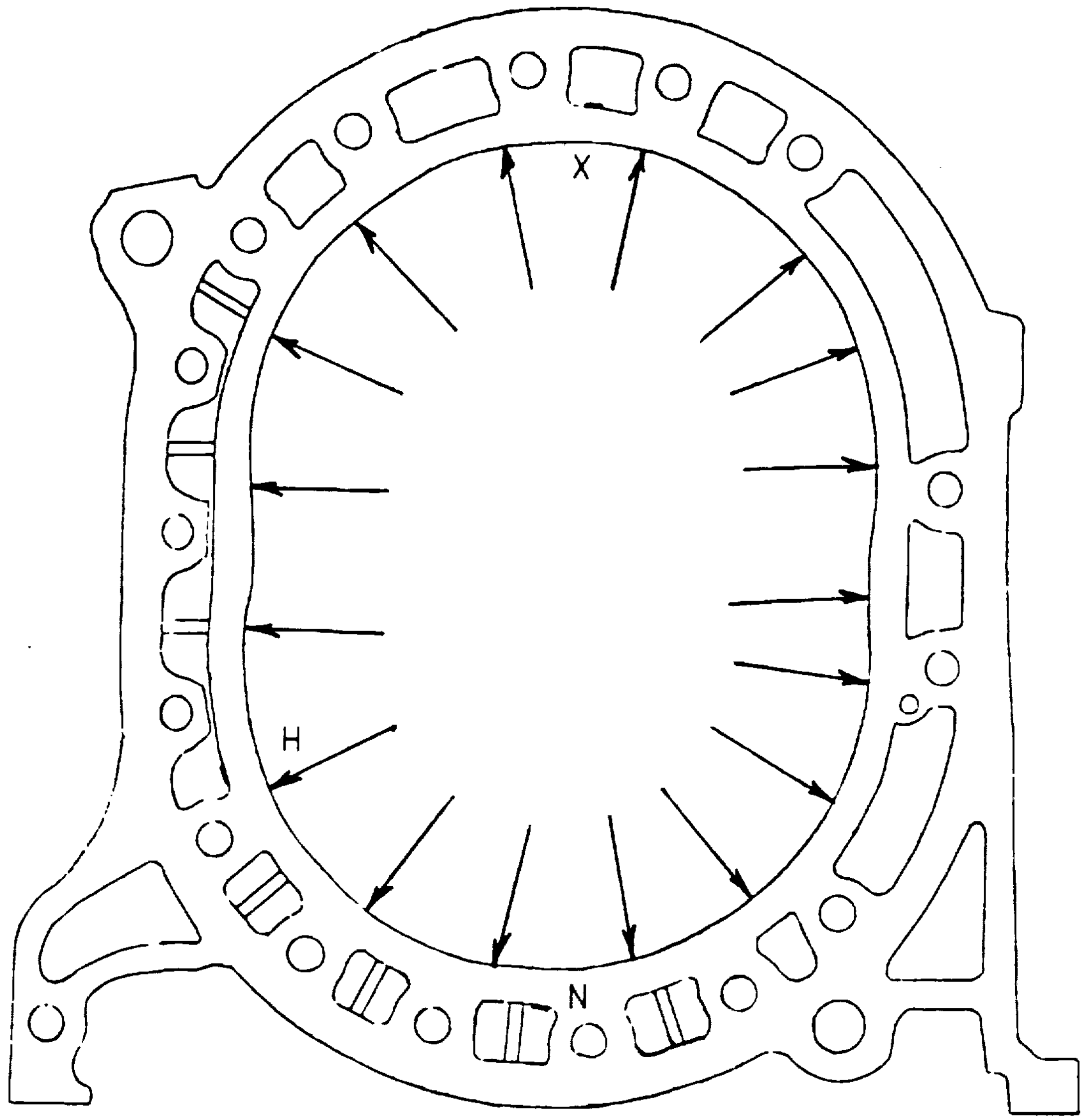


Figure 3.1: Illustration of the internal pressure test and locations of the strain gage rossettes.

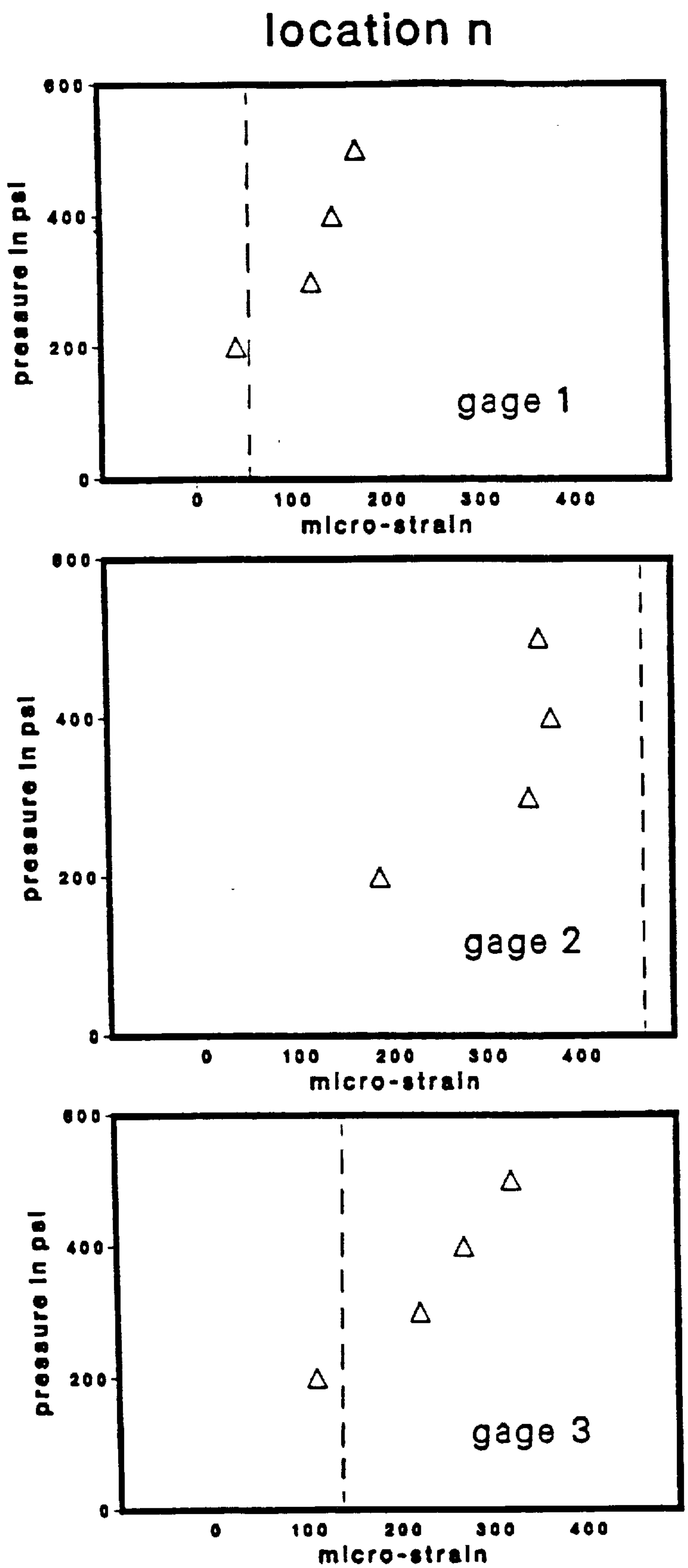


figure 3.2; internal pressure test.

location x

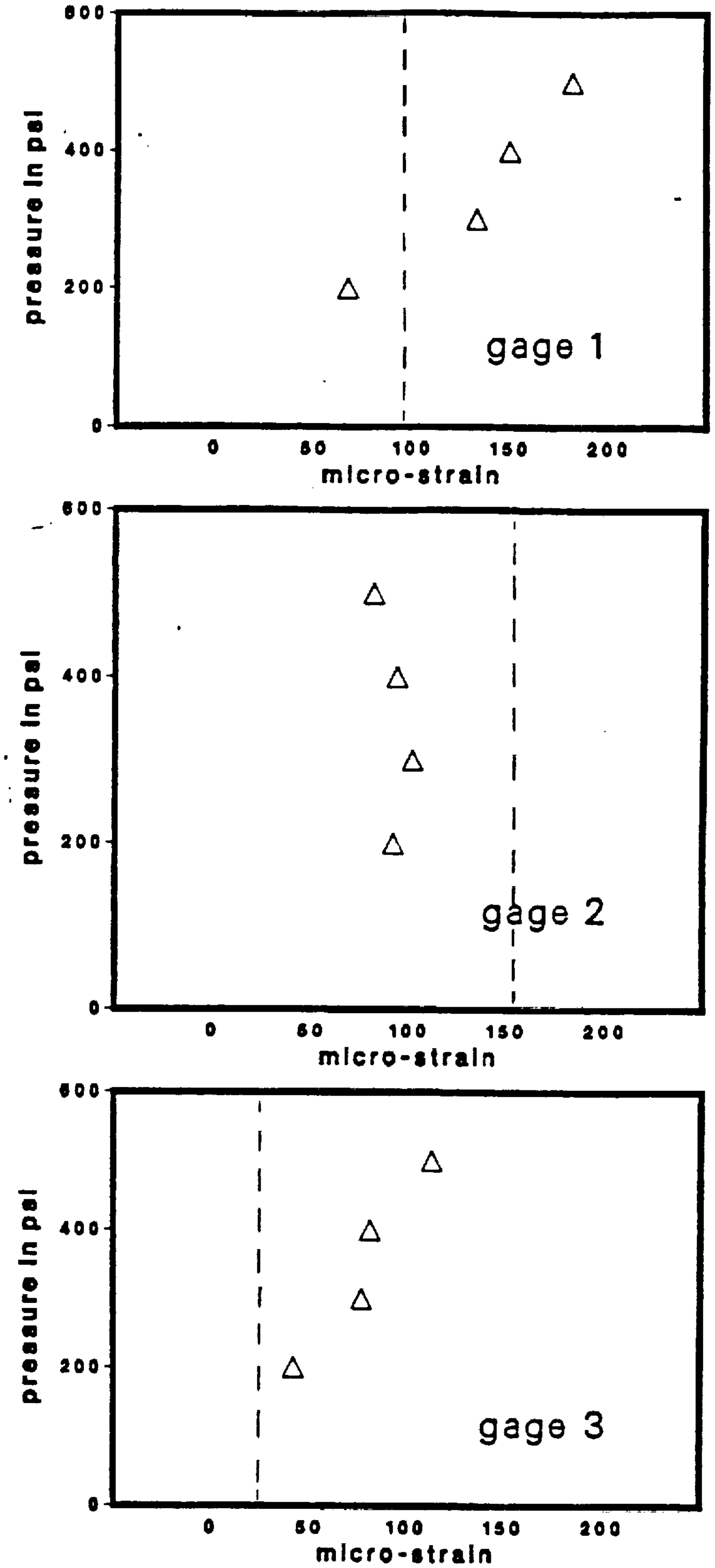


figure 3.4; internal pressure test.

location h

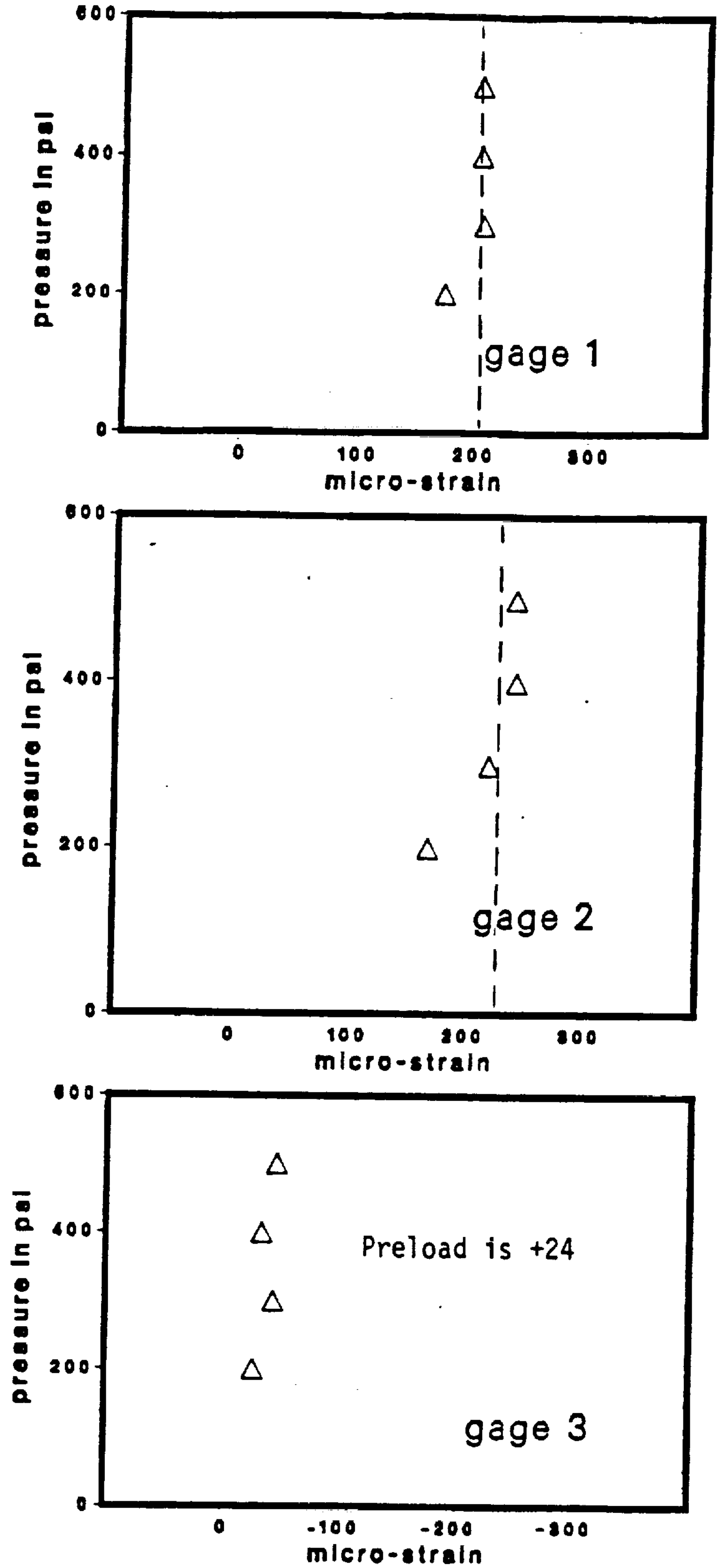


figure 3.3; internal pressure test.

hoop direction. Third and finally, the stress at location N, which is a heavily reinforced region, was computed to be larger than the stress at location H, which is a region with very little support. There were other puzzling features but these discrepencies alone forced further investigation into the test. Inevitably, this led to reviewing the test procedure and the manner in which it was executed.

The procedure was very straight forward. In order to pressurize the housing it needed to be sealed. This sealing was accomplished by bolting two aluminum plates to either side of the housing. After the bolts were tightened, the gages were zeroed and the housing was pressurized. Thus, the preload that initially existed on the housing from tightening the bolts had been neglected. When the housing was pressurized, the pressure proceded to relieve the compressive load which induced a tensile strain in the gages. The summation of the preload relief and the pressurization was what the gages actually read. In an attempt to grasp the effect of the preload, the bolts were loosened and resulting strains recorded. The vertical line on graphs 3.2-3.4 indicate these strains. As can be seen, the effect of the preload varied throughout the housing.

Theories were hypothesized on what was occuring within the housing. Summarized, these theories were that either the housing deformed until the clearance in the bolts was taken up, or, the preload was completely relieved and deformation of the endplates, simulated in figure 3.5, was occuring. Although this occurrence needed to be studied, it was not easily simulated and study of it was not proposed by this thesis. Therefore, a second test was needed which could be easily simulated and would also verify each model.

3.20 DISCUSSION OF THE TENSILE TEST

Due to the unpredictable parameters of the internal pressure test, an additional test was required. Realizing the undesirable

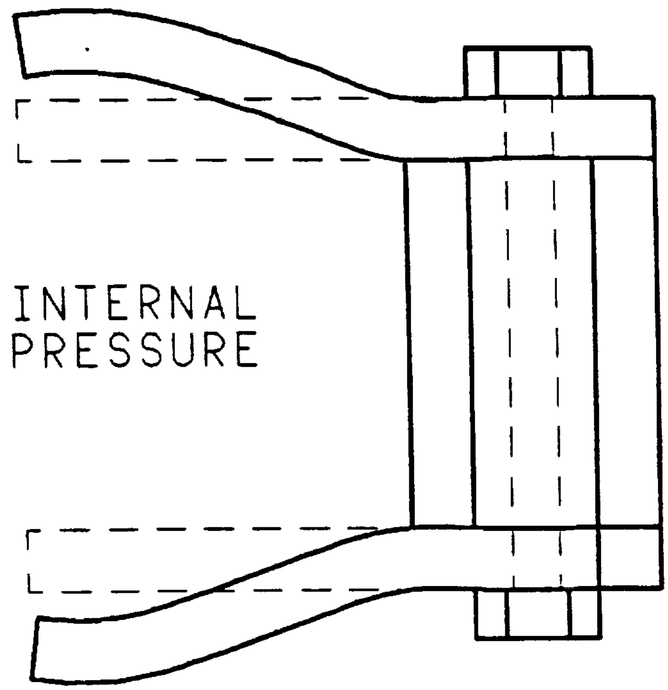


Figure 3.5: Simulated endplate deformation.

features of the first test, the second test had to not only be indicative of the conditions due to operation but also it had to be easily simulated. These two restrictions were satisfied by a simple tensile test. Even though the test would not actually simulate pressurization, the stresses incurred would be similar in various regions. Combinations of bending, membrane, and shearing actions would stress the model much like operating conditions.

To perform this test, the housing was obtained from NASA. Two devises were made (see figure 3.6) which enabled the housing to be tested using the University's Tinius Olsen testing machine. As seen in figures 3.7 and 3.8, the housing was loaded in two directions. Results were obtained by using the strain gage rosettes mounted by NASA. Loads were incremented until enough strain (15 micro) was acquired to overcome any errors involving gage sensitivity. By carefully monitoring each rossette, permanent deformation was avoided. Also, to aid in simulation, the FEM's global axes were placed on the housing.

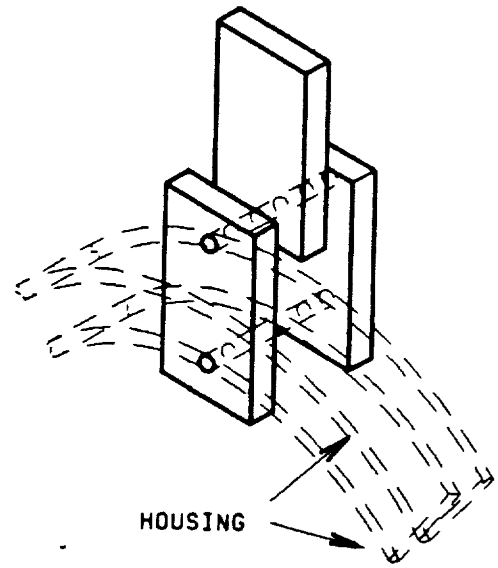


Figure 3.6: Illustration of devises used to test housing.

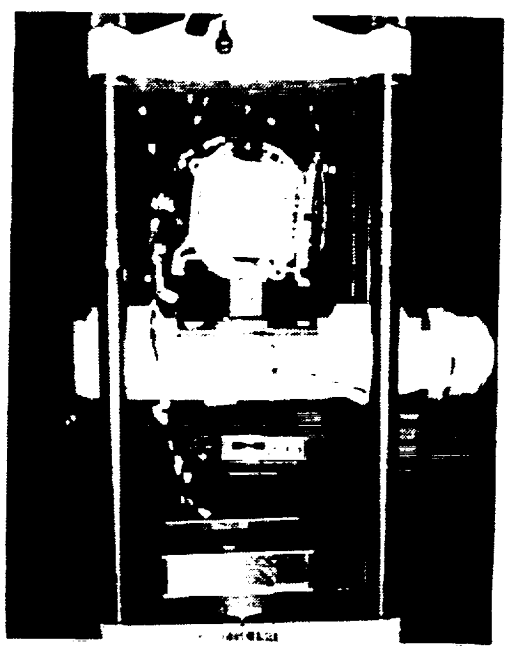


Figure 3.7
Longitudinal tensile test.

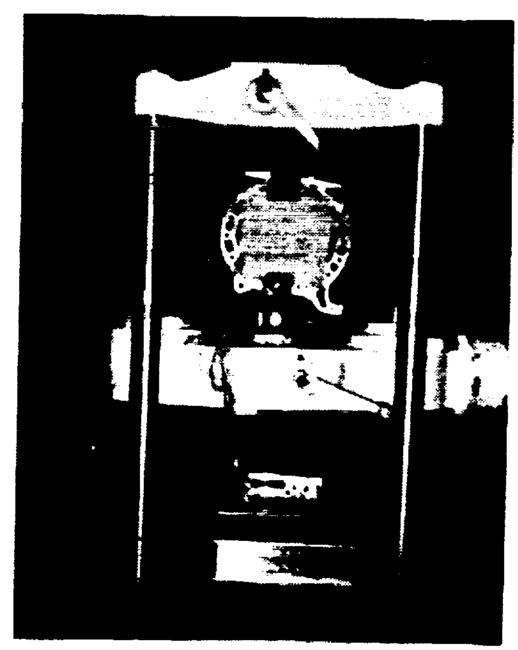


Figure 3.8
Transverse tensile test.

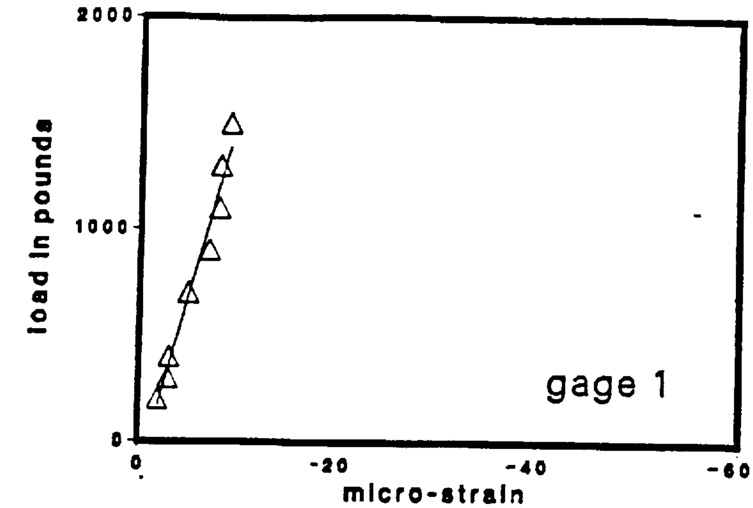
Using this as a reference, the loading directions were easily obtained.

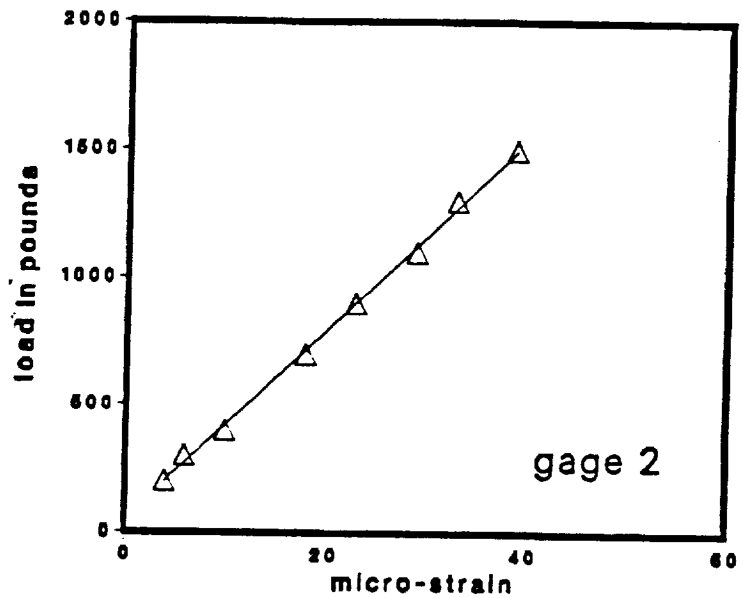
The data obtained from performing the tensile test was very consistent. Strains recorded were, as seen in figures 3.9-3.11 and 3.12-3.14, linear. The strains recorded for the tensile transverse test are reported, but due to inconsistencies, they were not relied upon for model verification. Furthermore, the stresses resulting from these strains were computed and are listed in table 1. In this table, principal stresses and directions are given, but also stresses are given in the model's expected coordinate system explained in appendix 1.

3.30 SIMULATION OF THE TENSILE TESTS

Once the results of both tensile tests were recorded and analyzed, simulation of the tests was completed. The orientation of the global axes during the test was known. From this information the longitudinal loading of each model was directed 5 degrees from the vertical and a Load of substantial magnitude, 1000 pounds, was applied to each model. Since clearance in the bolts was large, loading caused the bolts to deform (simulated in figure 3.15). Thus, distribution of the load varied. Deformation of the bolts applied the load symmetrically across the bolt hole's length ,but the effect of the added deformation could not be fully modeled. A load distributed across the bolt hole's length was utilized for verification but, in order to bracket the actual load application, two other loading distributions were investigated. One, a single point load located in the center of the bolt hole's length, and a second, which applied equal loads on either end of the bolt hole. The actual loading distribution was expected to occur between these two extremes.

To run the test, each model needed to be supported. However, location n





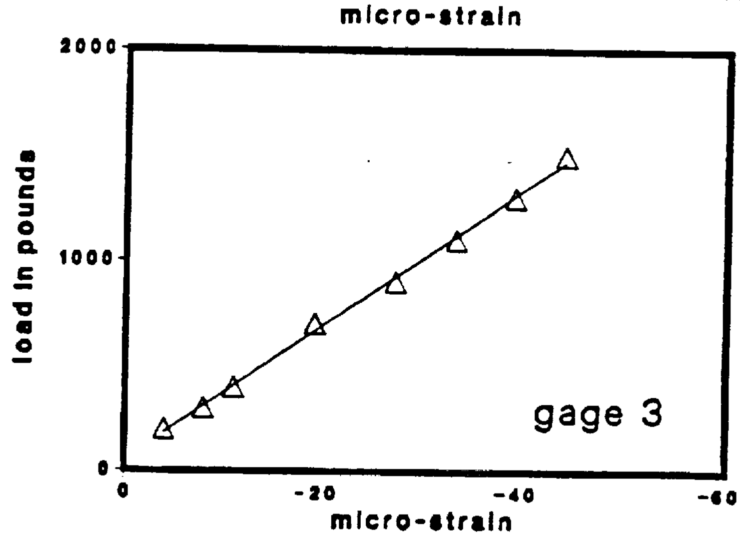


figure 3.9; longitudinal tensile test.

location h

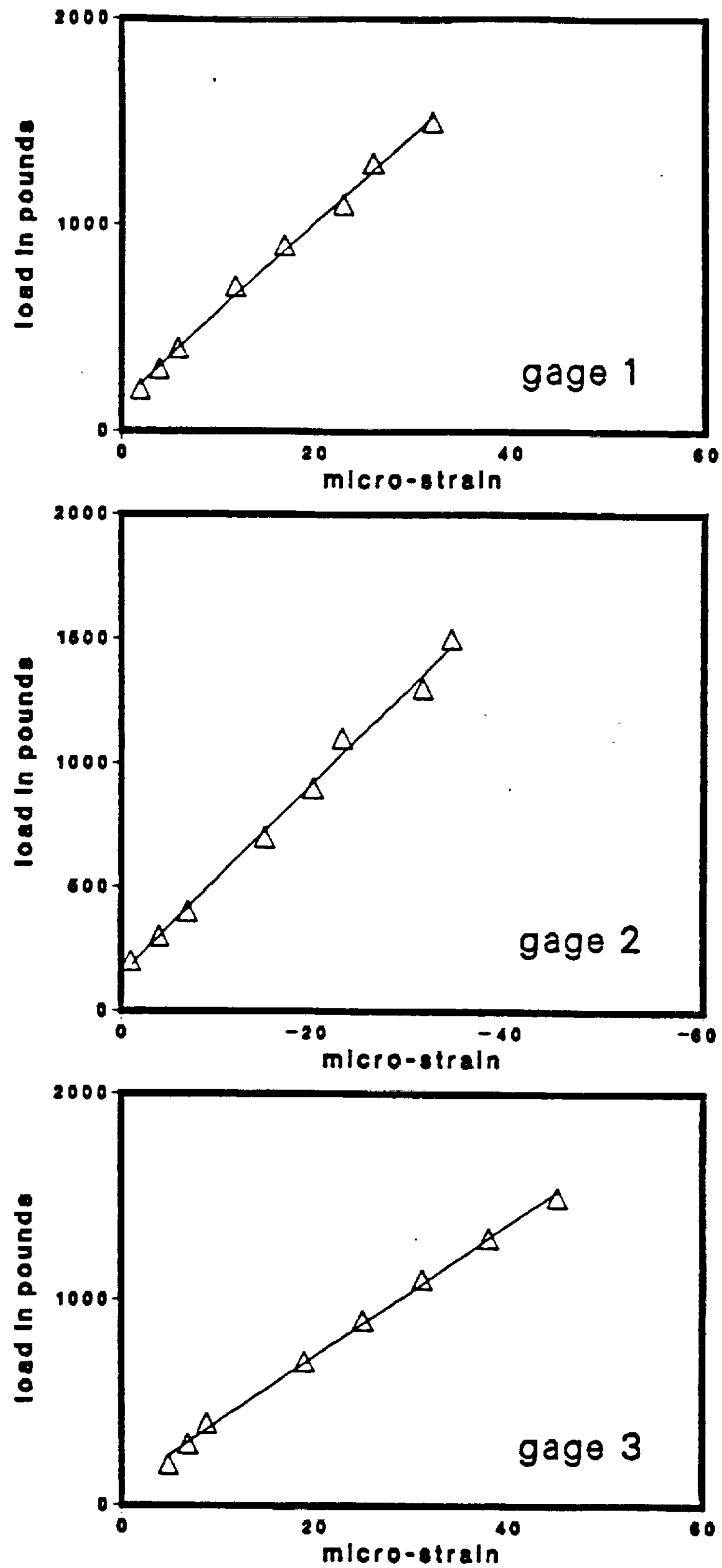


figure 3.10; longitudinal tensile test.

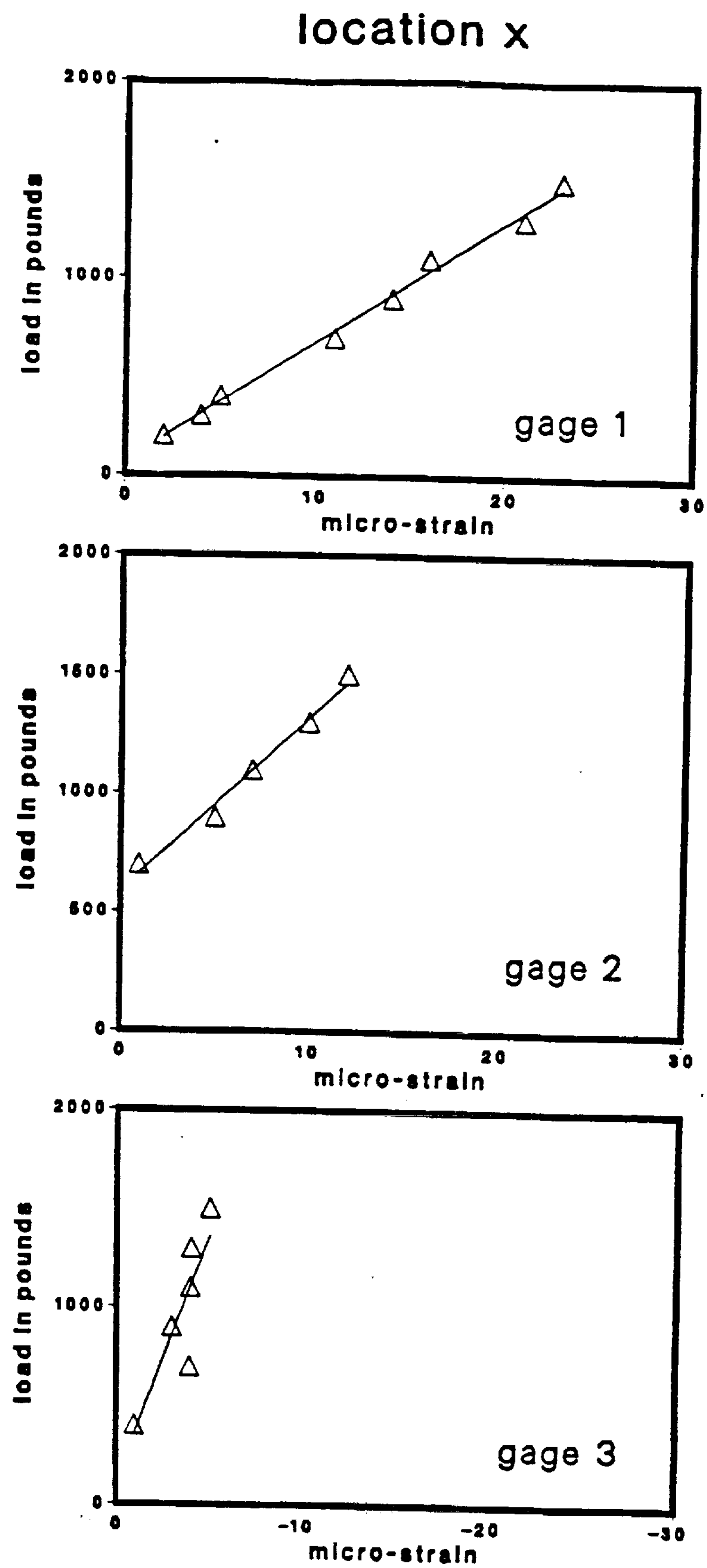


figure 3.11; longitudinal tensile test.

location n

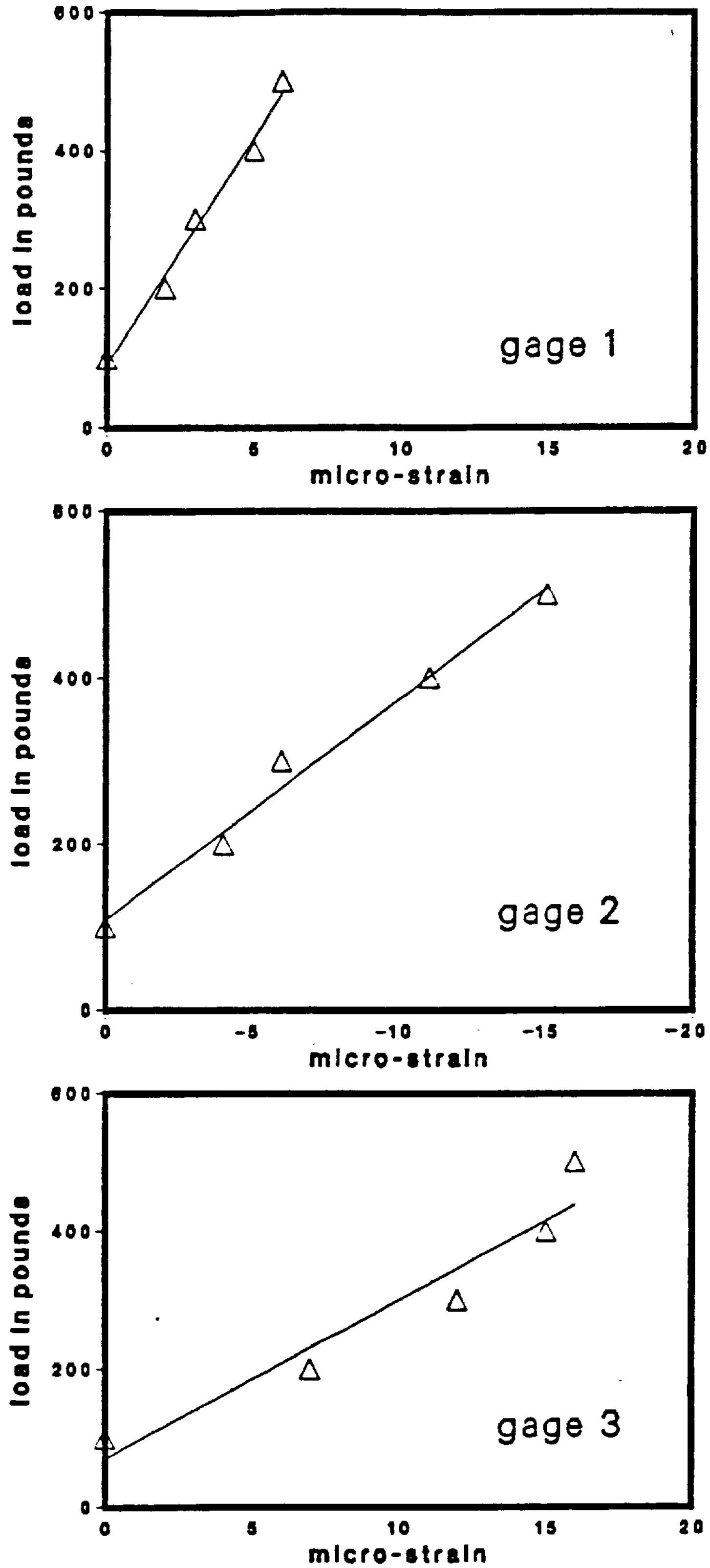


figure 3.12; transverse tensile test.

location h

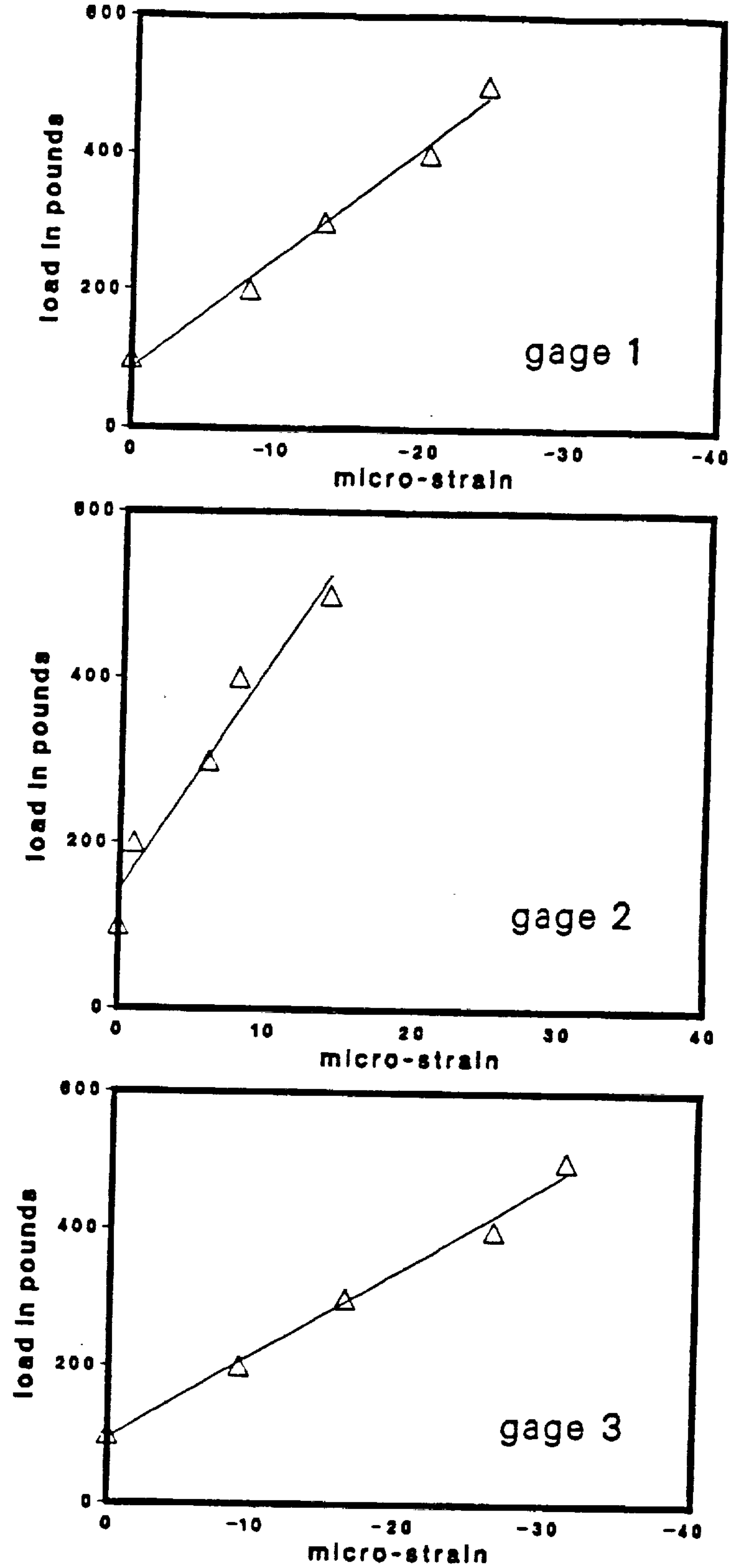


figure 3.13; transverse tensile test.

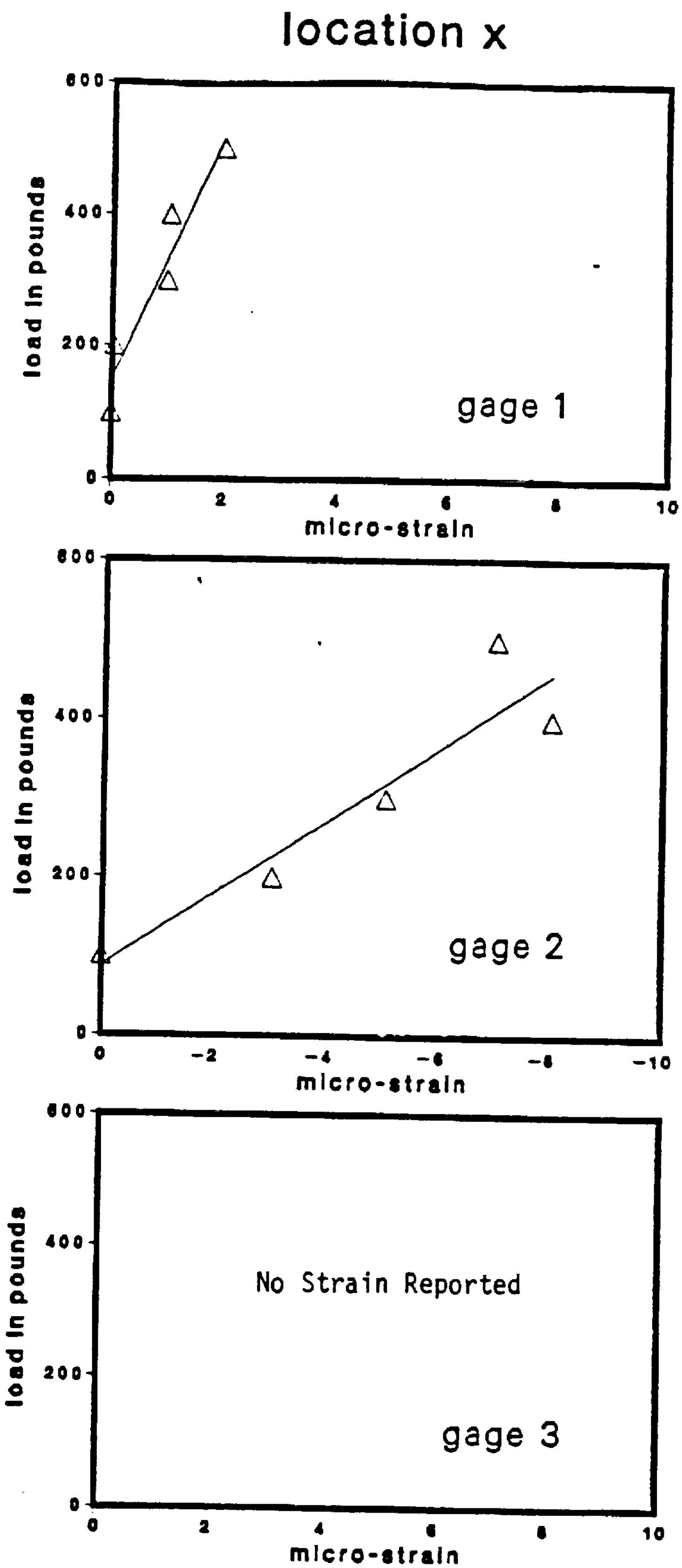


figure 3.14; transverse tensile test.

LOCATION	N	H	X
SIGMA-X	75	-9.3	85
SIGMA-Y	-597	702	100
TAU-XY	-85	36	-76
SIGMA-P1	86	704	170
SIGMA-P2	-607	-11	15
DIRECTION	7.1	2.9	47

*NOTE: POSITIVE DIRECTION INDICATES CLOCKWISE ROTATION OF AXES.

STRESSES GIVEN IN PSI

Table 1: Stresses from longitudinal tensile test (1000 lbs.).

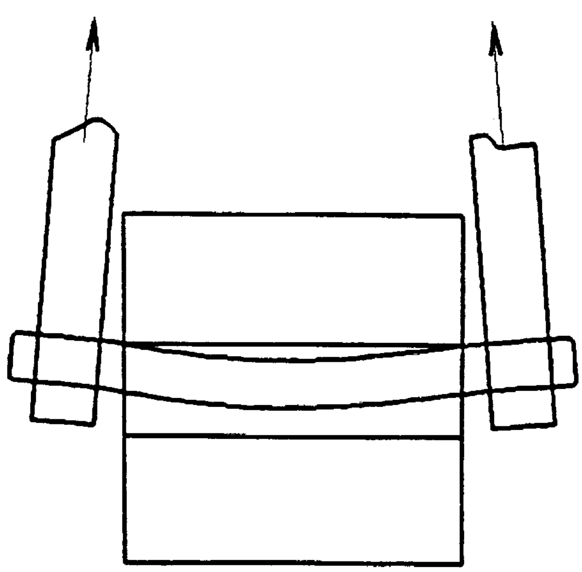


figure 3.15: Illustration of suspected bolt deformation.

support of the model in the regions around the rosette locations would affect the validity of the results. Consequently, support of the model was restricted to being located away from the gage locations, as shown in figure 3.16. In addition, actual support of the model had to inhibit the model as little as possible. Although two support systems, shown in figure 3.17, were utilized, various support systems were looked into. The first type was used primarily in the thin shell models as shown in figure 3.17a. It restrained all the degrees of freedom of one node. The remaining nodes in the vertical row were able to translate vertically and only the vertical rotation was constrained. This scheme enabled the model to deform in the most natural manner possible. The second type, shown in figure 3.17b, was designed for the thick shell models since nodal rotations in the models could not be restrained. In order to restrict the rotation of the model about its vertical axis, two rows of nodes were restrained.

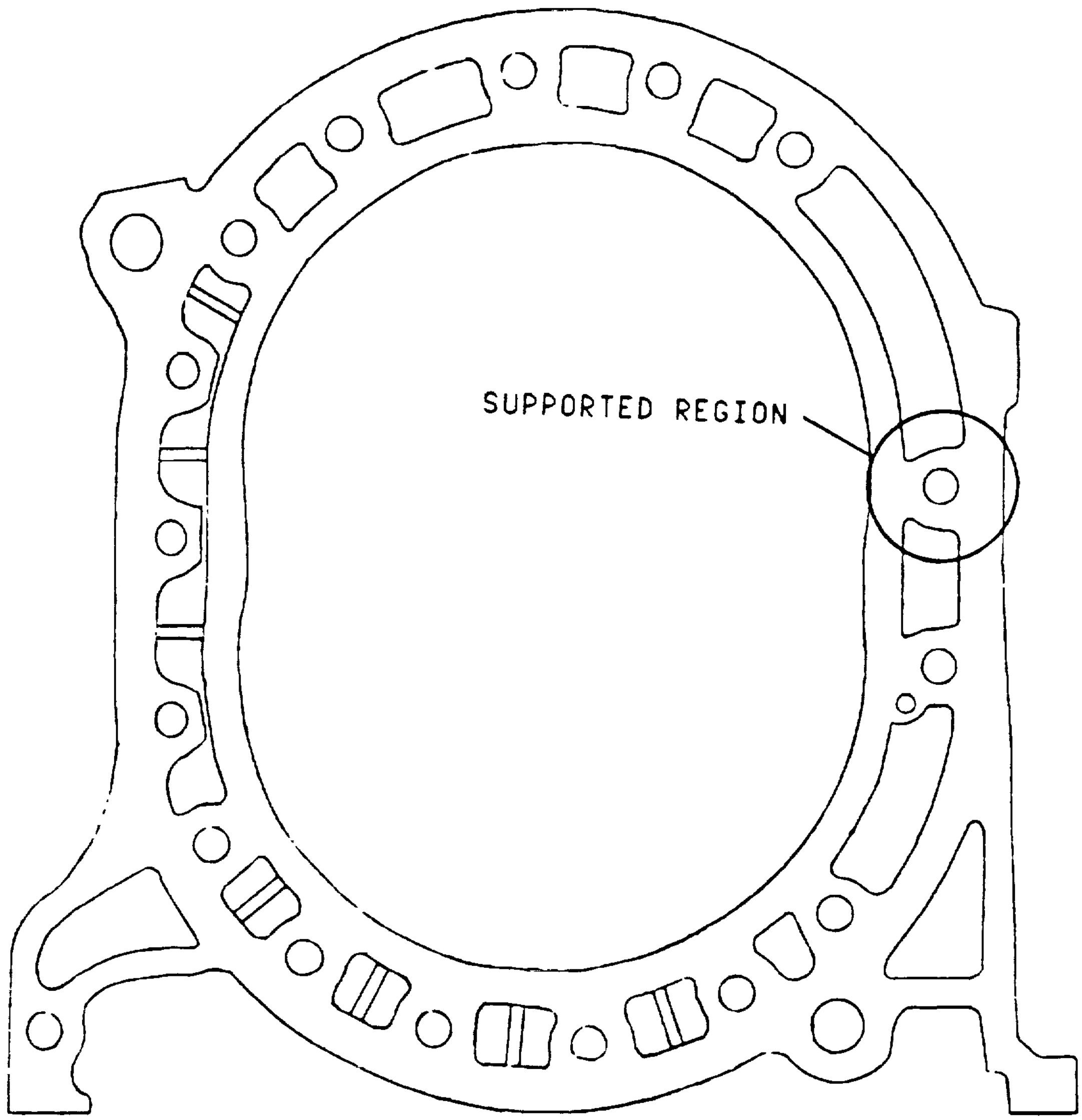
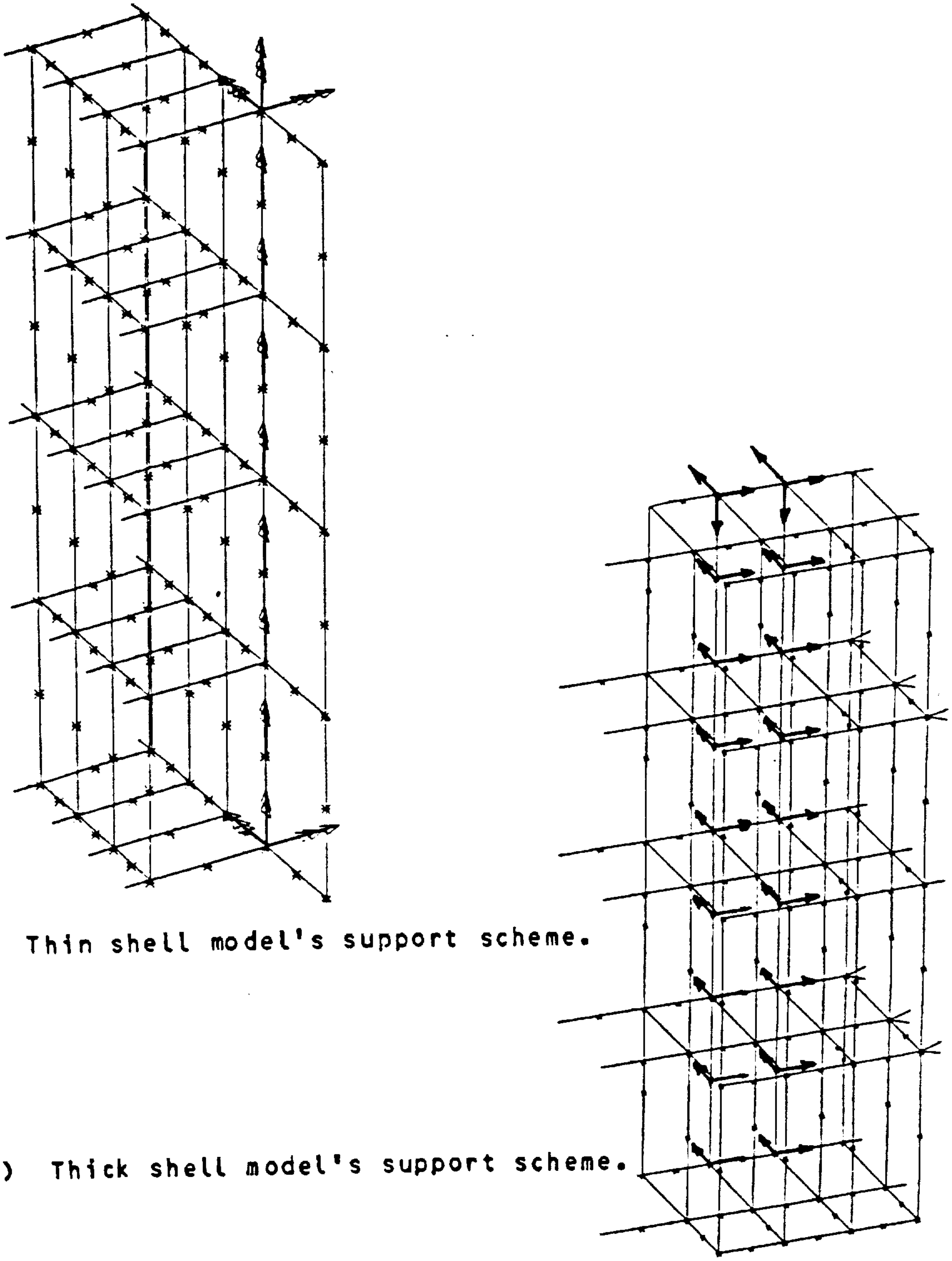


Figure 3.16: Location of model support.



(a) Thin shell model's support scheme.

(b) Thick shell model's support scheme.

Fijure 3.17: Illustration of support schemes for thin and thick shell models.

4.00 CONSTRUCTION METHODS AND MODEL VALIDITY

As mentioned earlier, the construction of the FEM of the housing was done with a series of models. Each model in the series utilized construction methods proven by the previous models. Initially, the sequence was to consist of four models: a linear thin shell, parabolic thin shell, parabolic thick shell, and the final model. However, as expected, modifications to this plan were required. Mesh sizes, element types, and nodal configurations were methodically modified until the final model, vaguely introduced in chapter 2, was constructed. By utilizing this sequence, convergence was checked and overall confidence in the model was gained. Thus, when it was completed, the final FEM not only accurately simulated the housing, but it also possessed a firm modeling foundation. In the next segments the construction methods and results of each model in the sequence are presented.

4.10 CONSTRUCTION OF THE LINEAR THIN SHELL MODEL

As the first model of the construction sequence, the linear thin shell model was to be a simple FEM of the housing. Since simplicity was to be its greatest attribute, the model was constructed using elements with linear nodal configurations. This made the model's size a modest 9200 degrees of freedom. Furthermore, as seen in figure 4.1, the actual geometry construction was not complicated. Thin shell elements were utilized for the inner and outer shells, thus, only the middle surface of either shell needed to be defined. In addition, certain pieces of geometry were not included. Since it was believed that the regions where the strain gage rosettes were located would not be affected by the placement of the spark

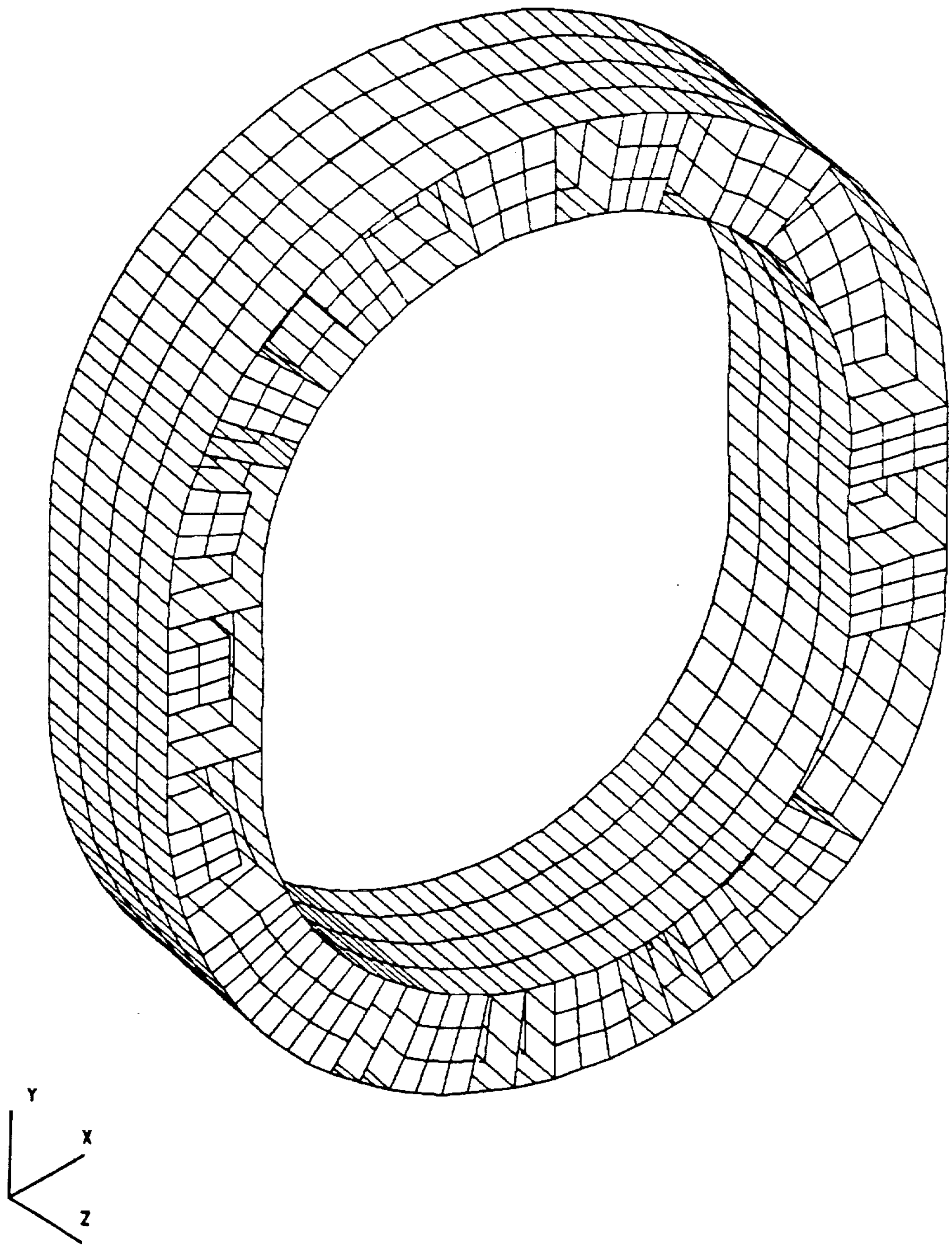


Figure 4.1: Illustration of linear thin shell model.

plugs, engine mounts, and both ports, this geometry was not modeled. Also, it should be noticed that the bolt holes were not modeled either. The significance of the bolt holes was considered to be negligible. However, in an endplate interaction study the significance of the bolt holes must be considered due to the tolerance of the bolts. Simplicity was further gained by using a mesh size that was uniform and considered coarse. Refinements of the mesh would be considered if necessary.

As construction of the linear thin shell model began, modeling methods materialized. Problems that arose were subsequentially solved by employing new methods. Of the problems encountered, only one major difficulty stood out, this was compatibility between solid and thin shell elements. Since solid elements do not support nodal rotations, a compatibility problem was introduced whenever thin shell elements were to be modeled with solid elements. This particular problem occurred in two regions of the linear thin shell model. In these regions, the solution used was similar. Both utilized a method in which certain nodes were coupled together. By coupling these nodes, the translational degrees of freedom of the associated nodes were forced to move together. It must be noted here that the nodal rotations of the thin shell elements were not affected by this coupling. Thus, moments could not be translated across the elements in question. Although this method seemed adequate, it did not model the regions exactly, so doubts were raised on its accuracy.

Both regions in question were located around the primary ribs. The first being where the inner and outer shells connected to a primary rib. In order to model this particular region, two fully closed cylinders were constructed. The solid ribs were then placed between the two shells. By defining common meshes, coincident nodes were assured. These nodes were then coupled creating a "spot welded" effect. Figure 4.2 illustrates this construction technique.

In the second region, where the mid-plane stiffeners connect to primary ribs, the coupling method was used in much the same

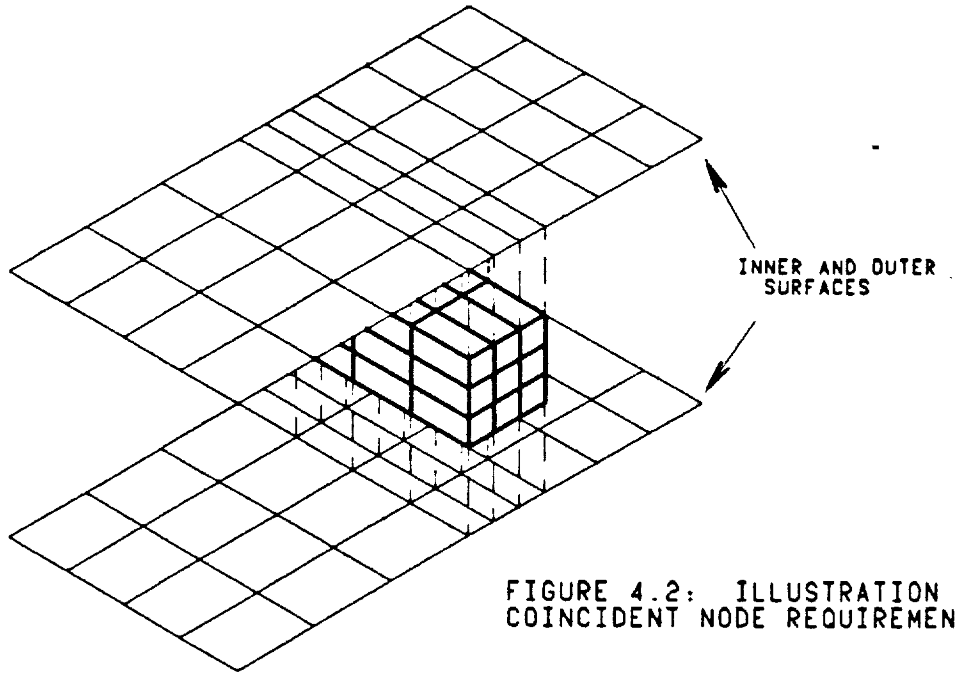


FIGURE 4.2: ILLUSTRATION OF
COINCIDENT NODE REQUIREMENT.

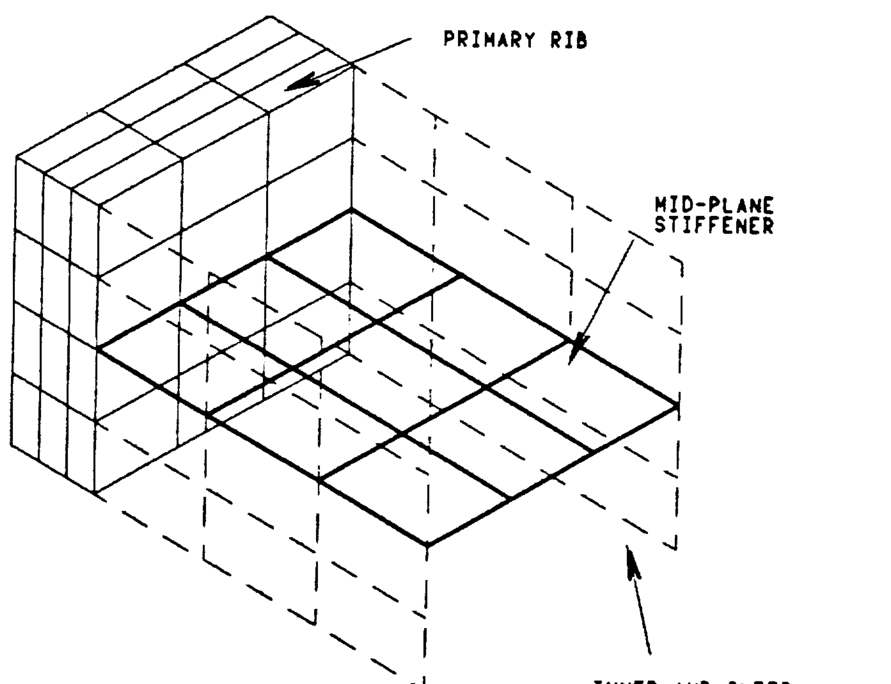


FIGURE 4.3: ILLUSTRATION OF
MID-PLANE STIFFENER LOCATION.

manner. A common mesh produced coincident nodes which were then coupled. However, as can be seen in figure 4.3, the major support of this region originated from the intersection of the stiffener and the inner and outer shells.

As stated, the accuracy of these modeling techniques was doubted. Since rotations of the thin shell elements in these regions was possible, deformations as seen in figure 4.4, could occur. Yet, the effect of such deformation was not known. Clearly, this effect was negligible for the mid-plane stiffeners since the expected loading direction would not produce undue deformation. Only if a pressure load, normal to the stiffeners' surface, were modeled would the effect be noticable. However, the deformation of the inner shell was not expected to be closely simulated by the coupling techniques.

Actual construction of these regions required that the inner and outer surfaces be made with a number of small segments. A segment would occur at each primary rib and associated cooling channel. This scheme would assure the existence of common meshes. To accomodate this, a number of construction points were required. A numerical routine was developed to provide consistent placement of these points. After the wire frame model was built the surfaces and volumes were defined such that tw surfaces existed at every rib location. From this, two continuous shells and a number of primary ribs were made. A further requirement was the use of an even number of elements in the housing's axial direction. This last restriction assured coincident nodes for the mid-plane stiffeners. Other than this restriction, the mesh configuration was optional, therefore, a simple uniform mesh was chosen. Due to the cylindrical shape of the housing, SUPERB's wavefront optimization routine performs very poorly. Thus, to optimize the elemental wavefront of the model, the element numbering must procede in a radial fashion.

The expectations for the linear thin shell model were limited. It was hoped that insight into the stressing mechanisms of the housing could be gained. Furthermore, familarization with the

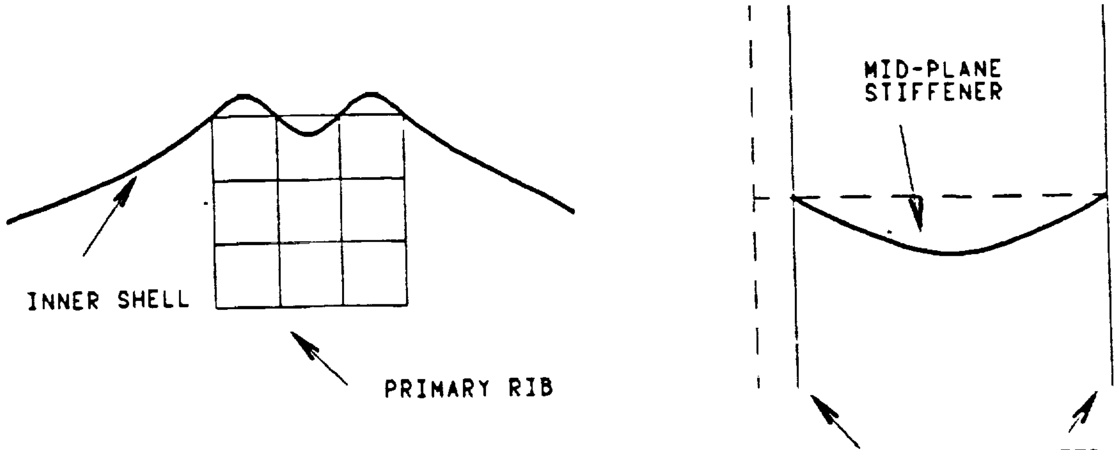


FIGURE 4.4: ILLUSTRATION OF POSSIBLE DEFORMATIONS BY UTILIZING THE COUPLING METHOD.

modeling system was anticipated. The compatibility problem, as well as the fact that linear thin shell elements could not model shearing actions, was expected to cause the stresses at locations X and N to be off the measured values. However, under longitudinal loading, location H was expected to model the experimental values quite closely since only a simple uniaxial load would be in play. Overall, the linear thin shell model was expected to provide an initial step in the construction sequence.

4.11 ANALYSIS OF THE LINEAR THIN SHELL MODEL

After its completion, the linear thin shell model was subjected to the longitudinal tensile test described in chapter 3. As also mentioned in chapter 3, the loading distribution was analyzed. However, after analysis, it was determined that the actual loading distribution had little effect on the regions around strain gage rossettes. Thus, verification was completed by simply distributing the load across the bolt hole's length. Furthermore, various supporting methods were also applied. Basically, three types of supports were examined. Each one utilized the restraints described in chapter 3. But, as seen in figure 4.5, each supported different regions of the housing. For

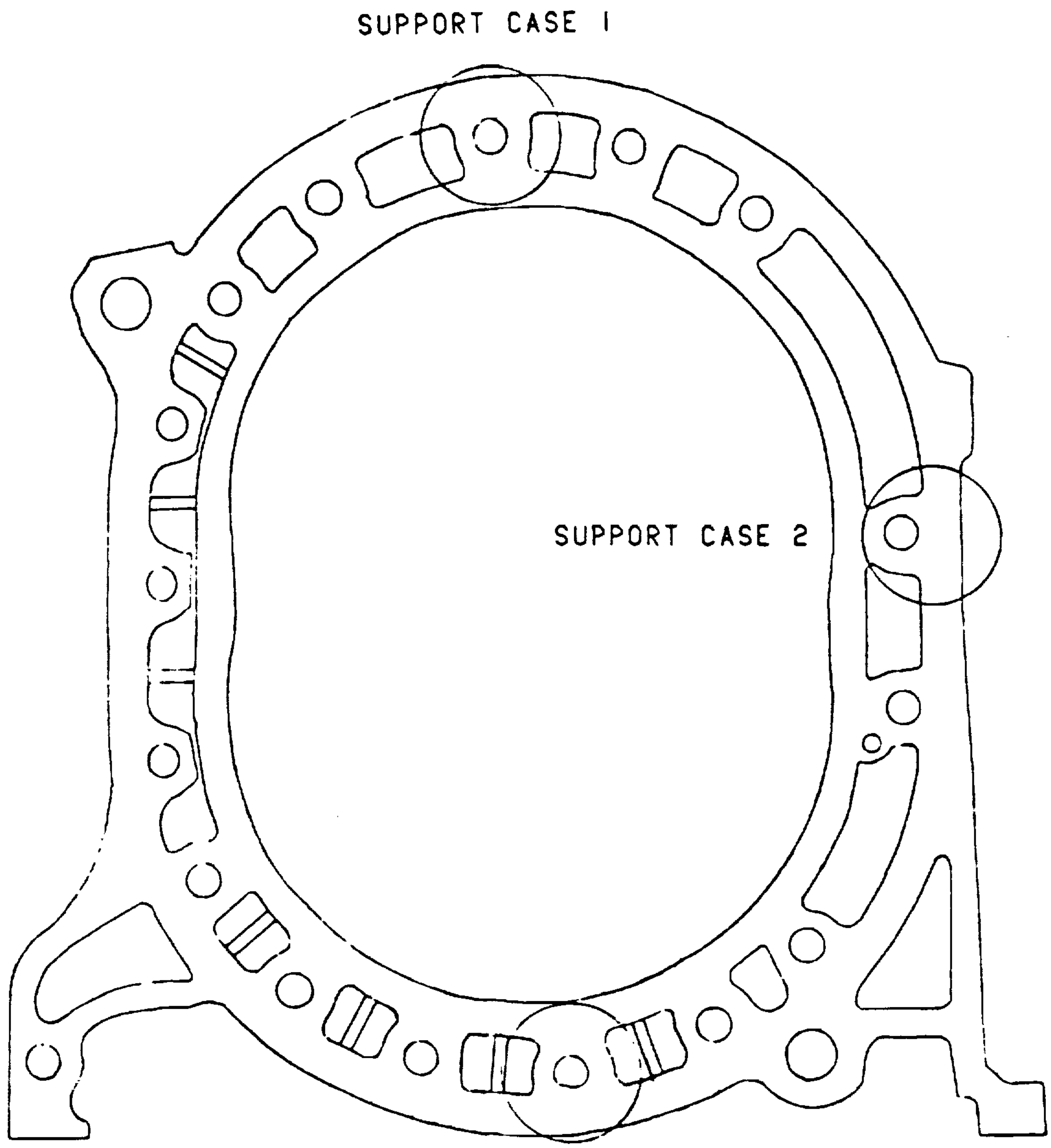


Figure 4.5: Various regions where support of the model was investigated.

simplicity, the groups will be referred to as support case 1, 2, and 3 as illustrated in figure 4.5. Results from this testing were obtained in the form of deformation plots and nodal stresses. For each support case, nodal stresses were compared to the experimental baseline data provided in table 1. Also, for each case, the particular loading and support of the model was confirmed through the use of the deformation plots (refer to figures 4.6, 4.7, and 4.8).

When testing was complete, the results were reviewed. From this, two conclusions were drawn. First, high stress gradients occurred around each rosette location, thus, comparison of stresses would need to be approximated. Second, of the three support cases, only case 2 enabled natural deformation of the housing to occur. The first of the conclusions is emphasized by figures 4.9-4.17. Even though only a few data points are available, the stress gradients are obvious. An illustration of the strain gage locations, shown in figure 4.18, will aid in the intrepetation of the grid figures. The approximate rosette locations are shown, and the estimated stresses at each location are given in table 2. From table 2, the second conclusion can be drawn. Clearly, the rosette locations were affected by the nearby support of the housing. Consequently, support case 2 was chosen for the remaining testing. Finally, close inspection of the results indicated the accuracy of the linear thin shell model. Stresses, in each location, matched the measured values quite closely. Also, changes in the support cases were very easily predicted and, the existence of the stress gradients provided proof of the adequacy of the testing used to exercise the model.

4.20 CONSTRUCTION OF THE PARABOLIC THIN SHELL MODEL

The second model in the construction sequence, the parabolic thin shell model, was basically chosen to analyze the mesh configuration. Since the same geometry could be utilized, the

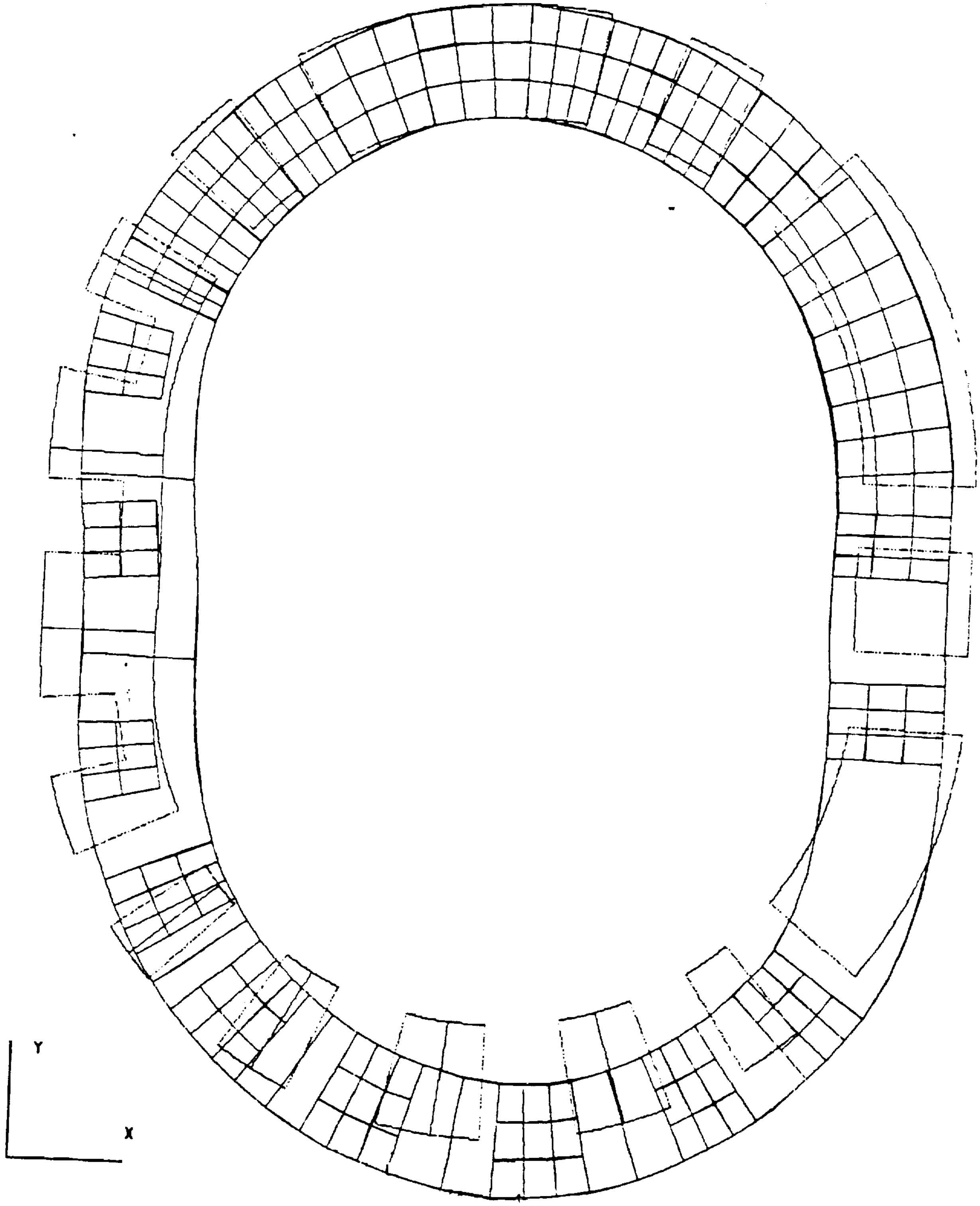


Figure 4.6: Deformation of linear thin shell model (support case 1).

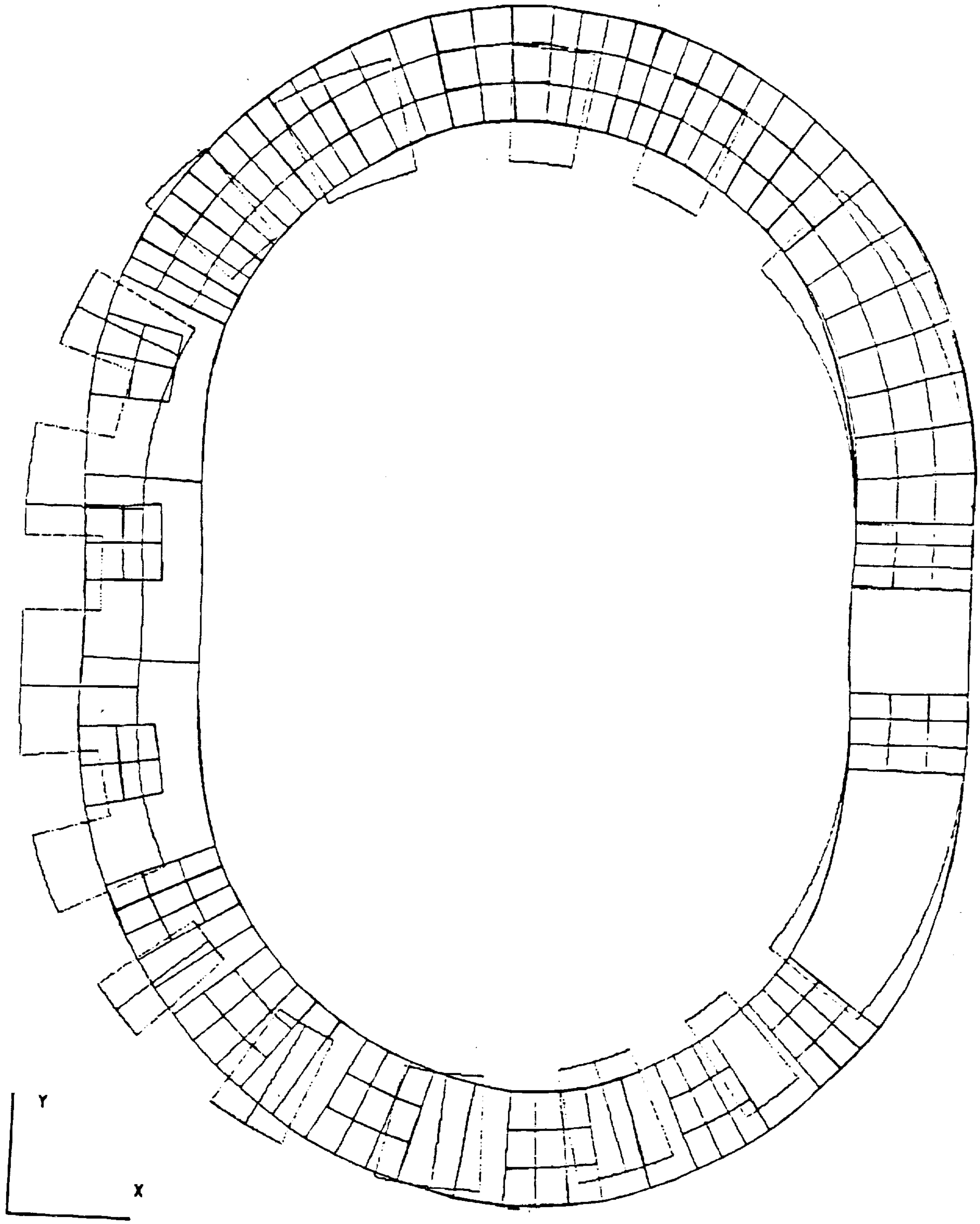


Figure 4.7: Deformation of linear thin shell model (support case 2).

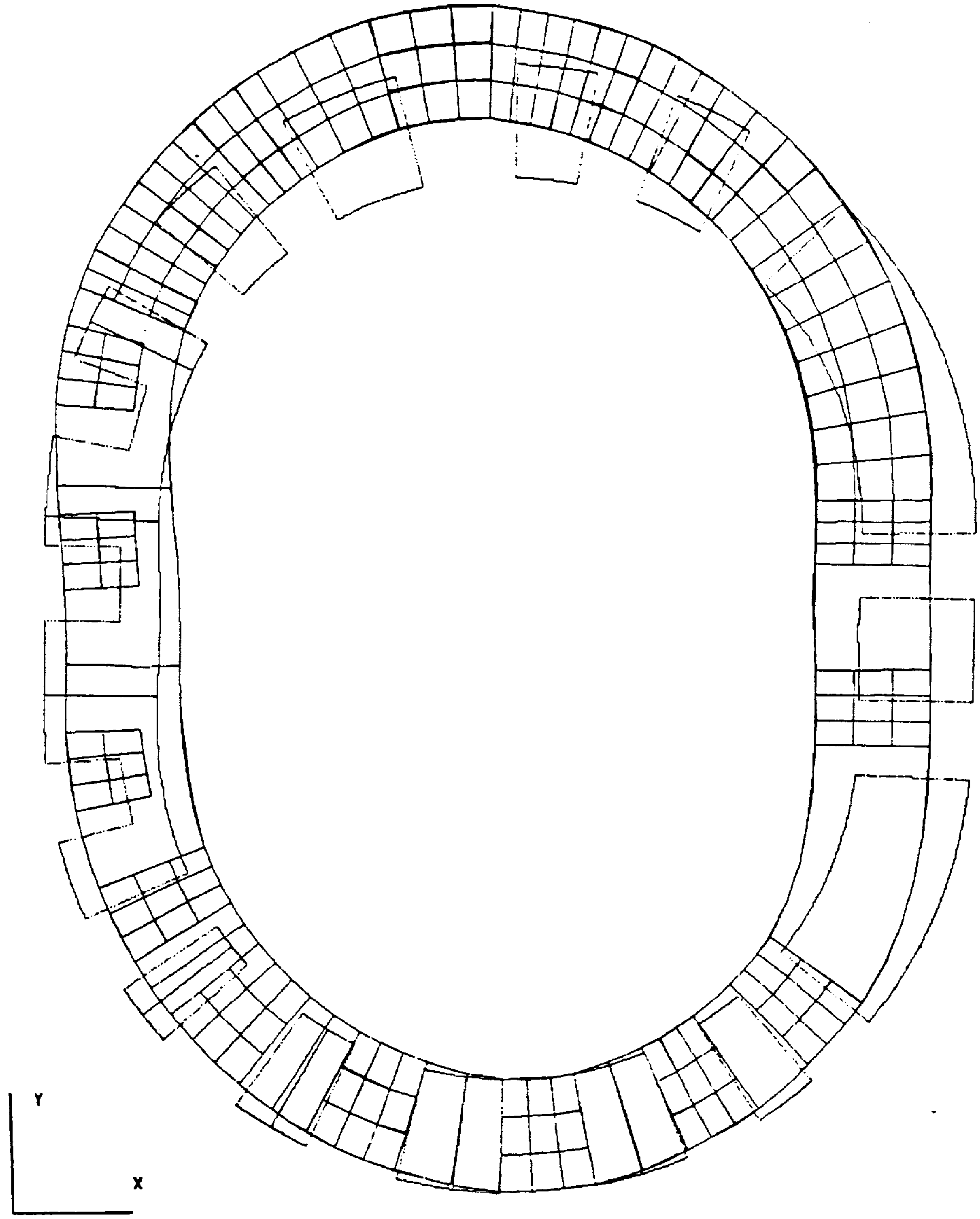
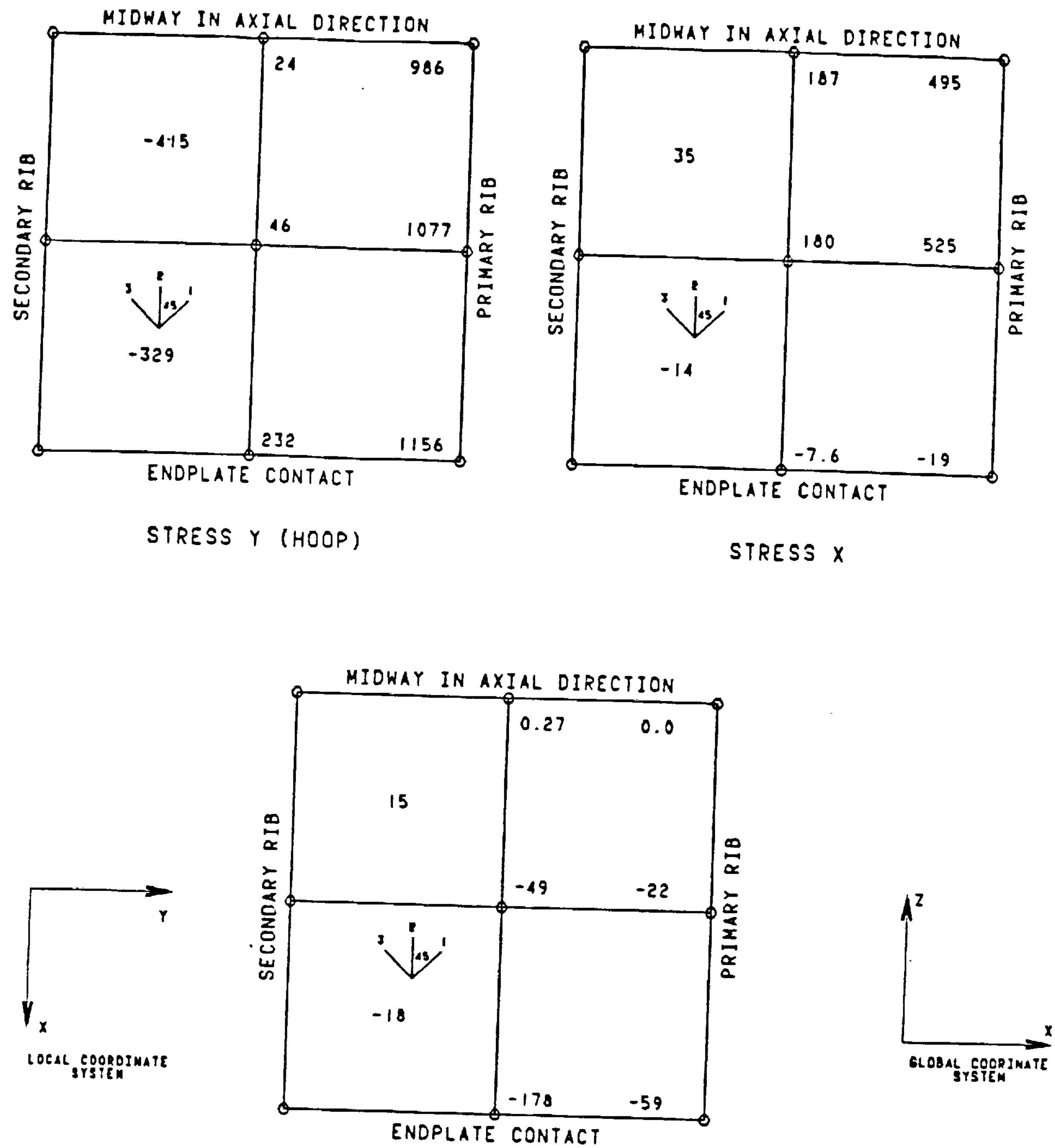


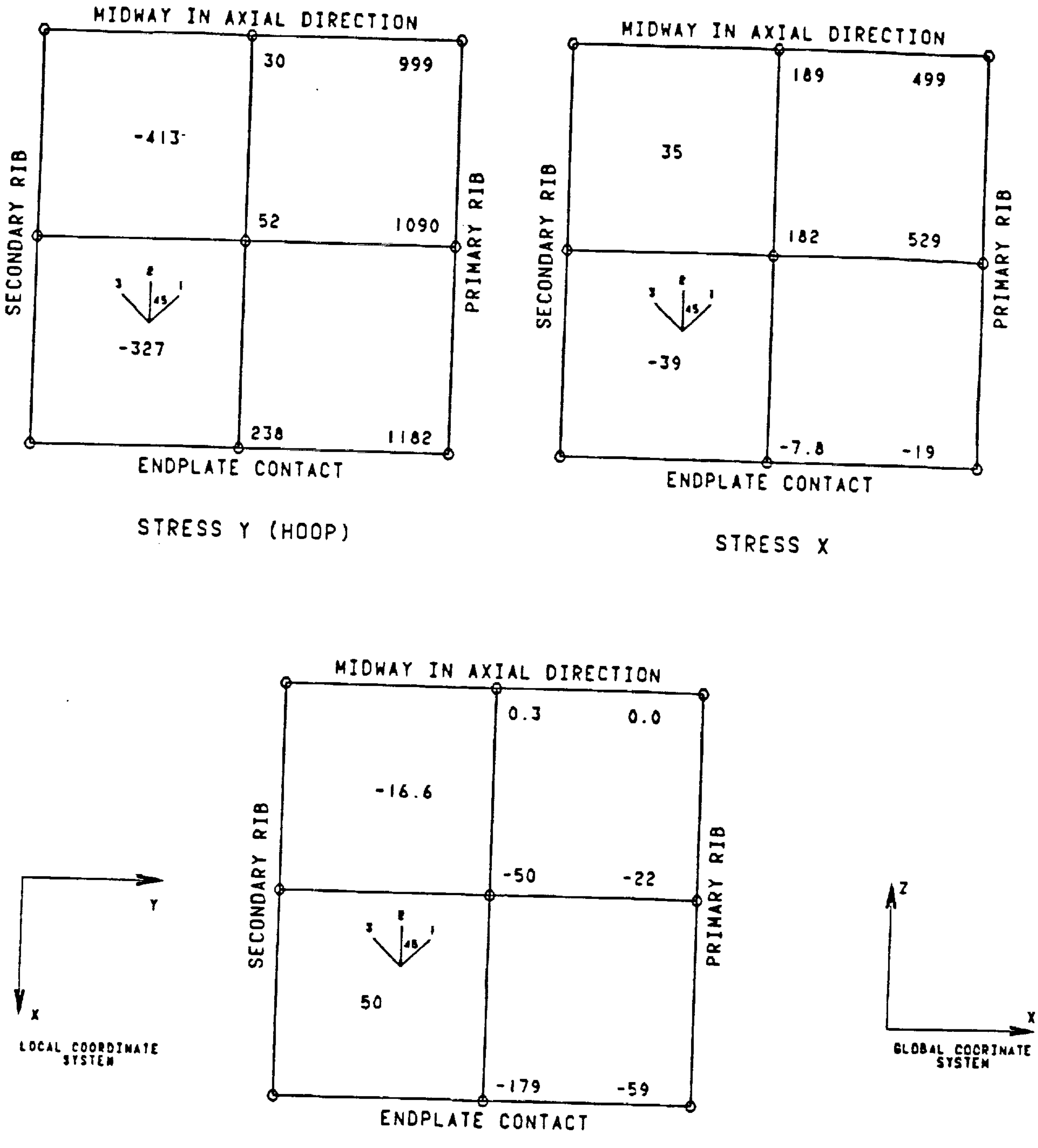
Figure 4.8: Deformation of linear thin shell model (support case 3).



STRESS TAU-XY

NOTE: STRESSES GIVEN IN LOCAL COORDINATE SYSTEM

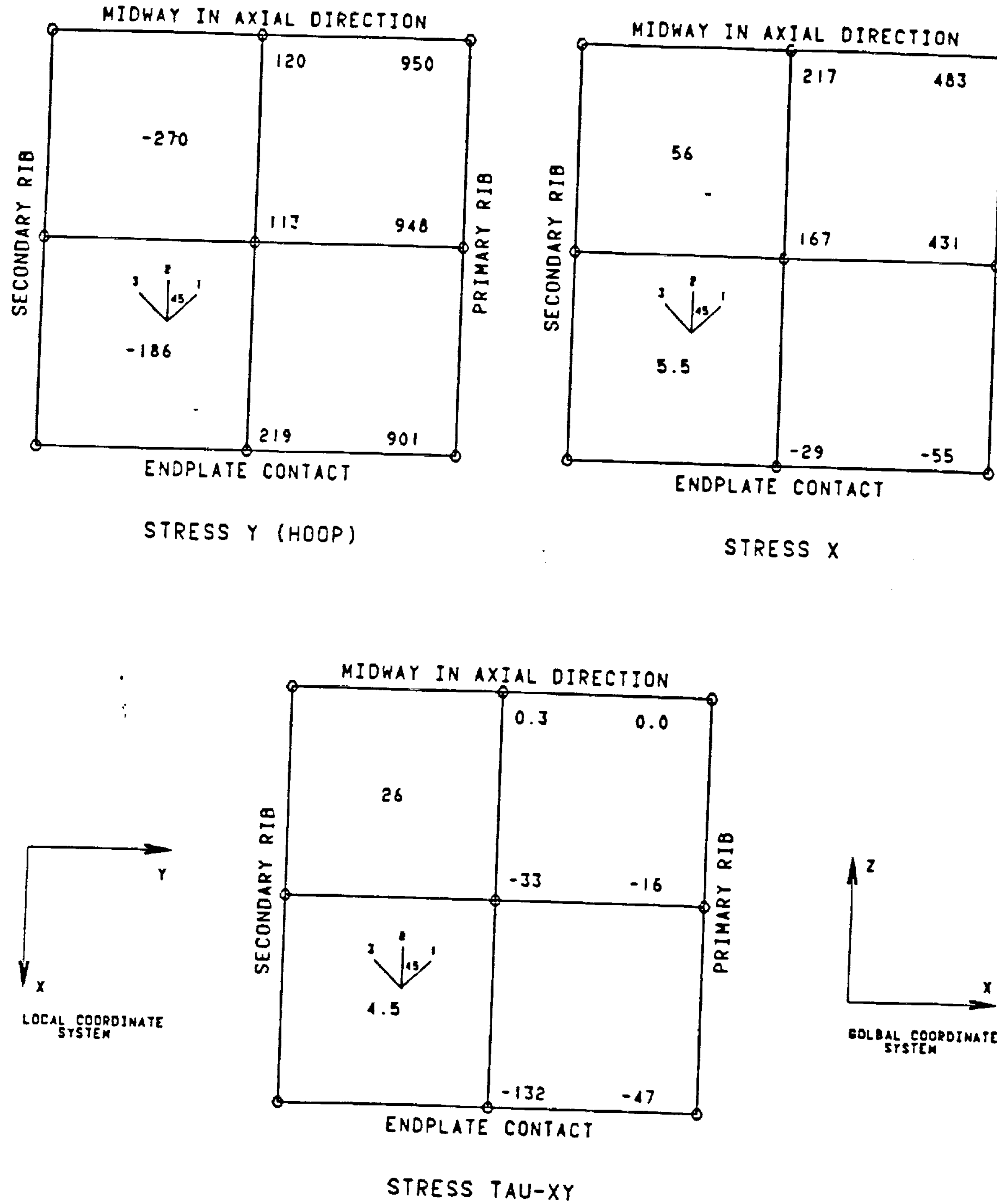
FIGURE 4.9: STRESSES AT LOCATION N
LINEAR THIN SHELL MODEL
RESTRAINT CASE I



STRESS TAU-XY

NOTE: STRESSES GIVEN IN LOCAL COORDINATE SYSTEM

FIGURE 4.10: STRESSES AT LOCATION N
LINEAR THIN SHELL MODEL
RESTRAINT CASE 2



NOTE: STRESSES GIVEN IN LOCAL COORDINATE SYSTEM

FIGURE 4.11: STRESSES AT LOCATION N
LINEAR THIN SHELL MODEL
RESTRAINT CASE 3

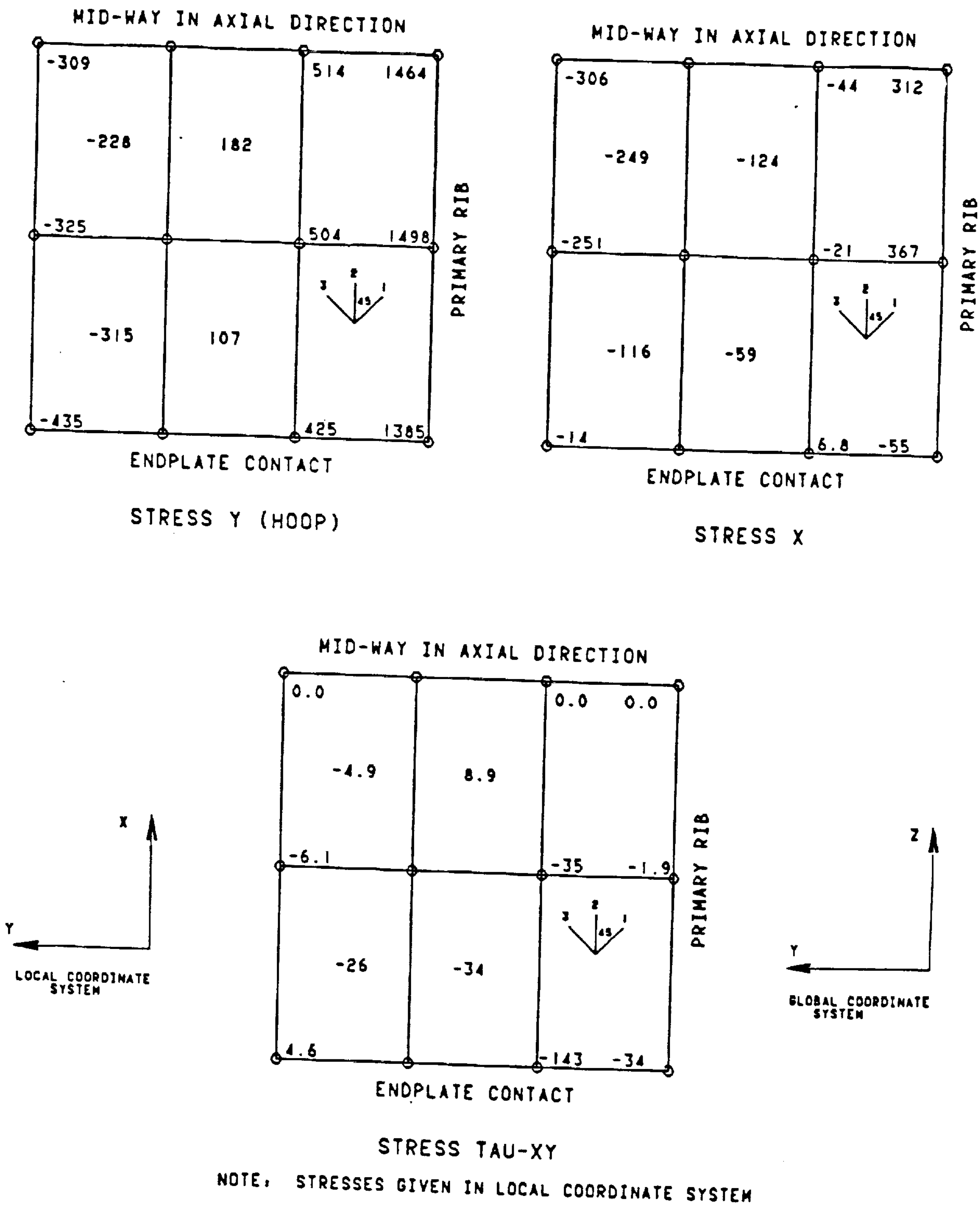
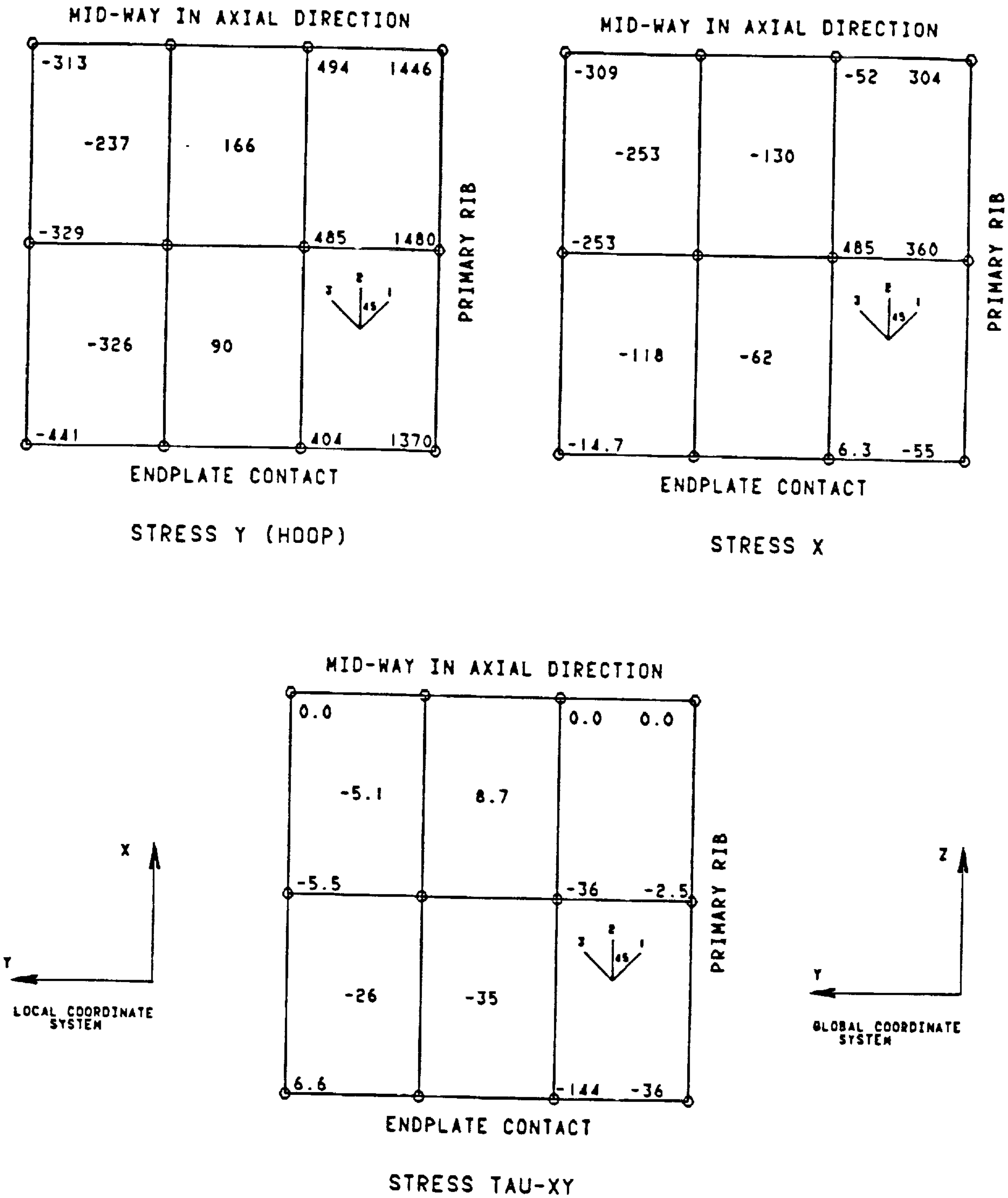


FIGURE 4.12: STRESSES AT LOCATION H
LINEAR THIN SHELL MODEL
RESTRAINT CASE I



NOTE: STRESSES GIVEN IN LOCAL COORDINATE SYSTEM

FIGURE 4.13: STRESSES AT LOCATION H
LINEAR THIN SHELL MODEL
RESTRAINT CASE 2

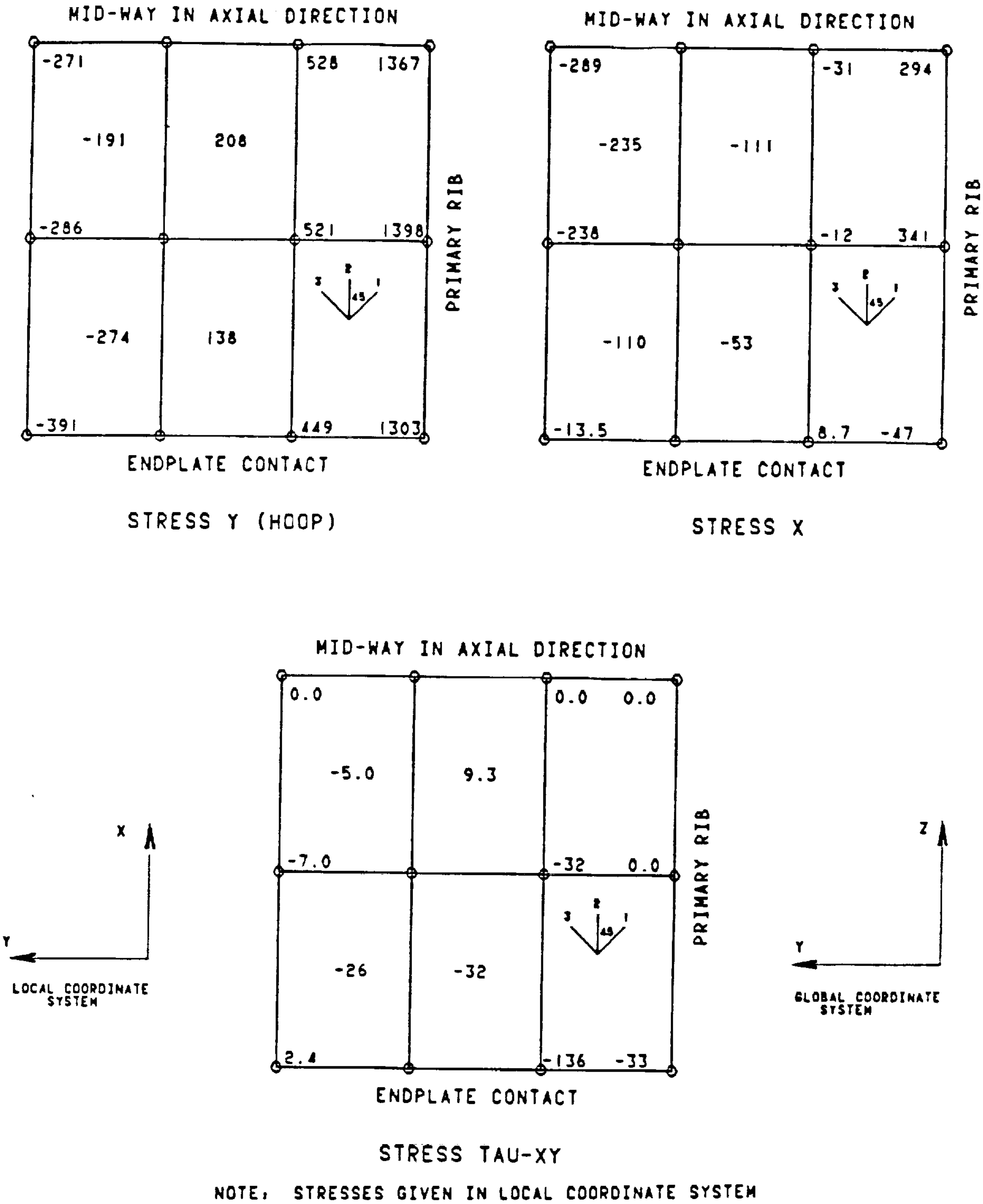


FIGURE 4.14: STRESSES AT LOCATION H
LINEAR THIN SHELL MODEL
RESTRAINT CASE 3

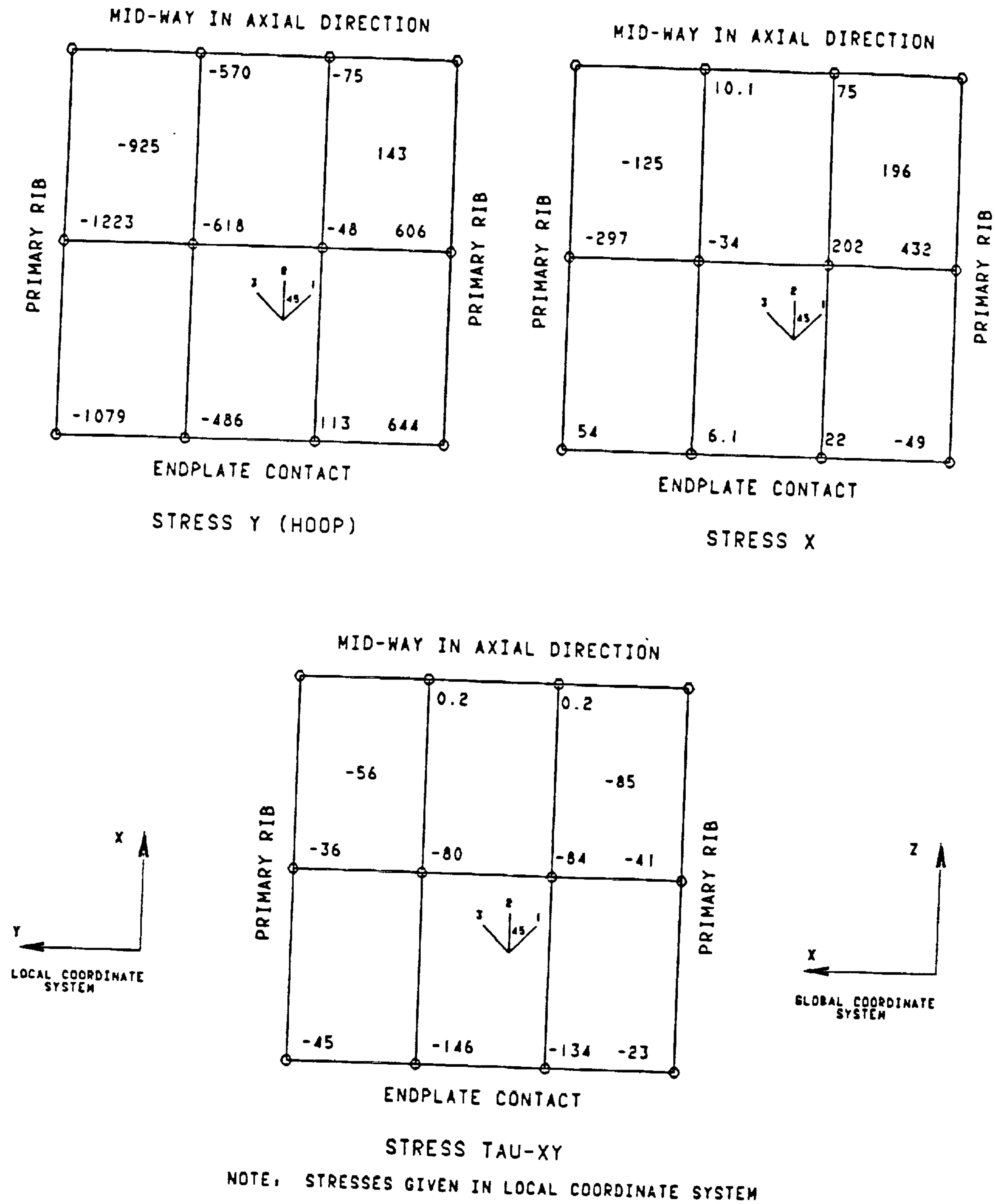


FIGURE 4.15: STRESSES AT LOCATION X
LINEAR THIN SHELL MODEL
RESTRAINT CASE I

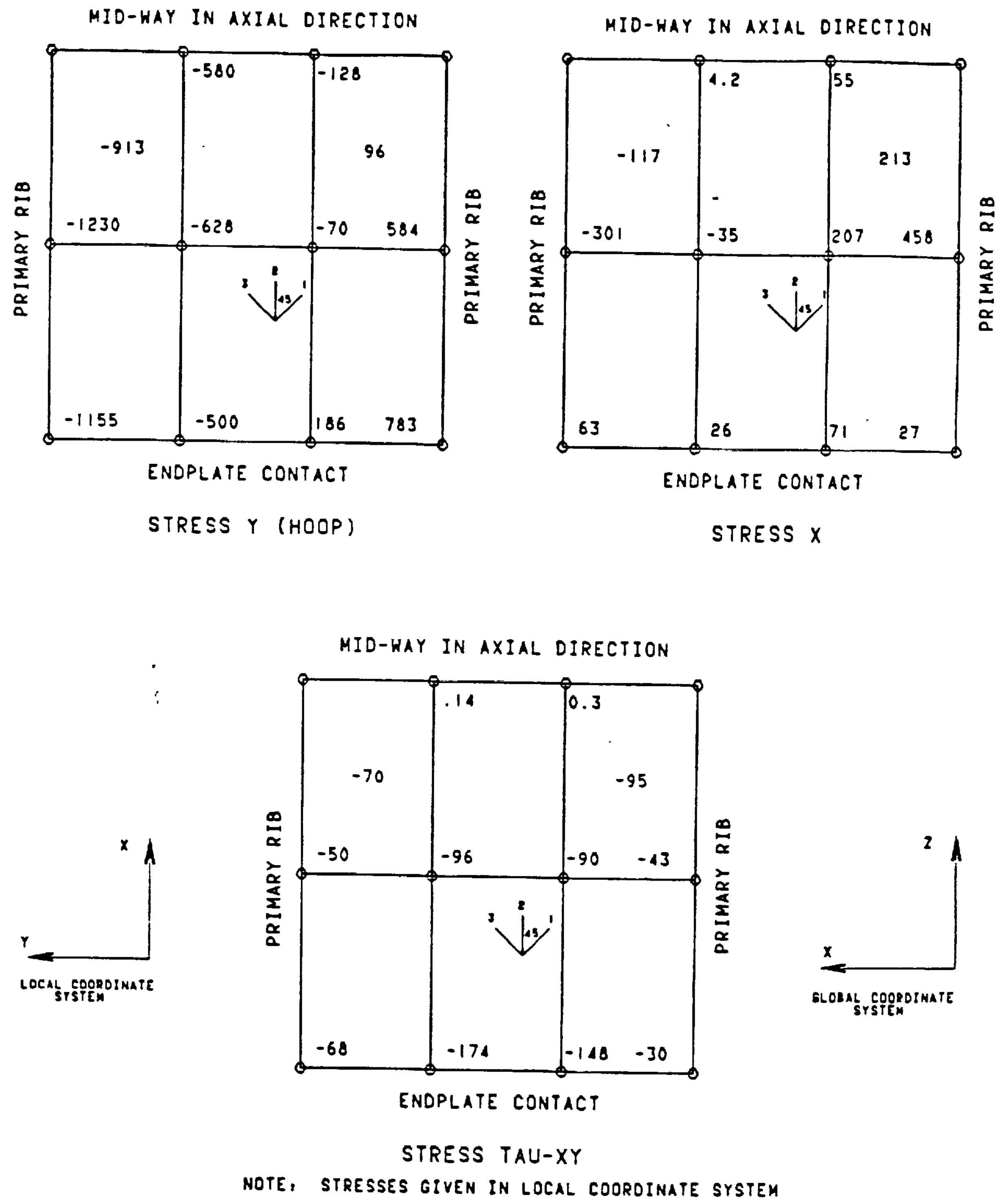


FIGURE 4.16: STRESSES AT LOCATION X
LINEAR THIN SHELL MODEL
RESTRAINT CASE 2

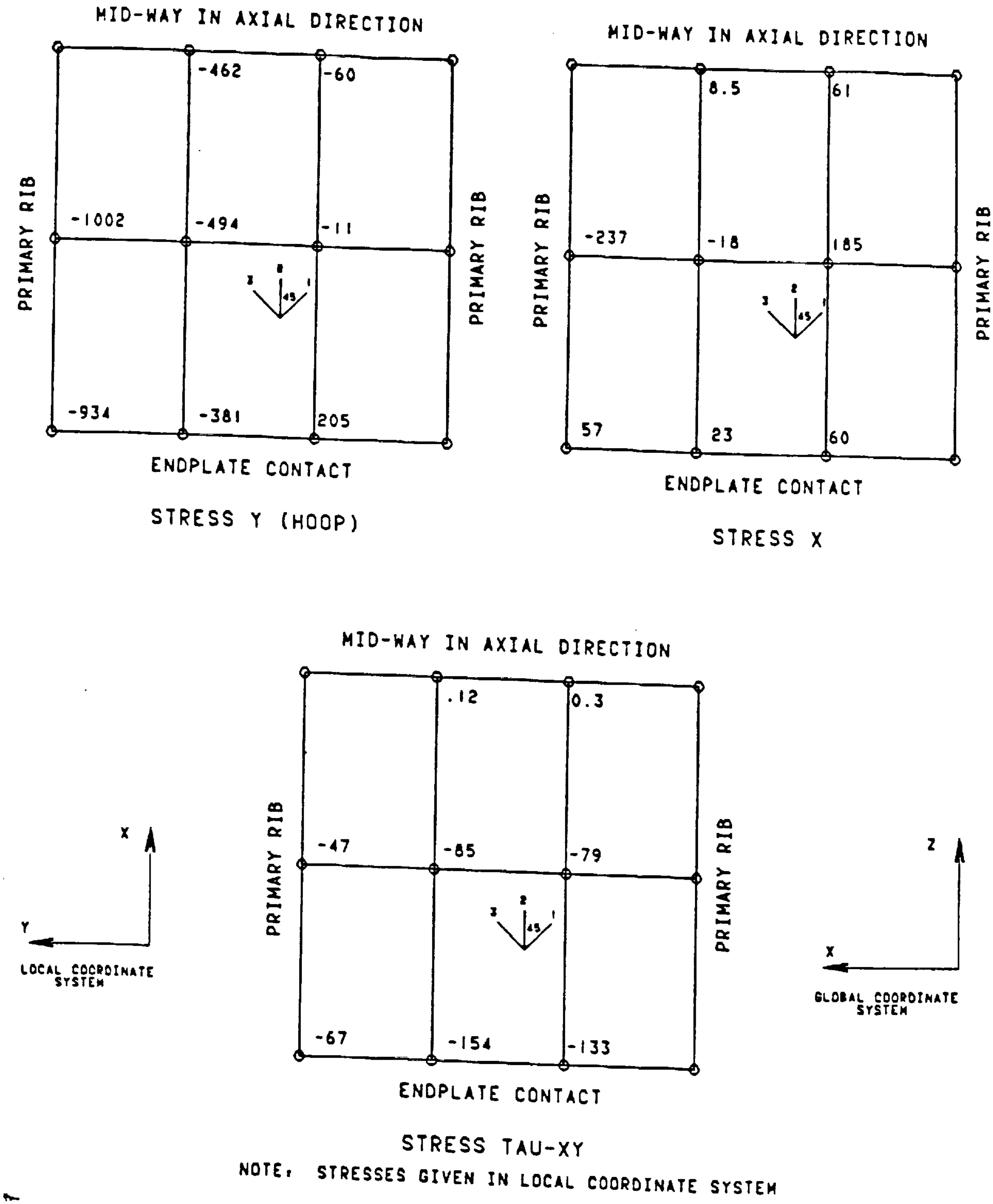


FIGURE 4.17: STRESSES AT LOCATION X
LINEAR THIN SHELL MODEL
RESTRAINT CASE 3

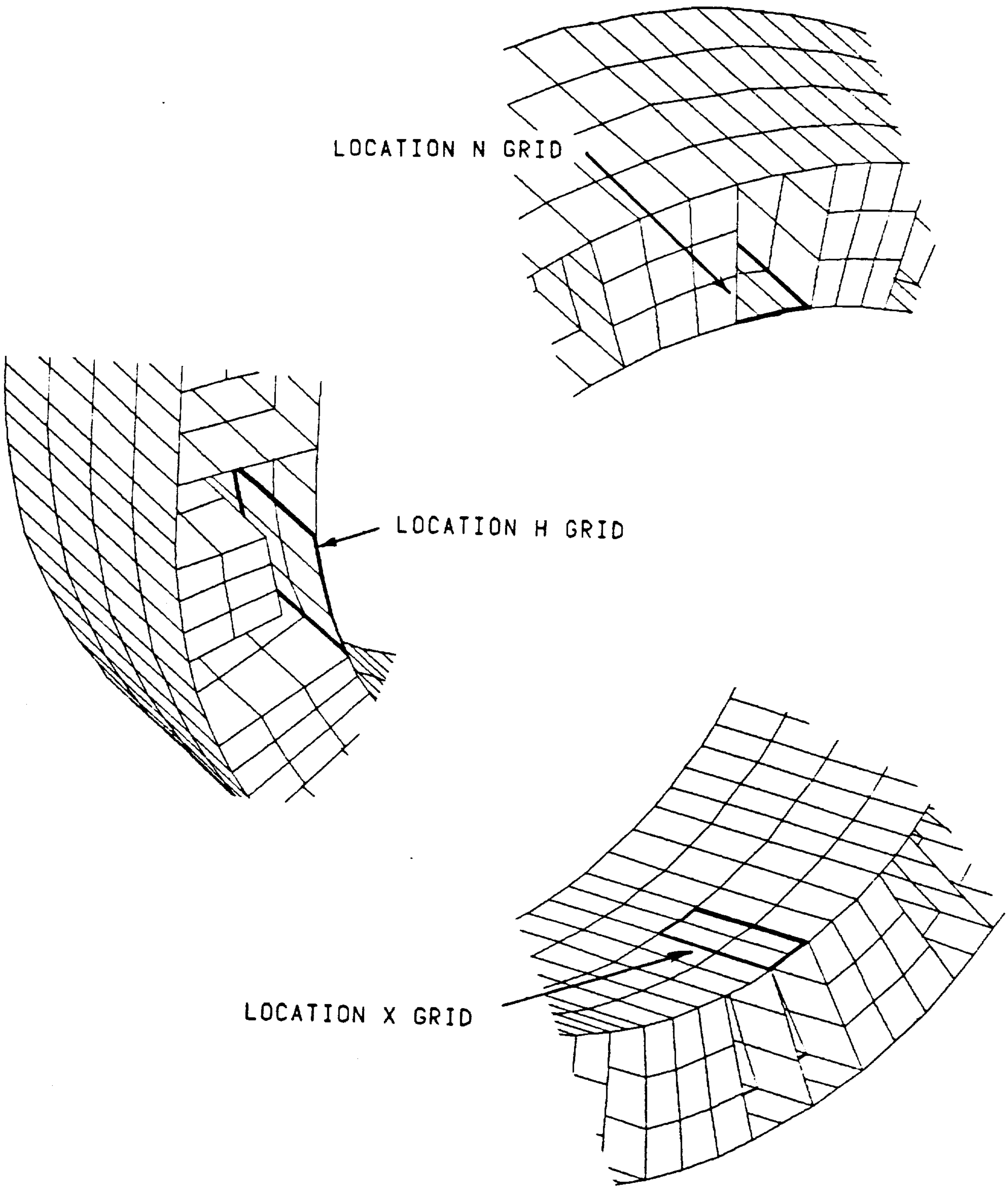


Figure 4.18: Illustration of strain gaye rossette and stress grid locations.

LOCATION	ITEM	TIED @ LOC. X	TIED @ TRANS	TIED @ LOC. N	EXP DATA
N	SIGMA-X	20	20	20	75
	SIGMA-Y	-350	-350	-200	-597
	TAU-XY	-15	30	0.0	-85
	SIGMA-P1	-4.3	22	20	86
	SIGMA-P2	-350	-352	-200	-608
	DIRECTION	2.5	-7.9	0.0	7.1
H	SIGMA-X	100	100	100	-9.3
	SIGMA-Y	700	700	700	702
	TAU-XY	-40	-40	-40	36
	SIGMA-P1	702	702	702	704
	SIGMA-P2	97	97	97	-11
	DIRECTION	-3.4	-3.4	-3.4	2.9
X	SIGMA-X	120	120	120	85
	SIGMA-Y	40	100	60	101
	TAU-XY	-100	-100	-100	-77
	SIGMA-P1	187	210	194	170
	SIGMA-P2	-27	9.5	-14	15
	DIRECTION	34	42	37	47

*NOTE: POSITIVE DIRECTIONS INDICATE CLOCKWISE ROTATION OF THE AXES.

STRESSES GIVEN IN PSI

Table 2: Results from linear thin shell model.

effect of modifications in the mesh could easily be investigated. However, the performance of the linear thin shell model indicated that the need for mesh refinements was limited. Therefore, the major drive behind the construction of the parabolic thin shell model was verification of the existing mesh. The introduction of parabolic elements provided the capability to model shearing actions as well as additional data points. Both of these features were expected to produce better agreement with the measured values. Furthermore, although stresses were expected to be slightly higher than those obtained previously, similar results were anticipated.

4.21 ANALYSIS OF THE PARABOLIC THIN SHELL MODEL

With the use of the same geometry, construction of the parabolic thin shell model was easily completed. When finished, the model was similarly subjected to the longitudinal tensile test. Support of the housing was achieved by utilizing support case 2. In addition, results were obtained in the same manner as before. As in the linear model, high stress gradients occurred around each rosette location. The results resembled the gradients obtained with the linear model. These results are shown in figures 4.20-4.22, and the approximated stresses are listed in table 3. Also, loading and support of the model was confirmed through the use of the deformation plot shown in figure 4.19.

The concurrence of the parabolic thin shell model's results to the results from the linear thin shell model proves convergence of the mesh used. Furthermore, the expected slight increase in the stresses obtained, furnishes added confidence in the model's accuracy. Therefore, at this point in the study, it was felt that the structural characteristics of the housing were modeled quite adequately.

LOCATION	ITEM	MODEL RESULTS	EXP. DATA
N	SIGMA-X	50	75
	SIGMA-Y	-300	-597
	TAU-XY	-55	-85
	SIGMA-P1	58	86
	SIGMA-P2	-308	-608
	DIRECTION	8.7	7.1
H	SIGMA-X	60	-9.3
	SIGMA-Y	700	702
	TAU-XY	-40	36
	SIGMA-P1	702	704
	SIGMA-P2	57	-11
	DIRECTION	-3.6	2.9
X	SIGMA-X	80	85
	SIGMA-Y	100	100
	TAU-XY	-75	-76
	SIGMA-P1	166	170
	SIGMA-P2	14	15
	DIRECTION	48	48

NOTE: POSITIVE DIRECTION INDICATES CLOCKWISE ROTATION OF AXES.

STRESSES GIVEN IN PSI

Table 3: Results obtained from parabolic thin shell model.

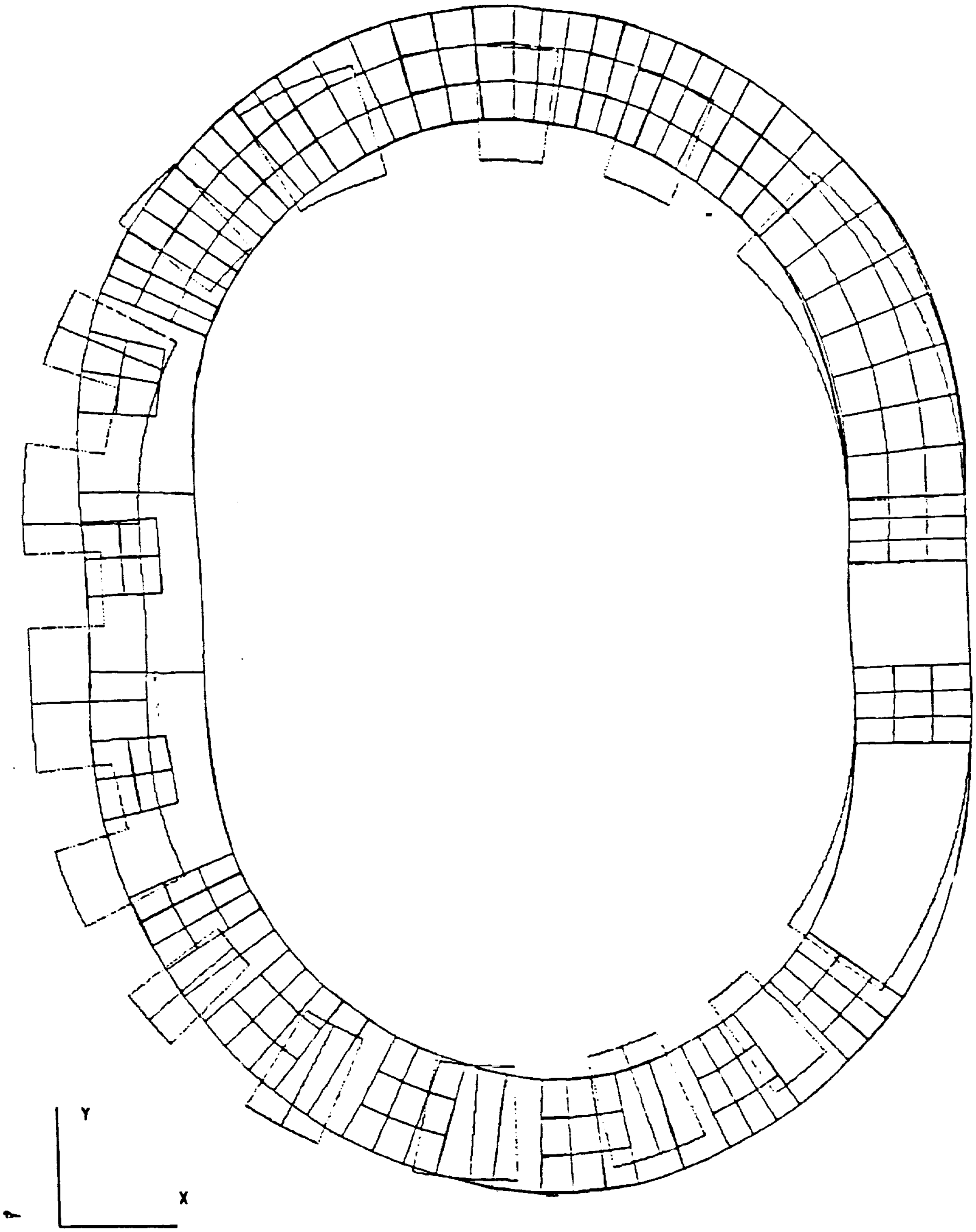


Figure 4.19: Deformation of the parabolic thin shell model subjected to a 1000 lb. tensile load

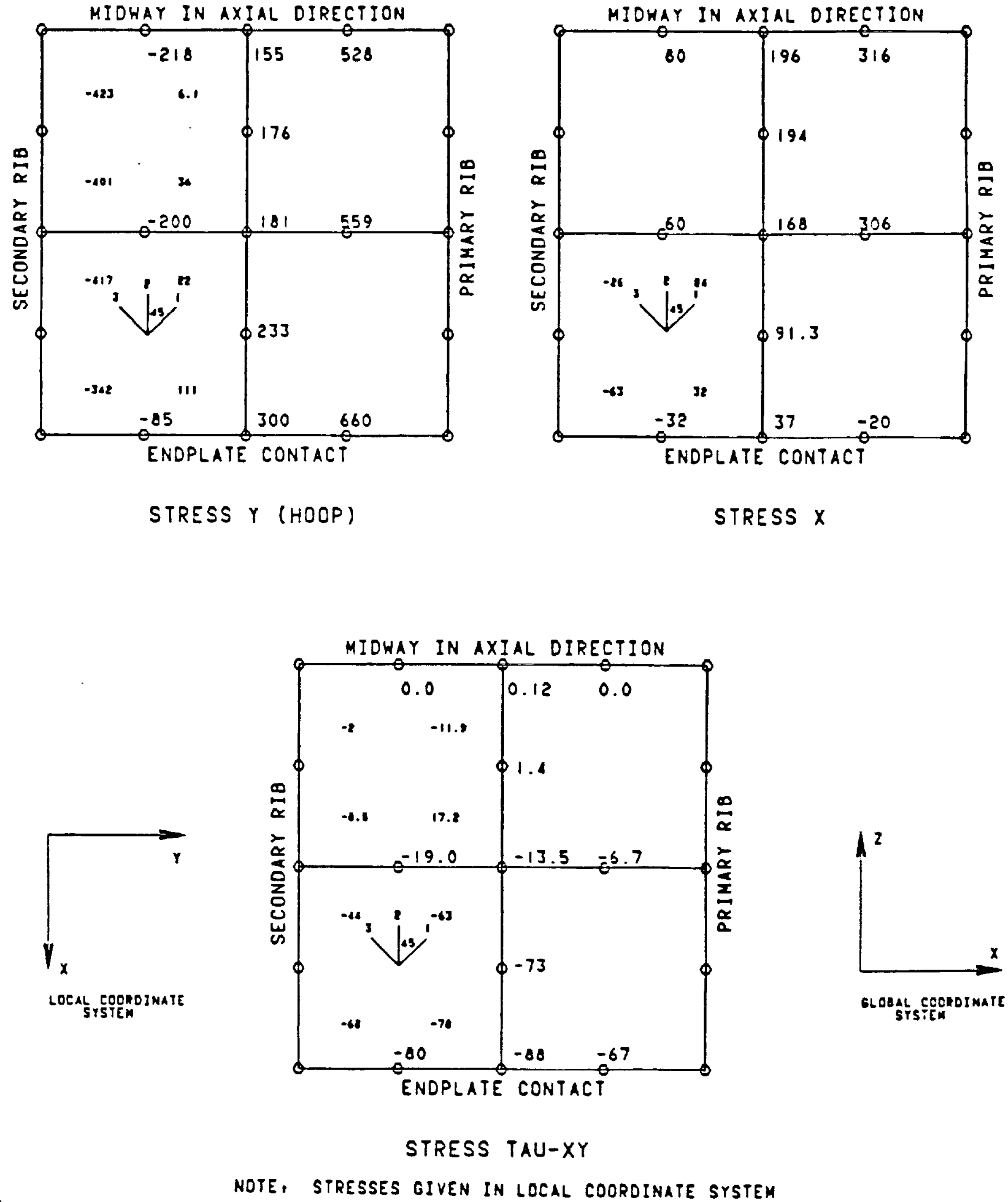


FIGURE 4.20: STRESSES AT LOCATION N
PARABOLIC THIN SHELL MODEL
RESTRAINT CASE 2

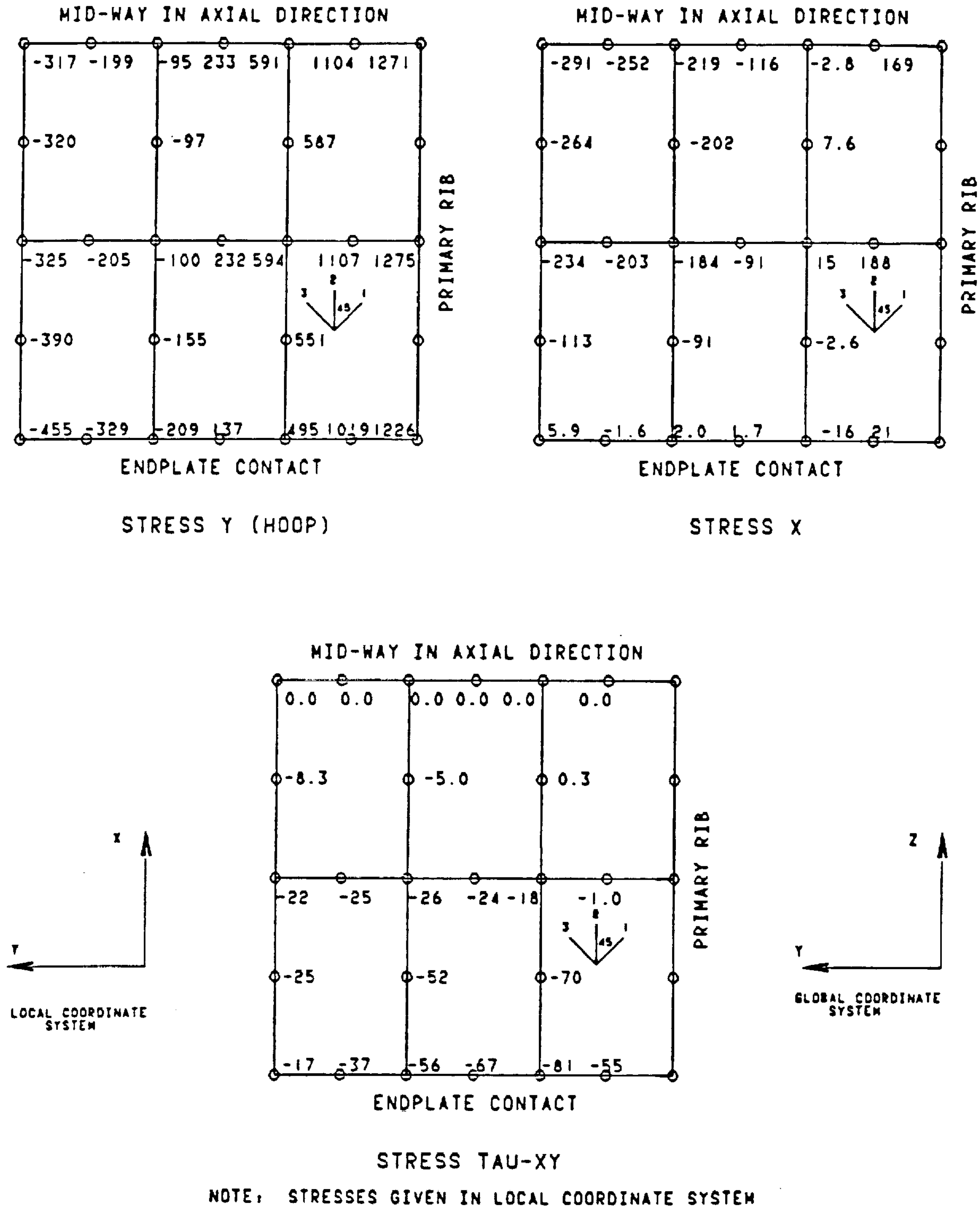


FIGURE 4.21: STRESSES AT LOCATION H
PARABOLIC THIN SHELL MODEL
RESTRAINT CASE 2

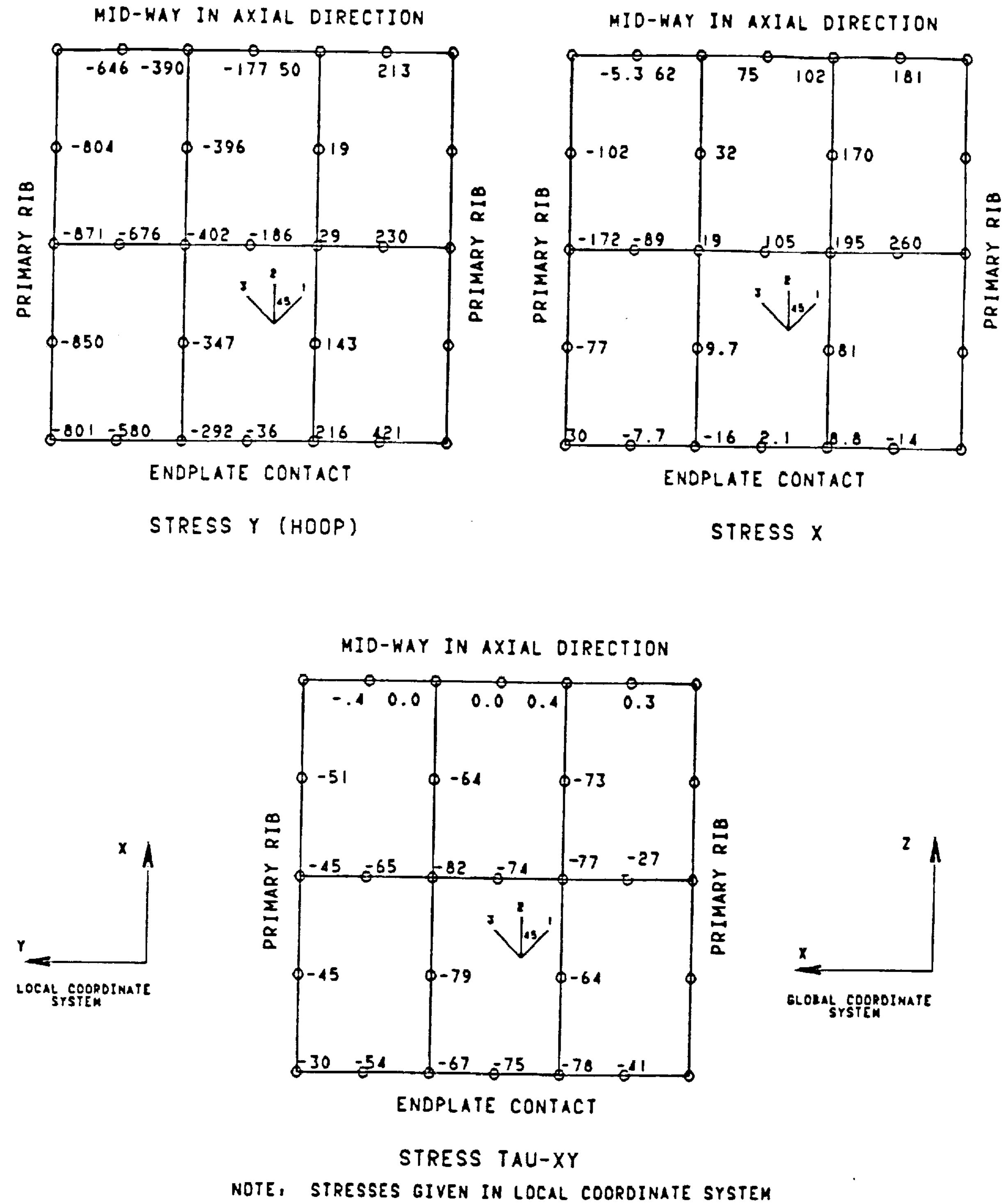


FIGURE 4.22: STRESSES AT LOCATION X
PARABOLIC THIN SHELL MODEL
RESTRAINT CASE 2

4.30 CONSTRUCTION OF THE PARABOLIC THICK SHELL MODEL

Once the adequacy of the modeling techniques used was confirmed, construction of a model that could support a thermal analysis was needed. The third model, the parabolic thick shell model, encompassed this requirement. With the introduction of the parabolic thick shell model, shown in figure 4.23, the configuration of the FEM of the housing approached the form that was introduced in chapter 2. In this model, SUPERB's thick shell elements replaced the thin shell elements previously used to model the inner and outer shells. The remaining geometry, however, continued to utilize the same element types. Also, the same mesh configurations were employed.

Construction of the parabolic thick shell model was noticebly more complicated than the thin shell models, due to the fact that 3-D representations of the inner and outer shells were required. Once more, the spark plugs, engine mounts, and both ports were not modeled. The mesh size remained the same and construction methods were unchanged. However, one major problem surfaced after analysis had begun. Although the compatibility problems that occurred in the thin shell models were solved by using thick shell elements, the use of these elements produced the same problems in the mid-plane stiffener and secondary rib region of the thick shell models. Since nodal rotations in these regions were no longer supported as they were in the thin shell models, deformations, as seen in figure 4.24, could occur. The solution to this problem was quite innovative. A beam element which resisted bending but offered no resistance to uniaxial loading was introduced. By "burying" the beams into the thick shell and solid elements, as shown in figure 4.25, the stiffness of the regions was greatly increased.

It was anticipated that the outcome of the parabolic thick shell model would concur with the previous thin shell models. Since the structural qualities of the thick shell elements

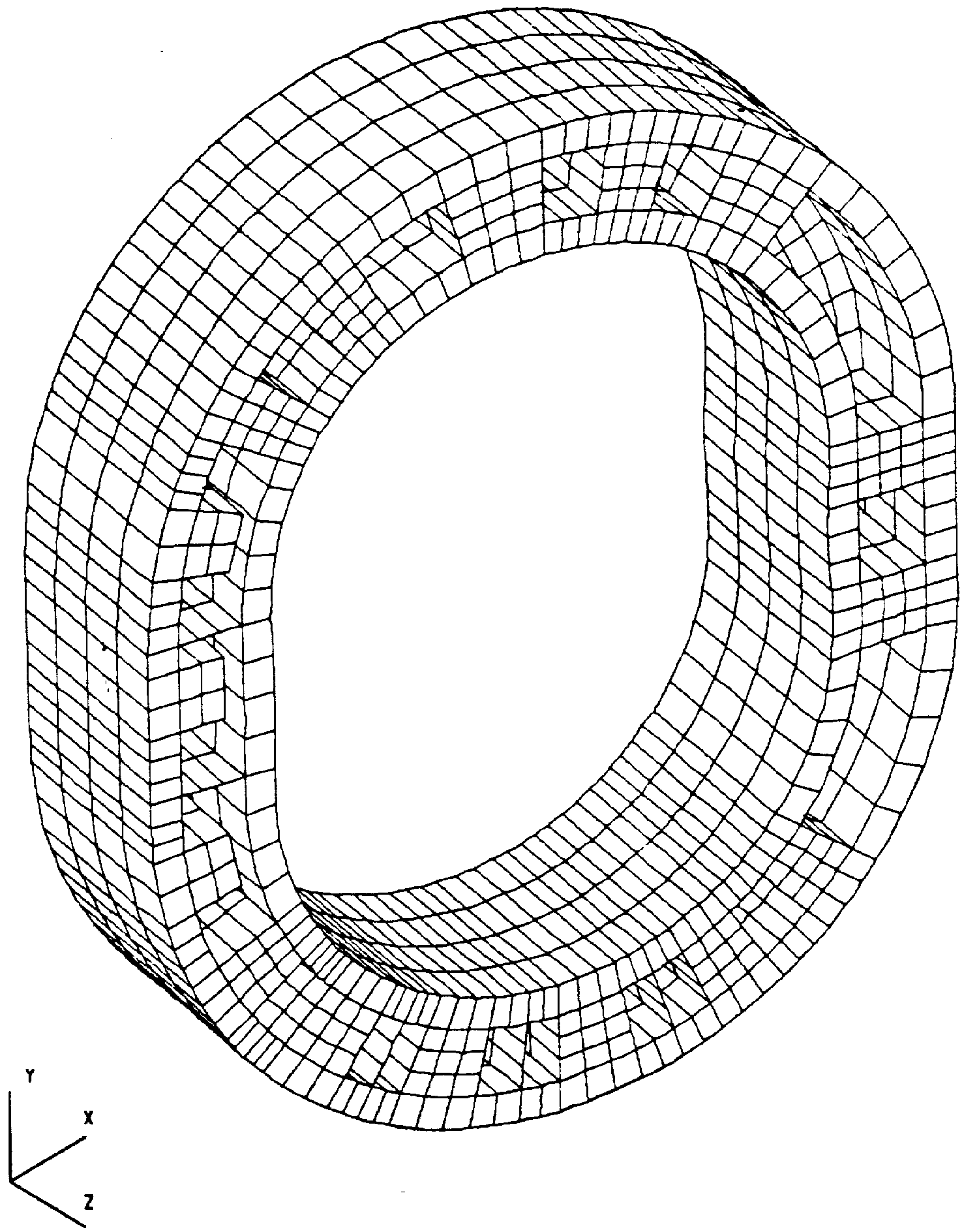


Figure 4.23: Illustration of the parabolic thick shell model.

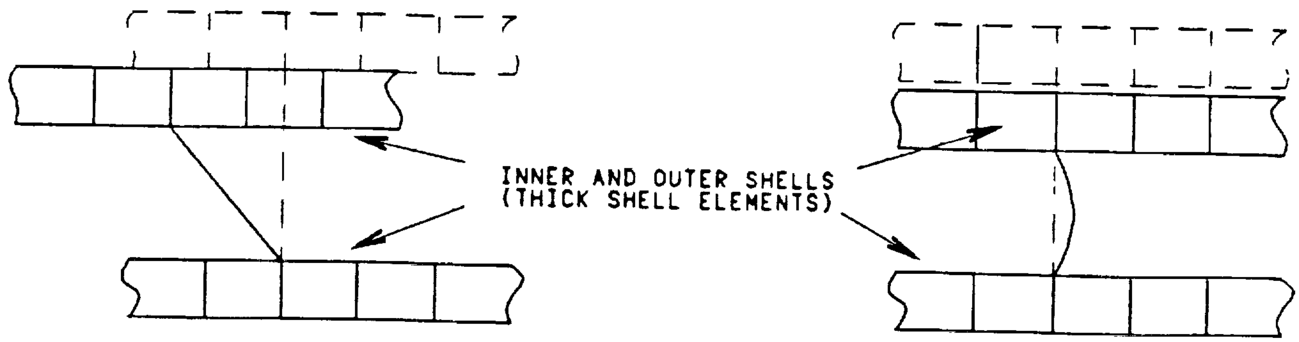


FIGURE 4.24: ILLUSTRATION OF POSSIBLE DEFORMATIONS OF HOUSING WITHOUT ELEMENTAL ROTATIONS SUPPORTED.

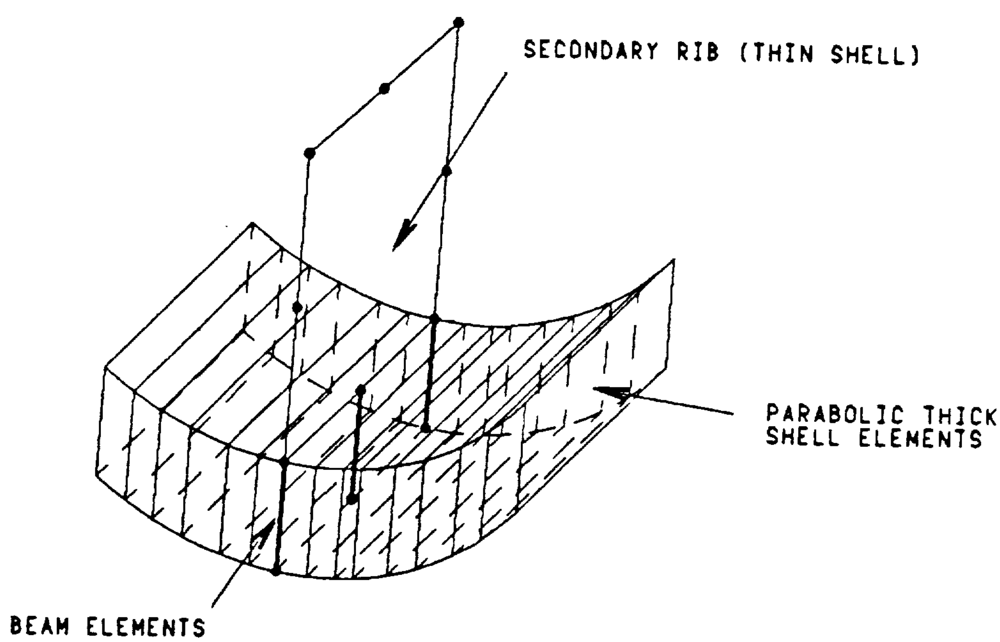


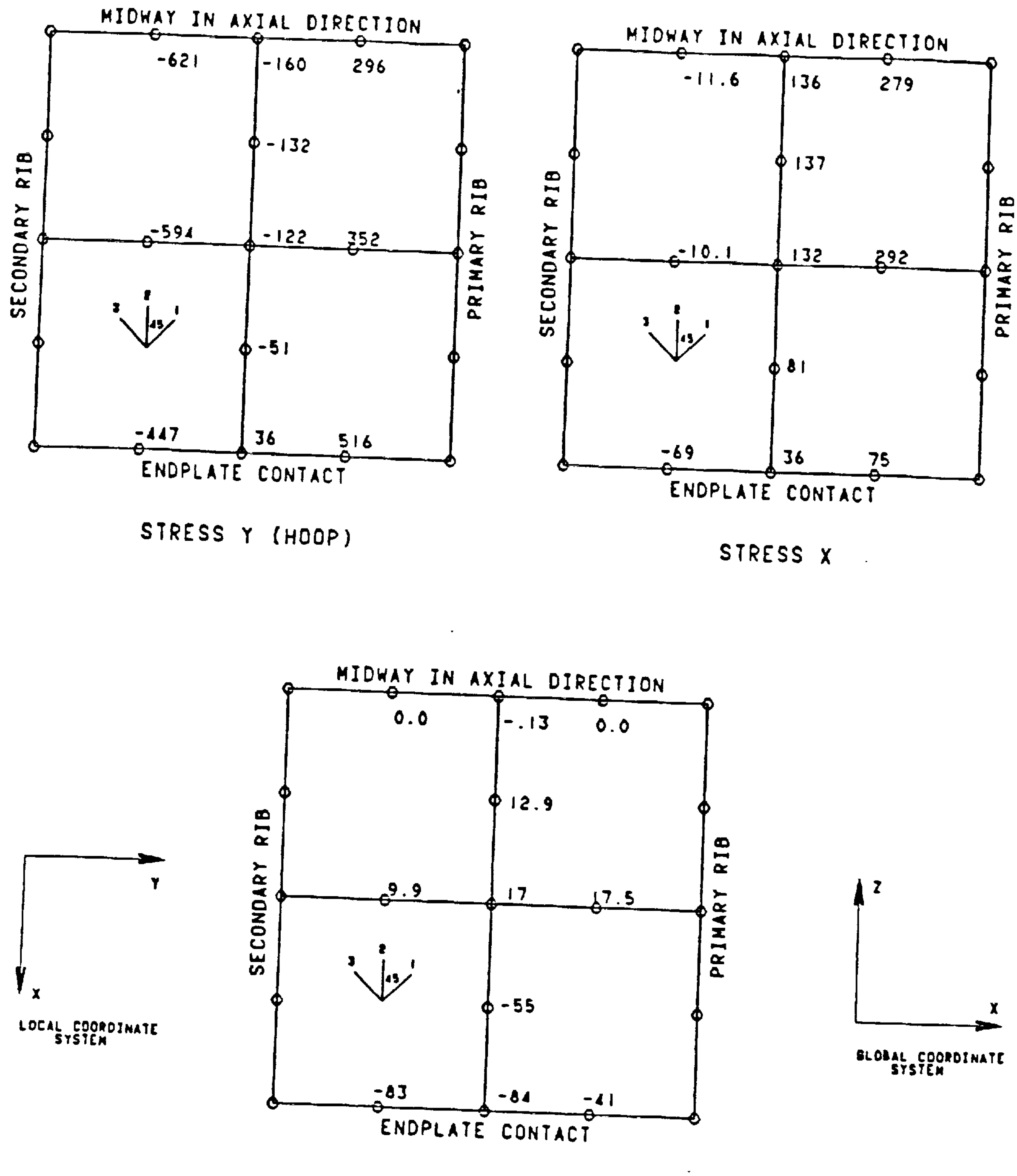
FIGURE 4.25: ILLUSTRATION OF THE INTRODUCTION OF BEAM ELEMENTS.

closely matched those of the thin shell elements, the stresses were not expected to be affected. The elimination of the compatibility problems around the primary ribs, however, would tend to make the model predict the deformation of the inner shell more closely. Also, it was predicted that the introduction of beam elements would not only provide adequate support in the regions involved, but also their effect on the inner and outer shells was expected to be negligible. This later expectation was due to the fact that the thick shell elements should "classically" deform, planes would remain planes. In conclusion, it was felt that the parabolic thick shell model would introduce thermal modeling capability but not effect the excellent structural modeling capabilities already shown.

4.31 ANALYSIS OF THE PARABOLIC THICK SHELL MODEL

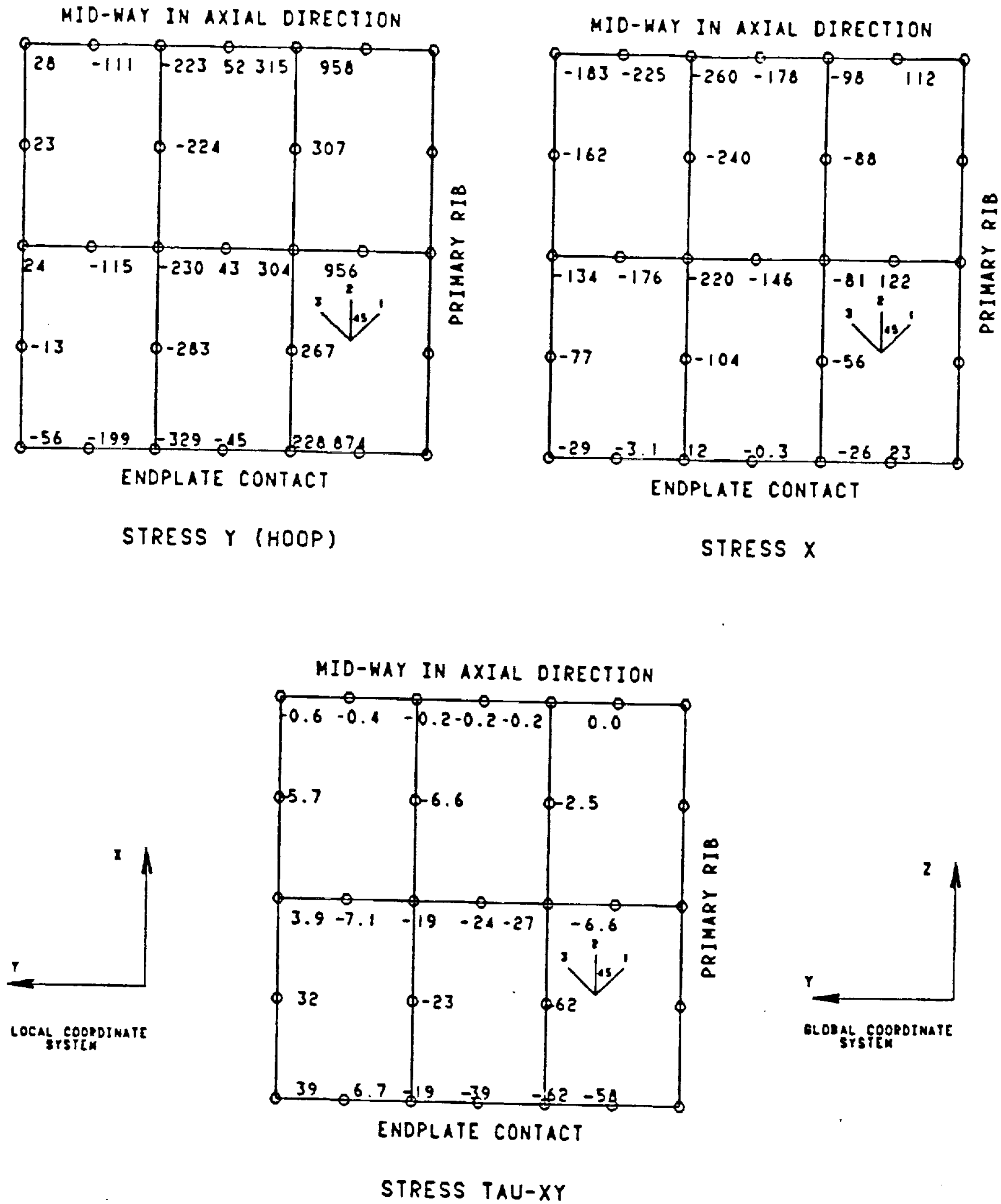
Testing of the parabolic thick shell model was completed in much the same manner as the previous models. The longitudinal tensile test was applied to the model and support case 2 again supported the model. A change in support case 2, however was required. The need for this change was described in chapter 2. Basically, in order to support axial rotation of the model, the translations of two rows of nodes were supported.

As predicted, the results of the parabolic thick shell model agreed with the previous thin shell models'. Displayed in figures 4.26-4.28, the stresses were again located in high stress gradient regions. The approximated stresses at each rosette location are listed in table 4. Comparison with the experimental data was quite good. Also, as can be seen in figure 4.29, deformation of the housing confirms the loading and support of the housing.



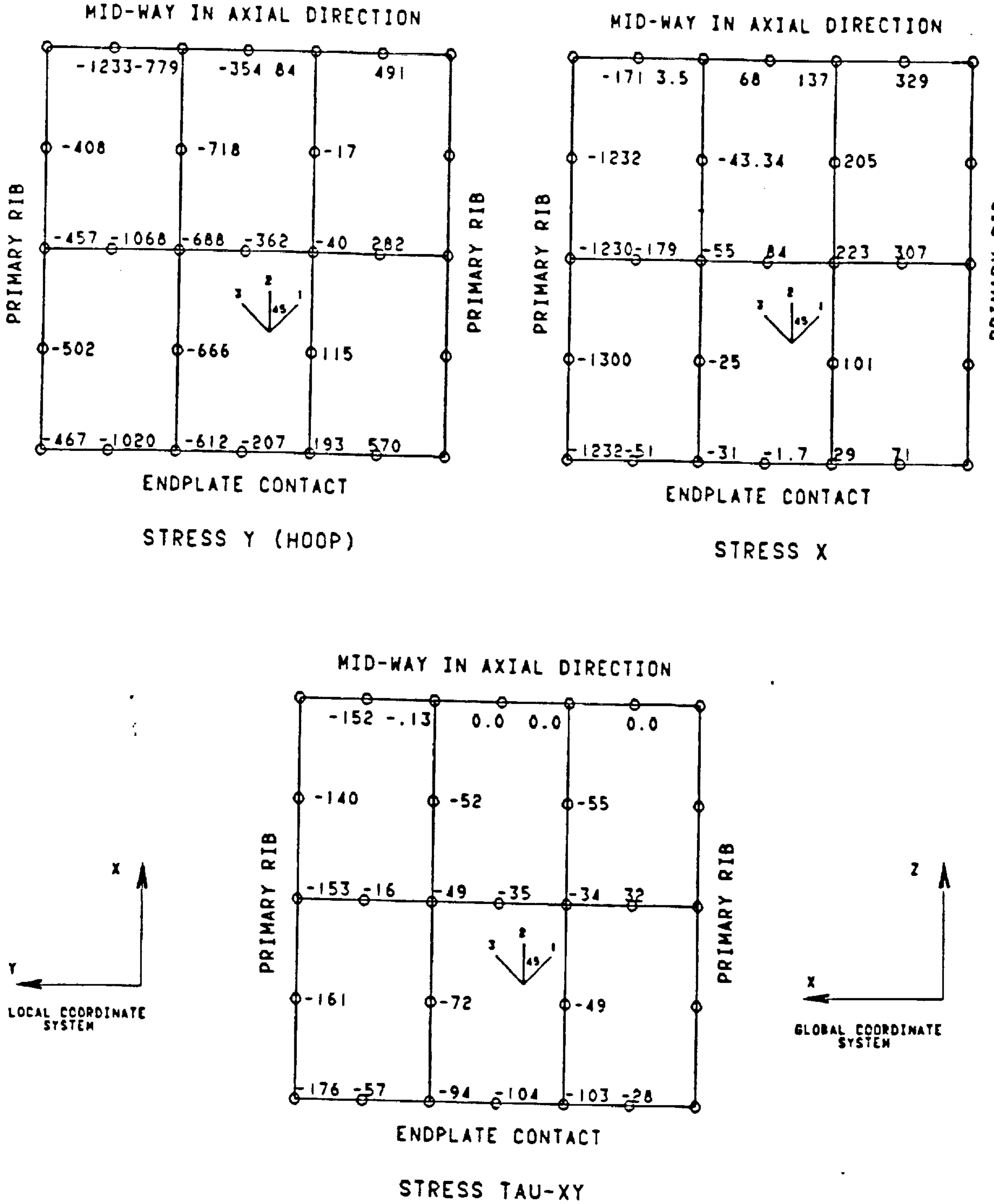
NOTE: STRESSES GIVEN IN LOCAL COORDINATE SYSTEM

FIGURE 4.26: STRESSES AT LOCATION N
PARABOLIC THICK SHELL MODEL
RESTRAINT CASE 2



NOTE: STRESSES GIVEN IN LOCAL COORDINATE SYSTEM

FIGURE 4.27: STRESSES AT LOCATION H
PARABOLIC THICK SHELL MODEL
RESTRAINT CASE 2



NOTE: STRESSES GIVEN IN LOCAL COORDINATE SYSTEM

FIGURE 4.28: STRESSES AT LOCATION X
PARABOLIC THICK SHELL MODEL
RESTRAINT CASE 2

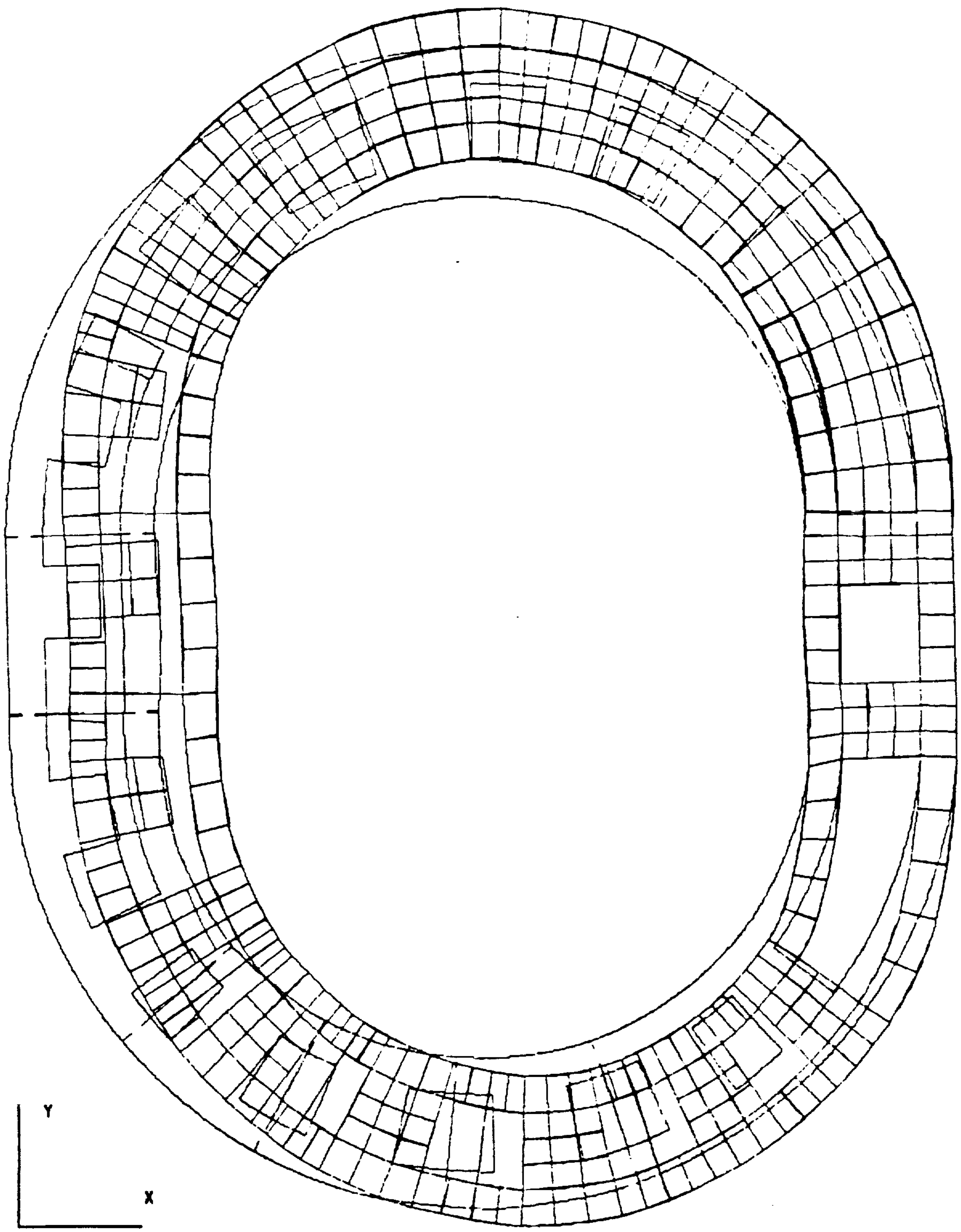


Figure 4.29: Deformation of the parabolic thick shell model subjected to a load of 1000 lbs.

LOCATION	ITEM	MODEL RESULTS	EXP DATA
N	SIGMA-X	0.0	75
	SIGMA-Y	-560	-597
	TAU-XY	-50	-85
	SIGMA-P1	4.4	86
	SIGMA-P2	-564	-608
	DIRECTION	5.1	7.1
H	SIGMA-X	50	-9.3
	SIGMA-Y	660	702
	TAU-XY	-50	36
	SIGMA-P1	664	704
	SIGMA-P2	45	-11.2
	DIRECTION	-4.7	2.9
X	SIGMA-X	90	85
	SIGMA-Y	50	100
	TAU-XY	-60	-76
	SIGMA-P1	133	170
	SIGMA-P2	6.8	15
	DIRECTION	36	48

NOTE: POSITIVE DIRECTION INDICATES CLOCKWISE ROTATION OF AXES.

STRESSES GIVEN IN PSI

Table 4: Results obtained from parabolic thick shell model.

5.00 PRESENTATION AND REVIEW OF THE FINAL MODEL

After the analysis of the parabolic thick shell model was completed, it was clear that a structurally sound model of the housing, potentially capable of accurately completing a thermal and dynamic analysis, had been constructed. Thus, the balance of this study included adding the additional geometry previously left out: the spark plug and port regions. Furthermore, performance of this model was checked against both experimental tensile tests. Finally, since this is the foundation of a much larger study, documentation on the construction and ability of the FEM of the housing is presented.

5.10 CONSTRUCTION

Construction of the final FEM, shown in figure 5.1, basically added the last pieces of geometry to the parabolic thick shell model. As can be seen in figures 5.2-5.4, the placement of the ports and spark plug holes was not an easy task. Basically, three cylinders and a box were inserted radially in the model such that the inner bore and outer shell remained smooth and continuous. Placement of this geometry had to not only be consistent with the housing's geometry, but also, the meshes used had to be compatible with the existing mesh. There was no method, other than monitoring the effect on the rest of the housing, in which the construction could be verified. Thus, the introduction of the spark plugs and port regions proceeded on the basis of previous knowledge.

In order to construct the geometry of the ports and spark plug holes so that they would lie on the inner and outer surfaces, a numerical routine was again inplemented (appendix 4). By utilizing this, consistent placement of the construction

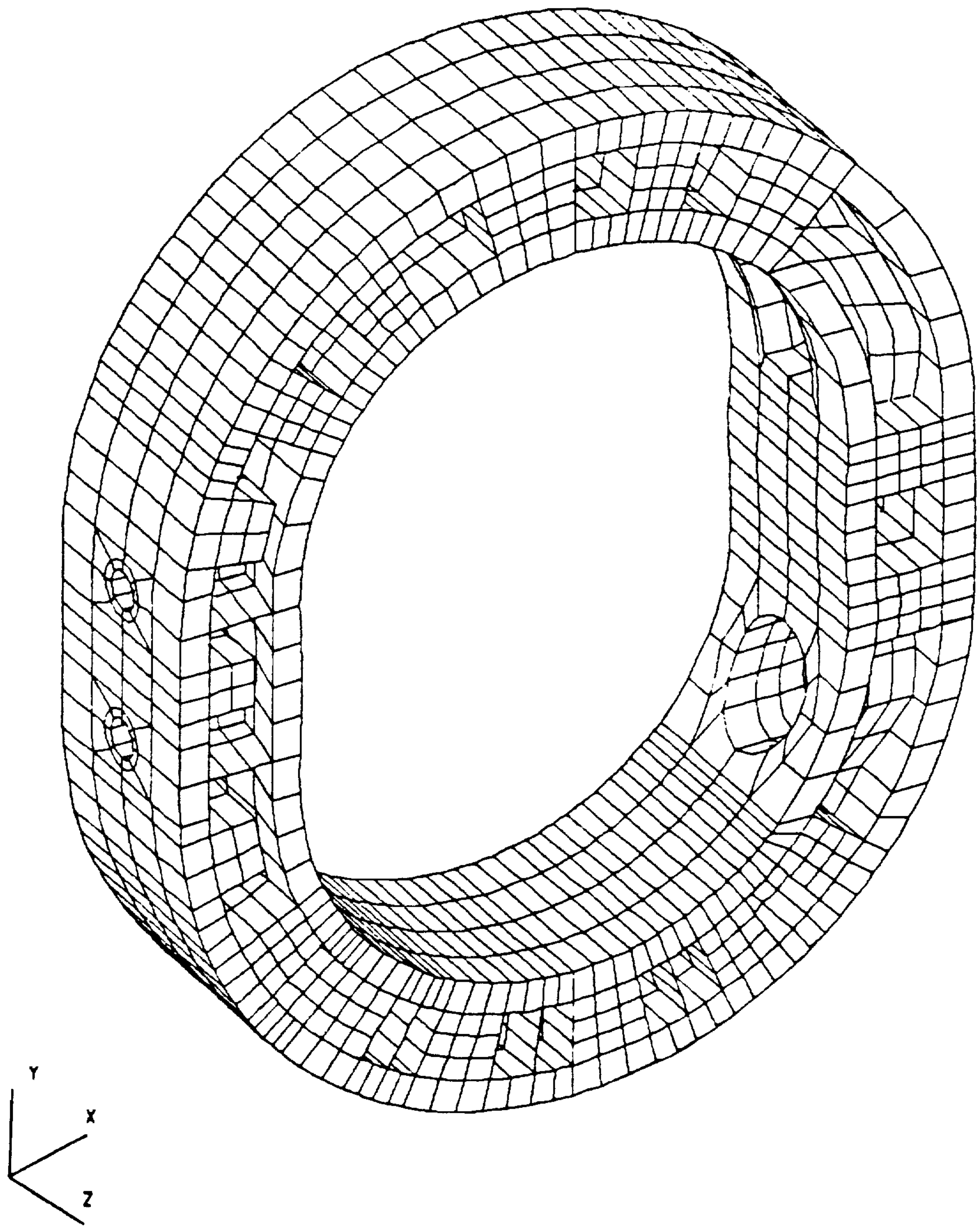


Figure 5.1: Illustration of the final FEM of the housing.

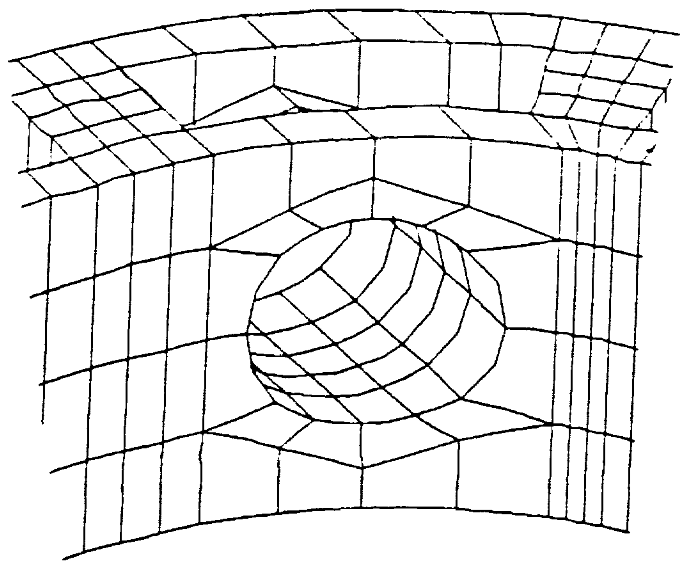


Figure 5.2: Illustration of exhaust port mesh.

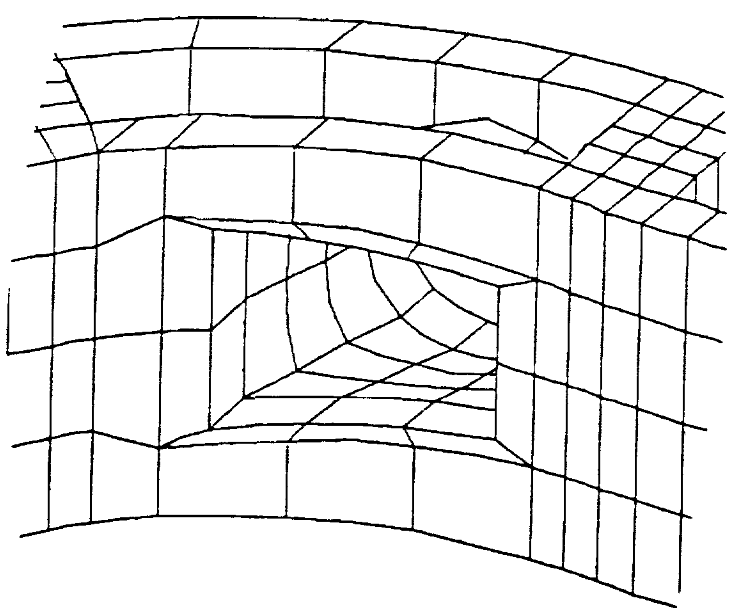
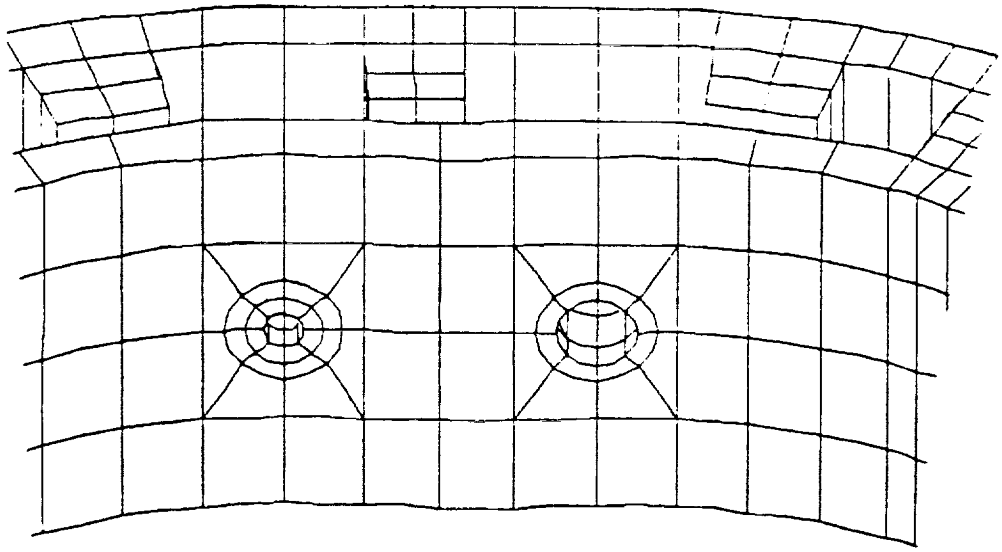
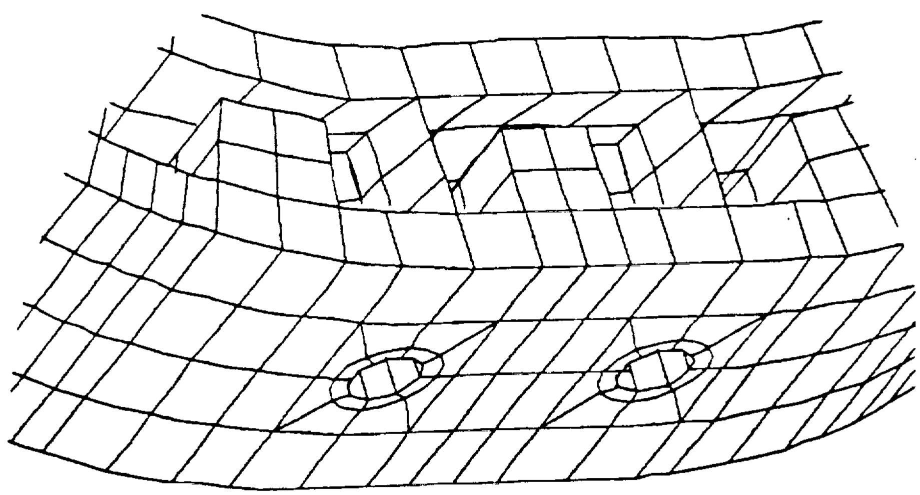


Figure 5.3: Illustration of intake port mesh.



(a) view of the inner bore.



(b) view of the outer surface.

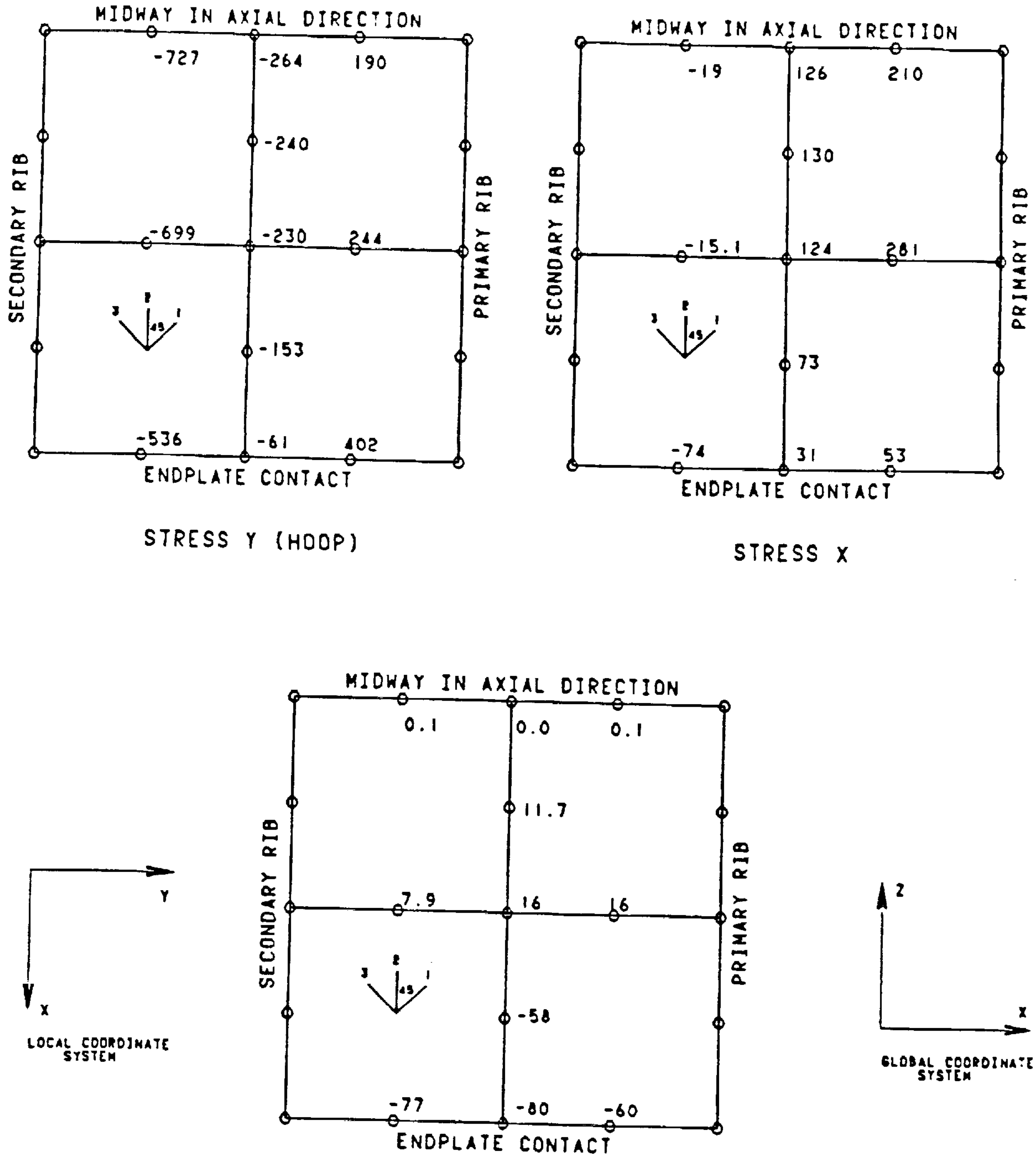
Figure 5.4: Illustration of spark plug region.

points was possible. After the wire frame geometry was completed, meshes were put in place. The resulting mesh was, again considered coarse and in actuality it consisted of a number of smaller meshes. Thick shell elements were employed for the port and spark plug walls. However, since the geometry was of odd shape, the actual connection between the ports and spark plug geometry and the housing was accomplished by using the coupling method. In these regions, though, the method did not affect the modeling capabilities at all because the elements were of the same or similar type and were therefore compatible.

In reviewing the final FEM's construction, a number of features must be noticed. First, the inner and outer surfaces were made of segments, one segment for each rib and associated cooling channel, which assured the existence of common meshes. Second, intricate geometry such as the bolt holes, rounds and fillets, and seal grooves was not included since the effect of this geometry was believed negligible. Third, there is an even number of elements in the axial direction. Fourth, elements are numbered using a radially advancing wavefront. Fifth, a coupling sequence for all coincident nodes was completed. Sixth and finally, four types of elements were used: parabolic thick shell, parabolic thin shell, parabolic solid, and linear beam.

5.20 PERFORMANCE AND ABILITY

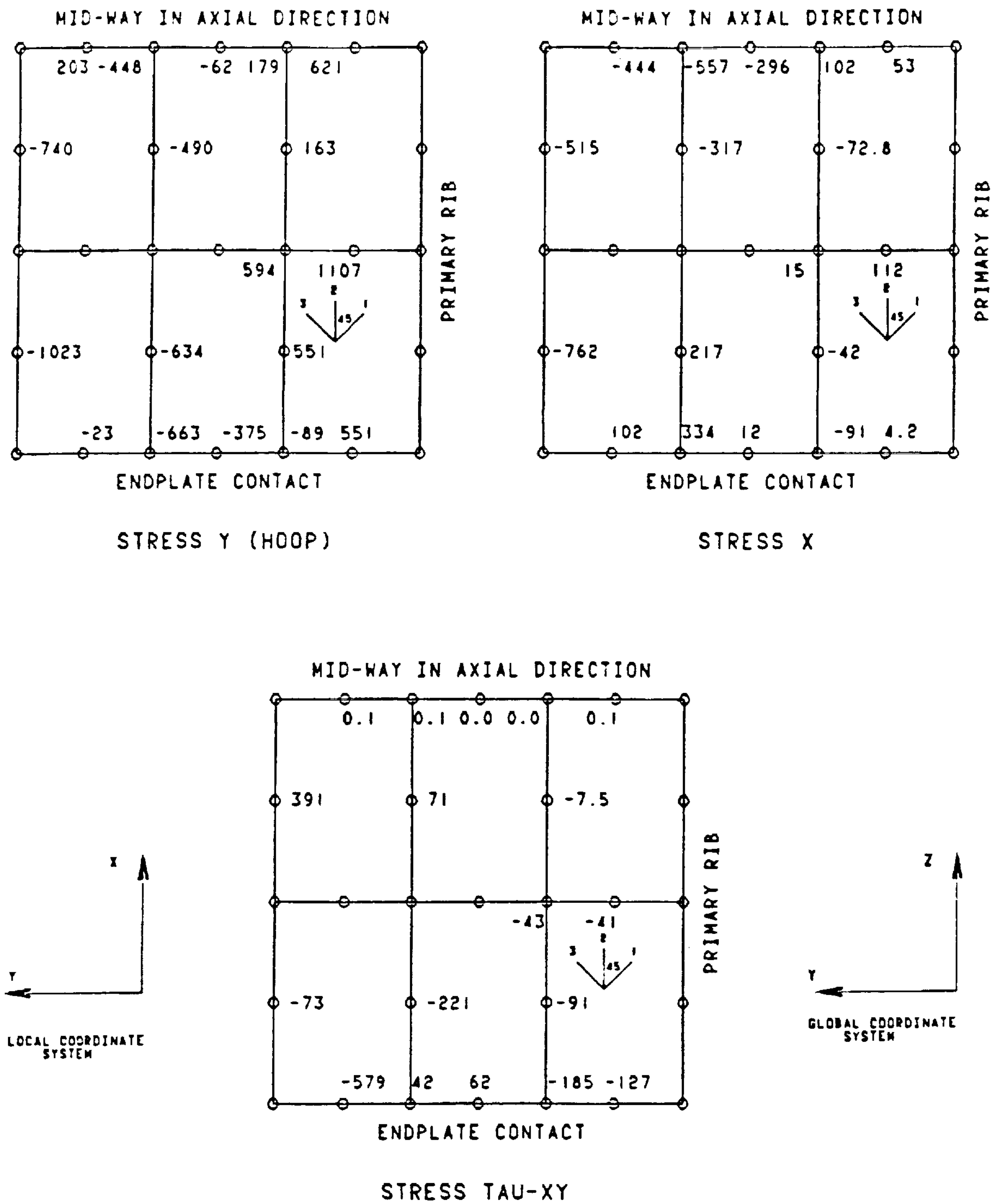
Performance of the final model was, again, matched against the experimental results obtained from the longitudinal tensile test. Restraint case 2 was used to support the model. As expected, the performance of the final FEM of the housing was excellent. From the results displayed in figures 5.5-5.7 and in table 5, it can be seen that the FEM not only matched the experimental data, but also concurred with the gradients determined in the previous models. It can also be seen, in figure 5.8, that the additional geometry's affect on the housing's deformation was minor. However, since a verification



STRESS TAU-XY

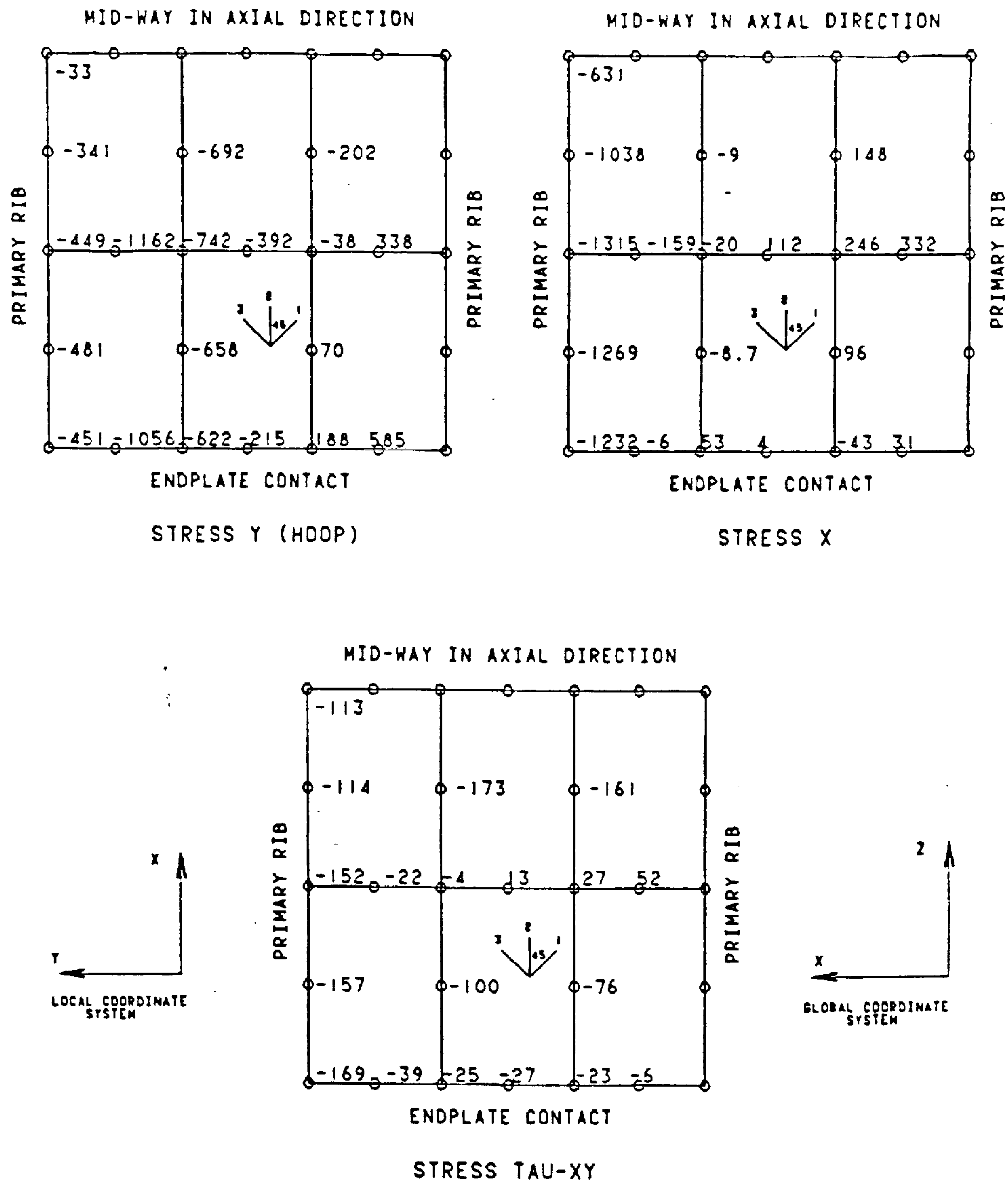
NOTE: STRESSES GIVEN IN LOCAL COORDINATE SYSTEM

FIGURE 5.5: STRESSES AT LOCATION N
FINAL FEM OF THE HOUSING
RESTRAINT CASE 2



NOTE: STRESSES GIVEN IN LOCAL COORDINATE SYSTEM

FIGURE 5.6: STRESSES AT LOCATION H
THE FINAL FEM OF THE HOUSING
RESTRAINT CASE 2



*NOTE: STRESSES GIVEN IN LOCAL COORDINATE SYSTEM

FIGURE 5.7: STRESSES AT LOCATION X
FINAL FEM OF HOUSING
RESTRAINT CASE 2

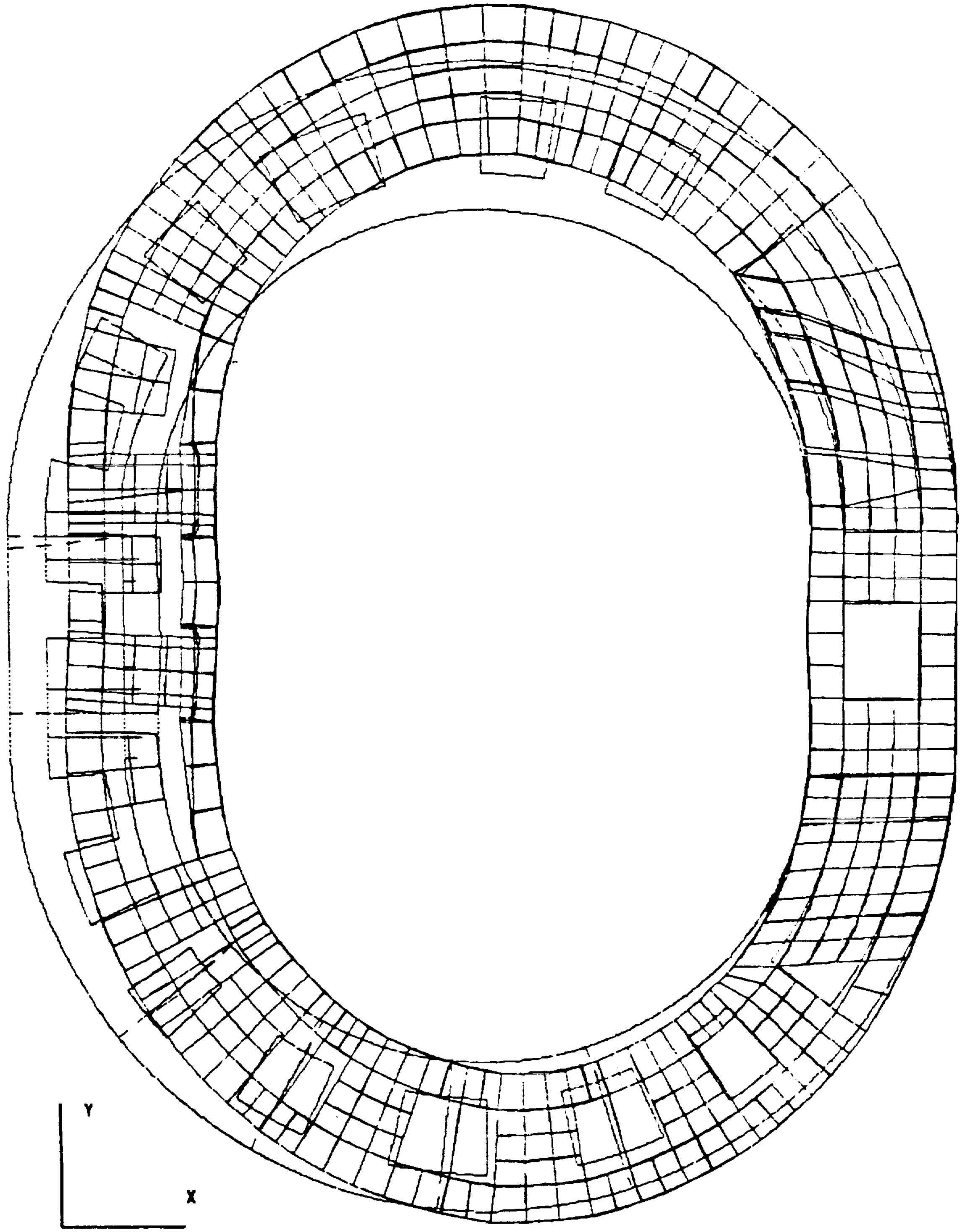


Figure 5.8: Deformation of final FEM subjected to a 1000 lb. tensile load.

LOCATION	ITEM	MODEL RESULTS	EXP DATA
N	SIGMA-X	10	75
	SIGMA-Y	-600	-597
	TAU-XY	-45	-85
	SIGMA-P1	13.3	86
	SIGMA-P2	-603	-608
	DIRECTION	4.2	7.1
H	SIGMA-X	30	-9.3
	SIGMA-Y	600	702
	TAU-XY	-80	36
	SIGMA-P1	611	704
	SIGMA-P2	19	-11.2
	DIRECTION	-7.8	2.9
X	SIGMA-X	100	85
	SIGMA-Y	30	100
	TAU-XY	-60	-76
	SIGMA-P1	135	170
	SIGMA-P2	-4.5	15.6
	DIRECTION	30	48

NOTE: POSITIVE DIRECTION INDICATES CLOCKWISE ROTATION OF AXES.

STRESSES GIVEN IN PSI

Table 5: Results obtained from the final FEM.

method was not available, the deformations and stresses in those regions must be further investigated.

With these results, it was felt that the purpose of this study had been attained. A structurally sound FEM of a Wankel engine center housing had been built and verified. Although it is considered to have a coarse mesh, it by no means a small FEM. It contains over 29,000 degrees of freedom and has a maximum wavefront of 588. It consists of 516 solid, 968 thick shell, 130 thin shell, and 167 beam elements. Along with the ability to perform a static analysis, the Wankel FEM is also capable of performing a steady-state thermal and/or dynamic analysis of the housing alone. In addition, since the FEM is a SUPERTAB base file, the model can be adjusted into NASTRAN, ANSYS, or SUPERB format. With this capability, the limitations to what can be performed with this as a base model are few.

6.00 CONCLUSIONS AND RECOMMENDATIONS

The basic result of this study was the presentation of a finite element model of a Wankel engine center housing that is valid for static loading conditions existing on the housing alone. The model performed well and is perceived to be an excellent foundation for further study. With the graphical preprocessor SUPERTAB as a base, changes in geometry and material can now be analyzed very quickly. Further analysis must be completed to verify the thermal and dynamic capabilities of the FEM. When completing the dynamic analysis, use of the previous thin shell models might be necessary. It should be realized, though, that a dynamic analysis of the housing alone should be completed first. In addition, the construction of the exhaust port and spark plug regions should be investigated before a thermal analysis is completed. As for completing an analysis on the interaction between the endplates and the housing, the actual endplates should be modeled. The deformation of the plates as well as the clearance in the bolt holes must be considered. The internal pressure test presented in this study would provide an excellent experimental baseline for such a study. Modeling of the preload and pressurization should obtain sufficient knowledge to enable an accurate prediction of the interaction. In conclusion, it is believed that all of the studies mentioned above can be performed on the FEM presented in this thesis with very few modifications required.

APPENDIX 1
SUPERB FINITE ELEMENT LIBRARY

A1.00 STRESS OUTPUT (COORDINATE SYSTEM OREINTATION)

From the nodal displacements computed, SUPERB determines stresses and strains at guass integration points within each element, the actual method will be discussed for each element type later. These guass point stresses are then extrapolated to the elements nodes. Thus, the nodal stress at a specific node is an average of the guass point stresses in the elements that are connected to the node. Nodal stresses are reported at each node except when a geometric discontinuity is encountered. When this occurs, guass point stresses are reported in the elements near the discontinuity. Furthermore, for thin shell elements, nodal stresses are reported at the top, middle, and bottom surfaces at each node. Typical stress outputs are shown in figures A1.1, A1.2, and A1.3.

Stresses can be reported in at least three different coordinate systems: global, local, or elemental. The global system refers to the coordinate system in which the geometry and nodal coordinates are defined. The local and elemental coordinate systems are defined by each element. Nodal stresses are reported in the global (G-STS), or local (L-STS), coordinate systems while the guass point stresses are reported in the elemental coordinate system. Exceptions to this are found when two different element types are connected (9).

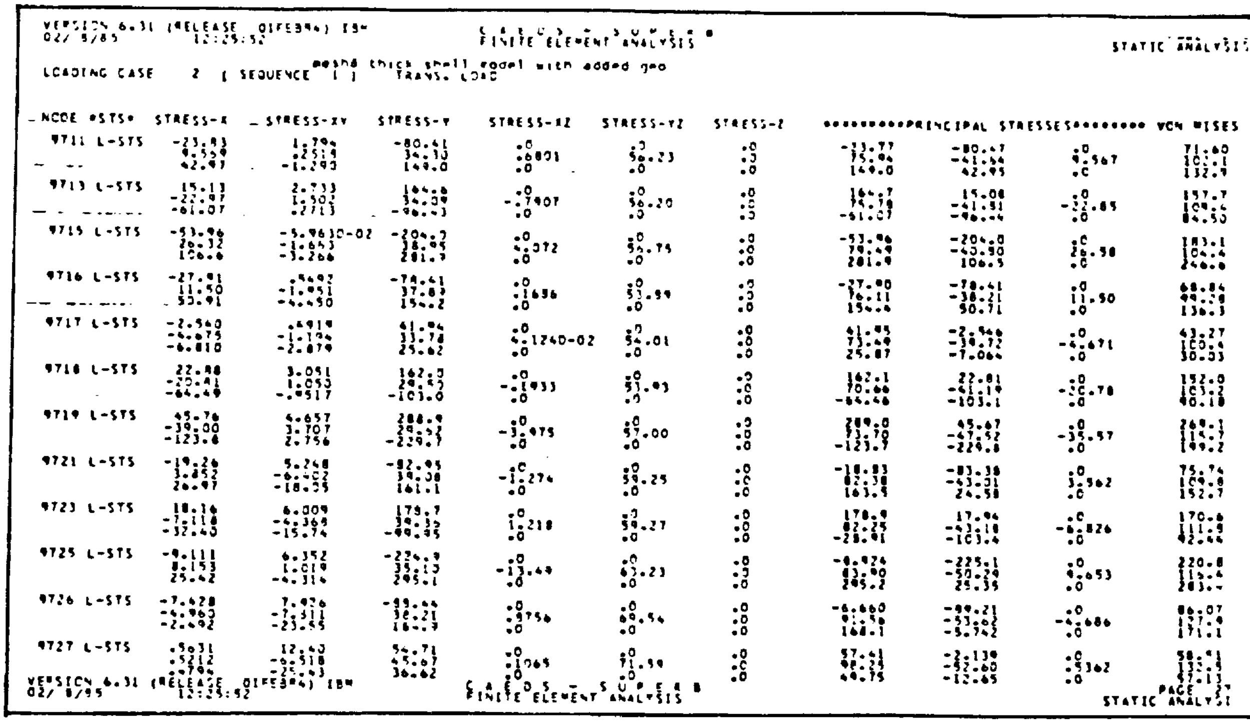


Figure A1.1: Typical thin shell element stress output.

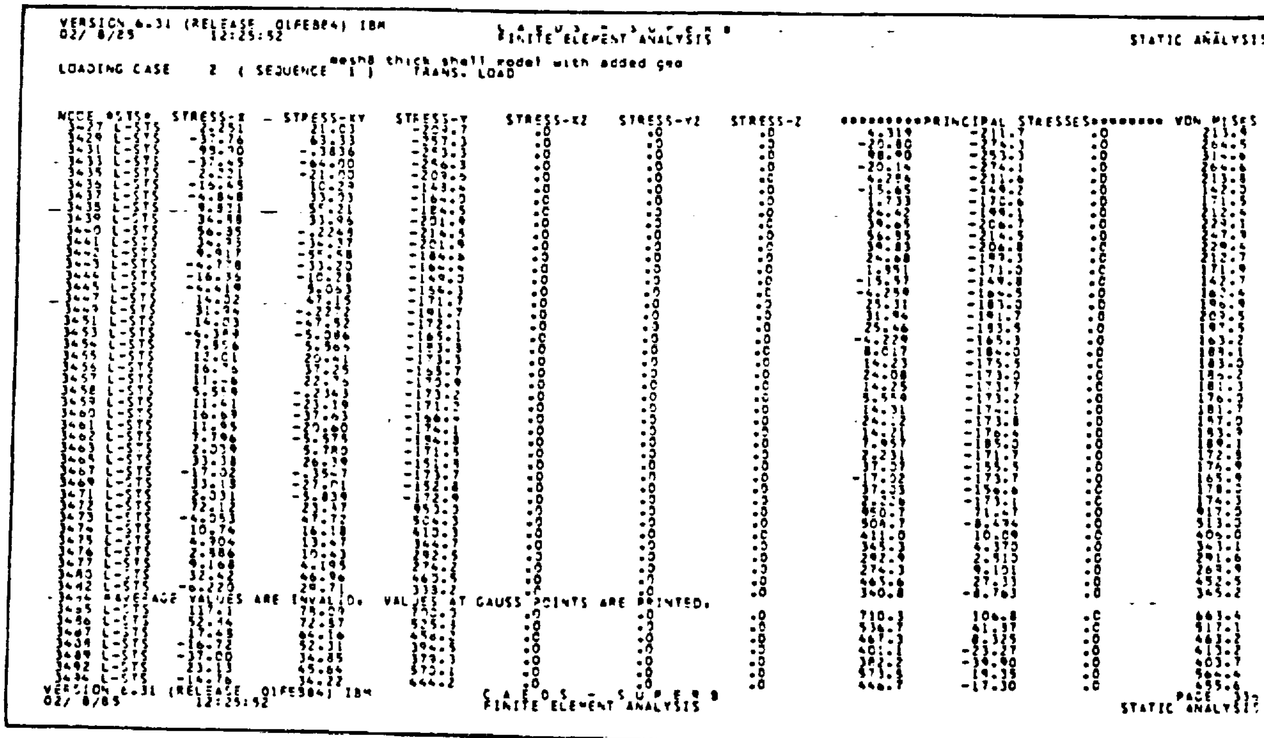
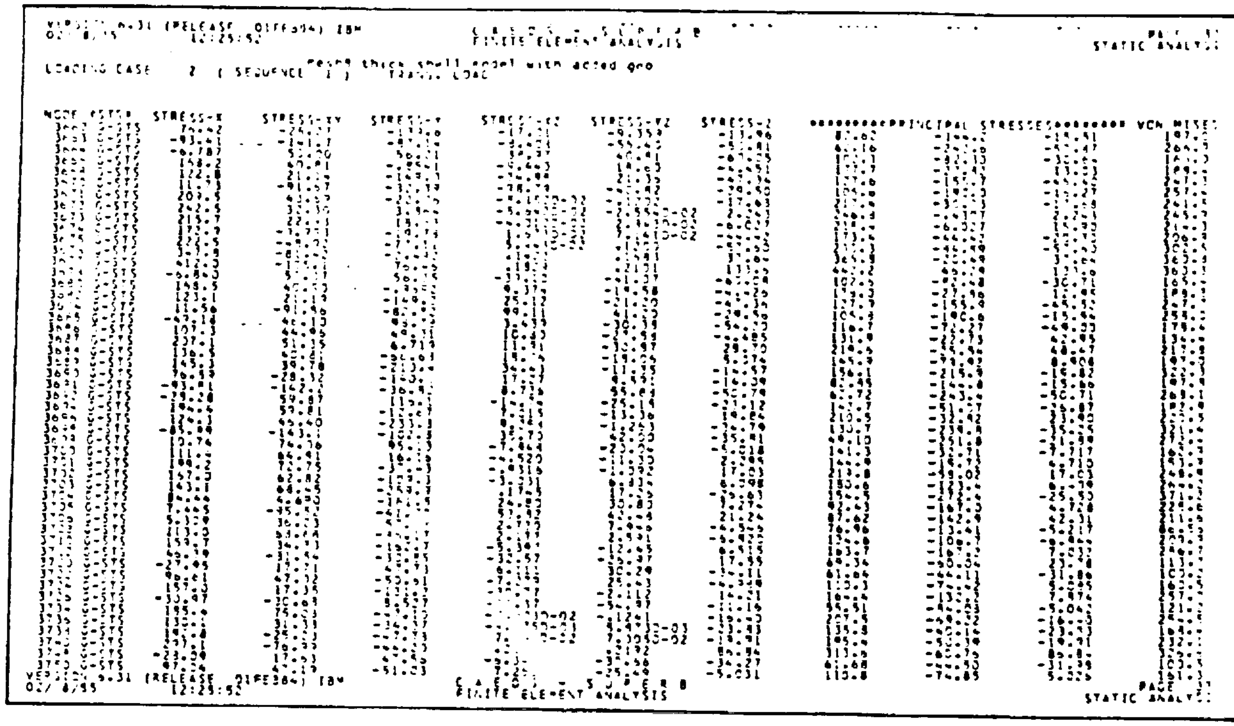


Figure A1.2: Typical thick shell element stress output.



STRESS-X STRESS-XY STRESS-Y STRESS-XZ STRESS-YZ STRESS-Z PRINCIPAL STRESSES VON MISES

Figure A1.3: Typical solid element stress output.

A1.10 THIN SHELL ELEMENT

As seen in figure A1.4, three basic nodal configurations are available with the thin shell: linear (4-node), parabolic (8-node), and cubic (12-node). Various transition elements are also available to aid in modeling. Each node of a thin shell element is assigned six degrees of freedom (3 translations and 3 rotations) and is used to define the element's middle surface. Thus, the element is capable of modeling bending and and membrane actions. Also, the parabolic and cubic elements are capable of modeling deformations due to shearing forces, where as the linear elements are not. In addition to this, each element can be assigned isotropic or orthotropic material properties. In computations the stress normal to the element's middle surface (sigma-z) is assumed zero. Nodal stresses are reported at the top, middle, and bottom surfaces at each node. The top and bottom surfaces of a thin shell element are defined by the element connectivity as shown in figure A1.5.

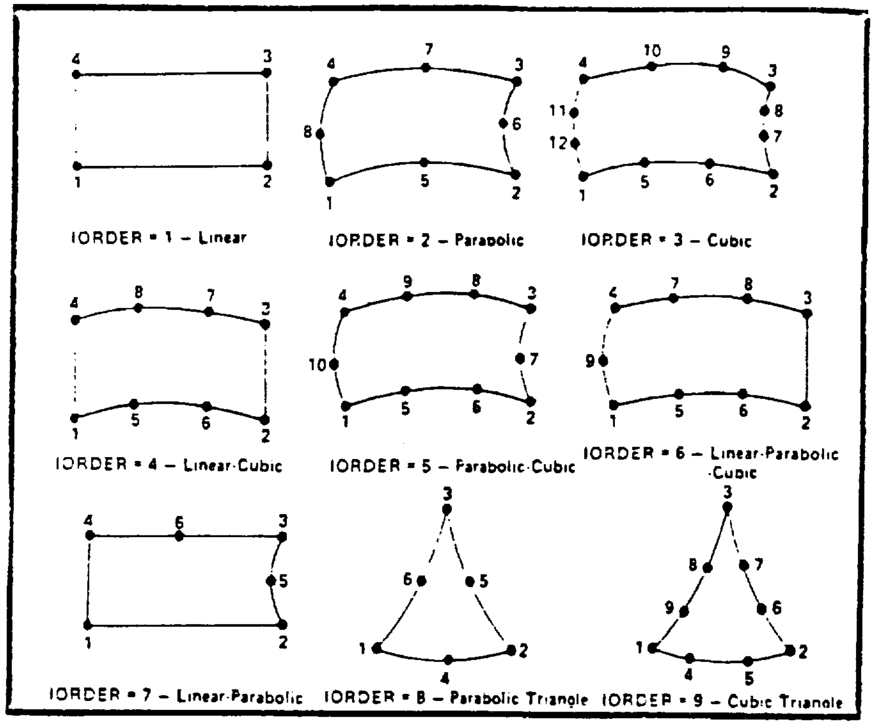


Figure A1.4: Various thin shell elements available.

(Courtesy of SDRC)

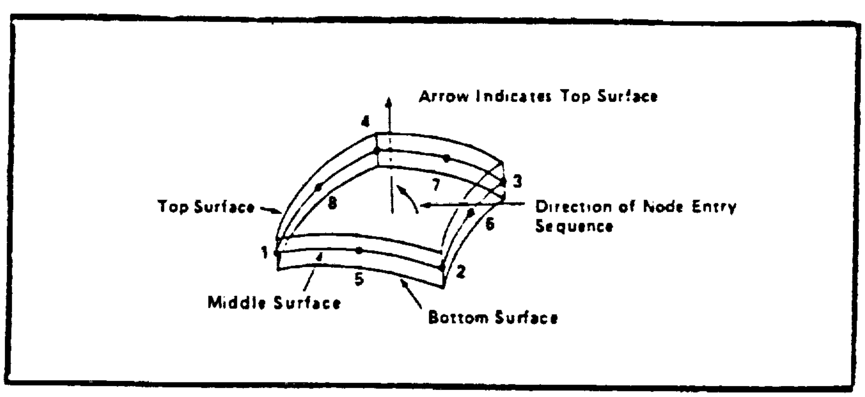


Figure A1.5: Thin shell surface definition.

(Courtesy of SDRC)

The thermal capability of the thin shell element is limited to in-plane steady state temperature distributions. Heat conduction prepindicular to the element's middle surface is assumed zero. Although each surface can have its own convective coefficent and ambient temperature, convective heat transfer through each surface uses the same body temperature. Furthermore, heat transfer through the element's free edges is assumed zero.

Element connectivity determines the local and elemental coordinates systems. In both systems the Z-axis is normal to the shell's surface and in the direction of the top surface. The local X-axis is defined by crossing the global Y-axis and the local Z-axis. If the angle between the local Z-axis and the global Y-axis is less than 18 degrees, the global X-axis is used to determine the local X-axis. Once the local Z and X axes are determined the local Y-axis is found using the right-hand rule. The process in illustrated in figure A1.6.

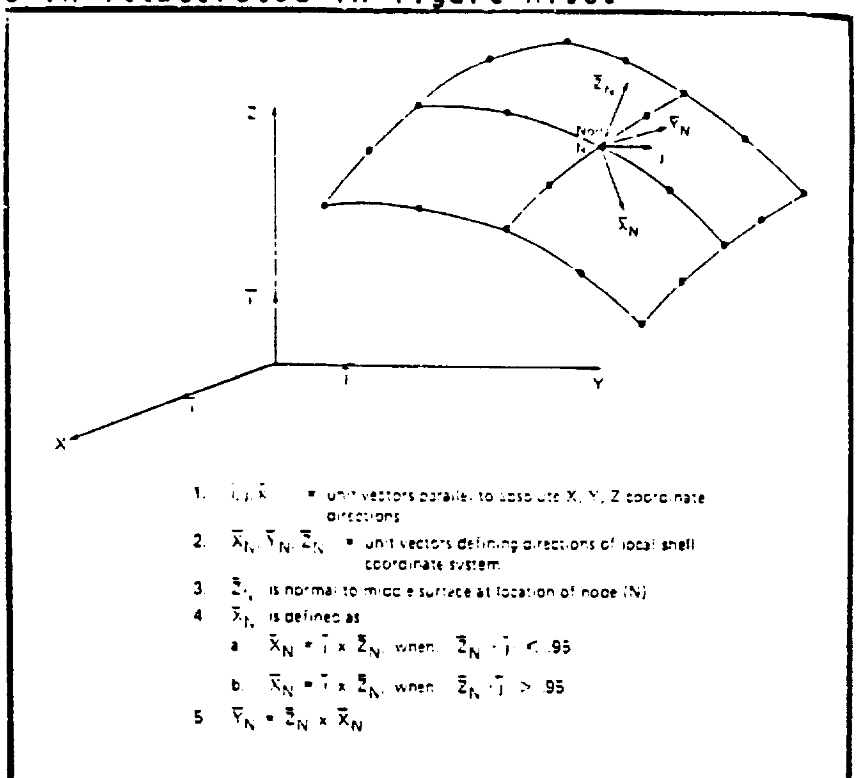


Figure A1.6: Local axis orientation.

(Courtesy of SDRC)

ORIGINAL PAGE IS
OF POOR QUALITY

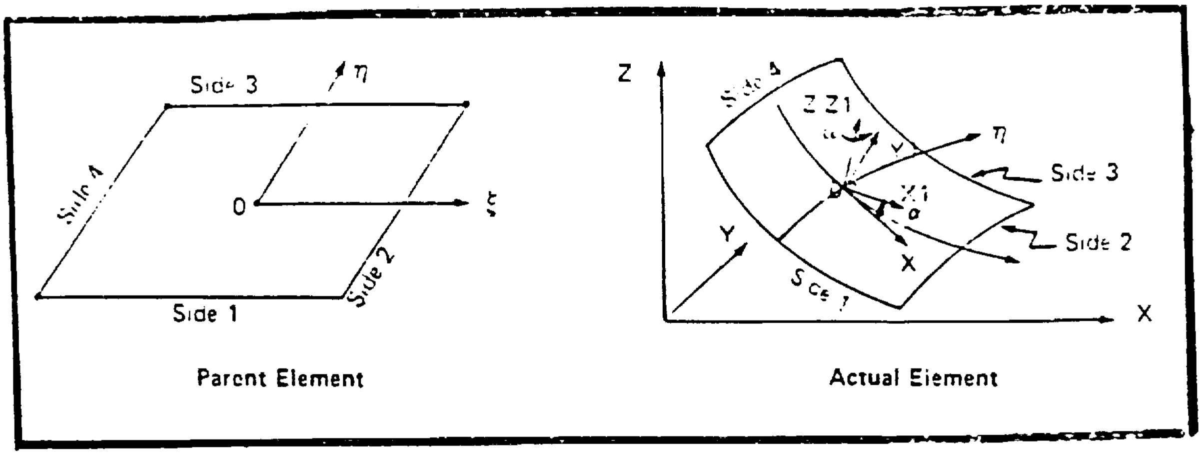


Figure A1.7: Elemental axis orientation.
(Courtesy of SDRC)

Continuing, the elemental X-axis is tangent to the element's middle surface and perpendicular to it's second side as defined by the element's connectivity. The element Y-axis is also tangent to the element's middle surface but is perpindicular to it's third side. The elemental coordinate system is displayed in figure A1.7.

A1.20 THICK SHELL ELEMENT

As seen in figure A1.8, the thick shell element is constructed using brick type geometry. Each node of the element is assigned three degrees of freedom, all translational, and as in the thin shell element the stress normal to the element's middle surface (sigma-z) is assumed zero. In addition, as in the thin shell element, the thick shell elements can be assigned isotropic or orthotropic material properties. The nodal configurations which are available include the linear (8-node), parabolic (16-node), and cubic (24-node), and various transition

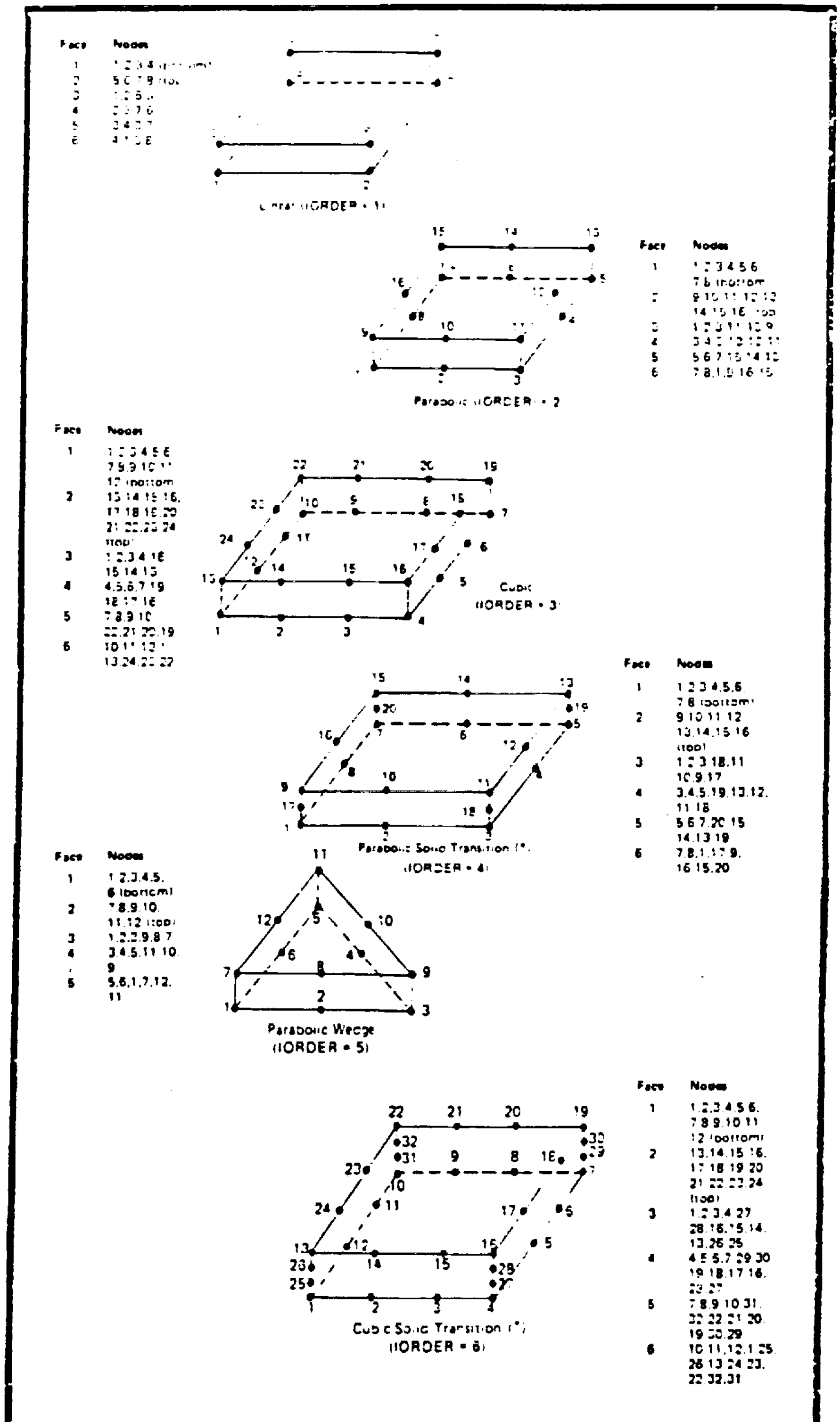


Figure A1.8: Various thick shell elements available.

(Courtesy of SDRC)

ORIGINAL PAGE IS
OF POOR QUALITY

elements. It is capable of modeling membrane, bending and shearing actions. Furthermore, heat transfer can occur through any of its six faces. Thus, the thick shell element is capable of modeling a steady-state temperature distribution through its thickness. As with the thin shell elements, the top and bottom surfaces are defined by the element connectivity as illustrated in figure A1.9. The local and elemental coordinate systems are also defined in a similar manner as the thin shell element. Nodal stresses are given in the local coordinate system (exceptions are given in reference 9).

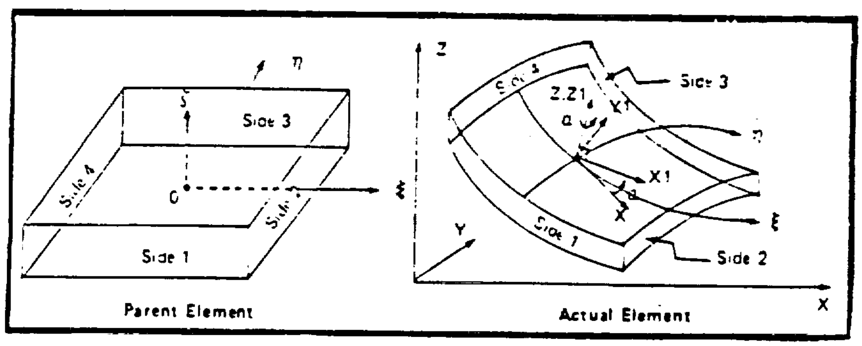


Figure A1.9: Thick shell element surface definition.
(Courtesy of SDRC)

A1.30 SOLID ELEMENT

As with the thick shell element, the solid element, shown in figure A1.10, uses 3-D "brick" geometry to describe itself. Each node is assigned 3 degrees of freedom, all translational, and can use isotropic or orthotropic material properties. The solid element, as determined by the benchmark study, is fully capable of modeling uniaxial loading. However, a number of elements are required for adequate modeling of bending actions. In addition, the solid element can model thermal distributions extremely well, since heat transfer can occur through each face of the element.

A1.40 BEAM ELEMENT

The final element type utilized was the beam element. Although each beam element possesses the ability of being linear, tapered, or curved, only linear beam elements were used in the modeling of the housing. Each node of the beam element supports 6 degrees of freedom (3 translation and 3 rotation). Thus, it is compatible with the thin shell element. By specifing no cross sectional area and large moments of inertia, a beam, extremely rigid to bending forces but able to offer no resistance to uniaxial loading can be obtained. Thus, the beam element was used primarily in stiffening the areas around the secondary ribs and mid-plane stiffeners. In conclusion, even though thermal capabilities are available, utilization of them was not necessary.

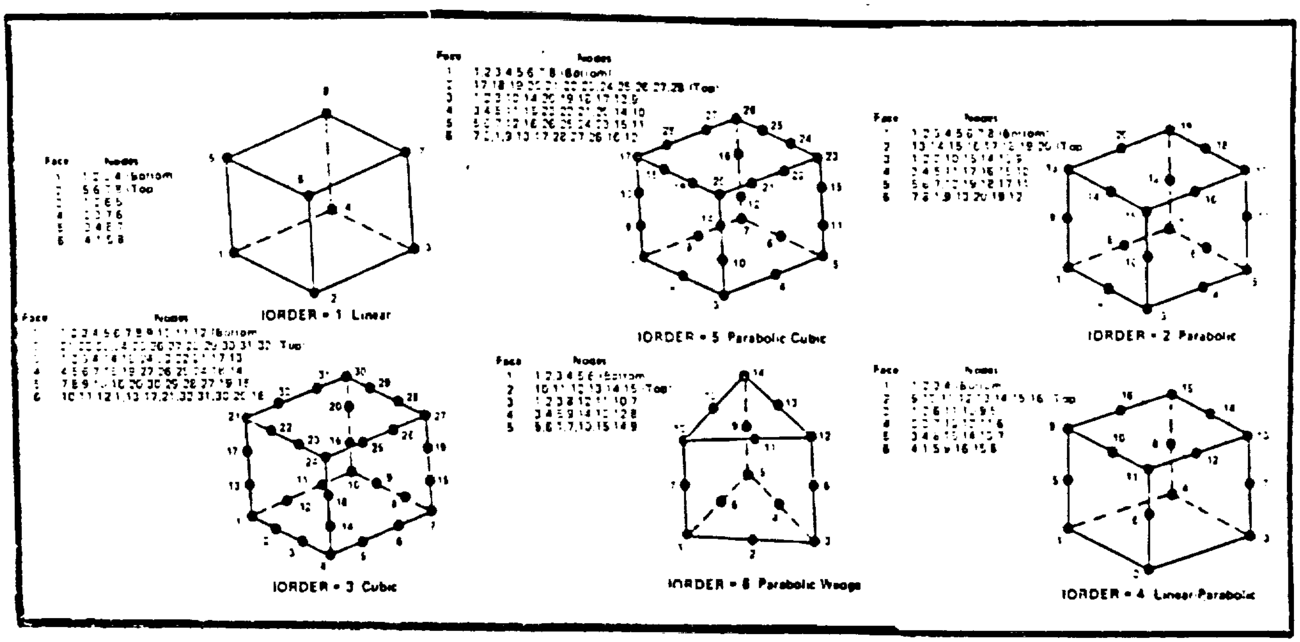


Figure A1.10: Various solid elements available.

(Courtesy of SDRC)

APPENDIX 2
ELEMENT BENCHMARK TESTS

In order to determine the capabilities of the element types available to model the housing, a benchmark study was completed for each type of element. Tests were devised to exercise the element's under loading conditions similar to those expected in the housing. Since the housing would be subjected to bending, membrane, and shearing actions, the shell elements were subjected to these types of loadings and their capabilities were determined. Various mesh sizes and aspect ratios were used and exact solutions were found to determine convergence and accuracy.

A2.00 SHELL ELEMENT TESTS

Two tests were devised to exercise the shell elements available. The flat plate was constructed as a thick, short plate to introduce a shearing action of substantial magnitude. Both tests were constrained on two opposing sides while the remaining two sides were left free.

An exact solution was found in reference (10) (see appendix 3) for a flat plate simply-supported on two opposite sides and free on the remaining two sides. Due to the way the thick shell elements are constructed, though, only the thin shell elements were tested against the exact solution. The simply-supported sides were then clamped and the thick shell elements were tested against the thin shell elements. This method insured that the same boundary conditions were applied to each element type. Furthermore, the exact solution did not account for deformation due to shear so a third test was devised to decrease the effect of the shearing action. This was done by decreasing the thickness of the plate. The tests are illustrated in figures A2.1, A2.2 and the results are given in tables 6 and 7.

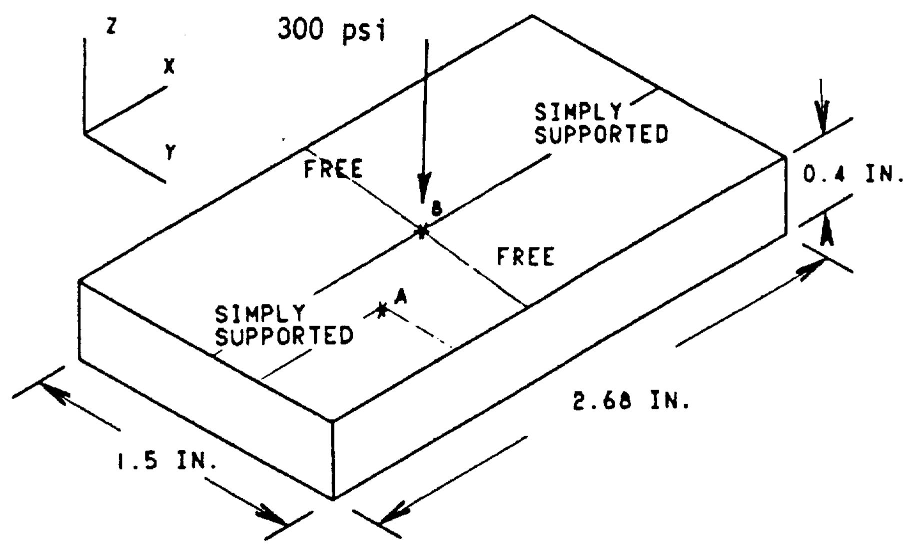


Figure A2.1: Illustration of simply-supported plate test.

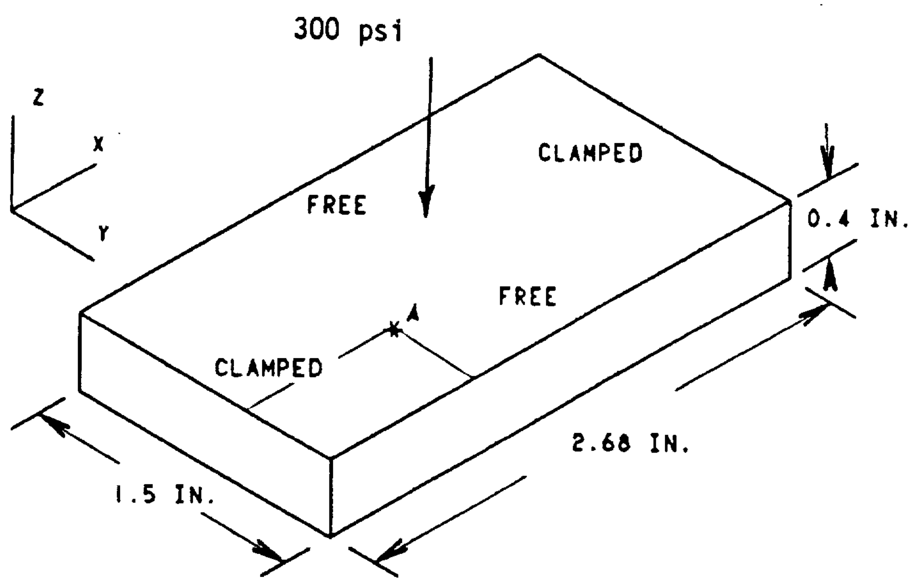


Figure A2.2: Illustration of clamped plate test.

ITEM	LOCATION	TEST	STRESS-X	STRESS-Y
(3X3) 4-NODE	A	SIMPLY-SUP.	8868	1422
(3X3) 8-NODE	A	SIMPLY-SUP.	9708	806.7
(6X6) 4 NODE	A	SIMPLY-SUP.	8904	1094
(6X6) 8-NODE	A	SIMPLY-SUP.	9121	668.8
(6X3) 8-NODE	A	SIMPLY-SUP.	9137	689.1
(12X6) 8-NODE	A	SIMPLY-SUP.	8983	655.8
(12X12) 8-NODE	A	SIMPLY-SUP.	8986	642.6
EXACT	A	SIMPLY-SUP	8949	970.7
(4X4) 8-NODE	B	SIMPLY-SUP	10400	917
(12X12) 8-NODE	B	SIMPLY-SUP	10070	831.7
(3X3) 8-NODE	A	CLAMPED	2926	507.6
(6X6) 8-NODE	A	CLAMPED	2368	343
(3X3) 8-NODE*	A	CLAMPED	256	30
(6X6) 8-NODE*	A	CLAMPED	1242	167
(12X12) 8-NODE*	A	CLAMPED	1870	248
(3X3) 16-NODE*	A	CLAMPED	2926	507
(6X6) 16-NODE*	A	CLAMPED	2368	343
(12X12) 16-NODE*	A	CLAMPED	2226	295

STRESSES GIVEN IN PSI

* THICK SHELL ELEMENT

Table 6: Results from plate test.

	EXACT		(12X12) 8-NODE	
THICKNES	STRESS-X	STRESS-Y	STRESS-X	STRESS-Y
0.4 IN	8949	970	8986	642
0.2 IN	35800	3882	35820	3193
0.1 IN	143200	15530	143100	14110

STRESSES GIVEN IN PSI

Table 7: Results from plate test (thickness reduced).

A2.10 SOLID ELEMENT TESTS

SUPERB's solid elements were also subjected to simple tests. The tests were to determine the element's ability to model both uniaxial and bending forces. In order to accomplish this a beam, shown in figure A2.3, and a column, shown in figure A2.4, were subjected to specific loadings. It was believed the beam test would induce the necessary bending action where as the column test would test the element's ability to model uniaxial loading. As expected, the solid elements modeled the beam test very poorly. Only by increasing the number of elements through the depth of the beam could the actual stress field be predicted. However, the solid elements did model the uniaxial loading quite well but this was not suprising. The results of both tests are given in tables 8 and 9.

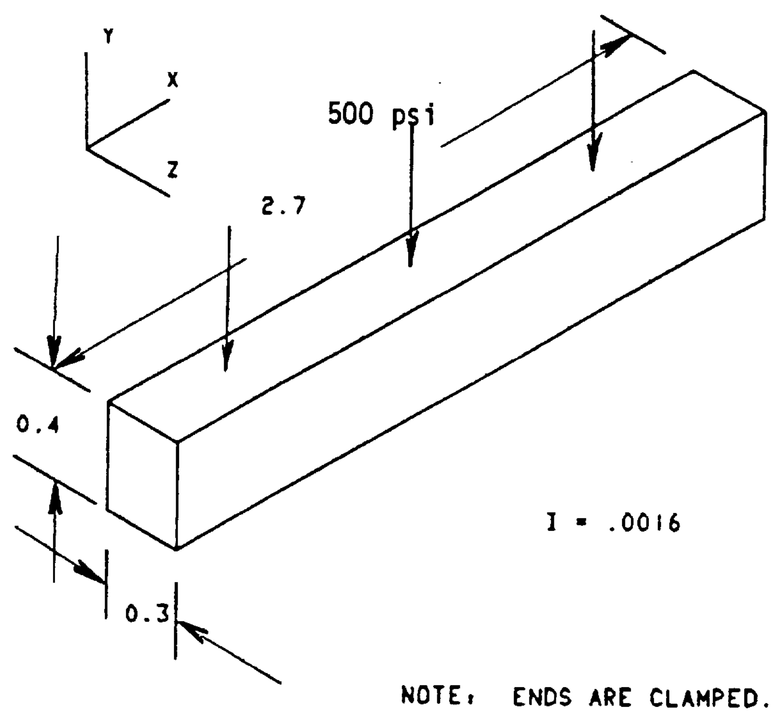


Figure A2.3: Illustration of beam test.

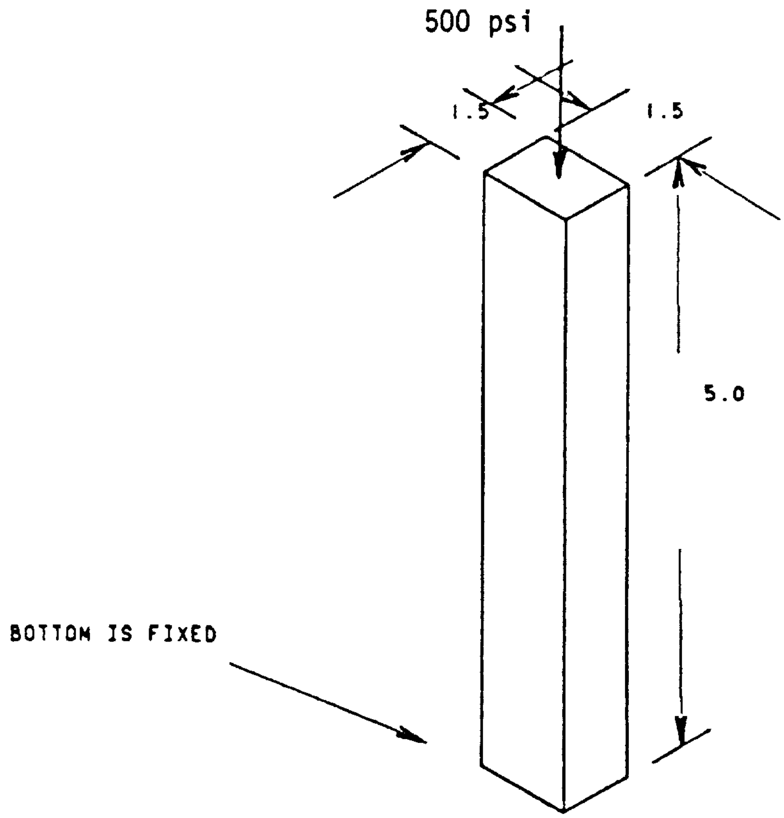


Figure A2.4: Illustration of column test.

FIBER STRESS IN BEAM TEST

ITEM	BEAM RESULTS	EXACT
(4X1X1) 8-NODE	-1851	-5695
(4X2X1) 8-NODE	-1774	-5695
(4X1X1) 20-NODE	-5800	-5695
(4X2X1) 20-NODE	-5900	-5695

STRESSES GIVEN PSI

Table 8: Results from beam test.

ITEM	COLUMN TEST	EXACT
(4X1X1) 8-NODE	-500 PSI	-500 PSI
(4X1X1) 20-NODE	-500 PSI	-500 PSI

STRESSES GIVEN IN PSI

Table 9: Results from column test.

APPENDIX 3
EXACT SOLUTION FOR
BENCHMARK TESTS

In reference (10), a series solution for a flat plate simply-supported on two opposing sides and free on the remaining two was found. By applying this solution to the benchmark plate geometry, accuarcy of the plate elements were checked. It must be noted, though, that this solution does not include shear deformation effects.

So,

$$w = \left[\frac{4pa^4}{\pi^5 D}\right] \sum_{m=1,2,3}^{\infty} \left[\frac{1}{m^5}\right] \sin\left[\frac{m\pi x}{a}\right] + \left[\frac{pa^4}{D}\right] \sum_{m=1,2,3}^{\infty} \left[A_m \cosh\left[\frac{m\pi y}{a}\right] + B_m\left[\frac{m\pi y}{a}\right] \sinh\left[\frac{m\pi y}{a}\right]\right] \sin\left[\frac{m\pi x}{a}\right]$$

Where,
$$D = \left[\frac{Et^3}{12(1-\nu^2)}\right]$$

$$A_m = \left[\frac{4}{m^5\pi^5}\right]\left[\frac{\nu(1+\nu)\sinh\alpha_m - \nu(1-\nu)\alpha_m\cosh\alpha_m}{(3+\nu)(1-\nu)\sinh\alpha_m\cos\alpha_m - (1-\nu)^2\alpha_m}\right]$$

$$B_m = \left[\frac{4}{m^5\pi^5}\right]\left[\frac{\nu(1-\nu)\sinh\alpha_m}{(3+\nu)(1-\nu)\sinh\alpha_m\cos\alpha_m - (1-\nu)^2\alpha_m}\right], \quad \alpha_m = \left[\frac{m\pi b}{2a}\right]$$

Where w is the displacement normal to the plate's surface. Then,

$$\sigma_x = -\left[\frac{Ez}{1-\nu^2}\right]\left[\frac{\partial^2 w}{\partial x^2} + \nu\frac{\partial^2 w}{\partial y^2}\right]$$

$$\sigma_y = -\left[\frac{Ez}{1-\nu^2}\right]\left[\frac{\partial^2 w}{\partial y^2} + \nu\frac{\partial^2 w}{\partial x^2}\right]$$

$$\tau_{xy} = -\left[\frac{Ez}{1+\nu}\right]\left[\frac{\partial^2 w}{\partial x \partial y}\right]$$

Thus, by implementing the necessary parameters,

$a=2.68$ in. $\quad b=1.5$ in.

$E=30\mathrm{x}10$ psi $\quad \nu=0.3$

$p=300$ psi

the solution was obtained.

APPENDIX 4
NUMERICAL ROUTINES USED
IN
FEM CONSTRUCTION

Two routinesw were required to aid in construction of the FEM. The first was necessary to place points on the trochoidal bore at a specific angle relative to the outer shell's center of radius. This requirement is shown in Figure A4.1.

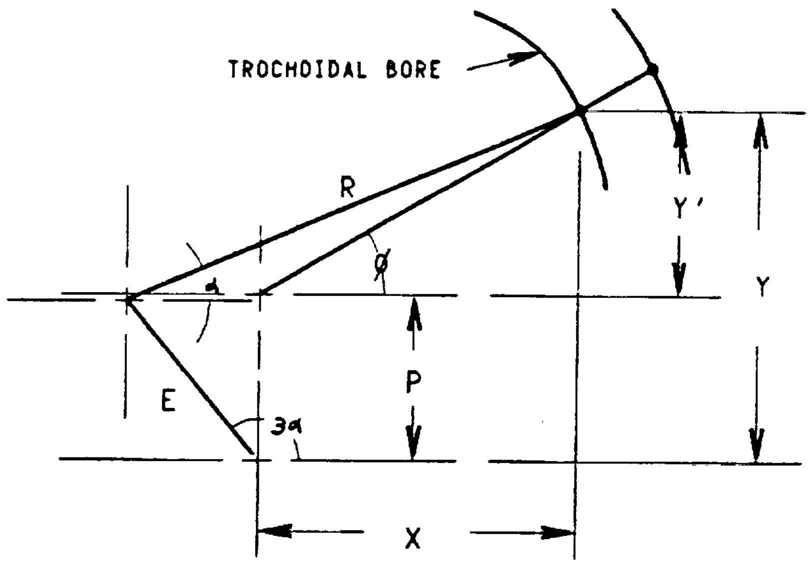


Figure A4.1: Illustration of the construction point placement.

In order to accomplish this a iteration routine was devised to solve equation 4.1:

$$\text{Eq. 4.1:} \qquad \alpha = \sin^{-1}\left[\frac{(e\cos 3\alpha + R\cos\alpha)\tan\phi + P - e\sin 3\alpha}{R}\right]$$

A second routine was needed to find the X coordinate of the trochoidal bore given any Y coordinate. Thus, the second routine solved Equation 4.2:

$$\text{Eq. 4.2:} \qquad \alpha = \sin^{-1}\left[\left[\frac{1}{R}\right]\left[y - e\sin(3\alpha)\right]\right]$$

REFERENCES

1) Jones, C., "Advanced Development of Rotary Stratified Charge 750 and 1500 HP Military Multi-fuel Engines at Curtis-Wright," SAE Paper 840460, presented at the International Congress and Exposition, Detroit, Feb. 1984.

2) Matsuura, K., et. al., "The Behavior of a Rotary Engine Apex Seal Against the Trochoidal Surface," Bull. of JSME, Vol. 21, No. 161, Nov. 1978.

3) Eberle, M. K., and Klomp, E. D., "An Evaluation of the Potential Performance Gain from Leakage Reduction in Rotary Engines," SAE Paper 730117, presented at the International Automotive Engineering Congress, Detroit, Jan. 1973.

4) Behling, R. D., and Weise, E., "Lubricant and Fuel Requirements and General Performance Data of Wankel Rotary Piston Engines," SAE Paper 730048, presented at the International Automotive Engineering Congress, Detroit, Jan. 1973.

5) Rodgers, T. W., et. al., "Lubricant Studies in Rotary Combustion Engines, "SAE Paper 720467, presented at the National Automotive Engineering Meeting, Detroit, May 1972.

6) Prasse, H. F., McCormac, H. E., and Anderson, R. D., "A Critical Analysis of the Rotary Engine Sealing Problem," SAE Paper 730118, presented at the National Autcmobile Engineering Meeting, Detroit, May 1972.

7) Matsuura, K., et. al., "The Relative Behavior of a Rotary Engine Apex Seal to the Walls of a Slot," Bull. of the JSME, Vol. 19, No. 137, Nov. 1976.

8) Ansdale, R. F., <u>The Wankel RC Engine</u>, A. S. Barnes & Co., 1969.

9) "SUPERB Users Manual," Structural Dynamics Reasearch Corporation, 1983.

10) Timoshenko, S., and Woinowsky-Krieger, S., <u>Theory of Plates and Shells</u>, Mcgraw-Hill Book Company, 1968.

11) "SUPERTAB Users Manual," Structural Dynamics Reasearch Corporation, 1983.

12) Jones, C., "Development of the Rotary Engine," SME Technical Paper MM73-650, presented at the Rotary Engine Symposium, Detroit, Nov. 1973.

13) Meng, P. R., Hady, W. F., Barrows, R. F., "An Overview of the NASA Rotary Engine Research Program," SAE Paper 841018, presented at the West Coast International Meeting & Exposition, San Diego, Aug. 1984.

14) Norbye, J. P., <u>The Wankel Engine Design Development Applications</u>, Chilton Book Company, 1971.

15) Knoll, J., "A Dynamic Analysis of Rotary Combustion Engine Seals," a thesis presented to Michigan Technological University, Nov. 1983.

16) Cook, R. D., <u>Concepts and Applications of Finite Element Analysis</u>, John Wiley & Sons, 1981.

17) Vinson, J. R., <u>Structural Mechanics: The Behavior of Plates and Shells</u>, John Wiley & Sons, 1974.

18) Ugural, A. C., and Fenster, S. K., <u>Advanced Strength and Applied Elasticity</u>, Elsevier, 1982.

Appendix

B

DEVELOPMENT OF A PREPROCESSOR THAT GENERATES FINITE ELEMENT MODELS OF ROTARY COMBUSTION ENGINE CENTER HOUSINGS

by

WILLIAM M. LYCHUK

A THESIS

submitted in partial fulfillment of the requirements

for the degree of

MASTER OF SCIENCE IN MECHANICAL ENGINEERING

MICHIGAN TECHNOLOGICAL UNIVERSITY

1985

This thesis, " Development of a Preprocessor that Generates Finite Element Models of Rotary Combustion Engine Center Housings, " is hereby approved in partial fulfillment of the requirements for the degree of MASTER OF SCIENCE IN MECHANICAL ENGINEERING.

DEPARTMENT: Mechanical Engineering - Engineering Mechanics

Carl Vilmann

Thesis Advisor

Harold W Lord

Head of Department

1-28-86

Date

Abstract

This thesis documents the development of a specialized preprocessor that will generate a finite element model of different rotary combustion engine center housing geometries. The specialized preprocessor has been written to be used in conjunction with General Electric's Computer Aided Engineering software, specifically, IDEAS. The only user-supplied inputs required by the specialized preprocessor are easily measured parameters that describe the center housing geometry.

When executed, the FORTRAN coded specialized preprocessor creates two files - a universal file and a program file. The universal file contains data in universal format that describes the housing geometry. Universal format is a standard that has been defined by General Electric's Computer Aided Engineering. The program file contains commands that are understood by the programmability allowed within IDEAS. The commands guide the algorithms in IDEAS through the generation of the meshes, nodes, and elements.

Acknowledgments

The author would like to express his thanks to Dr. Carl Vilmann and Dr. Chris Passerello for their guidance, encouragement and friendship throughout this project. Both have helped me to maintain my perspective and to understand what was to be accomplished.

Thanks is expressed to Dr. Thomas Grimm and Dr. William Bulleit for reviewing this paper and serving on the examination committee.

Love and affection to my fiance Katherine Drakos for her unending love and patience.

Most importantly, the author wishes to express his deepest love to his parents, to whom, he owes everything that he has achieved.

Table of Contents

List of Illustrations

Glossary of Abbreviations

Abbreviation	*Description*
AAD	Advanced Adiabatic Diesel
BMEP	Brake Mean Effect Pressure
FEM	Finite Element Model
GE/CAE	General Electric / Computer Aided Engineering
MPG	Miles Per Gallon
PF	Program File
RCE	Rotary Combustion Engine
RPE	Reciprocating Piston Engine
SP	Specialized Preprocessor
UF	Universal File

CHAPTER 1: Introduction

The Rotary Combustion Engine (RCE) was first introduced in the early 1950's by Felix Wankel, a German inventor (Figure 1).

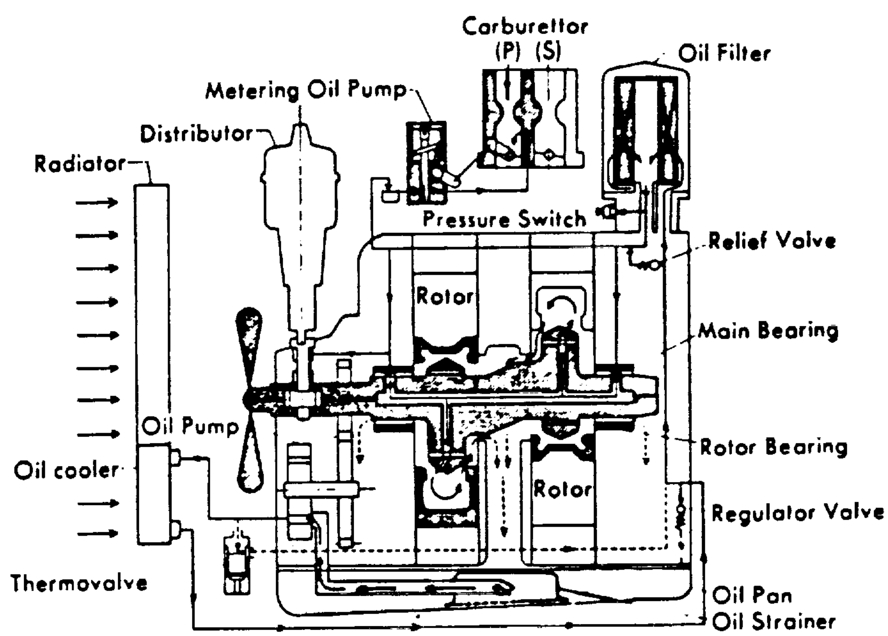


Figure 1. Rotary Combustion Engine

The RCE is an internal combustion engine which consists of three major components - rotor, center housing and end plates. The end plates are bolted to each side of the center housing. The triangularly shaped rotor is located in the trochoidally shaped chamber of the center housing (Figure 2).

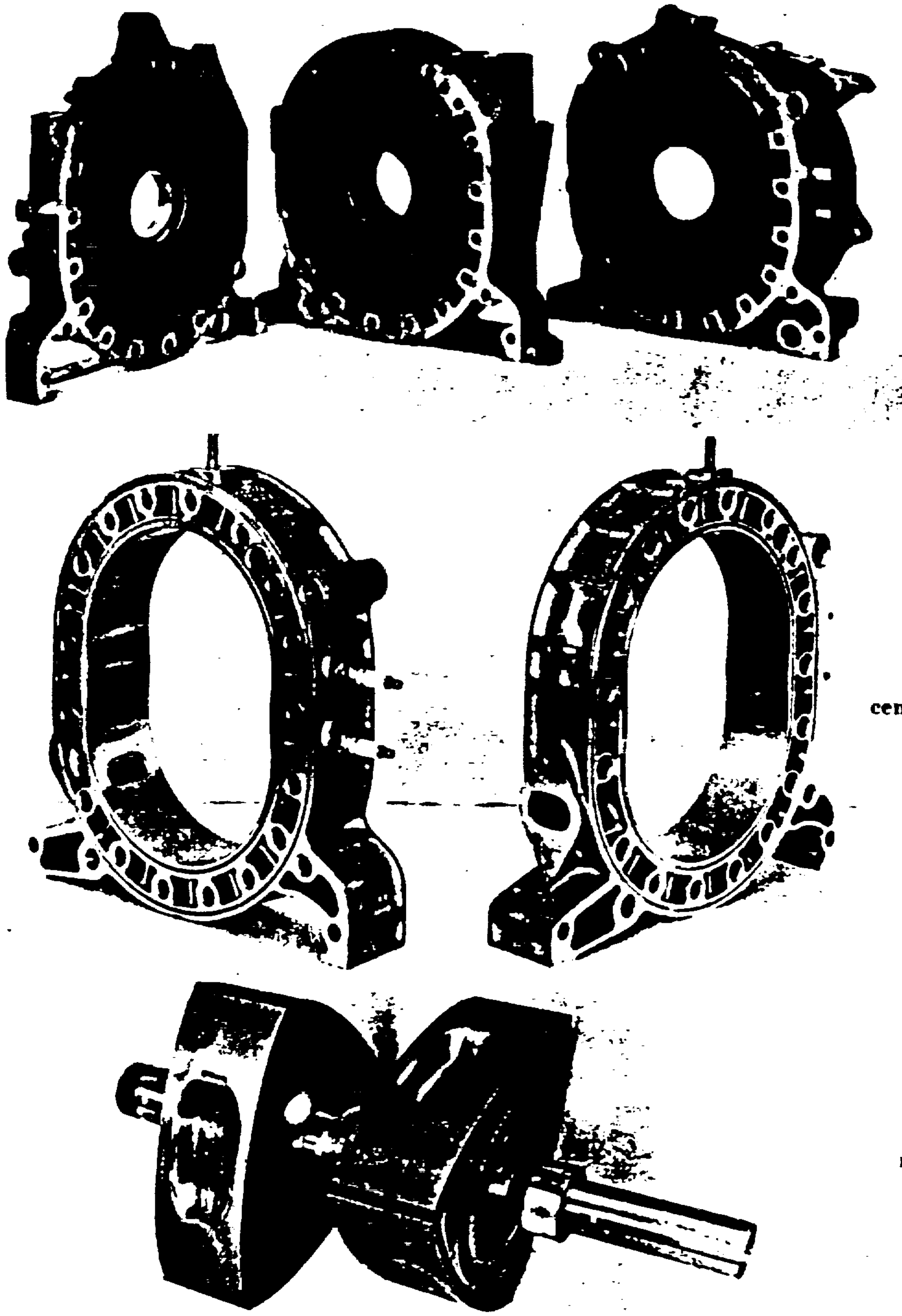


Figure 2. Major Components of the Rotary Combustion Engine

The rotation of the rotor causes different sized chambers to be created between each face of the rotor and the trochoidal surface. It is in these chambers that the different phases of the Otto cycle - intake, compression, combustion and exhaust - are executed (Figure 3). One Otto cycle is executed by each face of the rotor every revolution (i.e. three complete Otto cycle are executed per rotor revolution).

1.1 Rotary Combustion Engine Center Housing Geometry

A prominent feature of the center housing geometry is the trochoidal shape of the inner surface. The generation of this shape is accomplished by first constructing an epitrochoid. By definition, an epitrochoid is the locus of points created by a point on the radius of a circle which rolls without slip around a base circle. The familiar two-lobed epitrochoid is obtained when the base circle is equal to twice the rolling circle (Figure 4). The dimensions needed to describe the epitrochoid are the base circle radius, the rolling circle radius (one half of the base circle radius) and the distance of the generating point from the rolling circle's center, the eccentricity.

The epitrochoidally shaped inner bore of the center housing is constructed so that the rotor, when inserted, will create line contact at all three apexes. To enhance the sealing capacity between the rotor apexes and epitrochoidal surface, seals are placed at the apex of each rotor (Figure 5). To account for the extra space required by the seals, the epitrochoidal surface has to be expanded. The expanded curve is called a trochoid. During operation, the motion of the center of the radius of the apex seal traces an epitrochoidal curve and the motion of the apex seal tip traces a trochoidal curve.

The trochoidal shape is derived from expanding the epitrochoid a perpendicular distance equal to the radius of the apex seal. A family of curves can be created by specifying different magnitudes of the perpendicular distance (Figure 6).

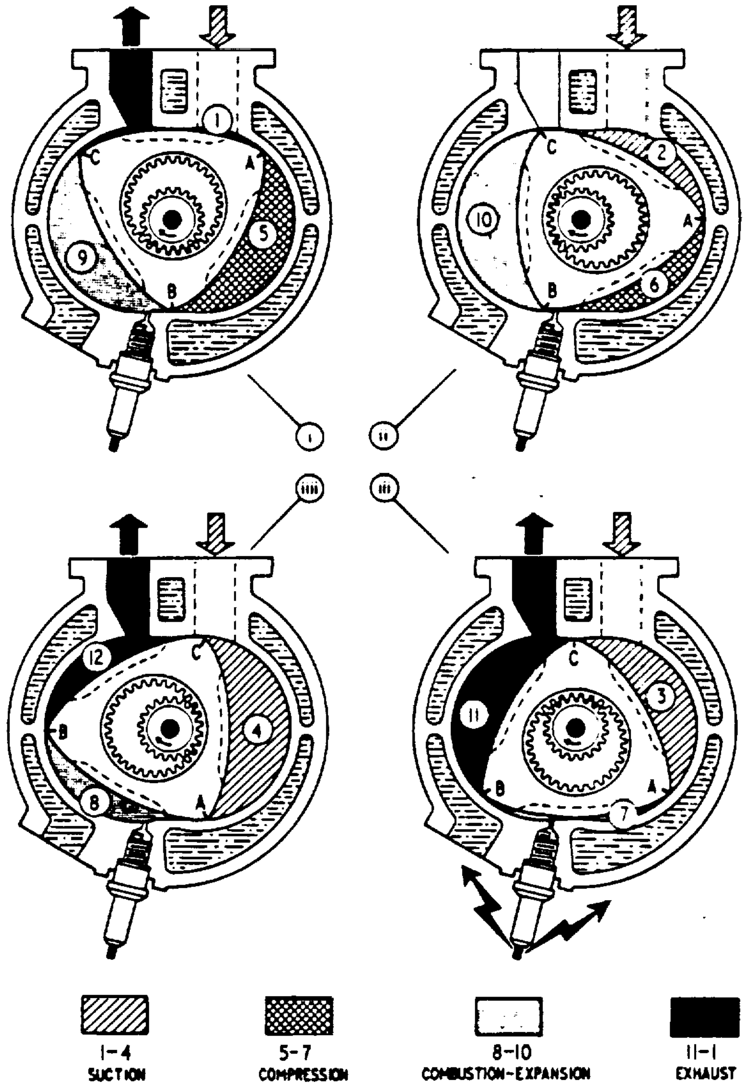


Fig. 5.3. 1–4 induction
5–7 compression
8–10 expansion
11–1 exhaust

Figure 3. The execution of the Otto cycle in the RCE

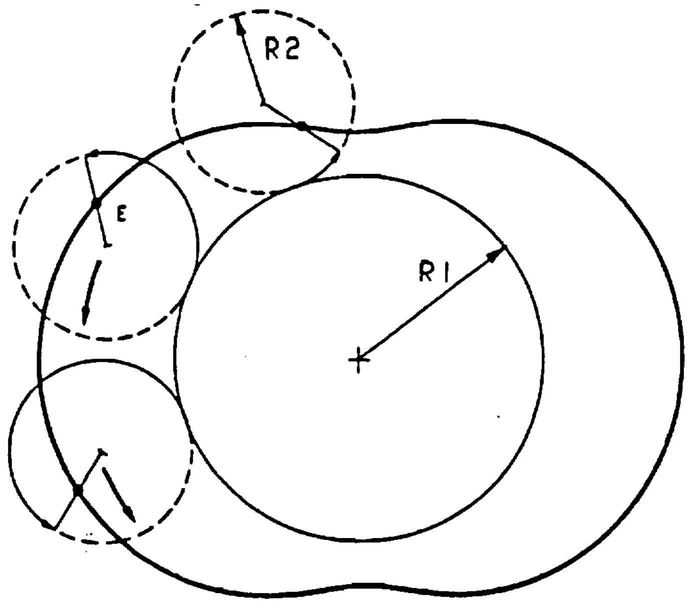


Figure 4. Generation of the Epitrochoid

Ansdale (1) gives the rectangular coordinates of any point on the trochoidal surface as:

$$x = E \times \cos(3\alpha) + R \times \cos(\alpha) + A \times \cos(\alpha + \theta) \quad (1.1a)$$

$$y = E \times \sin(3\alpha) + R \times \sin(\alpha) + A \times \sin(\alpha + \theta) \quad (1.1b)$$

where E is the eccentricity, R is the radius of the generating circle, A is the perpendicular distance between the epitrochoid and trochoid, and theta is the angle of obliquity. Theta is defined by:

$$\theta = \cos^{-1}\left[\frac{(R + 3 \times \cos(2 \times \alpha))}{\sqrt{9 \times E^2 + R^2 + 6 \times E \times R \times \cos(2 \times \alpha)}}\right] \quad (1.1c)$$

Theta is a measure of the angle, relative to the normal of the trochoidal curve, that the apex seal rotates through during operation. Theta is a minimum at the major and minor axes and reaches a maximum value midway between the two axes.

1.2 Rotary Combustion Engine Development

After Felix Wankel persuaded an obscure German motorcycle and small car manufacturer, NSU, to help develop the RCE, many other companies, including General Motors, Porshe, Rolls Royce, Curtis-Wright and Toyo Kogyo, have attempted to make the

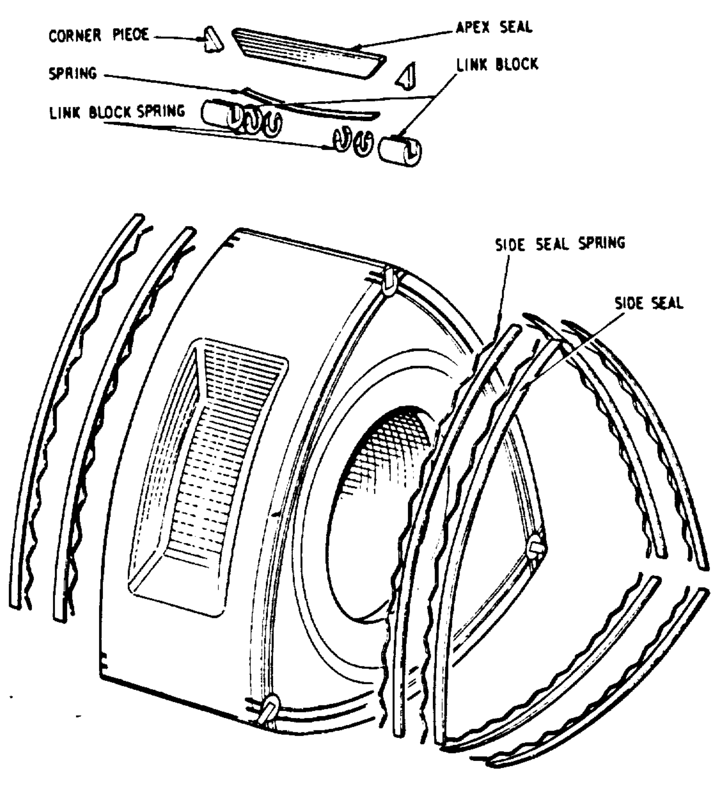


Figure 5. Rotor with Seals

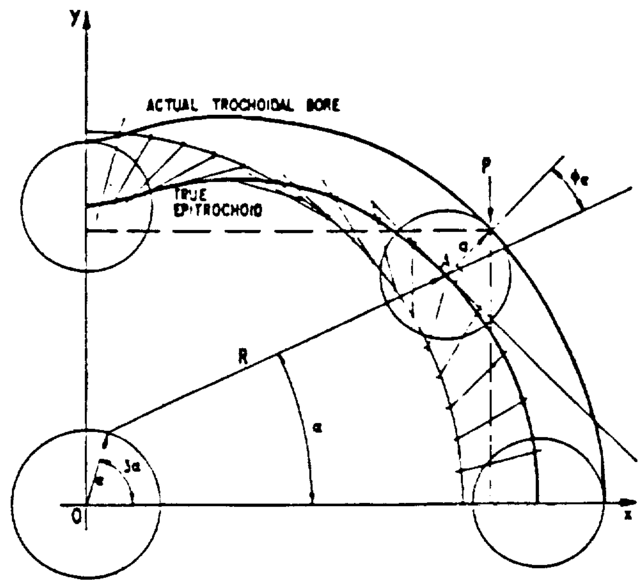


Figure 6. Generation of the Trochoid

RCE's actual performance characteristics approach its theoretical possibilities. To date in the automotive industry, Toyo Kogyo's Mazda RX7 is the only production automobile still using the RCE.

During its initial development, the RCE was hailed as the engine of the future that would replace the reciprocating piston engine (RPE). In many areas, the RCE has better performance characteristics than its RPE counterpart. For RCE and RPE engines of comparable power output, the RCE has less than half the weight, is almost half the size, produces less noise and vibration, and uses less than half the moving parts (2). But the RCE is plagued with a 12% higher specific fuel consumption and higher exhaust emissions (3). These are two of the major problems that have prevented the RCE from living up to its early expectations.

1.2.1 Major Sources of Problems in the Development of the RCE

The major sources of the problems preventing the widespread acceptance of the RCE include:

1. When combustion pressures are high, imperfect sealing of the apex seal on the trochoidal surface will permit leakage into the leading and/or trailing chamber. The leakage, because it comes from the hydrocarbon-rich end of the combustion chamber, leads to higher hydrocarbon emissions (4).

2. Residuals are the fraction of the gasses remaining in the combustion chamber at the end of the exhaust cycle. The residuals mix with the incoming air fuel mixture diluting it largely with inert gasses. The residuals cause cycle-to-cycle variations in the combustion mixture (5).

3. At high operating speeds, the apex seal is subjected to a high contact force from the trochoidal surface. The friction forces, resulting from the contact force, cause excessive wearing of the apex seal. As the apex seals wear, more leakage is allowed into the leading and/or trailing chambers (6).

4. About forty percent of the combustion chamber surface is on the rotor. The balance is on the trochoid. A layer of unburned hydrocarbons is formed on these surfaces as the flame front is quenched by these relatively cool surfaces. These unburned hydrocarbons contribute to the high exhaust emissions (7).

1.2.2 Modifications of the RCE

In an effort to enhance the operating conditions of the RCE many modifications have been proposed. Some of the significant modifications include:

1. In order to improve the apex seal conformability to the trochoidal surface, crowning of the apex seal in the radial direction was applied (8). Although the crowning is only ten to thirty microns, its effect on the brake mean effect pressure (BMEP) is an increase of two to eight percent (Figure 7).

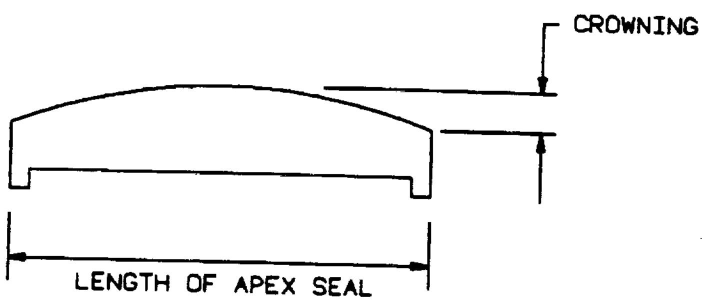


Figure 7. Crowning of the Apex Seal: The crowning has been amplified for illustrative purposes.

2. The shape of the combustion chamber recess and the spark plug location influences the amount of residuals (9). By increasing the distance between the two spark plugs and altering the combustion chamber recess, the BMEP is increased and the brake specific fuel consumption is reduced (Figure 8).

3. The shape and opening and closing timings of the port, especially the intake port, can affect the volumetric efficiency by as much as twenty percent (10).

4. Stratified charging of the RCE has increased the miles per gallon (MPG) rating by as much as fifty percent (11). (Figure 9)

Currently, there is an effort to develop advanced adiabatic diesel (AAD) engines. An AAD engine uses ceramic coatings on critical engine components to allow for higher operating temperatures. These higher operating temperatures significantly increase the

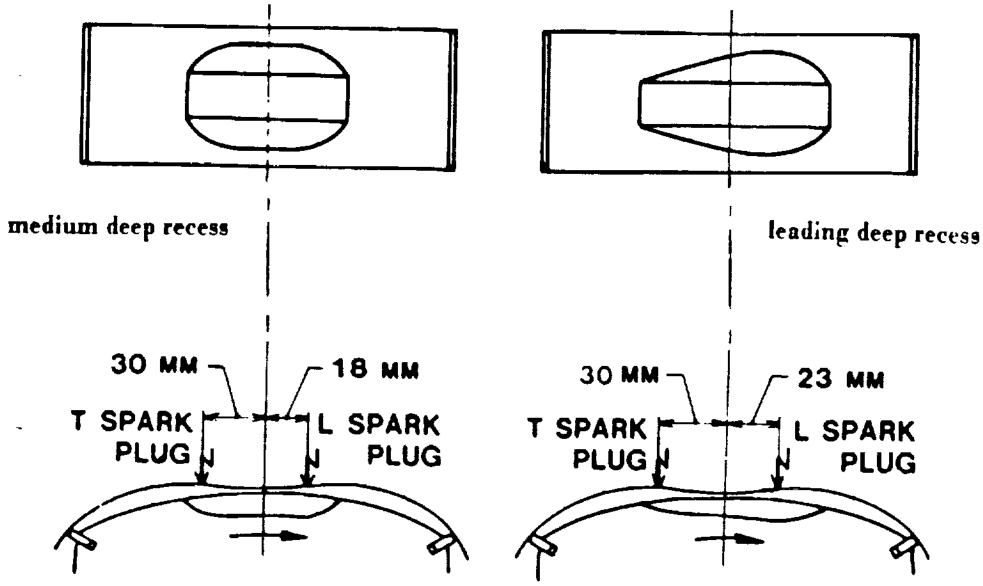


Figure 8. Spark Plug Spacing and Shapes of the Combustion Chamber

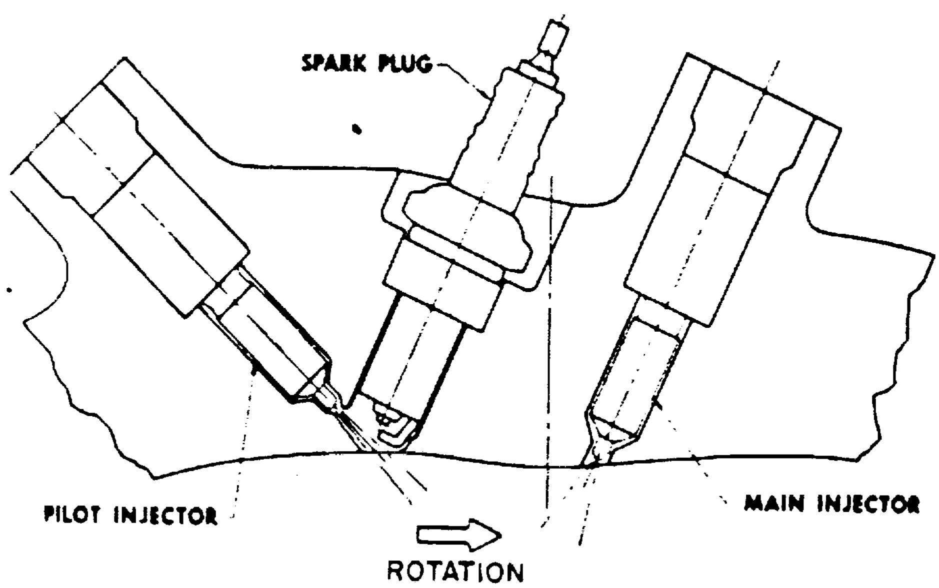


Figure 9. Stratified Charging of a Rotary Combustion Engine

engine's MPG rating and reduce certain exhaust emissions. The RCE is well suited for conversion to an AAD engine. Its multifuel tolerance allows it to burn diesel fuel and its fewer moving parts and simpler configuration make the application of a ceramic coating on the necessary components of the RCE easier than on the RPE.

1.3 Finite Element Analysis Background

Whenever any modifications or developments of the RCE are proposed, both the efficiency derived from these changes and their effect on the structural integrity of the engine must be evaluated. Typically, these types of evaluations have been made by construction and testing of engine prototypes. Since this construction and testing can become very costly, accurate analytical modeling of the proposed modifications can save both time and money. Instead of building a prototype engine out of metal, a mathematical model of the engine can be developed using finite elements.

The Finite Element Method (FEM) is a numerical procedure used for solving differential equations. The method involves dividing a physical continuum into a finite number of geometric units, the finite elements. The equations of the properties of the individual elements can be assembled using different approaches - direct, variational, weighted residuals, or energy balance. By solving these equations simultaneously using Gaussian Elimination or a similar method, an approximation of the exact solution can be obtained.

The solution of a general continuum problem by the finite element method follows an orderly step-by-step process. The first step of a finite element analysis is to determine how accurately the geometry needs to be modeled in order to obtain stresses and displacements that closely approximate the actual values. This determination is based primarily on preliminary analysis, previous experience and a physical understanding of the problem at hand. For the center housing, the most significant geometry that was not modeled was the bolt holes in the ribs. Bolts are passed through the holes and are used to hold the two end plates to the housing. It was determined in a previous study that the exclusion of the bolt holes from the finite element model did not significantly affect the stress and deformation results (11). Other noncritical geometry, such as the grooves for the gaskets, was not modeled because its effect on the results would be insignificant.

After the geometry is created, it is discritized into subdivisions or finite elements. Before the elements can be defined, nodes and meshes are described. Meshes define the boundaries within which the nodes and elements are located. The nodes, specified by spatial coordinates, define the elements. Loading and restraint conditions are applied at the nodes. Element parameters - number, size and type - significantly affect the accuracy of the stress and deformation results. The element parameters that were used to create the FEM of the housing were validated in a previous study (11).

The final steps in the finite element analysis of the center housing include assuming a displacement model, and from that model, deriving the individual element stiffness matrices. These individual matrices are assembled and then modified to account for the restraint conditions. The matrices were formulated according to the equation:

$$\{F\} = [K]\{X\} \tag{1.3a}$$

where $\{F\}$ is the column matrix of nodal loads, $\{K\}$ is the combined stiffness matrix and $\{X\}$ is the matrix of nodal dispalcements. When equation 1.3a was solved, $\{X\}$ being the unknown, the nodal displacements are known. The stresses are derived from the displacements by using the appropriate solid mechanics equations.

1.4 Development of Automatic Finite Element Model Generators

Due to the complex nature of the RCE center housing geometry, the amount of data necessary to create a FEM is enormous. To aid the user in the construction of the FEM. data preprocessors, such as IDEAS, have been written. Even with the aid of these powerful general purpose preprocessors. it can take hundreds of hours to build one model (12). This time intensive nature of the FEM creation greatly adds to the cost of the FEM design procedure. This can prohibit the analysis of many alternate designs.

Automated design and optimization programs have been developed to reduce the time-intensive nature of finite element model creation. Typically, the programs consist of two major components - the controller and the analyzer. The analyzer performs two functions. The first function is to accept the user supplied design variables - maximum allowable stresses or critical dimensions - that are not altered during the optimization. The second function of the analyzer is to calculate the stresses and deflections of the structure. The optimizer provides input to the analyzer. and based on the results of the analyzer, determines what changes should be made to reduce the mass of the geometry being optimized.

The major drawbacks of the optimization programs include the requirement on the user to create the initial model and the limitations of the type of geometry applicable for optimization. In general. geometries that can be stamped. rather then cast, are better suited for automated design. Bennett (13) cites several reasons for this. They include: problems maintaining an adequate finite element mesh on boundaries that are varied during the optimization, defining general shapes using a reasonable number of design variables, and imposing the proper constraints so that a realistic design results. Because of these restrictions, the automated design and optimization programs are best suited for only relatively complex structures that can be modeled with beams, thin shells or a combination of the two element types. Currently, the programs are used in the optimization of automobile and aircraft bodies.

1.4.1 Definition of the Specialized Preprocessor

There is a need to develop a program that can automatically create models of complex three dimensional geometry with a minimum of user supplied inputs. The current automated design and optimization programs are insufficient in that they not only require the user to create the initial model, but also prohibit the use of three-dimensional elements in the finite element model. This thesis outlines the development of a specialized preprocessor (SP) that will reduce the time necessary to create FEM's of different RCE center housing geometries. With a minimum of user supplied inputs, the SP will automatically generate a finite element model of the specified housing geometry. The automatic optimization of the model, as in the current programs, is beyond the scope of this thesis; it is left up to the user. The SP is designed only to build finite element models of different RCE center housing geometries. Being geared towards the building of a specific geometry, the specialized preprocessor requires only a minimum knowledge of the system by the user. With the automatic capability of the SP significantly reducing the time necessary to construct the FEM, alternate RCE center housing designs can be analyzed without time or cost becoming prohibitive.

The SP was developed to be used in conjunction with General Electric / Computer Aided Engineering's (GE/CAE) graphics package, specifically, IDEAS. The SP is a FORTRAN coded program. When executed, the SP creates two different files. The first file is formatted such that it can be read by the GE/CAE package. This file contains all the data necessary to describe the geometry of the RCE center housing. The file is in universal format; hence the name, Universal File (UF). The universal format is a standard that has been defined by GE/CAE (14-15). See Appendix 1 for an example of a universally formatted file. The second file created by the SP contains commands that can be executed within IDEAS. The commands are necessary to guide the IDEAS subroutines through the generation of the meshes, nodes and elements on the geometry. The file uses the programming capabilities within IDEAS; hence the name Program File (PF).

There were several reasons why it was necessary to write two files - one UF and one PF - instead of just one UF or just one PF. The PF was necessary because certain entities that are needed to define a FEM, namely the meshes, can not be defined in the UF because GE/CAE did not include meshes when they defined this universal format. The PF contains commands that use the programmability allowed within IDEAS to guide the complicated algorithms in IDEAS through the generation of the meshes, nodes and elements. The UF is still necessary, although limited to defining geometry, because the programmability that is allowed within IDEAS is limited to two hundred variables - not sufficient to describe the complex geometry of the center housing, but powerful enough to guide the algorithms in IDEAS through the generation of the meshes, nodes and elements. In summary, the UF is needed to define the geometry, but it can not define the meshes, nodes and elements. The PF is needed to define the meshes, nodes and elements, but it can not be used to program the different geometric possibilities of the center housing.

CHAPTER 2: Universal File Development

Before the development of the UF is described, the reader should have a good understanding of the universal format. For the RCE center housing, six geometric entities are required to completely describe the RCE center housing geometry. They are points, lines, splines, edges, surfaces, and volumes. GE/CAE has established a hierarchy that must be followed when describing these geometric entities in universal format. From lowest order to highest order entity, the hierarchy is:

- Points are used to define lines and splines.
- Lines and splines are used to define edges.
- Edges are used to define surfaces.
- Surfaces are used to define volumes.

As all these geometric entities are being created, they are assigned numeric labels. The universal format uses the labels of lower order entities to define upper order entities. As examples, two point labels could be used to define a line or six surface labels could be used to define a volume.

2.1 Geometric Entity Definition

The first step required in the development of the UF was the decomposition of the complicated geometry of the center housing into smaller, simpler components. The most fundamental components are the inner and outer shells of the center housing and the space between the shells (Figure 10). These shell components can be combined to form geometric building blocks that represents small but repeated sections of the center housing. These geometric building blocks are:

- Ribs.
- Channels.
- Intake port.
- Exhaust port.
- Spark plug ports.

By combining the building blocks in different combinations, the complicated center housing geometry can be formed. The ribs and channels are the primary building blocks. The majority of the center housing is formed by alternating ribs and channels. The intake and exhaust ports exist only once, if at all, in the center housing (in some engine configurations, the intake port is located in the end plates not in the center housing). The spark plug ports exist twice.

2.1.1 Rib Definition

There are two different types of ribs - normal and recessed (Figure 12). A normal rib is defined as the inner and outer shell of the center housing and the material between the shells. A rib is designated as recessed when there is a gap between the material and the inner shell. This type of rib is used in areas where relatively high temperatures are expected - the spark plug region. The increased coolant flow allowed by the recess dissipates more heat in this area.

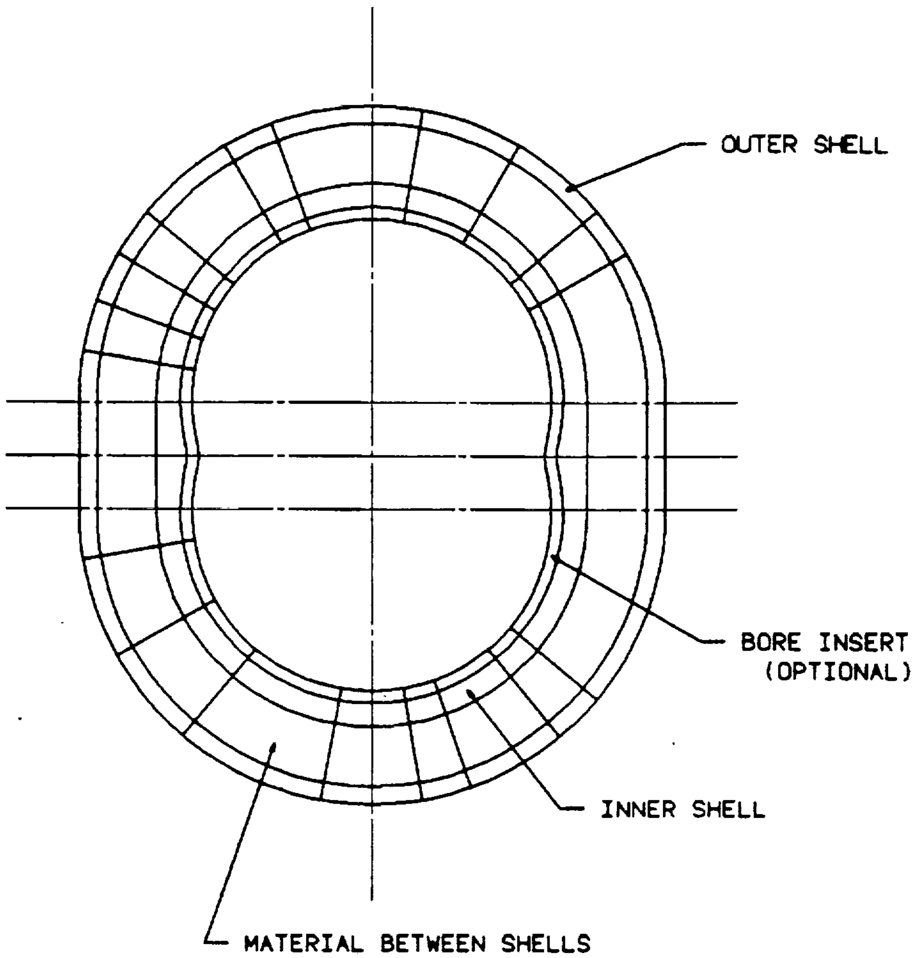


Figure 10. Fundamental Components of the Center Housing

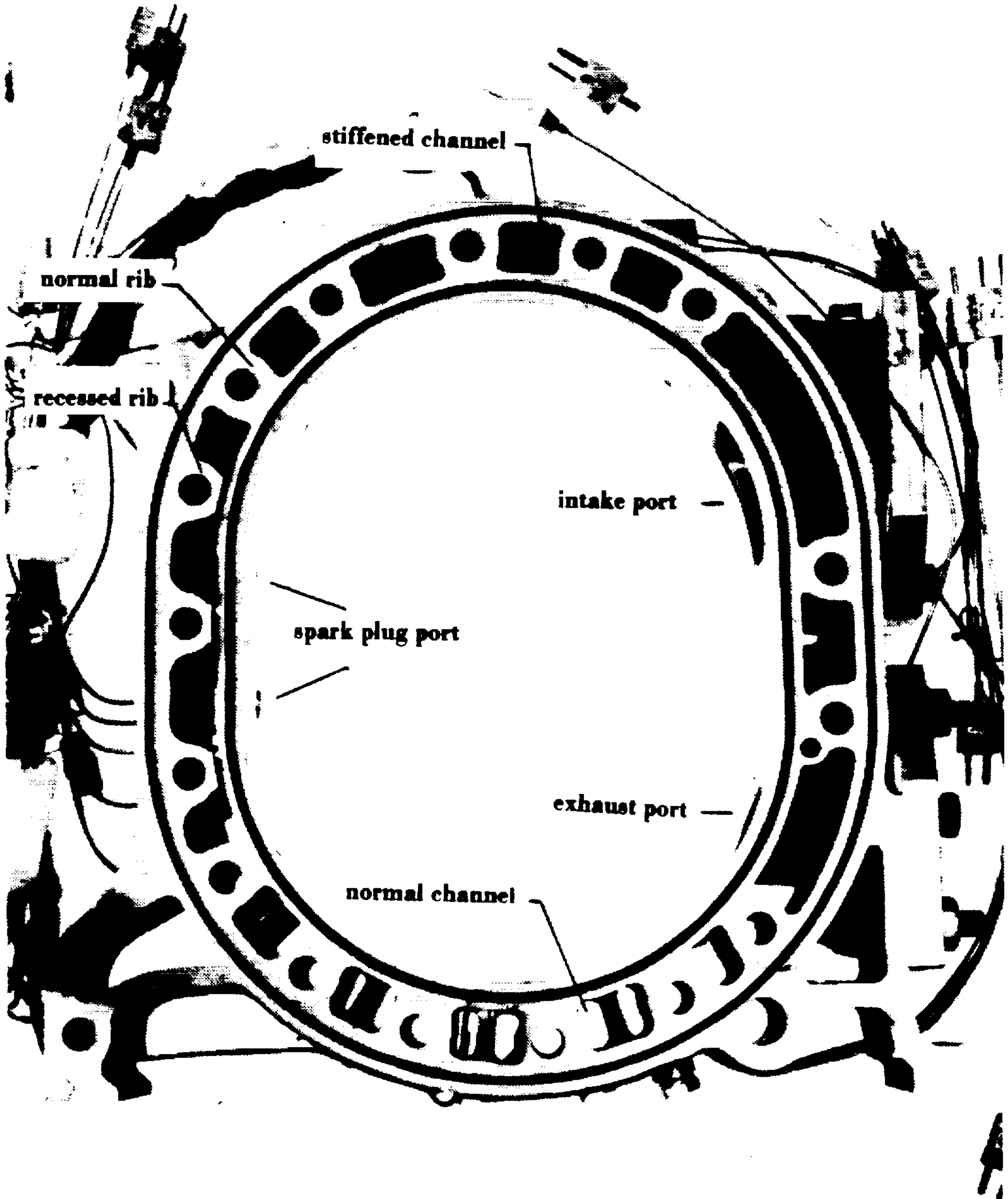


Figure 11. Ribs, Channels and Ports: as they appear in the Center Housing.

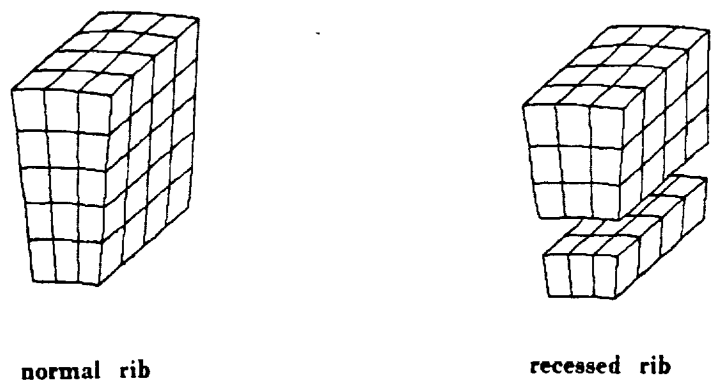


Figure 12. Modeling of the Two Rib Types

2.1.2 Channel Definition

There are two types of channels - normal and stiffened (Figure 13). A normal channel is defined as the inner and outer shell of the center housing. There is no material between the shells. Engine coolant flows through the vacancy between the shells. A channel is designated as stiffened when a thin plate at the midplane of the channel blocks the coolant flow. This type of channel is used in areas where relatively large deformations and low temperatures are expected - the compression region. The increased stiffness caused by the plate decreases the deformations in this area. Figure 14 illustrates both rib and channel types as they may occur in the center housing geometry.

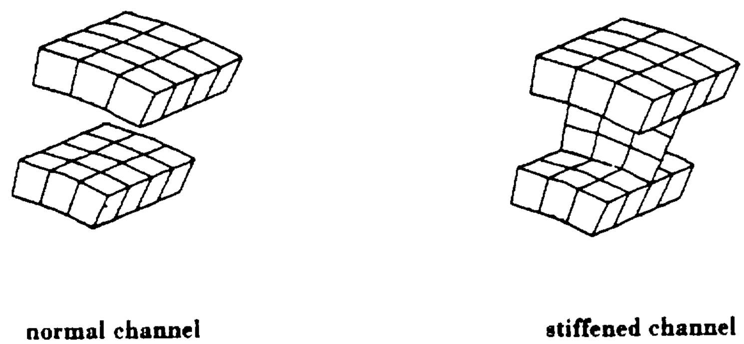


Figure 13. Modeling of the Two Channel Types

2.1.3 Bore Insert Definition

There has been increasing interest in converting the RCE to an AAD engine. To accommodate this interest, the SP will, if the user specifies, place a bore insert in the center housing. The bore insert could consists of a ceramic based material that reduces the heat rejection of the RCE.

In terms of the definitions of the building blocks of the housing geometry, the insert is the addition of a shell on the trochoidal surface. Instead of being defined as **one** inner shell, outer shell, and the material between shells, the rib is defined as **two adjacent** inner shells, an outer shell, and the material between the shells. The insert has a similar affect on the definition of a channel. (See Figure 14).

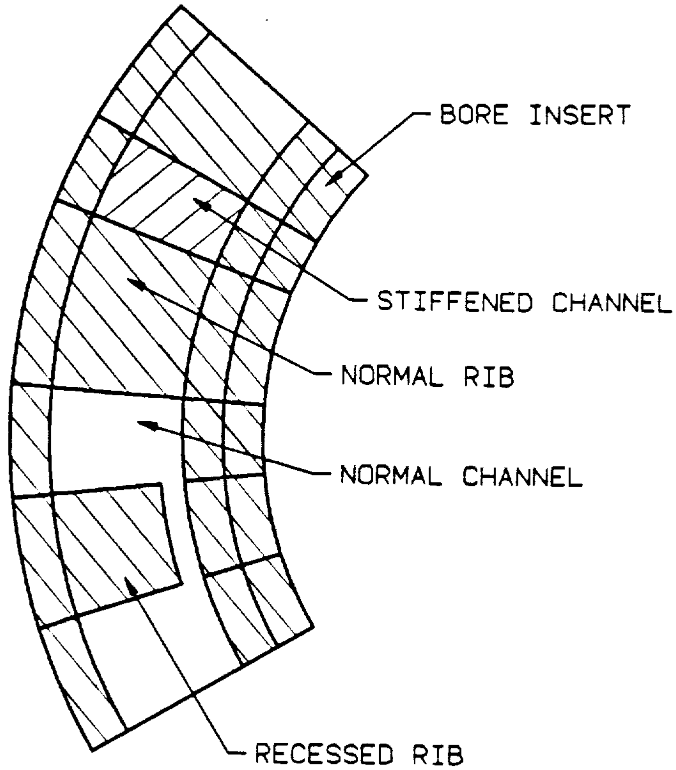


Figure 14. Example of Combined Rib and Channel Types: This figure illustrates how the rib and channel types can be combined to form the center housing geometry.

2.1.4 Intake Port Definition

The intake port is a hole that runs radially through the center housing. At the inside edge of the center housing the hole is rectangular. At the outside edge, the hole is circular (Figure 15). The air and fuel mixture is injected from this port into the trochoidal chamber of the center housing. The SP has been written so that the intake port can be omitted from the center housing geometry. This option has been included because in some RCE configurations, the intake port is located in the end plate, not the center housing.

2.1.5 Exhaust Port Definition

The exhaust port, like the intake port, is a hole in the center housing that runs radially through the center housing. The hole is circular at the intersection of both the inside and outside edges of the center housing (Figure 16). The residue of the combustion process is ejected out of this port.

2.1.6 Spark Plug Port Definition

The spark plug ports are, as expected, where the spark plugs are screwed into the center housing. Two spark plugs are used in the RCE to make the combustion process more efficient thus producing less exhaust emissions. The second spark plug burns some of the air fuel mixture not burned by the first spark plug. Like the intake and exhaust ports, the spark plug ports are holes that run radially through the center housing (Figure 17).

2.2 Geometric Parameter Definition

The second step required in the development of the SP was to define the parameters that would be needed to fully describe the building blocks. In order that these user supplied parameters are as easily obtainable as possible, all the data required by the SP can

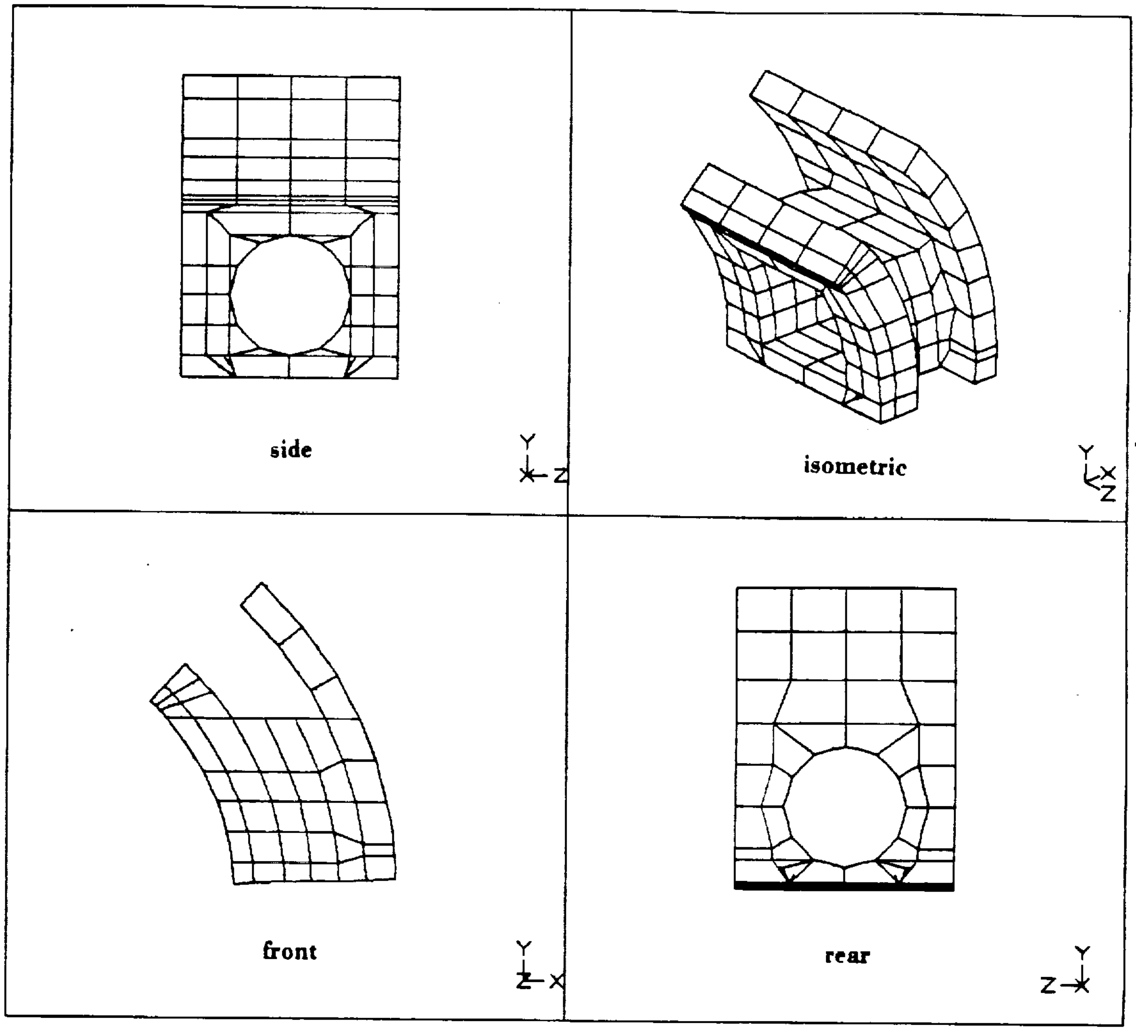


Figure 15. Modeling of the Intake Port

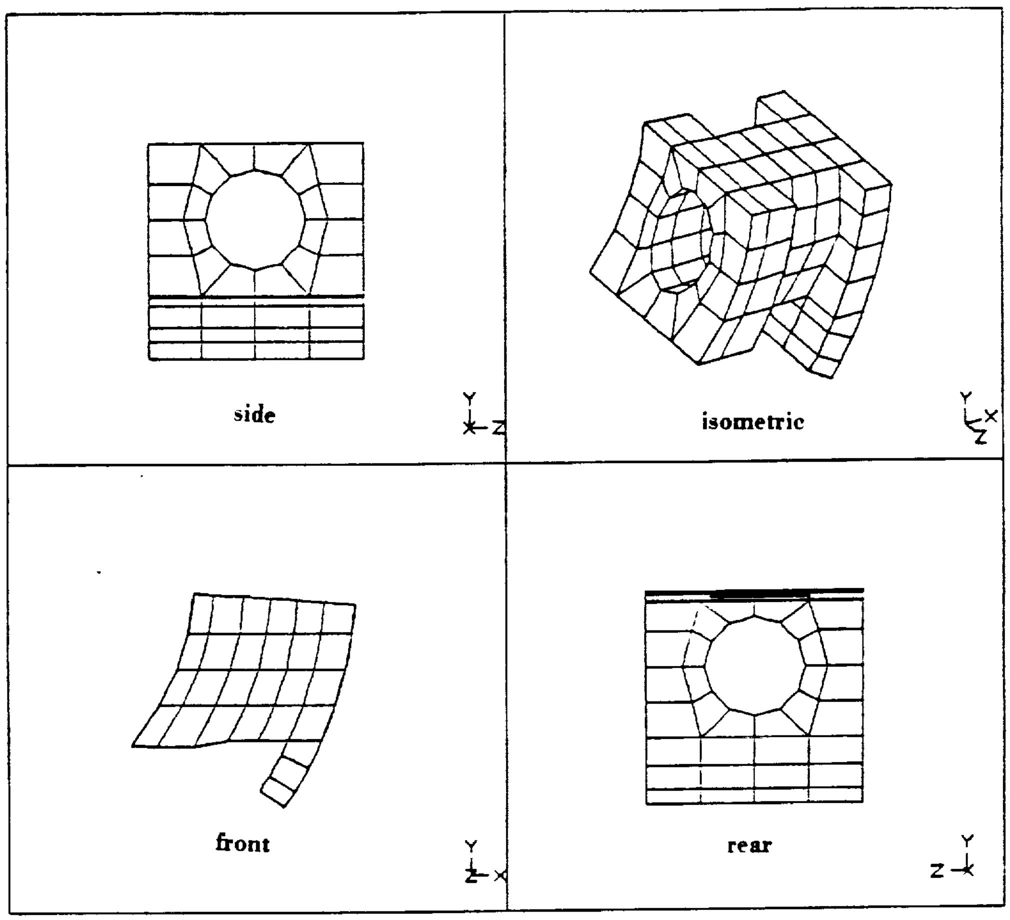


Figure 16. Modeling of the Exhaust Port

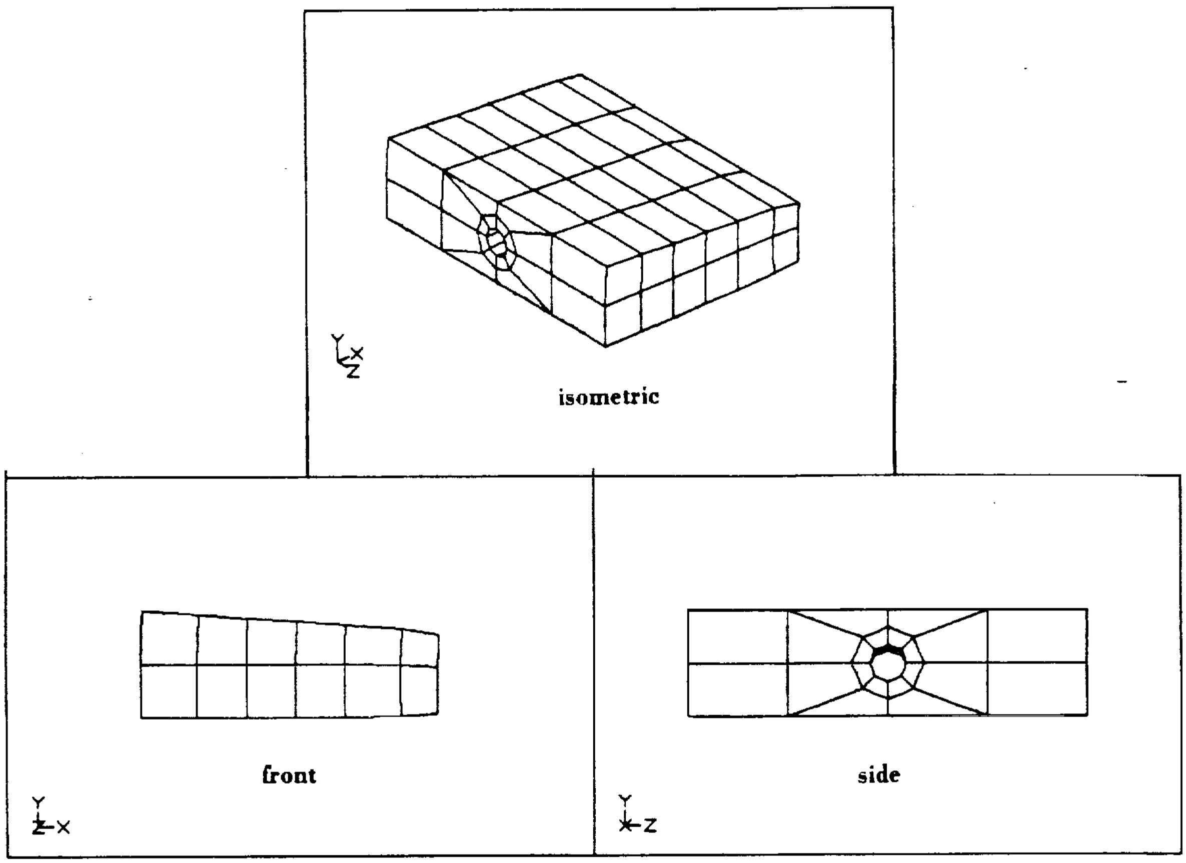


Figure 17. Modeling of the Spark Plug Port

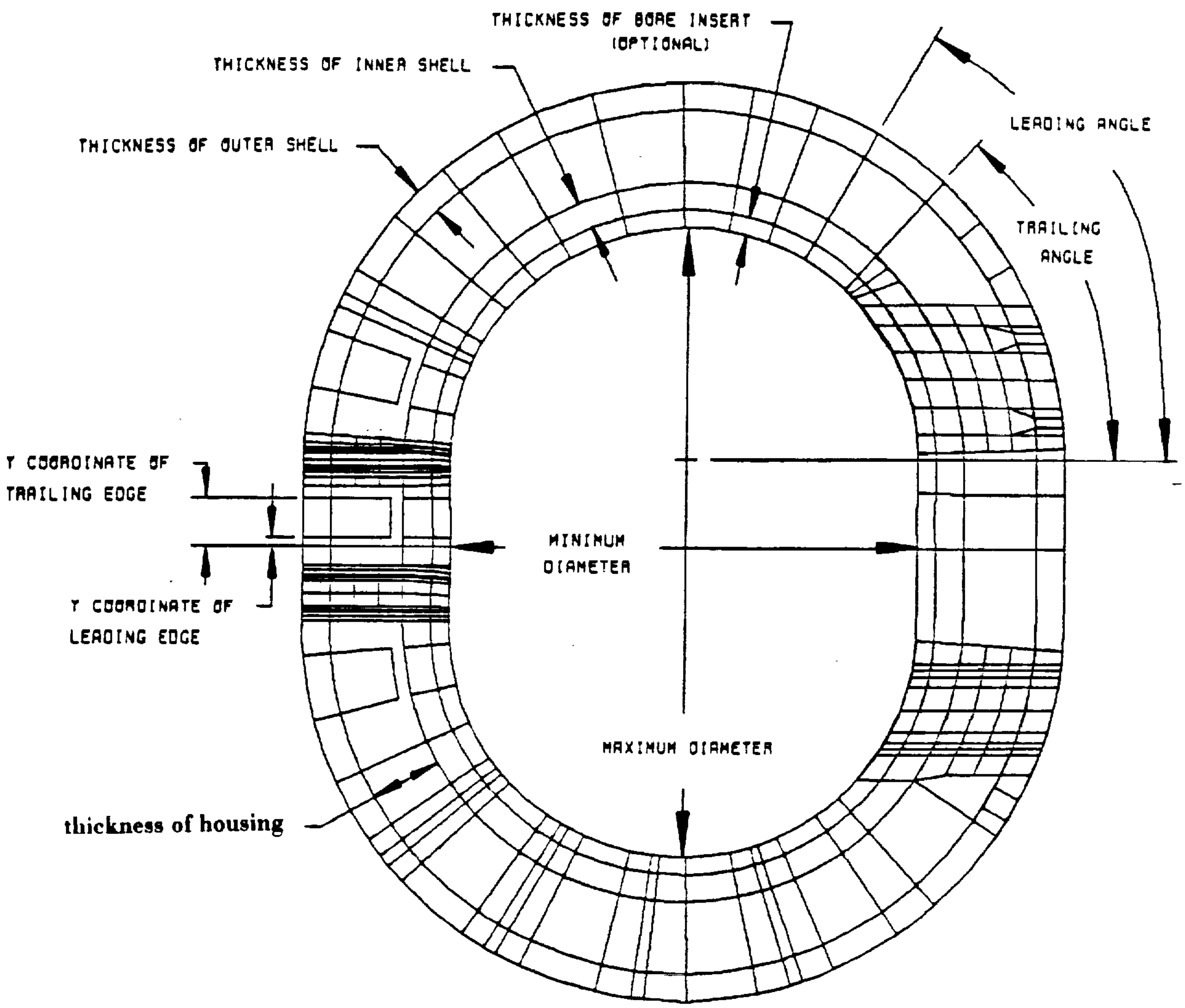


Figure 18. Typical Drawing of the Center Housing: All the parameters needed to describe the housing geometry can be measured off of this drawing.

The inputs supplied by the user are:

- The maximum and minimum diameters of the trochoidal bore.
- The thickness and axial depth of the center housing.
- The thickness of both the inner and outer shells.
- The thickness of the bore insert. (optional)
- The size, type and location of each rib.
- The type of each channel.
- The size and location the intake port.
- The size and location of the exhaust port.
- The size and location of the spark plug ports.

be measured from a drawing of the center housing. All parameter definitions are illustrated in Figure 18.

A detailed explanation of these parameters can be found in following sections.

The manner in which the user supplied parameters are defined changes depending on which region of the center housing is being constructed. There are two types of regions in the center housing - cylindrical and rectangular (Figure 19). In the cylindrical region, the inputs are in polar coordinates and in the rectangular region, the inputs are in Cartesian coordinates. Different coordinate systems are used so that the different parameters can be more easily obtained by the user.

There is a possibility that a rib or channel lies, not entirely within the cylindrical or rectangular region, but overlaps the two regions. When this overlapping of regions occurs, the SP must know what percent of the rib or channel lies in each region. The percentage of overlap must be known so that the SP can use the appropriate coordinate system - polar or Cartesian - to calculate the point coordinates of the overlapping rib or

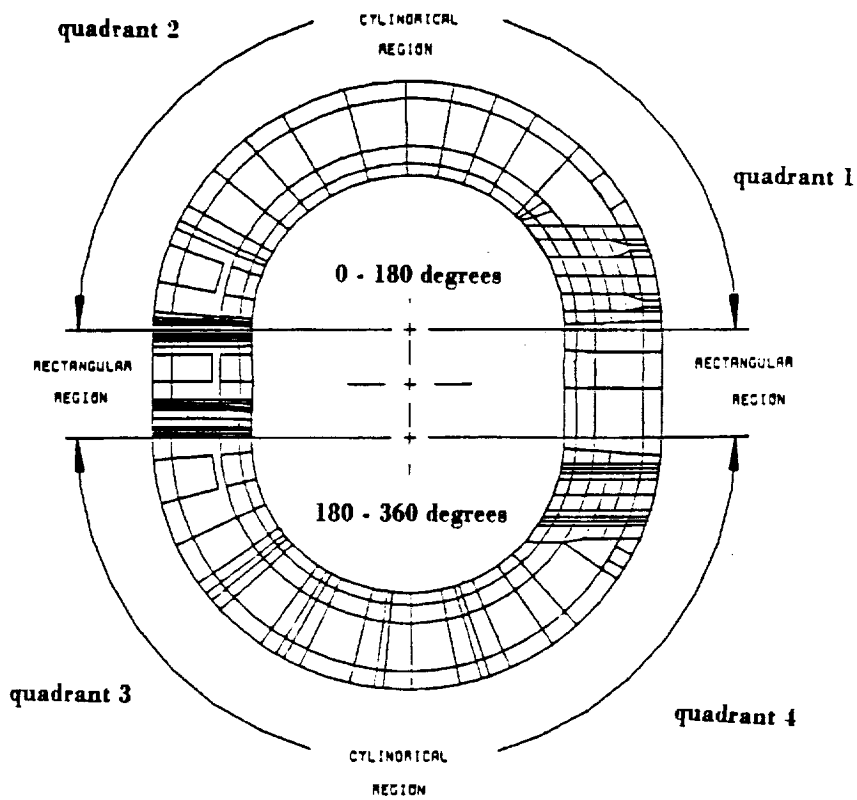


Figure 19. Illustration of the Different Regions in the Center Housing

channel. The amount of overlap is calculated from the user supplied leading and trailing edge input that defines the location of the ribs and channels.

The overlap conditions were classified into six categories.

The overlap categories are:

1. Twenty percent or less of the **rib** lies in the cylindrical region
2. Eighty percent or more of the **rib** lies in the cylindrical region
3. Between twenty and eighty percent of the **rib** lies in the cylindrical region
4. Twenty percent or less of the **channel** lies in the cylindrical region
5. Eighty percent or more of the **channel** lies in the cylindrical region
6. Between twenty and eighty percent of the **channel** lies in the cylindrical region

Based upon the category that the overlapping rib or channel falls under, the SP calls a routine that uses the appropriate coordinate system to calculate the point coordinates that lie in the different regions.

2.2.1 Rib and Channel Input Parameter Definition

Because there are two different regions, the first input the user must specify is in which of these regions the rib lies or whether the rib overlaps the regions. If it is in the rectangular region, the y-coordinate of the leading and trailing edges must be specified. The x-coordinate is unnecessary because it is required that the ribs must be specified in a counterclockwise order starting in the first quadrant. This allows the x-coordinate to be defined by the housing input data. In the cylindrical region, the user must specify the angles of the leading and trailing edges of the rib. The last input needed to describe a rib is its type - normal or recessed. If a rib is recessed, the user must specify the magnitude of the recess.

Because the dimensions of the channels are defined by the borders of the ribs, the only input that is needed to describe the channel is its type - normal or stiffened. If a channel is stiffened, a thin plate is placed at its midplane blocking the coolant flow. The thickness of the plate does have to be input because the plate is modeled with thin-shell elements. For thin-shell elements the thickness does not have to be defined by the geometry. The thickness is, however, defined in the finite element input data file and it is considered in the creation of the element stiffness matrix.

2.2.2 Intake and Exhaust Port Input Parameter Definition

In order to maintain a uniformity in the generation of the housing geometry, the center housing is initially constructed only with ribs and channels. The intake and exhaust ports are then inserted into the channels specified by the user. When the ports are inserted, they do not occupy the entire channel; a small portion of the channel remains. To define where in the channel the ports are to be inserted and how much of the channel is to remain, the user must specify the leading and trailing angle of the ports at the intersection of the outer edge of the center housing (see Figure 20 for an illustration of these angles). The outer edge of the housing is chosen because the radius to this edge is known; the radius to the trochoidal edge is not. This trochoidal radius must be known when the input angles are being reduced into a form that can be used in the equations that calculate the point coordinates of the ports. These angles, as well as locating the port within the channel, also define the diameter of the circular end of the port. For the intake port, because one end of the port hole is rectangular, the user must specify the length and width of the rectangular hole. The last parameter needed to describe the intake or exhaust port is a wall thickness of the port. This input is needed because, in the axial direction, the ports are not flush with the edges of the housing. The indentation created by the port not being flush with the housing is termed the wall thickness of the port.

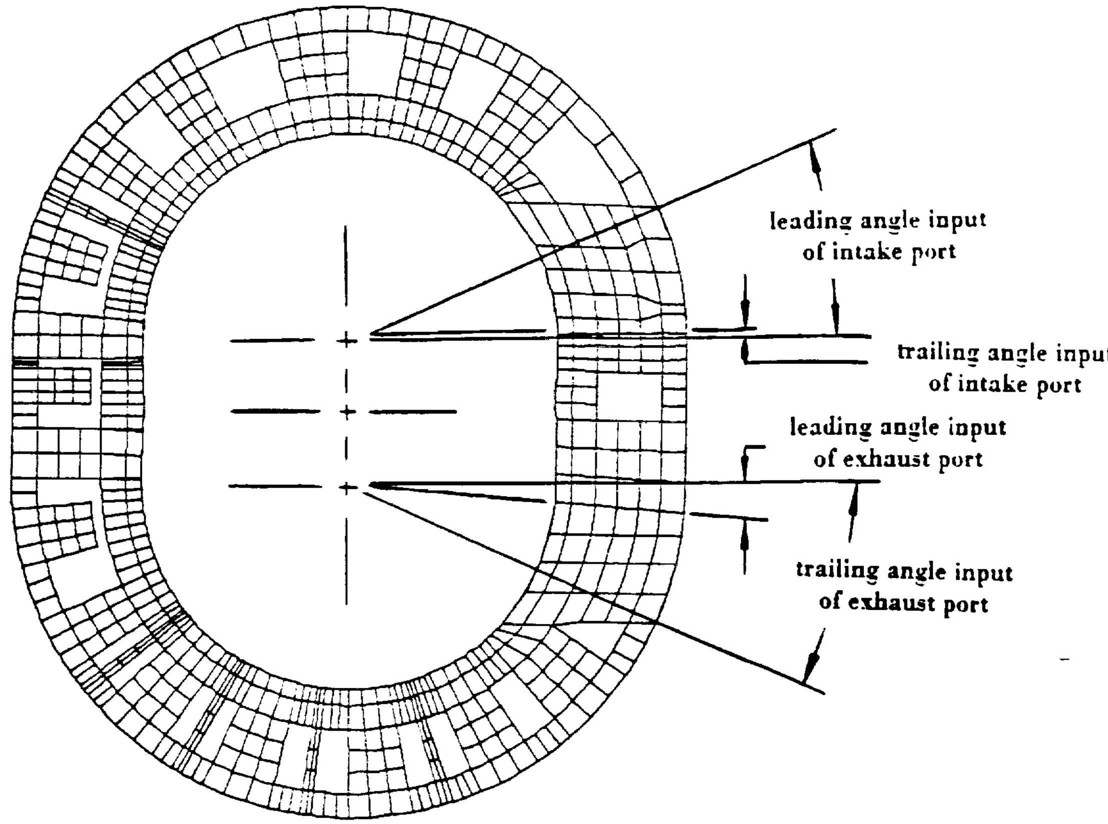


Figure 20. Illustration of the Intake and Exhaust Port Input Data

2.2.3 Spark Plug Port Input Parameter Definition

The spark plug can be thought of as a specialized rib: it is a rib that has a hole through it. The user has to specify which rib location is to have a circular hole in it and the radius of that hole. The user does not have to specify the location of the hole in the rib; it is always centered. The program has been written so that any number of spark plugs can be specified. Some RCE configurations have one spark plug and others have two.

2.3 Numbering Patterns of the Geometric Entities

The difficulty in developing the UF arises in keeping the program general. All possible geometric configurations of the housing must be considered. The different types of ribs and channels must be able to be input in any combination. For example, one section of the housing might consist of a normal rib joined to a stiffened channel joined to a second normal rib while another section of the housing could consist of a different pattern of alternating rib and channel types. All possible combinations of the building blocks must be accounted for in order that their individual geometric entity label numbering patterns can be written into the UF. The numbering patterns of the geometric entity labels result from the order in which the geometric entities are used to create the building blocks. The generation order of the geometric entities that was chosen to create the building blocks progressed first radially, then axially and finally angularly. This generation order created more repetitive numbering patterns than any other generation order. The more repetitive the numbering patterns were, the easier it was to organize them into a FORTRAN program (Figure 21 and Figure 22). It is critical that the numbering patterns of the entities be regular so that the UF maintains its ability to generate any user specified housing geometry.

To minimize the number of different numbering patterns needed to define the housing geometry, the ribs and channels were constructed together as rib/channel segments. There are three types of rib/channel segments - normal rib/normal channel, normal rib/stiffened channel, and recessed rib/normal channel (Figure 23). Associated with each of these three segments is a numbering pattern of the geometric entity labels. The user specifies which type of segment they are building and the SP calls the correct

line label	point labels	
	start	stop
1	9	10
2	10	11
3	11	12
4	9	33
5	10	34
6	11	35
7	12	36
8	33	34
9	34	35
10	35	36
11	21	22
12	22	23
13	23	24
14	21	45
15	22	46
16	23	47
17	24	48
18	45	46
19	46	47
20	47	48

Figure 21. Example of a Numbering Pattern: This pattern creates the lines of a normal rib/normal channel segment

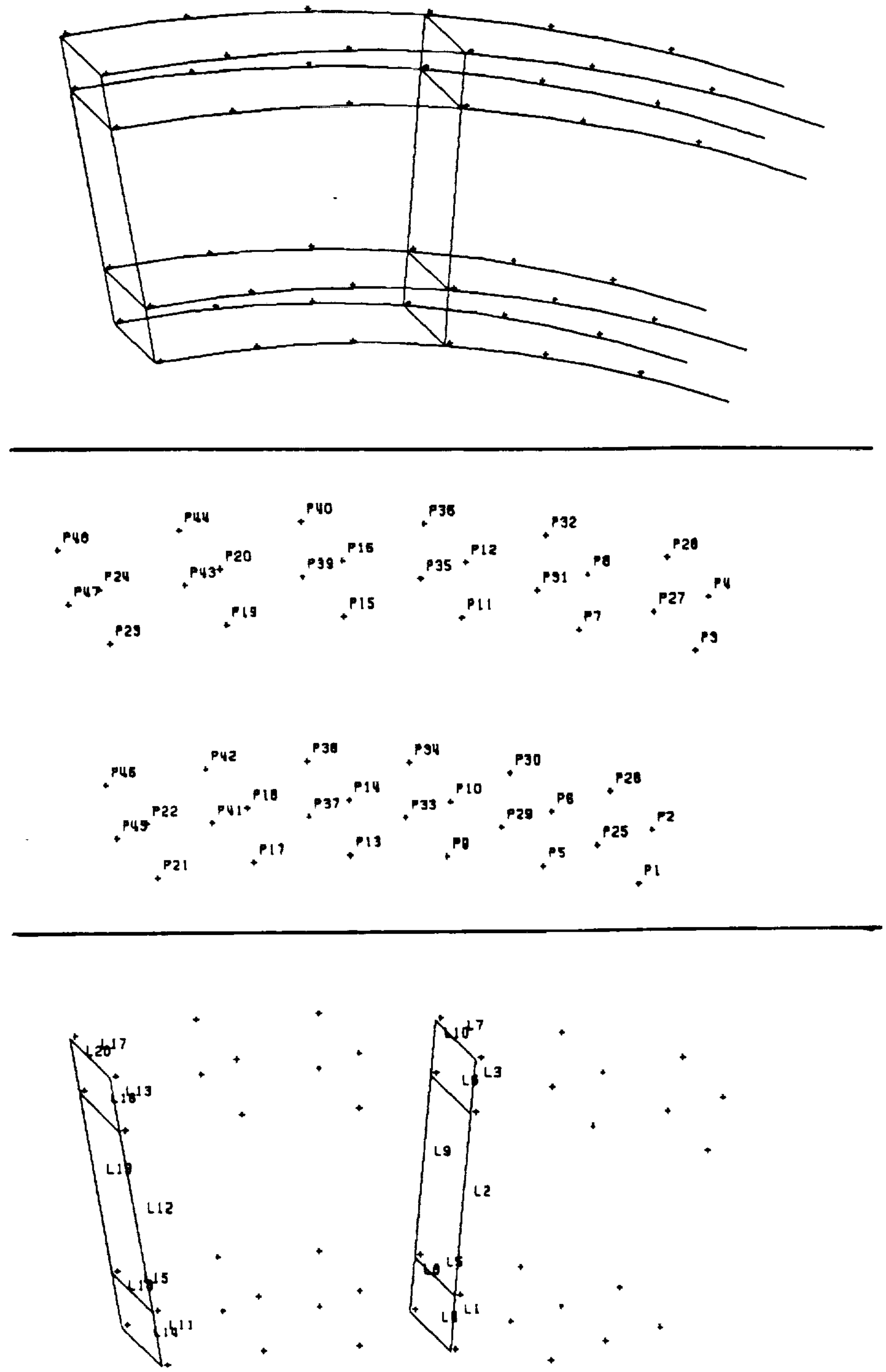


Figure 22. Pictures Generated from Numbering Pattern: The top figure is a reference for the the bottom two figures. The middle figure is points and their labels. The bottom figure is the lines generated by the numbering pattern of the point labels.

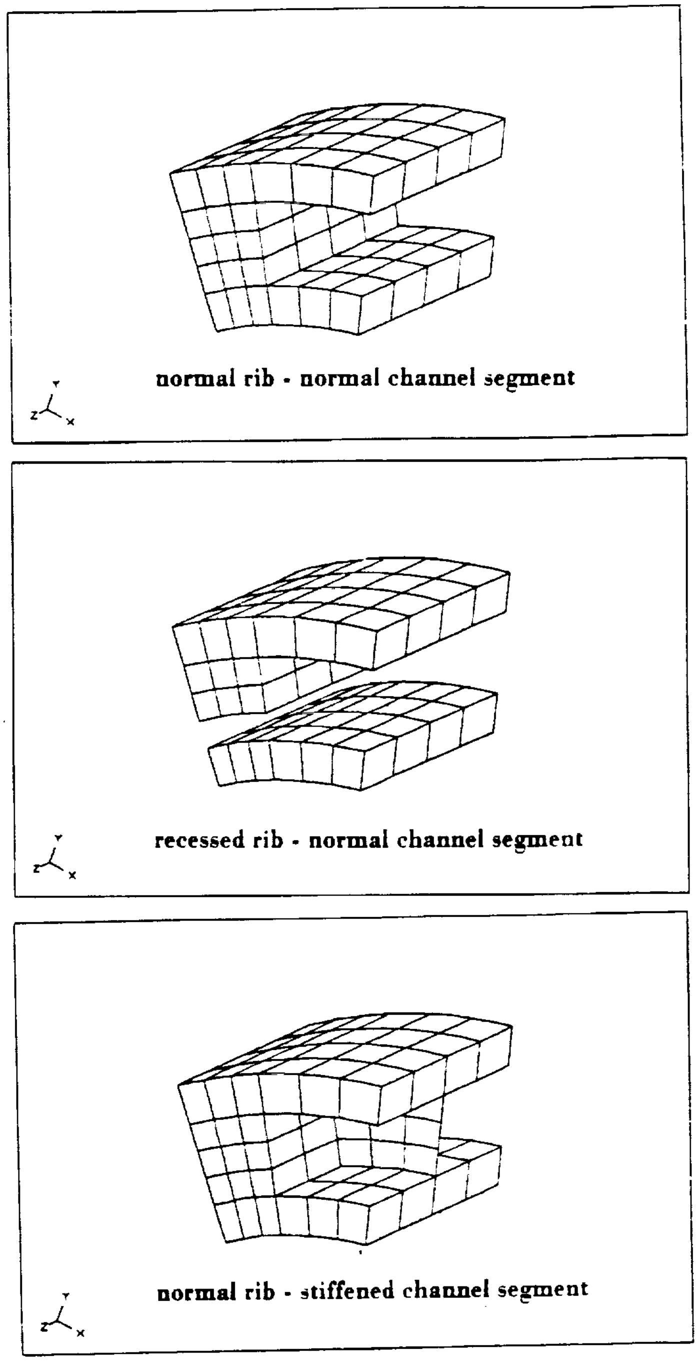


Figure 23. Illustration of the Three Rib/Channel Segment Types

2.4 Reduction & Implementation of Input for Universal File Generation

The input parameters can be categorized into three types. The first type are inputs that can be used directly by the SP, no conversion is necessary. For example, the axial depth of the housing is used unchanged to give the x and y point coordinates that are calculated by equations 1.1a and 1.1b a z dimension (depth). The second type of input is used to set flags in the SP. All the rib and channel type inputs fall into this category. For example, since different routines in the SP are used to generate the two rib types - normal and recessed - flags have to be used to ensure that the correct routine is called. The third and final type of input must be converted into a form that the SP can use to calculate the point coordinates of the geometry. For example, the maximum and minimum diameter inputs that define the trochoidal bore must be converted to the radius and eccentricity that are used in equations 1.1a and 1.1b. The following sections outline the necessary reductions for the third type of input.

2.4.1 Housing Dimension Input Reduction and Implementation

The housing parameters are used to calculate the constants, radius, eccentricity and perpendicular distance, that are used in equations 1.1a and 1.1b to define the trochoidal shape. The maximum and minimum diameters of the housing are used to calculate the radius and eccentricity of the trochoidal surface by applying the boundary conditions:

$$\text{minimum diameter} = \text{radius} - \text{eccentricy} \qquad \text{when} \quad \alpha = 90° \tag{2.4.1b}$$

$$\text{maximum diameter} = \text{radius} + \text{eccentricy} \qquad \text{when } \alpha = 0° \qquad (2.4.1a)$$

to equations 1.1a and 1.1b. Note that the perpendicular distance, 'A' in equations 1.1a and 1.1b, was set equal to zero for the generation of the inner edge of the trochoidal surface. This is why it does not appear in equations 2.4.1a and 2.4.1b. The thicknesses of the inner shell, outer shell and the housing that the user supplies define the different perpendicular distances that are used to generate the family of trochoidal curves that describe the housing geometry. The last housing parameter, the axial depth, is used to project the two dimensional model into three dimensions.

2.4.2 Cylindrical Region Input Reduction and Implementation

To define a rib/channel segment that lies in the cylindrical region or overlaps the cylindrical and rectangular regions, the user supplies the angles of the leading and trailing edges of the rib. The leading and trailing angles, called phi, are measured from the center of the cylindrical region. The cylindrical region, as its name implies, is cylindrical. Because it is cylindrical, the center of the region can be located by subtracting the maximum y-coordinate of the trochoidal surface from the maximum x-coordinate. The difference of the two coordinates is the distance measured along the y-axis between the origin of the center housing and the center of the cylindrical region. (Figure 24).

The maximum y-coordinate is easily determined; it is the radius plus the eccentricity of the trochoid. The x-coordinate was more difficult of determine. A numerical routine was written that, starting from zero and incrementing by one tenth of a degree to ninety degrees, would input alpha into equation 1.1a. The x-coordinates from equation 1.1b were compared until the largest value was found. Only one quadrant had to be

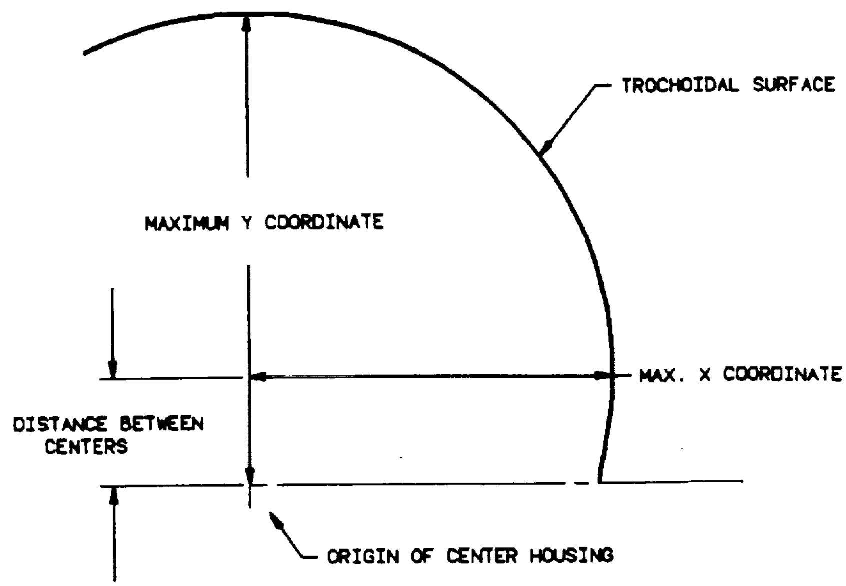


Figure 24. Determination of the Center of the Cylindrical Region

checked for the maximum x-coordinate since the trochoidal surface is symmetric about both axes.

The input angles, phi, must be converted to the angles, alpha, that are used in equations 1.1a and 1.1b. To accomplish this conversion, a numerical procedure was written that calculates an alpha, given a phi. The procedure uses the incremental search method to 'trap' the solution between an upper and lower value and the bisection method to solve for the root explicitly. The bisection method solves the equation:

$$\alpha = \tan(\varphi) \times [(E \times \cos(3\alpha) + R \times \cos(\alpha)) - (E \times \sin(3\alpha) + R \times \sin(\alpha))] + T \quad (2.4.2a)$$

where E and R are the eccentricity and radius of the housing respectively, and T is the distance between centers. The bisection method was used because its convergence to the root of the equation was not affected by the inflection points that occur in the plot of alpha versus phi (Figure 25). So, given an angle phi, a corresponding angle alpha can be solved for. Alpha is used in equations 1.1a and 1.1b to solve for the x and y coordinate of a point on the trochoidal surface.

Now, after alpha has been obtained, theta is the only unknown in equations 1.1a and 1.1b. Theta is given by:

$$\theta = \cos^{-1}\left[\frac{(R + 3 \times \cos(2 \times \alpha))}{\sqrt{9 \times E^2 + R^2 + 6 \times E \times R \times \cos(2 \times \alpha)}}\right] \tag{2.4.2b}$$

After theta is known, the x and y coordinates of the trochoidal surface can be calculated. The x and y-coordinates are given a z-coordinate equal to the axial depth to create three-dimensional point coordinates.

Seven different values of phi are needed to calculate the point coordinates that describe a rib/channel segment. Two of the seven values of phi come from the leading and trailing angle input. A third comes from the leading angle of the previous rib. The remaining four values of phi are calculated by dividing both the rib and channel into three equal parts (Figure 26). The seven values of phi are used to calculate seven two-dimensional point coordinates on the outside of the housing. The values of phi are converted to values of alpha by the numerical routine previously mentioned. The values of alpha are used to solve for values of theta and then both the values of alpha and theta are used to calculate seven two-dimensional point coordinates on the trochoidal surface. To create three dimensional model, these fourteen two-dimensional point coordinates (seven describing the outside edge of the housing and seven describing the trochoidal surface) are given a third dimension equal to the axial depth of the center housing. This brings the

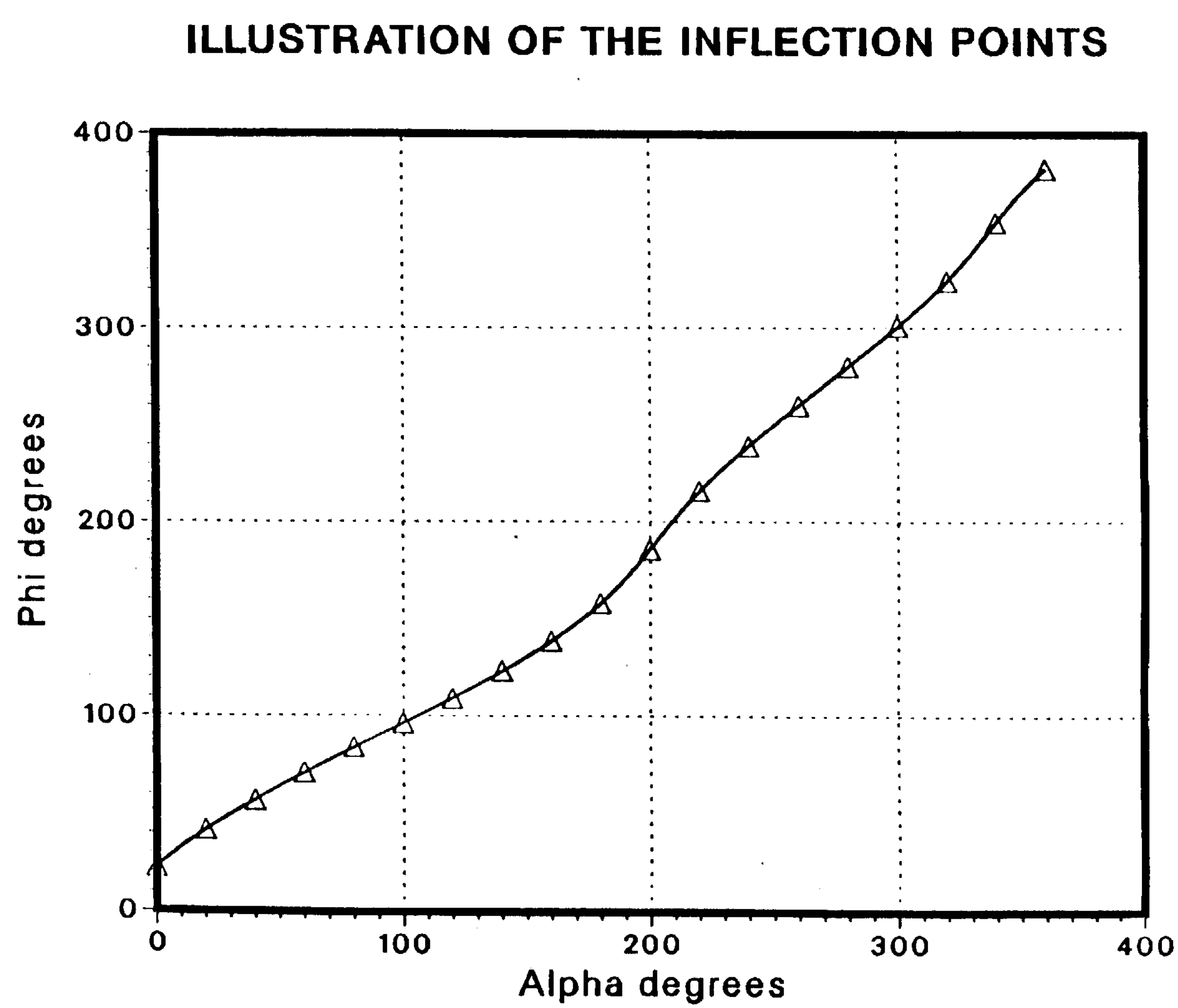
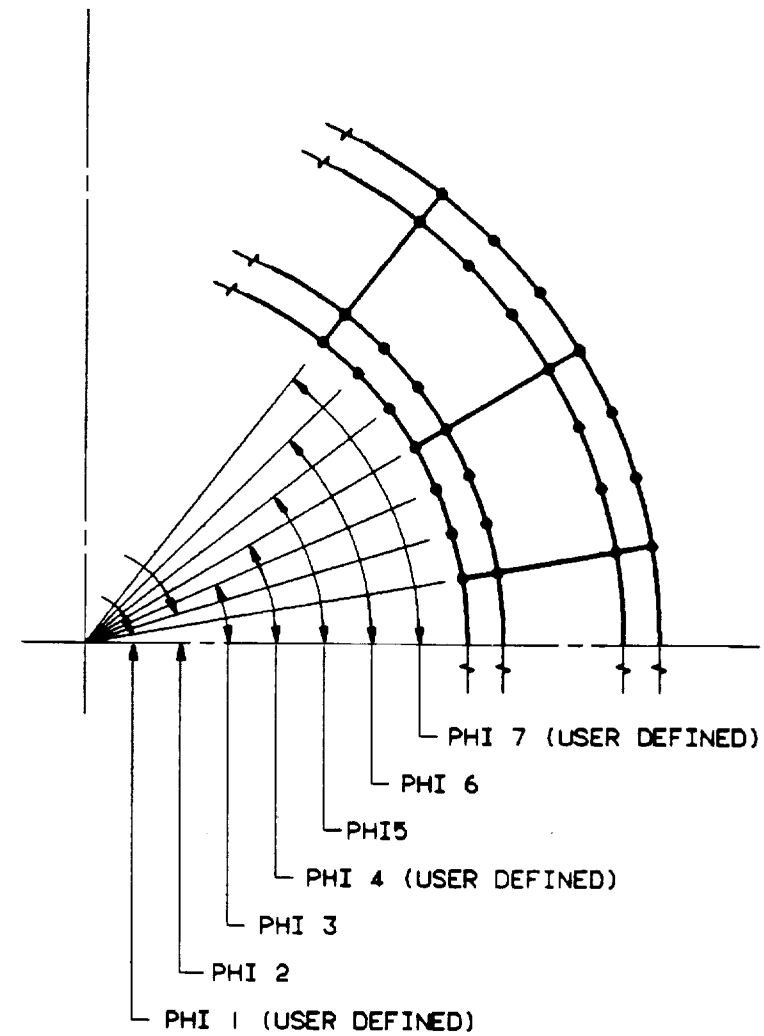


Figure 25. Graph of Alpha versus Phi



NOTE: PHI 2 = PHI 1 + (PHI 4 - PHI 1)/3

PHI 3 = PHI 1 + 2(PHI 4 - PHI 1)/3.0

PHI 5 = PHI4 + (PHI 7 - PHI 4)/3

PHI 6 = PHI 4 + 2(PHI 7 -PHI 4)/3

Figure 26. Illustration of the Seven Phi Angles: In the cylindrical region, seven phi angles are needed to define a rib/channel segment.

total number of points that describe a normal rib - normal channel segment to twenty eight.

Thirty-six point coordinates are needed to describe a rib/channel segment if either the rib is recessed or the channel is stiffened. To create the recessed rib, the eight point coordinates defining the trochoidal surface of the rib are radially expanded a distance equal to the magnitude of the recess. This provides the gap between the rib and inner shell that allows for more coolant flow. To create the stiffened channel, the four two-dimensional point coordinates defining the trochoidal surface of the channel are given a third dimension equal to one half of the axial depth. This defines the bottom spline of the stiffening plate in the channel. To define the top spline of the plate, the four point coordinates on the outer surface of the center housing are also given a third dimension equal to one half of the axial depth.

2.4.3 Rectangualr Region Input Reduction and Implementation

If the rib/channel segment lies entirely within the rectangular region, the user supplies the y-coordinate of the leading and trailing edges of a rib. To calculate the coordinates of the points on the trochoidal surface, alpha, of equations 1.1a and 1.1b, must be known. To solve for alpha, given a y-coordinate, a numerical procedure, was written. The procedure uses the incremental search and the bisection method to solve the equation:

$$\alpha = E \times \sin(3\alpha) + R \times \sin(\alpha) - YCOR \qquad (2.4.3a)$$

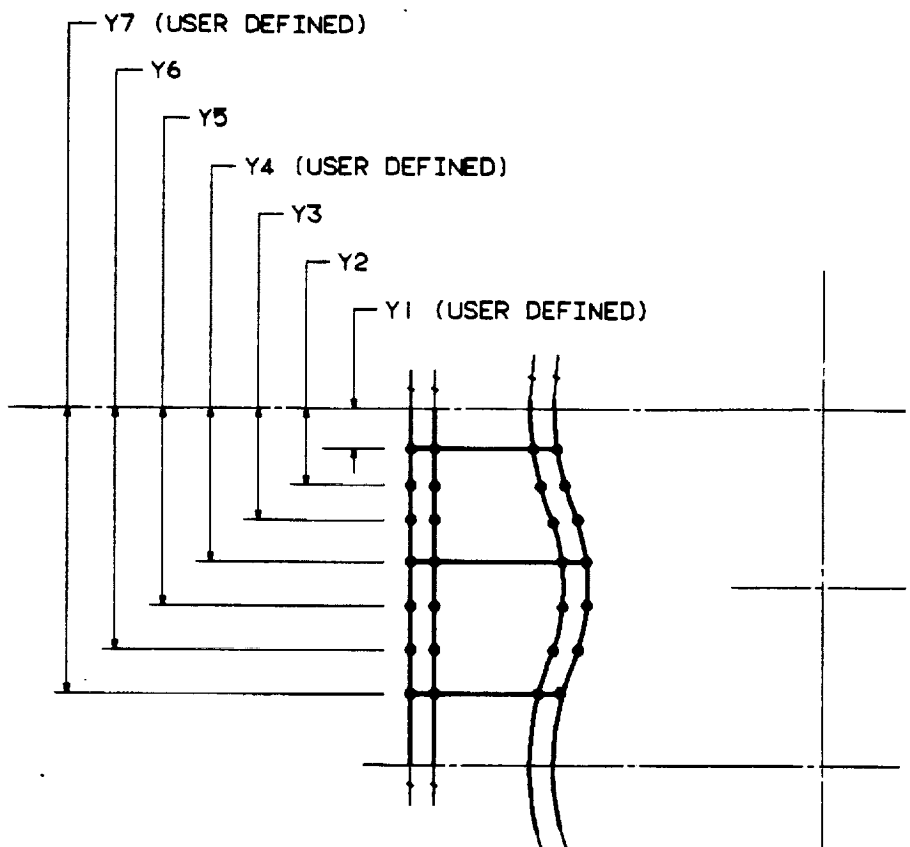
where E and R are the eccentricity and radius of the center housing respectively and YCOR is the user supplied y-coordinate.

Seven y-coordinates are needed to define a rib/channel segment in the rectangular region. The seven y-coordinates are obtained in the same manner that was described for obtaining the seven phi angles. Two of the seven y-coordinates come from the leading and trailing edge input. A third comes from the leading edge of the previous rib. The remaining four y-coordinates are calculated by dividing both the rib and channel into three equal parts (Figure 27).

It takes 28 point coordinates to define a normal rib/normal channel segment. The labels of the 28 point coordinates are used in groups of two to define 30 lines. The point labels are also used in groups of four to define the 16 splines. Each line and spline label defines an edge. The edge labels are used in groups of four to define 29 surfaces. The surface labels are used in groups of six to define the five volumes of the rib/channel segment. It takes 8 more point coordinates, two more lines, two more splines, four more edges, and one more surface to define a normal rib/stiffened channel or recessed rib/normal channel segment than it does to define a normal rib/normal channel segment.

2.4.4 Intake Port Input Reduction and Implementation

To define the intake port, the user specifies into which channel the port is to be placed, the length and width of the rectangular hole that intersects the trochoidal surface, and the diameter of the circular hole that intersects the outer surface. The diameter of the circular hole is defined by the leading and trailing angle of the edges of the hole (See Figure 20). These inputs are used unaltered to calculate the point coordinates that are used to define the intake port. The first points calculated define the outer edge of the center housing. Because the port is oriented horizontally in the housing, the remaining points defining the port are calculated by subtracting from the X-coordinates defining the outer edge; the Y and Z coordinates remain constant. The distance subtracted from the



NOTE: $Y2 = Y1 + (Y1 - Y4)/3$

$Y3 = Y1 + 2(Y1 - Y4)/3$

$Y5 = Y4 + (Y4 - Y7)/3$

$Y6 = Y4 + 2(Y4 - Y7)/3$

Figure 27. Illustration of the Seven Y-Coordinates: In the rectangular region, seven y-coordinates are needed to define a rib channel segment.

X-coordinates is calculated from the user-supplied inputs that define the thicknesses of the housing and shells.

It takes 317 points to define the intake port. The numeric labels of the 317 points are used in groups of two to define 768 lines in the intake port. Each numeric label of a line defines an edge. The edge labels are used in groups of three or four to define the 596 surfaces of the intake port. The surfaces are used in groups of five or six to define the 150 volumes of the intake port.

Due to the complicated nature of the intake port geometry, irregularly shaped elements can be created for a given set of input dimensions. Distorted elements cause the finite element analysis to yield erroneous results. To eliminate the possibility of distorted elements being created, a critical area was isolated where highly distorted elements were likely to occur if certain input dimensions were used. The critical area was found at the bottom edge of the rectangular hole (Figure 28). To prevent the creation of distorted elements, the volumes were created at a size where only one element was placed on each volume. This allowed the elements in the critical areas to be created with triangular elements. The generated shape of the triangular elements was easier to control. Although distorted elements may still be created in this area, it is not as probable now that triangular elements are employed.

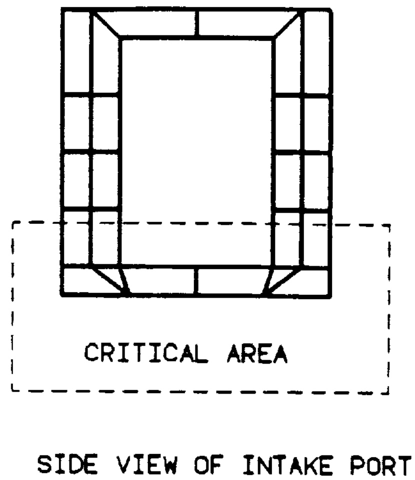


Figure 28. Critical Area for Distorted Elements

2.4.5 Exhaust Port Input Reduction and Implementation

To define the exhaust port, the user specifies into which channel the port is to be placed and the diameter of the port. The diameter of the port is defined by the leading and trailing angle of the hole (See Figure 20). These inputs are used unaltered to calculate the point coordinates that are used to define the exhaust port. The points are calculated first at the outer edge of the center housing and, working horizontally inward in a manner similar to that described for the intake port point calculation, to the trochoidal edge.

It takes 248 points coordinates to define the exhaust port. The numeric labels of the 248 points are used in groups of two to define 585 lines. Each numeric label of a line defines an edge. The edge labels are used in groups of four to define the 766 surfaces. The surfaces are used in groups of six to define the 108 volumes of the exhaust port.

2.4.6 Spark Plug Port Input Reduction and Implementation

To define the spark plug port, the user specifies in which rib the port is to be placed and the radius of the port. Even though the spark plug port can either lie within the rectangular region or overlap the rectangular and cylindrical regions, the user specifies the same inputs. These inputs are used unaltered to calculate the point coordinates of the port.

It takes 168 points coordinates to define each spark plug port. The numeric labels of the 168 points are used in groups of two to define 585 lines. Each numeric label of a line defines an edge. The edge labels are used in groups of four to define the 426 surfaces. The surfaces are used in groups of six to define the 96 volumes of a spark plug port.

CHAPTER 3: Program File Development

The PF contains the commands that are used to guide the complicated algorithms in IDEAS through the generation of the meshes, nodes and elements. The problem encountered with creating the PF is the fact that the programmability allowed in IDEAS can not read externally defined variables. An externally defined variable is a variable that is defined in a program other than that which is currently being executed. This means that variables in IDEAS can only be assigned values by IDEAS. An example of externally defined variables that IDEAS must know before it can create the PF is the element type of a given mesh or the material type of a given element.

To overcome the limitations of the programmability in IDEAS, an innovative approach was used. It involved the using of the FORTRAN program to create the PF. The PF that the FORTRAN program file creates, looks exactly like a PF that would be created by using the programmability in IDEAS. The difference being, the FORTRAN program has read variables that were calculated in the UF to create the PF. The programmability in IDEAS does not have this capability.

3.1 Mesh Definition

Before the meshes are defined, the PF must define the type of element to be placed on the mesh. Once the element type parameter has been defined, the algorithms in IDEAS are set to receive the first command that begins the mesh generation. The first command indicates which volume is having a mesh placed upon it. The remaining commands involve the number and placement of the elements that are to be generated on the mesh. (Note that the elements are not generated at this point, but the mesh generating algorithms in IDEAS must know something about the elements before the meshes can be generated.) For each surface or volume that is to have a mesh placed on it, the PF contains a set of commands that indicates which surface or volume is currently having a mesh placed on it and the number and placement of the elements on the mesh. For a typical geometry, the PF creates approximately 600 meshes.

3.2 Node and Element Definition

Before the nodes and elements can be defined, certain parameters must be set. The parameters indicate that the nodes and elements are auto numbered by IDEAS and that nontriangular elements are to be generated. Also, the material constants - modulus of elasticity and poisson's ratio - and the physical properties of the elements must be provided. An example of a physical property is the thickness of a thin shell element. Once the parameters are set and the meshes are described, the nodes and elements can be defined. The first command in the PF that begins the node and element generation indicates which mesh is to have the nodes and elements placed upon it. The remaining commands in the PF set which material and physical properties are associated with the nodes and elements being generated. In summary, for each mesh, the PF contains a set of commands that indicates which mesh is currently having nodes and elements generated on it and which material and physical properties are associated with the nodes and elements. For a typical geometry, the PF creates over 3000 elements and 20,000 nodes.

CHAPTER 4: User's Guide

The user supplied input needed by the SP were designed to be easily obtainable. To obtain the required input, the user needs either a drawing of the housing or an actual housing. It should be noted that the user does not have to obtain the input data before he/she executes the SP. The user can execute the SP and obtain each input as the SP prompts for it.

4.1 Obtaining the Input

The first input required by the SP are the maximum and minimum diameters of the trochoidal bore. Next the thicknesses of the housing's, inner shell and outer shell is required. All of the input can be measured directly off the housing or drawing such as Figure 18 on page 27. This input, along with any other input defining length, width or depth, must be input in inches.

Before the user can continue obtaining input data, the center of the cylindrical regions, shown in Figure 19 on page 29, must be located. The distance from the origin, the geometric center of the trochoid, to the center of the cylindrical regions is one half of the difference between the maximum and minimum diameter. For example, if the maximum diameter was 8.0 and the minimum diameter was 6.0, then the distance from the origin to the centers of the cylindrical regions would be:

$$1.0 = \frac{(8.0 - 6.0)}{2.0} \tag{4.1}$$

Now that the centers of the cylindrical region has been established. the data describing the ribs, channels and ports can be measured and input. The SP requires that the user begin the rib/channel input with the rib/channel segment which overlaps the intersection of the rectangular and cylindrical region in the first quadrant (see Figure 19 on page 29). The data for each remaining rib/channel segment must be input proceeding in a counterclockwise direction around the housing. This starting point and counterclockwise direction was chosen arbitrarily as no point or direction is better than any other.

The first input needed to describe a rib/channel segment is the region it lies in or whether it overlaps the two regions. Each region has been assigned a numeric label. The labels have been assigned as follows:

1. cylindrical region
2. rectangular region
3. overlaps the two regions

To specify a region, the user inputs the appropriate number associated with that region. Note that this input in is integer format (no decimal point).

The second input required to describe each rib/channel segment is the location of the leading and trailing edge of the rib. If the rib/channel segment overlaps the two region or lies entirely within the cylindrical region, the leading and trailing edges are defined by the angle the edges make with respect a horizontal line running through the center of the cylindrical region (seeFigure 20 on page 32 - all angles must be measured in degrees). If the rib/channel segment lies entirely within the rectangular region. the rib is described by the y-coordinates of its edges. See Figure 18 on page 27 for an illustration of these inputs.

The third input required to describe a rib/channel segment is the type of the rib and channel. There are three rib types - normal. recessed, or a spark plug port - and there are four channel types - normal, stiffened, intake port or exhaust port. Each rib and channel type has been assigned a numeric label. The labels have been assigned as follows:

1. normal rib
2. recessed rib
3. spark plug port rib

1. normal channel
2. recessed channel
3. exhaust port channel
4. intake port channel

To specify a certain rib or channel type, the user inputs the appropriate number associated with that type. Note that this input in is integer format (no decimal point).

If the rib or channel is not normal, than input other than the location of the leading and trailing edges is required. If the rib is recessed, then the magnitude of the recess must be measured and input. If the rib is a spark plug port, then the radius of the port must be input. If the channel is an exhaust port, then the radius of the port must be supplied. The radius of the ports is defined by angles similar to those shown in Figure 29. If the channel is an intake port, then the length and width of the rectangular portion of the port must also be specified.

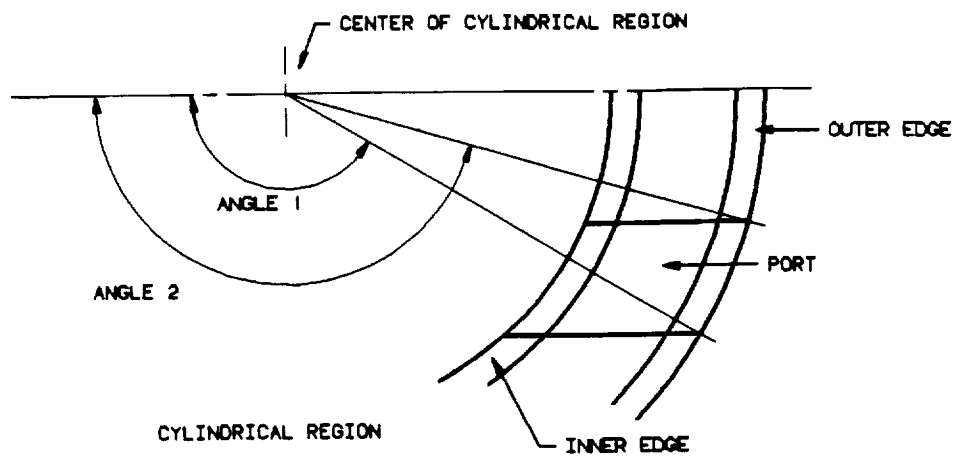


Figure 29. Angles Defining Radius of the Intake or Exhaust Port

4.2 Input Data File

The user has the choice of either answering the prompts as they appear on the screen or assembling a data file with all the input. If the user chooses to answer the prompts as they appear on the screen, then a data file of these responses will be created. This data file can be used on later executions of the SP. If the data file of input exists, then the prompts will not appear on the screen; the input needed for the SP will be read from the data file.

CHAPTER 5: Conclusions

The SP will automatically generate a FEM from a minimal number of user supplied inputs that describe a particular RCE center housing geometry. The SP, significantly reducing the many hours it takes to create a FEM, makes the design process and analysis of alternate housing designs more cost effective. Figure 30 and Figure 31 illustrate two different FEM's created by the SP.

The SP consists of almost 10,000 lines of FORTRAN code. The two files - the UF and the PF - that the SP creates, contain a total of approximately 25,000 lines of data. The UF consists of approximately 20,000 lines of data that describes the center housing geometry. The PF consists of approximately 3000 lines of commands that guide the algorithms in IDEAS through the generation of the meshes nodes and elements.

It takes an IBM 4341 computer approximately seven hours to generate the FEM. This total execution time is significantly affected by the system utilization while the SP is being executed. A majority of the execution time - approximately eighty percent - is used to generate the nodes and elements. Approximately ten percent of the execution time is used to generate the meshes. The remaining approximate ten percent of the execution time is used to execute the SP, read the UF and PF, store files, and perform other miscellaneous tasks. The user does not have to be present during the execution time. He/she has only to input the parameters that define the housing geometry and the SP automatically generates the finite element model.

The SP is limited in one key area; the number and placement of the nodes and elements is not variable. Two steps can be taken to reduce this limitation. The step first involves using the option of exiting the SP at any point during its execution. The gives the user the freedom to exit the SP after the generation of the geometry so that the nodes

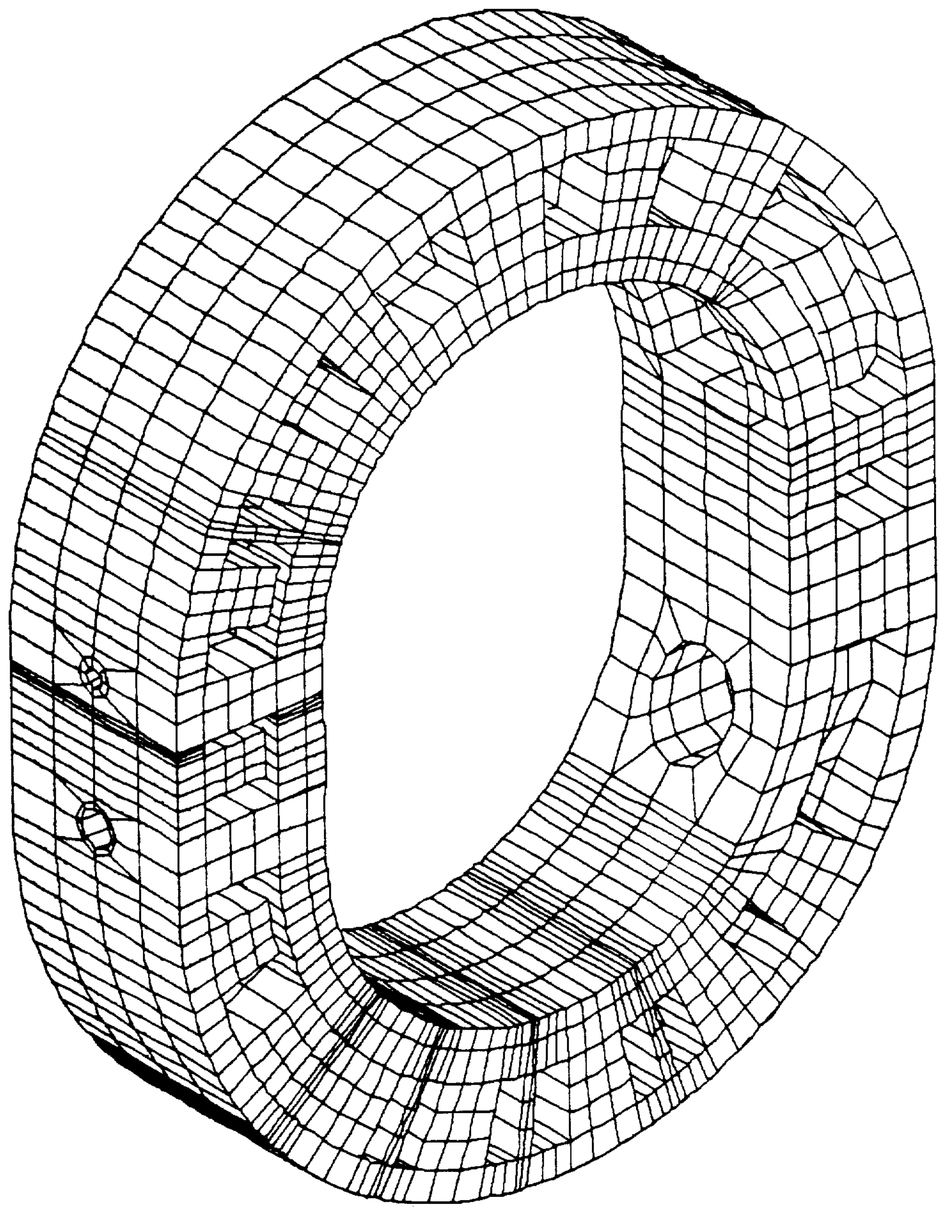


Figure 30. FEM Generated by the SP

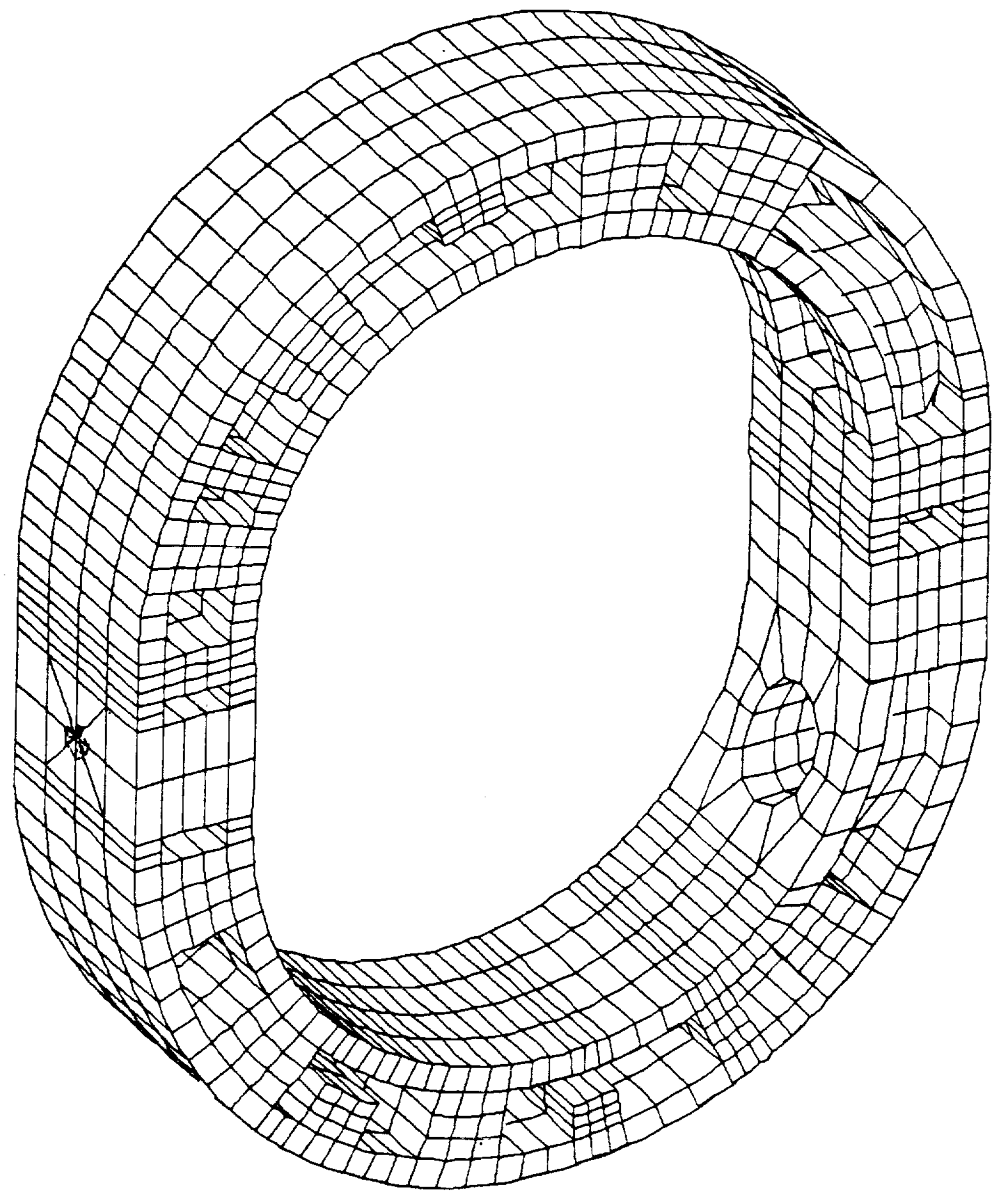


Figure 31. A second FEM Generated by the SP: Note that there is no bore insert and the location, size and type of the rib/channel segments is different.

and elements can be generated manually. The opportunity to exit the SP at any point in the finite element model creation also allows the user to verify that the SP generated the desired geometry before the time consuming generation of the nodes and elements begins.

The second and more practical step that can be used by the user to reduce the limitation of the SP is to use the "add element" capability offered in the IDEAS software. The "add element" capability allows the user to create elements from existing nodes. The user can use the SP to create a FEM and then use the "add element" capability to refine the finite element mesh in areas of interest.

References

1. Ansdale, A. F., *The Wankel RC Engine*, A. S. Barnes & Company, 1969.

2. Norbye, J. P., *The Wankel Engine: Design Development Applications*, Chilton Book Company, 1971.

3. Jones, C. & M. Berkwoitz, "Multifuel Rotary Aircraft Engine," SAE Paper No. 801237. Presented at AIAA/SAE/ASME Joint Propulsion Congress, Hartford, CT, July, 1980.

4. Matsuura, K., K. Terasaki & I. Watanabe, "The Behavior of a Rotary Apex Seal Against the Trochoid Surface," Bulletin of the JSME, Vol. 21 No. 161, November 1978.

5. Eberle, M. K. & E. D. Klomp, "An Evaluation of the Potential Performance Gain from Leakage Reduction in Rotary Engines," SAE Paper No. 730117. Presented at the International Automotive Engineering Congress, Detroit, MI, January 10-15, 1973.

6. Yamamoto, K., K. Kenichi & K. Takashi, "Toyo Kogyo's Research and Development on Major Rotary Engine Problems," SAE Paper No. 700079. Presented at the International Congress & Exposition, Detroit, MI, January 12-16, 1970.

7. Burley, H. A., M. R. Meloeny & T. L. Stark, "Sources of Hydrocarbon Emissions in Rotary Engines," SAE Paper No. 780419. Presented at the International Congress & Exposition, Detroit, MI, February 27 - March 3, 1978.

8. Yamamoto, K. & T. Mukoki, "Development on Exhaust Emissions & Fuel Economy of the Rotary Engine at Toyo Kogyo," SAE Paper No. 780417. Presented at the International Congress and Exposition, Detroit, MI, February 27 - March 3, 1978.

9. Muroki, T., "Recent Technology Development of High-Powered Rotary Engine at Mazda," SAE Paper No. 841017. Presented at West Coast International Meeting & Exposition, San Diego, CA, August 6-9, 1984.

10. Jones, C. "An Update of Applicable Automative Engine Rotary Stratified Charge Developments," SAE Paper No. 820347. Presented at the International Congress & Expostiiton, Detroit, MI, February 22-26, 1982.

11. Bradley, S. "Finite Element Model Development of a Rotary Combustion Engine Center Housing." A thesis presented at Michigan Technological University, Houghton, MI, February, 1985.

12. Krouse, J. K., "Reducing the Labor content in Finite Element Analysis," *Machine Design*, Vol. 19, No. 9, September 8, 1983, pp.64-69.

13. Bennett, J. A. & M. F. Nelson, "An Optimization Capability for Automotive Structures," SAE Paper No. 790972. 1979.

14. "User Manual for Supertab," Structural Dynamics Research Corporation, Vol 1, Level 2, 1983.

15. "Supertab User Manual Supplment," Structural Dynamics Research Corporation, Vol 1, Level 2, 1984.

Appendix 1: Example of a Universally Formatted Data File

This universal file generates the cube pictured in Figure 32

nodes

-1
15

label	rectangular coordinate system		color	coordinates		
1	0	0	11	-1.00E+00	-1.00E+00	0.00E+00
2	0	0	11	-9.99E-01	-9.99E-01	4.99E-01
3	0	0	11	-9.99E-01	-9.99E-01	9.99E-01
4	0	0	11	-9.99E-01	-9.99E-01	1.50E+00
5	0	0	11	-9.99E-01	-9.99E-01	2.00E+00
6	0	0	11	-9.99E-01	-9.99E-01	2.50E+00
			⋮			
688	0	0	11	9.99E-01	9.99E-01	7.50E-01
689	0	0	11	9.99E-01	9.99E-01	1.00E+00
690	0	0	11	9.99E-01	9.99E-01	1.50E+00
691	0	0	11	9.99E-01	9.99E-01	2.00E+00
692	0	0	11	9.99E-01	9.99E-01	2.50E+00
693	0	0	11	9.99E-01	9.99E-01	3.00E+00

-1

points

-1
25

label	rectangular coordinate system	color	coordinates		
1	0	7	-1.000E+00	-1.000E+00	0.000E+00
2	0	7	1.000E+00	-1.000E+00	0.000E+00
3	0	7	1.000E+00	1.000E+00	0.000E+00
4	0	7	-1.000E+00	1.000E+00	0.000E+00
5	0	7	-1.000E+00	-1.000E+00	3.000E+00
6	0	7	1.000E+00	-1.000E+00	3.000E+00
7	0	7	1.000E+00	1.000E+00	3.000E+00
8	0	7	-1.000E+00	1.000E+00	3.000E+00

-1

lines

-1
26

label	color	line style (solid)	point label start	point label stop
1	8	1	5	6
2	8	1	6	7
3	8	1	7	8
4	8	1	8	5
5	8	1	5	1
6	8	1	6	2
7	8	1	7	3
8	8	1	8	4

	9	8	1	1	2
	10	8	1	2	3
	11	8	1	3	4
	12	8	1	4	1
-1					

	label	color	number of entities deining edge	entity type 1 line 2 arc 3 spline
-1				
29				
	1	9	1	1
	1	1		
	2	9	1	1
	1	2		
	3	9	1	1
	1	3		
	4	9	1	1
	1	4		
	5	9	1	1
	1	9		
	6	9	1	1
edges	1	10		
	7	9	1	1
	1	11		
	8	9	1	1
	1	12		
	9	9	1	1
	1	5		
	10	9	1	1
	1	6		
	11	9	1	1
	1	7		
	12	9	1	1
	1	8		
-1				

	label	color	line style (dashed)		number of edges defining surface	
-1						
30						
	1	4	2	4	4	
	4	1	2	3		
	2	4	2	4	4	edge labels
	9	1	10	5		
	3	4	2	4	4	
surfaces	10	2	11	6		
	4	4	2	4	4	
	11	7	12	3		
	5	4	2	4	4	
	12	8	9	4		
	6	4	2	4	4	
	5	6	7	8		

-1

		label	color	line style (dashed)		number of surfaces defining volume	
	-1						
	39						
volumes		1	5	2	2	6	
		2	5	1	3	6	surface labels
	-1						

	label		element type parabolic solid			color			
	-1								
	71								
	1	20	116	1	1	7	20		
	1	2	3	10	17	16	15	8	
	64	66	80	78	127	128	129	136	
	143	142	141	134					node labels
	2	20	116	1	1	7	20		
	3	4	5	12	19	18	17	10	
	66	68	82	80	129	130	131	138	
	145	144	143	136					
	3	20	116	1	1	7	20		
	5	6	7	14	21	20	19	12	
	68	70	84	82	131	132	133	140	
	147	146	145	138					
element connectivity				⋮					
	58	20	116	1	1	7	20		
	547	548	549	556	563	562	561	554	
	610	612	626	624	673	674	675	682	
	689	688	687	680					
	59	20	116	1	1	7	20		
	549	550	551	558	565	564	563	556	
	612	614	628	626	675	676	677	684	
	691	690	689	682					
	60	20	116	1	1	7	20		
	551	552	553	560	567	566	565	558	
	614	616	630	628	677	678	679	686	
	693	692	691	684					
	-1								

		label	element type parabolic solid		physical property	material property
	-1					
	74					
		1	116	1	1	1
		2	116	2	1	1
		3	116	3	1	1
elements			⋮			
		58	116	58	1	1
		59	116	59	1	1
		60	116	60	1	1
	-1					

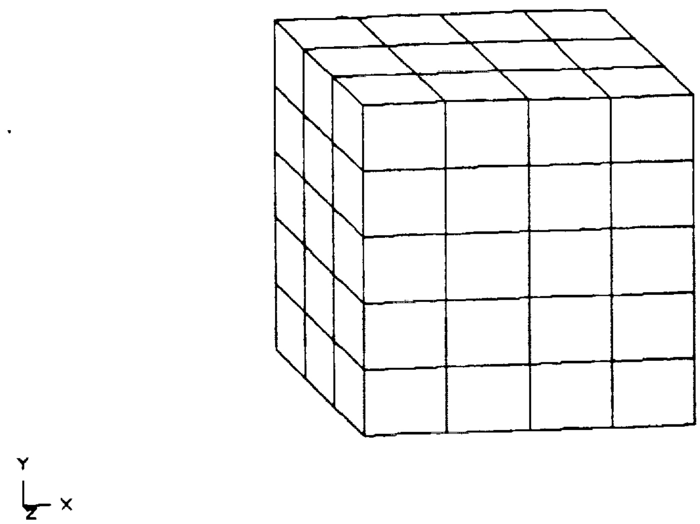


Figure 32. Example Model Generated by the Universal File

Note that although there are nodes and elements defined in universal format, there are no meshes. The mesh data is used to create the nodes and elements. Once the nodes and elements are created, the mesh serves no purpose so it is not stored in universal format.

Note also, that the original universally formatted file contained none of the annotative comments. These comments have been added to give the reader a better understanding of the format.

Appendix 2: Example of an Input Data File

This appendix is an example data file that creates a finite element model of the axially cooled rotary combustion engine center housing that was sent to Michigan Technological University by the NASA/ Lewis Research Center.

5.95,3.80	.....	maximum and minimum diameters
1.00	....	thickness of housing
2.68	....	depth of housing
0.375	....	thickness of outer shell
0.375	....	thickness of inner shell
0.25	....	thickness of bore insert
3	....	overlappin region
-5.0,2.0	......	phi 1 and phi 2 of rib edges
1	....	normal rib
2	....	stiffened channel
0.75	....	thickness of first channel
1	....	cylindrical region
47.5,60.0	.....	phi 1 and phi 2 of rib edges
1	....	normal rib
4	....	intake port channel
2.0,23.0	......	angles defining radius of port
0.275	....	wall thickness of port
1.50,1.50	.....	length and width of port
1	....	cylindrical region
70.0,80.0	.....	phi 1 and phi 2 of rib edges
1	....	normal rib
2	....	stiffened channel
1	....	cylindrical region
91.0,105.0	....	phi 1 and phi 2 of rib edges
1	....	normal rib
2	....	stiffened channel
1	....	cylindrical region
120.0,130.0	...	phi 1 and phi 2 of rib edges
1	....	normal rib

2	stiffened channel
1	cylindrical region
140.0,150.0	phi 1 and phi 2 of rib edges
1	normal rib
2	stiffened channel
1	cylindrical region
154.0,156.0	phi 1 and phi 2 of rib edges
1	normal rib
2	stiffened channel
1	cylindrcial region
160.0,169.0	phi 1 and phi 2 of rib edges
2	recessed rib
0.175	magnitude of recess
3	overlapping region
176.0,184.0	phi 1 and phi 2 of rib edges
3	rib is spark plug port
0.125	radius of port
2	rectangular region
0.60,0.05	y coordinate 1 and y coordinate 2 of rib edges
2	recessed rib
0.175	magnitude of recess
2	rectangular region
-0.33,-1.10	y coordinate 1 and y coordinate 2 of rib edges
3	rib is spark plug port
0.25	radius of port
3	overlapping region
185.0,195.0	phi 1 and phi 2 of rib edges
2	recessed rib
0.175	magnitude of recess
1	rectangular region
205.0,215.0	phi 1 and phi 2 of rib edges
1	normal rib
1	cylindrical region
218.0,220.0	phi 1 and phi2 of rib edges
1	normal region
1	normal channel
1	cylindrical region
223.0,235.0	phi 1 and phi 2 of rib edges
1	normal rib
1	normal channel
1	cylindrical region
239.0,241.0	phi 1 and phi 2 of rib edges

1	....	normal rib
1	....	normal channel
1	....	cylindrical region
245.0,257.0	....	phi 1 oand phi 2 of rib edges
1	....	normal rib
1	....	normal channel
1	....	cylindrical region
262.5,264.5	...	phi 1 and phi 2 of rib edges
1	....	normal rib
1	....	normal channel
1	....	cylindrical region
271.0,282.0	...	phi 1 and phi 2 of rib edges
1	....	normal rib
1	....	normal channel
1	....	cylindrical region
287.0,289.0	...	phi 1 and phi 2 of rib edges
1	....	normal rib
1	....	normal channel
1	....	cylindrical region
294.0,305.0	...	phi 1 and phi 2 of rib edges
1	....	normal rib
1	....	normal channel
1	....	cylindrcial region
312.0,326.0	...	phi 1 and phi 2 of rib edges
1	....	normal rib
1	....	normal channel
3	....	overlapping region
356.0,365.0	...	phi 1 and phi 2 of rib edges
1	....	normal rib
3	....	channel is exhaust port
340.0,354.0	...	angles defining radius of port
0.275	....	wall thickness of port
4	....	end prompted input
1	...	entities to edges generated - continue model generation
1	...	entities to meshes generated - continue model generation

Appendix 3: Examples of Numbering Patterns

The examples of the numbering patterns contained in this appendix are used in the SP to generate the geometric entities - splines, edges, surfaces, and and volumes - that are used to define a normal rib / normal channel segment.

APPENDIX 3.1: Numbering Pattern of the Splines

spline label	point labels			
	pnt 1	pnt 2	pnt 3	pnt 4
1	x	1	5	9
2	x	2	6	10
3	x	3	7	11
4	x	4	8	12
5	x	25	29	33
6	x	26	30	34
7	x	27	31	35
8	x	28	32	36
9	9	13	17	21
10	10	14	18	22
11	11	15	19	23
12	12	16	20	24
13	33	37	41	45
14	34	38	42	46
15	35	39	43	47
16	36	40	44	48

Figure 33. Numbering Pattern of the Splines: this pattern creates a normal rib / normal channel segment

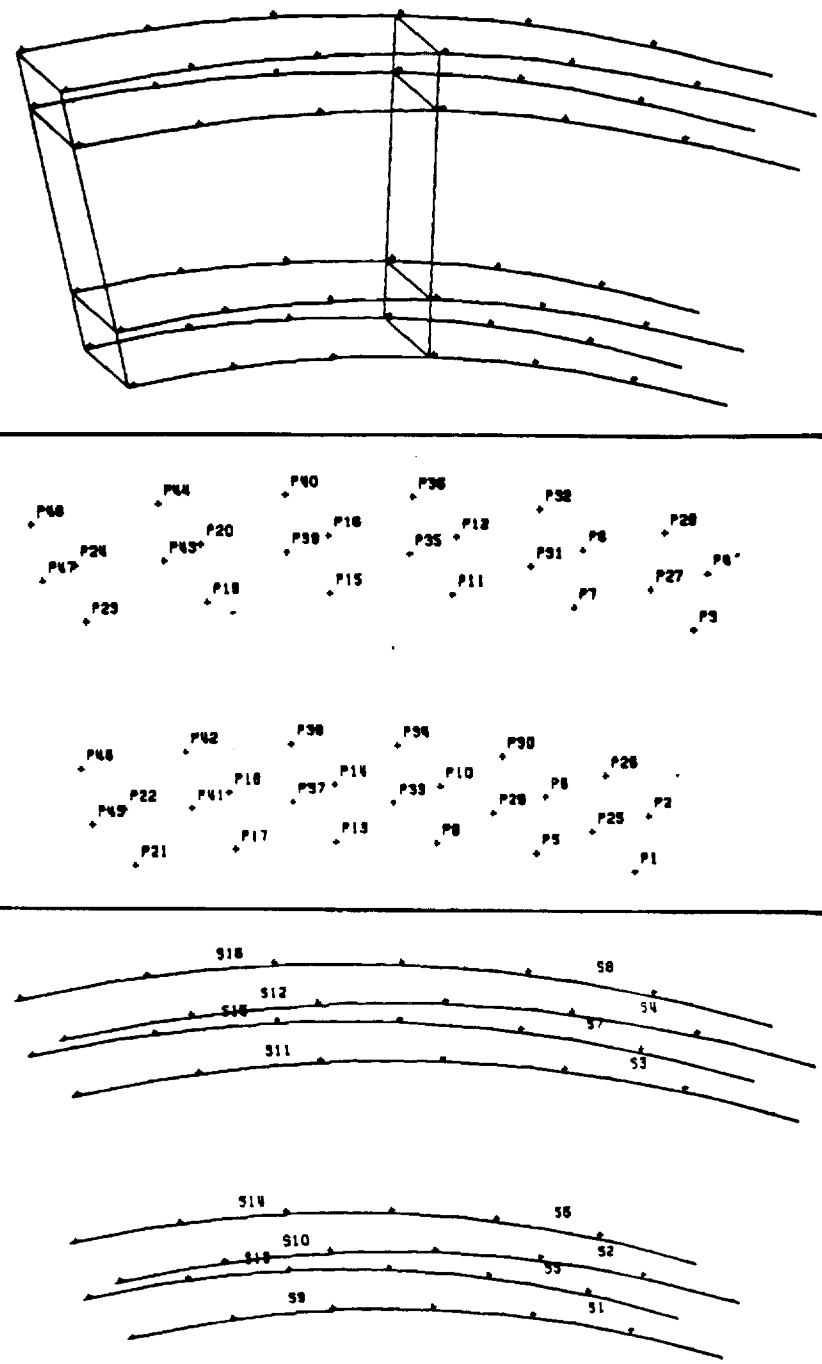


Figure 34. Illustration of the Numbering Pattern of the Splines

APPENDIX 3.2: Numbering Pattern of the Edges

edge label	line & spline labels
1	spline 1
2	spline 2
3	spline 3
4	spline 4
5	spline 5
6	spline 6
7	spline 7
8	spline 8
9	line 1
10	line 2
11	line 3
12	line 4
13	line 5
14	line 6
15	line 7
16	line 8

17	spline 9
18	spline 10
19	spline 11
20	spline 12
21	spline 13
22	spline 14
23	spline 15
24	spline 16
25	line 9
26	line 10
27	line 11
28	line 12
29	line 13
30	line 14
31	line 15
32	line 16

Figure 35. Numbering Pattern of the Edges: this pattern creates a normal rib : normal channel segment

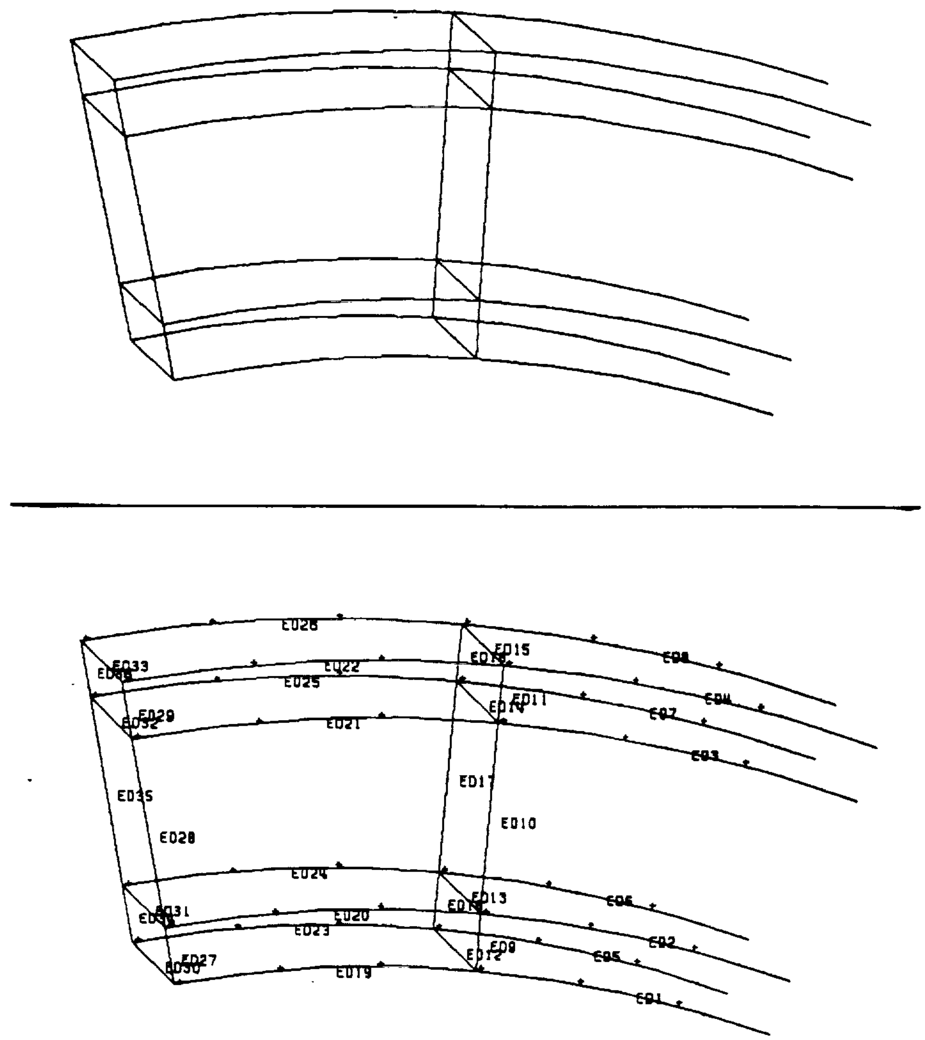


Figure 36. Illustration of the Numbering Pattern of the Edges

APPENDIX 3.3: Numbering Pattern of the Surfaces

surface label	edge labels			
	edg 1	edg 2	edg 3	edg 4
1	1	x	2	9
2	3	x	4	11
3	1	x	5	12
4	2	x	6	13
5	3	x	7	14
6	4	x	8	15
7	5	x	6	16
8	7	x	8	18
9	12	9	13	16
10	13	10	14	17
11	14	11	15	18
12	19	9	20	27
13	20	10	21	28
14	21	11	22	29
15	19	12	23	30
16	20	13	24	31
17	21	14	25	32
18	22	15	26	33
19	23	16	24	34
20	24	17	25	35
21	25	18	26	36
22	30	27	31	34
23	31	28	32	35
24	32	29	33	36

Figure 37. Numbering Pattern of the Surfaces: this pattern creates a normal rib , normal channel segment

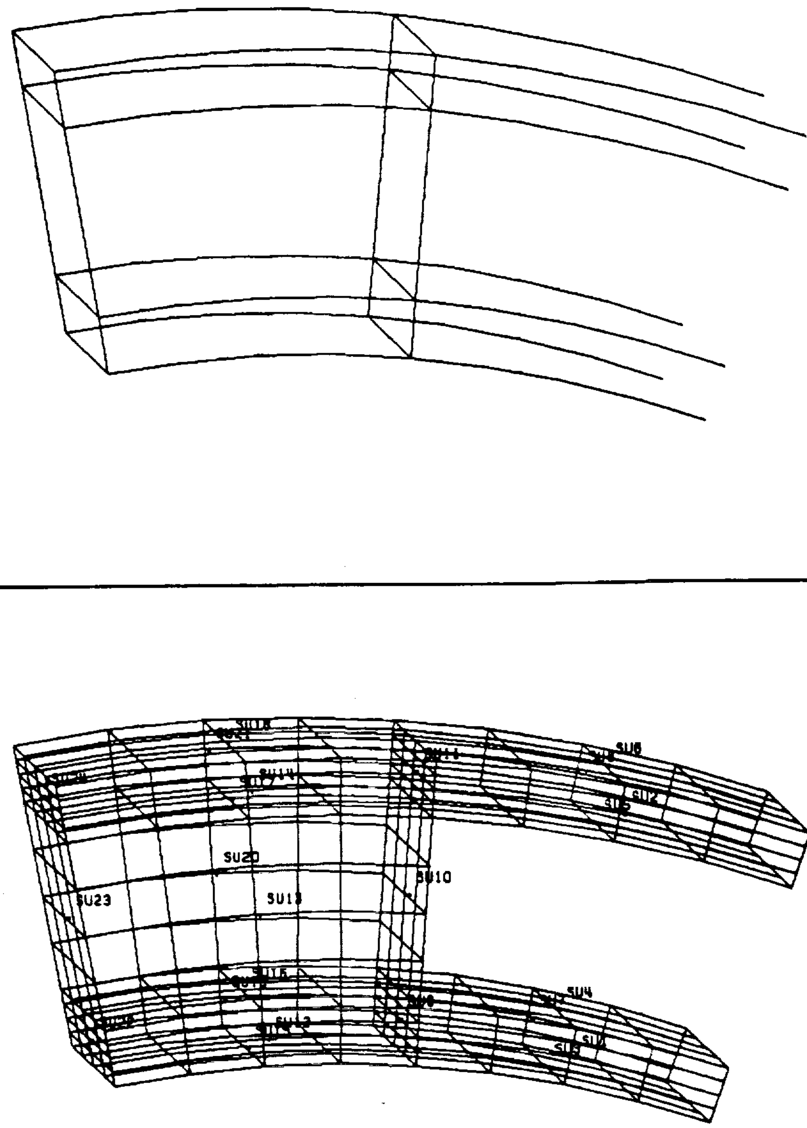


Figure 38. Illustration of the Numbering Pattern of the Surfaces

APPENDIX 3.4: Numbering Pattern of the Volumes

volume label	surface labels					
	sur 1	sur 2	sur 3	sur 4	sur 5	sur 6
1	3	1	x	7	9	4
2	5	2	x	8	11	6
3	15	12	9	19	22	16
4	16	13	10	20	23	17
5	17	14	11	21	24	18

Figure 39. Numbering Pattern of the Volumes: this pattern creates a normal rib / normal channel segment

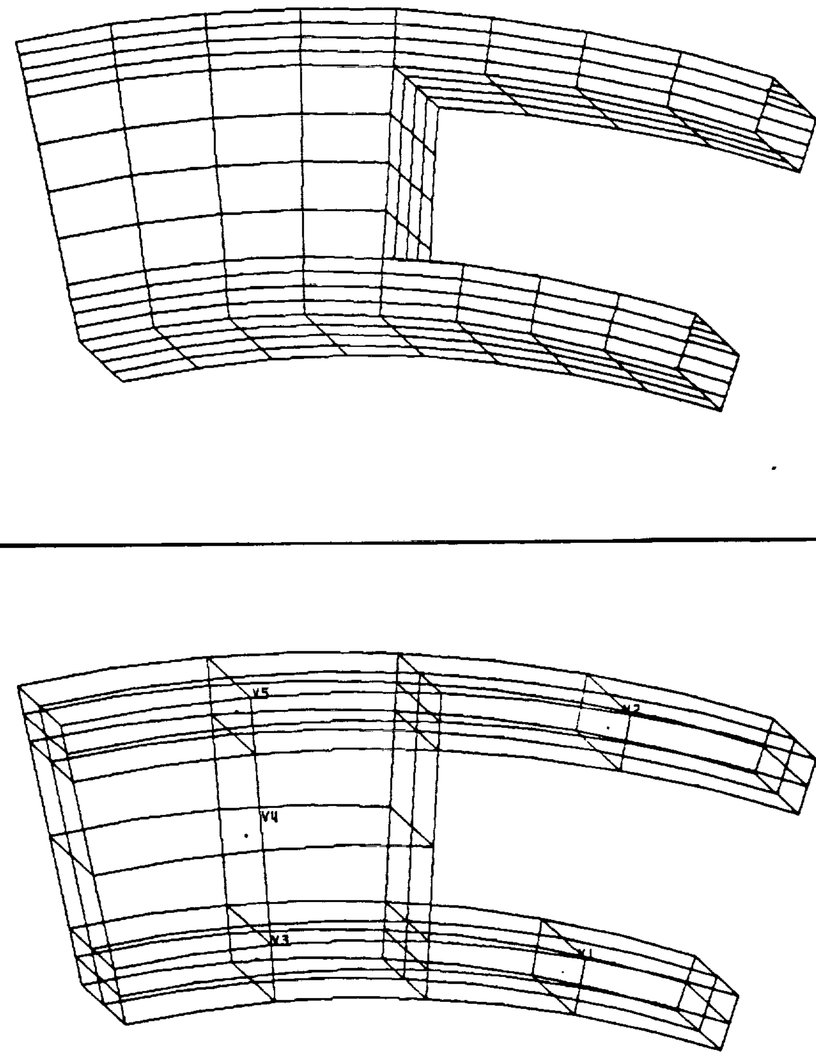


Figure 40. Illustration of the Numbering Pattern of the Volumes

Appendix

C

```
C     ************************************************************
C     *                                                          *
C     *    THIS PROGRAM GENERATES FINITE ELEMENT MODELS OF       *
C     *    USER DEFINED  ROTARY COMBUSTION ENGINE CENTER         *
C     *    HOUSING GEOMETRIES.  THE USER IS PROMPTED FOR THE     *
C     *    INPUT DATA.  THIS PROGRAM WILL OUTPUT SEVERAL FILES.  *
C     *    THE FILE CONTAINING THE DATA THE DESCRIBES THE GEOM-  *
C     *    TRY IS CALLED "GEOMETRY".   THE FILES CONTAINING THE  *
C     *    COMMANDS THAT GENERATE THE MESHES, NODES AND ELEMENTS *
C     *    IS CALLED "MSNDEL".                                  *
C     *                                                          *
C     ************************************************************

C     ************************************************************
C     *                                                          *
C     *                 VARIABLE DEFINITION                      *
C     *                                                          *
C     *    THE VARIABLES THAT BEGIN WITH THE SAME FIRST THREE    *
C     *    LETTERS ARE THE POINT COORDINATES OF THE BORE.        *
C     *    INSERT (AAA - GGG).  THE VARIABLES THAT BEGIN WITH    *
C     *    THE SAME FIRST TWO LETTERS ARE THE POINT COORDINATES  *
C     *    OF THE INNER EDGE OF THE INNER SHELL (AA - GG).  THE  *
C     *    VARIABLES THE BEGIN WITH THE LETTERS (A -G) ARE THE   *
C     *    POINT COORDINATES OF THE OUTER EDGE OF THE INNER      *
C     *    SHELL.  THE VARIABLES THAT BEGIN WITH THE LETTERS     *
C     *    (H - N) ARE THE POINT COORDINATES OF THE INNER EDGE   *
C     *    OF THE OUTER SHELL.  THE VARIABLES THAT BEGIN WITH    *
C     *    THE LETTERS (HH-NN) ARE THE POINT COORDIANTES OF THE  *
C     *    OUTER EDGE OF THE OUTER SHELL.                        *
C     *                                                          *
C     ************************************************************


      DIMENSION AIX(100),AIY(100),BIX(100),BIY(100),CIX(100),CIY(100),
     # DIX(100),DIY(100),EIX(100),EIY(100),FIX(100),FIY(100),GIX(100),
     # GIY(100),HIX(100),HIY(100),IIX(100),IIY(100),JIX(100),JIY(100),
     # KIX(100),KIY(100),LIX(100),LIY(100),MIX(100),MIY(100),NIX(100),
     # NIY(100),AAIX(100),AAIY(100),BBIX(100),BBIY(100),
     # CCIX(100),CCIY(100),DDIX(100),DDIY(100),EEIX(100),EEIY(100),
     # FFIX(100),FFIY(100),GGIX(100),GGIY(100),HHIX(100),HHIY(100),
     # IIIX(100),IIIY(100),JJIX(100),JJIY(100),KKIX(100),KKIY(100),
     # LLIX(100),LLIY(100),MMIX(100),MMIY(1000),NNIX(1000),NNIY(100),
     # DRIX(100),DRIY(100),ERIX(100),ERIY(100),FRIX(100),FRIY(100),
     # GRIX(100),GRIY(100),REGION(100),RSP(10),PHI1SP(10),PHI2SP(10),
     #AAAIX(100),AAAIY(100),
     #BBBIX(100),BBBIY(100),CCCIX(100),CCCIY(100),DDDIX(100),DDDIY(100),
     #EEEIX(100),EEEIY(100),FFFIX(100),FFFIY(100),GGGIX(100),GGGIY(100),
     #Y1SP(10),Y2SP(10),ISPRK(10),PHI1(1000),XX(1000),ZZZ(100),RZZZ(100)

C     ************************************************************
C     *                                                          *
C     *    THIS COMMON BLOCK CONTAINS THE POINT COORDINATES      *
C     *    OF THE INTAKE, EXHAUST, AND SPARK PLUG PORTS.         *
C     *                                                          *
C     ************************************************************

      COMMON / PORT / X1(7),Y1(7),Z1(7),X2(7),Y2(7),Z2(7),X3(7),Y3(7),
     #   Z3(7),X4(7),Y4(7),Z4(7),X5(7),Y5(7),Z5(7),X6(7),Y6(7),Z6(7),
     #   X7(7),Y7(7),Z7(7),X8(7),Y8(7),Z8(7),X9(7),Y9(7),Z9(7),
     #   X10(7),Y10(7),Z10(7),X11(7),Y11(7),Z11(7),X12(7),Y12(7),Z12(7),
     #   X13(7),Y13(7),Z13(7),X14(7),Y14(7),Z14(7),X15(7),Y15(7),Z15(7),
     #   X16(7),Y16(7),Z16(7),X17(7),Y17(7),Z17(7),X18(7),Y18(7),Z18(7),
     #   X19(7),Y19(7),Z19(7),X20(7),Y20(7),Z20(7),X21(7),Y21(7),Z21(7),
     #   X22(7),Y22(7),Z22(7),X23(7),Y23(7),Z23(7),X24(7),Y24(7),Z24(7),
```

```
     #    X25(7),Y25(7),Z25(7),X26(7),Y26(7),Z26(7),X27(7),Y27(7),Z27(7),
     #    X28(7),Y28(7),Z28(7),X29(7),Y29(7),Z29(7),X30(7),Y30(7),Z30(7),
     #    X31(7),Y31(7),Z31(7),X32(7),Y32(7),Z32(7),X33(7),Y33(7),Z33(7),
     #    X34(7),Y34(7),Z34(7),X35(7),Y35(7),Z35(7),X36(7),Y36(7),Z36(7),
     #    X37(7),Y37(7),Z37(7),X38(7),Y38(7),Z38(7),X39(7),Y39(7),Z39(7),
     #    X40(7),Y40(7),Z40(7),X41(7),Y41(7),Z41(7),X42(7),Y42(7),Z42(7),
     #    X43(7),Y43(7),Z43(7),X44(7),Y44(7),Z44(7),X45(7),Y45(7),Z45(7),
     #    X46(7),Y46(7),Z46(7),X47(7),Y47(7),Z47(7),X48(7),Y48(7),Z48(7),
     #    X49(7),Y49(7),Z49(7),X50(7),Y50(7),Z50(7),X51(7),Y51(7),Z51(7),
     #    X52(7),Y52(7),Z52(7),X53(7),Y53(7),Z53(7)

      COMMON / MAIN2 / CNLTYP(100),RIBTYP(100),NPTEP(7),NPTIP(7)
     #   ,NPTSP(7),NPTTL,ISP,IVOLR(25),IVOLC(25),TCHNL,CCC

      COMMON / SUB / ICONTI

      CHARACTER STRING*40,FILE1*40,NAME*40
C     ***************************************************************
C     *                                                             *
C     *     THIS INTEGER STATEMENT CONVERTS THESE REAL VARI-        *
C     *     ABLES INTO INTEGERS SO THAT THEY CAN BE USED AS         *
C     *                      COUNTERS                               *
C     *                                                             *
C     ***************************************************************

      INTEGER PT1,PT2,PT3,PT4,RIBTYP,CNLTYP,REGION,SDC,
     #        RGNCTR,SOLID,COLOR,START,STOP,COUNT,SPLINE,
     #        SPENTC,LNENTC,DASH,GRADE,SURFAC,ED1,ED2,ED3,ED4,
     #        VOLUME,SUR1,SUR2,SUR3,SUR4,SUR5,SUR6,PHCHCK,SLSLVR,
     #        SURT,SURB,EDGE,PNTT,PNTB,STATUS
C     ***************************************************************
C     *                                                             *
C     *     THIS REAL STATEMENT CONVERTS THESE INTEGER VARI-        *
C     *     ABLES INTO REALS SO THAT THEY CAN BE USED TO AS         *
C     *          ARRAYS TO HOLD THE POINT COORDINATES               *
C     *                                                             *
C     ***************************************************************

      REAL    IIX,IIY,IIIX,IIIY,JIX,JIY,JJIX,JJIY,KIX,KIY,
     #        KKIX,KKIY,LIX,LIY,LLIX,LLIY,MIX,MIY,MMIX,MMIY,
     #        NIX,NIY,NNIX,NNIY,LINPT
C     ***************************************************************
C     *                                                             *
C     *     THESE FUNCTION STATEMENTS CALCUALTE RECTANGULAR         *
C     *       POINT COORDINATES GIVEN TEH REQUIRED ANGLES           *
C     *                                                             *
C     ***************************************************************

      XXX(Z,BETA)=Z*COS(BETA)

      YYY(ZZ,BETA)=ZZ*SIN(BETA)

      FNX(GAMMA)=EE*COS(3.0*GAMMA)+RR*COS(GAMMA)

      FNY(GAMMA)=EE*SIN(3.0*GAMMA)+RR*SIN(GAMMA)

      FINERX(GAMMA,RLAMB,FF)=EE*COS(3.0*GAMMA)+RR*COS(GAMMA)+
     #             FF*COS(GAMMA+RLAMB)

      FINERY(GAMMA,RLAMB,FF)=EE*SIN(3.0*GAMMA)+RR*SIN(GAMMA)+
     #             FF*SIN(GAMMA+RLAMB)
```

```
C     **************************************************************
C     *                                                            *
C     *                END FUNCTION STATEMENTS                     *
C     *                                                            *
C     **************************************************************

C     **************************************************************
C     *                                                            *
C     *    CALL THE SUBROUTINE THAT CHECK FOR THE EXISTANCE        *
C     *          OF THE FILE "GENERATE DATA A"                      *
C     *                                                            *
C     *    IF THE FILE EXISTS, THEN READ INPUTS FOR THE PRE-       *
C     *    PROCESSOR FROM THAT FILE.  IF THE FILE DOES NOT         *
C     *       EXIST, THEN CREATE IT AND STORE THE USER'S           *
C     *             RESPONSES IN IT FOR LATER USE                  *
C     *                                                            *
C     **************************************************************

      NO=6
      CALL NEWFIL (STATUS,NO)

      IF (STATUS.EQ.0) THEN

       PRINT*,' '
       PRINT*, '  THE FILE "GENERATE DATA A" WAS FOUND ON YOUR DISK.'
       PRINT*, '  THEREFORE, YOU WILL NOT BE PROMPTED FOR THE INPUTS.'
       PRINT*, ' THE INPUTS WILL BE READ DIRECTLY FROM "GENERATE DATA".'
       PRINT*, '  IF YOU WANT TO BE PROMPTED FOR THE INPUTS, THEN '
       PRINT*, '  HALT THE EXECUTION OF THIS PROGRAM AND ERASE THE'
       PRINT*, '       FILE "GENERATE DATA" FROM YOUR DISK.'
       PRINT*,' '

      STRING='FILEDEF 5 DISK GENERATE DATA A '
      STATUS=CMSCMD(STRING)

             IF(STATUS.NE.0)THEN
               PRINT*,' FILEDEF ERROR - FILEDEF ON UNIT 5 FAILED'
               PRINT*,' RETURN CODE = ',STATUS
             ENDIF

      NO=7

      NAME='FILEDEF 7 DISK STORAGE DATA A '
      STATUS=CMSCMD(NAME)

             IF(STATUS.NE.0)THEN
               PRINT*,' FILEDEF ERROR - FILEDEF ON UNIT 7 FAILED'
               PRINT*,' RETURN CODE = ',STATUS
             ENDIF

      ENDIF

C     **************************************************************
C     *                                                            *
C     *                BEGIN PROMPTED INPUT                        *
C     *                                                            *
C     **************************************************************

C     **************************************************************
C     *                                                            *
C     *          ASK FOR THE HOUSING PARAMETERS                    *
C     *                                                            *
C     **************************************************************
```

```
      WRITE(NO,*) ' '
      WRITE(NO,*) ' '
      WRITE(NO,7)
 7    FORMAT('0','INPUT THE ECCENTRICITY AND RADIUS OF THE ENGINE HOUSIN
     #G. (EE AND RR)')
      WRITE(NO,8)
 8    FORMAT('0','      OR.....  ')
      WRITE(NO,9)
 9    FORMAT('0','INPUT THE MAX. X AND MAX. Y DIAMETER.')

      READ(5,*) TEMP1,TEMP2
      IF (STATUS.NE.0) WRITE(8,*) TEMP1,TEMP2
      WRITE(NO,*) TEMP1,TEMP2

         IF(TEMP1.LT.2.0) THEN
               EE=-TEMP1
               RR=TEMP2
         ELSE
               RR=(TEMP1+TEMP2)/4.0
               EE=-(TEMP2-TEMP1)/4.0
         ENDIF
          WRITE(NO,*) EE,RR

      WRITE(NO,10)
 10   FORMAT('0','ENTER HOUSING THICKNESS. (D)')
      READ(5,*) D
      IF(STATUS.NE.0) WRITE(8,*) D
      WRITE(NO,*) D

      WRITE(NO,11)
 11   FORMAT('0','INPUT THE DEPTH OF THE HOUSING. (DEPTH)')
      READ(5,*) DEPTH
      IF (STATUS.NE.0) WRITE(8,*) DEPTH
      WRITE(NO,*) DEPTH
          DEPTH=-DEPTH

      WRITE(NO,13)
 13   FORMAT('0','ENTER THE THICKNESS OF THE OUTER SHELL. (AA)')
      READ(5,*) AA
      IF (STATUS.NE.0) WRITE(8,*) AA
      WRITE(NO,*) AA

      WRITE(NO,14)
 14   FORMAT('0','ENTER THE THICKNESS OF THE INNER SHELL. (BB)')
      READ(5,*) BB
      IF (STATUS.NE.0) WRITE(8,*) BB
      WRITE(NO,*)BB
      BB=-BB

      WRITE(NO,15)
 15   FORMAT('0','ENTER THE THICKNESS OF THE INSERT. (CC)')
      WRITE(NO,16)
 16   FORMAT('0','IF THIS MODEL IS NOT TO HAVE AN INSERT, ENTER "0.0" FO
     #R (CC).')
      READ(5,*) CC
      IF (STATUS.NE.0) WRITE(8,*) CC
      WRITE(NO,*) CC
      CCC=CC
      CC=-CC
      CC=BB+CC

         PI=3.14159266
```

```
C     **************************************************************
C     *                                                            *
C     *    CALCULATE THE MAXIMUM X COORDINATE OF THE TROCHOIDAL    *
C     *    BORE AND THE DISTANCE FROM THE CENTER OF THE CENTER     *
C     *    HOUSING TO THE CENTER OF THE CYLINDRICAL REGIONS        *
C     *                                                            *
C     **************************************************************

               ZETA=0.0
               XX(1)=RR+EE

          DO 25 JJ=2,900

          XX(JJ)=ABS(FNX(ZETA))
          IF(XX(JJ-1)-XX(JJ)) 20,20,18
   18     XX(JJ)=XX(JJ-1)
   20     ZETA=ZETA+0.10*PI/180.0
   25     CONTINUE


               YMAX=RR-EE
               XMAX=XX(JJ-1)
               TRANS=YMAX-XMAX
               WRITE(NO,*) 'TRANS = ',TRANS
               TRANSB=TRANS
               TRANSC=TRANS
               TRANSD=TRANS
               TRANSE=TRANS
               TRANSF=TRANS
               TRANSG=TRANS

C     **************************************************************
C     *                                                            *
C     *                    INITIALIZE VALUES                       *
C     *                                                            *
C     **************************************************************

        NN=1
        SCALE=0.1E01
        ICS=0
        ICT=0
        ICTA=0
        ITOTAL=0
        PHCHCK=0
        SLSLVR=0
        COLOR=8
        SOLID=1
        SDC=1
        ISP=0
        IL=1
        IEPQUE=0
        IIPQUE=0
        IPRTCK=0
        EPSI=0.001
        DUMMY=1.0
        SIGN=1.0
        RGNCTR=0
        ICT1=0
        ICT2=0
        ICT3=0
        ICT4=0
        R=RR-EE+D-TRANS
        T=RR-EE+D-TRANS+AA
        XMAX=RR-EE-TRANS
```

```
C     ************************************************************
C     *                                                          *
C     *              END INITIALIZTION OF VALUES                 *
C     *                                                          *
C     ************************************************************

            WRITE(13,29) PI,R,TRANS
 29         FORMAT(F13.5,2X,F13.5,2X,F13.5)

       DO 130 J=1,200

          IT=1
          IVOLC(J)=0
          IVOLR(J)=0

C     ************************************************************
C     *                                                          *
C     *              ASK FOR THE REGION TYPE                     *
C     *                                                          *
C     ************************************************************

            WRITE(NO,30)
 30    FORMAT('0','IN WHICH REGION DOES THE RIB-CHANNEL SEGMENT LIE?')
       WRITE(NO,32)
 32    FORMAT('0','   ENTER  1  FOR PHI-SOLVER REGION...')
       WRITE(NO,33)
 33    FORMAT('0','   ENTER  2  FOR STRAIGHT-LINE SOLVER REGION...')
       WRITE(NO,34)
 34    FORMAT('0','   ENTER  3  IF THE RC SEGMENT OVERLAPS THE REGIONS...
      #')
       WRITE(NO,36)
 36    FORMAT('0','OR ENTER  4  TO END INPUT.')

            READ(5,*) REGION(J)
            IF (STATUS.NE.0) WRITE(8,*) REGION(J)
            WRITE(NO,*) REGION(J)

       IF(REGION(J).EQ.4) GO TO 135

            WRITE(13,*) REGION(J)

       IF(REGION(J).EQ.3) THEN

            RGNCTR=RGNCTR+1

       IF(RGNCTR.EQ.1) THEN
            ICT1=J
       ELSE IF(RGNCTR.EQ.2) THEN
            ICT2=J
       ELSE IF (RGNCTR.EQ.3) THEN
            ICT3=J
       ELSE IF (RGNCTR.EQ.4) THEN
            ICT4=J

       ENDIF
       ENDIF

C     ************************************************************
C     *                                                          *
C     *   IF THE REGION IS CYLINDRICAL, THEN ASK FOR THE TWO     *
C     *   ANGLES THAT DEFINE THE LEADING AND TRAILING EDGE       *
C     *                 OF THE RIB                               *
C     *                                                          *
```

```
C     ***********************************************************

              PHI1OD=PHIONE
              PHI2OD=PHITWO
          IF(REGION(J).EQ.1) THEN
              WRITE(NO,40)
 40           FORMAT('0','ENTER A VALUE FOR PHIONE AND PHITWO (DEGREES).')
              READ(5,*) PHIONE,PHITWO
              IF (STATUS.NE.0) WRITE(8,*) PHIONE,PHITWO
              WRITE(NO,*) PHIONE,PHITWO
              WRITE(13,42) PHIONE,PHITWO
 42           FORMAT(F13.5,5X,F13.5)
              PHI1(J)=PHIONE

C     ***********************************************************
C     *                                                         *
C     *    IF THE REGION IS RECTANGULAR, THEN ASK FOR THE TWO   *
C     *    Y-COORDINATES THAT DEFINE THE LEADING ANB TRAILING   *
C     *                   EDGE OF THE RIB                       *
C     *                                                         *
C     ***********************************************************

          ELSE IF(REGION(J).EQ.2) THEN
              WRITE(NO,44)
 44           FORMAT('0','ENTER THE YONE AND YTWO COORDINATE OF THE RIB CEN
         #TER.')
              READ(5,*) YONE,YTWO
              IF (STATUS.NE.0) WRITE(8,*) YONE,YTWO
              WRITE(NO,*) YONE,YTWO
              WRITE(13,42) YONE,YTWO
              YI=YONE

C     ***********************************************************
C     *                                                         *
C     *    IF THE REGION IS OVERLAPPING, THEN ASK FOR THE TWO   *
C     *    ANGLES THAT DEFINE THE LEADING AND TRAILING EDGE     *
C     *                   OF THE RIB                            *
C     *                                                         *
C     ***********************************************************

          ELSE IF (REGION(J).EQ.3) THEN

              WRITE(NO,55)
 55           FORMAT('0','ENTER A VALUE FOR PHIONE AND PHITWO (DEGREES).')
              READ(5,*) PHIONE,PHITWO
              IF (STATUS.NE.0) WRITE(8,*) PHIONE,PHITWO
              WRITE(NO,*) PHIONE,PHITWO
              WRITE(13,42) PHIONE,PHITWO
              PHI1(J)=PHIONE
          ENDIF

C     ***********************************************************
C     *                                                         *
C     *                 ASK FOR THE RIB TYPE                    *
C     *                                                         *
C     ***********************************************************

              WRITE(NO,60)
 60           FORMAT('0','WHAT TYPE OF RIB IS TO BE INPUT?')
              WRITE(NO,62)
 62           FORMAT('0','    ENTER  1  FOR NORMAL RIB...')
              WRITE(NO,64)
 64           FORMAT('0','    ENTER  2  FOR RECESSED RIB...')
              WRITE(NO,66)
```

```
 66       FORMAT('0','OR ENTER  3  IF THIS RIB IS A SPARK PLUG.')
          READ(5,*) RIBTYP(J)
          IF (STATUS.NE.0) WRITE(8,*) RIBTYP(J)
          WRITE(NO,*) RIBTYP(J)
          WRITE(13,43) RIBTYP(J)
 43       FORMAT(I2)

C    ************************************************************
C    *                                                          *
C    *    IF THE RIB IS RECESSED, ASK FOR MAGNITUDE OF RECESS   *
C    *                                                          *
C    ************************************************************

        IF(RIBTYP(J).EQ.2) THEN
          WRITE(NO,68)
 68       FORMAT('0','ENTER THE MAGNITUDE OF THE RECESS.')
          READ(5,*) RECES
          IF (STATUS.NE.0) WRITE(8,*) RECES
          WRITE(NO,*) RECES

C    ************************************************************
C    *                                                          *
C    *    IF THE RIB IS A SPARK PLUG PORT, ASK FOR THE RADIUS   *
C    *                     OF THE PORT                          *
C    *                                                          *
C    ************************************************************

        ELSE IF (RIBTYP(J).EQ.3) THEN

           WRITE(NO,69)
 69     FORMAT('0','ENTER THE RADIUS OF THE SPARK PLUG.')
        READ(5,*) RSPPG
        IF (STATUS.NE.0) WRITE(8,*) RSPPG
        WRITE(NO,*) RSPPG
        RIBTYP(J)=1
        IVOLR(J)=J

          ISP=ISP+1
          RSP(ISP)=RSPPG
          PHI1SP(ISP)=PHIONE*PI/180.0
          PHI2SP(ISP)=PHITWO*PI/180.0
          Y1SP(ISP)=YONE
          Y2SP(ISP)=YTWO
          ISPRK(ISP)=J

        ENDIF

C    ************************************************************
C    *                                                          *
C    *               ASK FOR THE CHANNEL TYPE                   *
C    *                                                          *
C    ************************************************************

        IF((J.EQ.1).OR.((RIBTYP(J).EQ.1).AND.(RIBTYP(J-1).EQ.1))) THEN

          WRITE(NO,70)
 70       FORMAT('0','WHAT TYPE OF CHANNEL IS TO BE INPUT?')
          WRITE(NO,72)
 72       FORMAT('0','   ENTER  1  FOR NORMAL CHANNEL...')
          WRITE(NO,74)
 74       FORMAT('0','   ENTER  2  FOR STIFFENED CHANNEL ...')
          WRITE(NO,75)
 75     FORMAT('0','   ENTER  3  IF THIS CHANNEL IS THE EXHAUST PORT ...')
          WRITE(NO,78)
```

```
 78   FORMAT('0','OR  ENTER  4  IF THIS CHANNEL IS THE INTAKE PORT.')

           READ(5,*) CNLTYP(J)
           IF (STATUS.NE.0) WRITE(8,*) CNLTYP(J)
           WRITE(NO,*) CNLTYP(J)

      IF(IEPQUE.EQ.1)  GO TO 84

C   ************************************************************
C   *                                                          *
C   *    IF THE CHANNEL IS AN EXHAUST PORT, THEN ASK FOR THE   *
C   *    TWO ANGLES THAT DEFINE THE SIZE AND LOCATION OF       *
C   *                     THE PORT                             *
C   *                                                          *
C   ************************************************************

           IF(CNLTYP(J).EQ.3) THEN
              IEPQUE=1
              IVOLC(J)=J
              CNLTYP(J)=1
              IEP=J
              P1ODEP=2.0*PI-PHI1OD
              P2ODEP=2.0*PI-PHI2OD
              P1NWEP=2.0*PI-(PHIONE*PI/180.0)

      WRITE(NO,79)
 79   FORMAT('0','ENTER PHIONE AND PHITWO OF THE EXHAUST PORT.')
              READ(5,*) PHI1EP,PHI2EP
              IF (STATUS.NE.0) WRITE(8,*) PHI1EP,PHI2EP
      PHI1EP=2.0*PI-(PHI1EP*PI/180.0)
      PHI2EP=2.0*PI-(PHI2EP*PI/180.0)
              WRITE(NO,*) PHI1EP,PHI2EP

      WRITE(NO,80)
 80   FORMAT('0','ENTER THE RADIUS OF THE EXHAUST PORT.')
              READ(5,*) REXPT
              IF (STATUS.NE.0) WRITE(8,*) REXPT
              WRITE(NO,*) REXPT

      WRITE(NO,82)
 82   FORMAT('0','ENTER THE THICKNESS OF THE EXHAUST PORT.')
              READ(5,*) TEXPT
              IF (STATUS.NE.0) WRITE(8,*) TEXPT
              WRITE(NO,*) TEXPT

           ENDIF

C   ************************************************************
C   *                                                          *
C   *    IF THE CHANNEL IS AN INTAKE PORT, THEN ASK FOR THE    *
C   *    TWO ANGLES THAT DEFINE THE SIZE AND LOCATION OF       *
C   *                     THE PORT                             *
C   *                                                          *
C   ************************************************************

 84   IF(IIPQUE.EQ.1) GO TO 96

           IF(CNLTYP(J).EQ.4) THEN
              IIPQUE=1
              IVOLC(J)=J
              CNLTYP(J)=1
              IIP=J
              P2ODIP=PHI2OD
              P1NWIP=PHIONE*PI/180.0
```

```
               PHIZ=P1NWIP*180.0/PI

         WRITE(NO,86)
  86     FORMAT('0','ENTER PHIONE AND PHITWO OF THE INTAKE PORT.')
               READ(5,*) PHI1IP,PHI2IP
               IF (STATUS.NE.0) WRITE(8,*) PHI1IP,PHI2IP
               PHI1IP=PHI1IP*PI/180.0
               PHI2IP=PHI2IP*PI/180.0
               WRITE(NO,*) PHI1IP,PHI2IP

         WRITE(NO,88)
  88     FORMAT('0','ENTER THE RADIUS OF THE INTAKE PORT.')
               READ(5,*) RINPT
               IF (STATUS.NE.0) WRITE(8,*) RINPT
               WRITE(NO,*) RINPT

         WRITE(NO,89)
  89     FORMAT('0','ENTER THE THICKNESS OF THE INTAKE PORT.')
               READ(5,*) TINPT
               IF (STATUS.NE.0) WRITE(8,*) TINPT
               WRITE(NO,*) TINPT

         WRITE(NO,91)
  91     FORMAT('0','ENTER THE LENGTH AND WIDTH OF THE INTAKE PORT.')
               READ(5,*) LINPT,WINPT
               IF (STATUS.NE.0) WRITE(8,*) LINPT,WINPT
               WRITE(NO,*) LINPT,WINPT

             ENDIF

  96     CONTINUE
         ELSE

             WRITE(NO,98)
  98         FORMAT('0','THE CHANNEL MUST BE NORMAL.')
             CNLTYP(J)=1

         ENDIF

             WRITE(13,*) IVOLC(J),IVOLR(J)

C     *************************************************************
C     *                                                           *
C     *                  END PROMPTED INPUT                       *
C     *                                                           *
C     *************************************************************

            IF((PHIONE.GT.180.0).AND.(PHIONE.LT.270.0)) PHCHCK=PHCHCK+1
            IF((PHITWO.GT.180.0).AND.(PHITWO.LT.270.0)) PHCHCK=PHCHCK+1
            IF(PHIONE.GT.360.0) PHCHCK=PHCHCK+1
            IF(PHITWO.GT.360.0) PHCHCK=PHCHCK+1

             PHIONE=PHIONE*PI/180.0
             PHITWO=PHITWO*PI/180.0

C     *************************************************************
C     *                                                           *
C     *      DEPENDING UPON WHICH REGION WAS SPECIFIED BY THE     *
C     *      USER, THE APPROPRIATE SECTION OF THE PROGRAM IS      *
C     *                       CALLED                              *
C     *       EITHER THE OVERLAP, RECTANGULAR, OR CYLINDRICAL     *
C     *                                                           *
C     *************************************     *****************
```

```
      IF(REGION(J).EQ.1) THEN

           GO TO 120

      ELSE IF(REGION(J).EQ.2) THEN

           GO TO 115

      ELSE IF((RGNCTR.EQ.1).OR.(RGNCTR.EQ.3)) THEN

           GO TO 100

      ELSE IF((RGNCTR.EQ.2).OR.(RGNCTR.EQ.4)) THEN

           GO TO 105

      ENDIF

C    ************************************************************
C    *                                                          *
C    *    THIS SECTION CALCULATES THE POINT COORDINATES OF THE  *
C    *    OVERLAPPING RIB/CHANNEL SEGMENTS.  THERE ARE 12       *
C    *    DIFFERENT OVERLAP CONDITIONS DEPENDING UPON THE       *
C    *    PERCENT A GIVEN RIB OR CHANNEL OVERALPS A REGION      *
C    *                                                          *
C    ************************************************************

 100  IF(J.EQ.1) THEN

C    ************************************************************
C    *                                                          *
C    *    FOR THE FIRST RIB/CAHNNEL SEGMENT, ASK FOR THE        *
C    *            THICKNESS OF THE CHANNEL                      *
C    *                                                          *
C    ************************************************************

       WRITE(NO,102)
 102   FORMAT('0','ENTER THE THICKNESS OF THE CHANNEL.')
         READ(5,*) TCHNL
         IF (STATUS.NE.0) WRITE(8,*) TCHNL
         WRITE(NO,*) TCHNL

      ENDIF

          IF((PHIONE.LT.0.0).OR.((SLSLVR.GT.1).AND.(PHIONE.LT.PI))) THEN

      IF(SLSLVR.GT.1) THEN

              SIGN=-SIGN
              TRANS=-TRANS
              RECES=-RECES

      ENDIF

              ICT=ICT+1
              IF(ICT.EQ.1) ICHK=0
              IF(ICT.GT.1) ICHK=1

                   PHIONE=-PHIONE
                   VRTICL=T*TAN(PHIONE)
                   KKIX(J)=T*SIGN
                   KKIY(J)=(ABS(TRANS)-VRTICL)*SIGN
                   KIX(J)=R*SIGN
                   KIY(J)=KKIY(J)
```

```
            DIX(J)=XMAX*SIGN
            DIY(J)=KKIY(J)
         IF(RIBTYP(J).EQ.2) THEN

            DRIX(J)=DIX(J)+RECES
            DRIY(J)=DIY(J)+RECES

         ENDIF

         CALL YALPSL (EE,RR,DIY(J),ICHK,DALPHA)
         CALL THTASL (EE,RR,PI,DALPHA,DTHETA)

            NNIX(J)=XXX(T,PHITWO)
            NNIY(J)=(ABS(YYY(T,PHITWO))+ABS(TRANS))*SIGN
            NIX(J)=XXX(R,PHITWO)
            NIY(J)=(ABS(YYY(R,PHITWO))+ABS(TRANS))*SIGN
        PHIG=PHITWO

         CALL PALPSL (EE,RR,PI,PHITWO,TRANS,GALPHA)
         CALL THTASL (EE,RR,PI,GALPHA,GTHETA)

            GIX(J)=FNX(GALPHA)
            GIY(J)=FNY(GALPHA)

         IF(RIBTYP(J).EQ.2) THEN

            GRIX(J)=GIX(J)+RECES*COS(PHIG)
            GRIY(J)=GIY(J)+RECES*SIN(PHIG)

         ENDIF

       CHECK=ABS(NNIY(J))-ABS(TRANS)+ABS(VRTICL)

C    **************************************************************
C    *                                                            *
C    *                 OVERLAP CONDITION ONE                      *
C    *                                                            *
C    *        20% OF THE RIB LIES IN THE CYLINDRICAL REGION       *
C    *               (QUADRANTS ONE AND THREE)                    *
C    *                                                            *
C    **************************************************************

      IF(VRTICL.LT.CHECK/5.0) THEN
      WRITE(NO,*) 'PART 1'
       IF(PHITWO.LT.PI) THEN

         PHIE=PHITWO/3.0
         PHIF=2.0*PHITWO/3.0

       ELSE IF(PHITWO.GT.PI) THEN

         PHIE=PI+(PHITWO-PI)/3.0
         PHIF=PI+2.0*(PHITWO-PI)/3.0

       ENDIF

            LLIX(J)=XXX(T,PHIE)
            LLIY(J)=YYY(T,PHIE)+TRANS
            LIX(J)=XXX(R,PHIE)
            LIY(J)=YYY(R,PHIE)+TRANS

          CALL PALPSL (EE,RR,PI,PHIE,TRANS,EALPHA)
          CALL THTASL (EE,RR,PI,EALPHA,ETHETA)
```

```
               EIX(J)=FNX(EALPHA)
               EIY(J)=FNY(EALPHA)

            IF(RIBTYP(J).EQ.2) THEN

               ERIX(J)=EIX(J)+RECES*COS(PHIE)
               ERIY(J)=EIY(J)+RECES*SIN(PHIE)

            ENDIF


               MMIX(J)=XXX(T,PHIF)
               MMIY(J)=YYY(T,PHIF)+TRANS
               MIX(J)=XXX(R,PHIF)
               MIY(J)=YYY(R,PHIF)+TRANS

            CALL PALPSL (EE,RR,PI,PHIF,TRANS,FALPHA)
            CALL THTASL (EE,RR,PI,FALPHA,FTHETA)

               FIX(J)=FNX(FALPHA)
               FIY(J)=FNY(FALPHA)

            IF(RIBTYP(J).EQ.2) THEN

               FRIX(J)=FIX(J)+RECES*COS(PHIF)
               FRIY(J)=FIY(J)+RECES*SIN(PHIF)

            ENDIF

C     *************************************************************
C     *                                                           *
C     *                 OVERLAP CONDITION TWO                     *
C     *                                                           *
C     *         80% OF THE RIB LIES IN THE CYLINDRICAL REGION     *
C     *               (QUADRANTS ONE AND THREE)                   *
C     *                                                           *
C     *************************************************************

         ELSE IF (VRTICL.GT.4.0*CHECK/5.0) THEN
            WRITE(NO,*) 'PART 2'
               LLIX(J)=T*SIGN
               LLIY(J)=(ABS(KKIY(J))+VRTICL/3.0)*SIGN
               LIX(J)=R*SIGN
               LIY(J)=LLIY(J)
               EIX(J)=XMAX*SIGN
               EIY(J)=LLIY(J)

            IF (RIBTYP(J).EQ.2) THEN

               ERIX(J)=EIX(J)+RECES
               ERIY(J)=EIY(J)+RECES

            ENDIF

            CALL YALPSL (EE,RR,LIY(J),ICHK,EALPHA)
            CALL THTASL (EE,RR,PI,EALPHA,ETHETA)

               MMIX(J)=T*SIGN
               MMIY(J)=(ABS(KKIY(J))+2.0*VRTICL/3.0)*SIGN
               MIX(J)=R*SIGN
               MIY(J)=MMIY(J)
               FIX(J)=XMAX*SIGN
               FIY(J)=MMIY(J)

```

```
                IF (RIBTYP(J).EQ.2) THEN

                   FRIX(J)=FIX(J)+RECES
                   FRIY(J)=FIY(J)+RECES

                ENDIF

                CALL YALPSL(EE,RR,MIY(J),ICHK,FALPHA)
                CALL THTASL (EE,RR,PI,FALPHA,FTHETA)

C    **************************************************************
C    *                                                            *
C    *                 OVERLAP CONDITION THREE                    *
C    *                                                            *
C    *          BETWEEN 20% AND 80% OF THE RIB LIES IN THE        *
C    *                    CYLINDRICAL REGION                      *
C    *                 (QUADRANTS ONE AND THREE)                  *
C    *                                                            *
C    **************************************************************

           ELSE IF((VRTICL.GT.CHECK/5.).AND.(VRTICL.LT.4.*CHECK/5.))THEN
              WRITE(NO,*) 'PART 3'
                   LLIX(J)=T*SIGN
                   LLIY(J)=(ABS(KKIY(J))+VRTICL/2.0)*SIGN
                   LIX(J)=R*SIGN
                   LIY(J)=LLIY(J)
                   EIX(J)=XMAX*SIGN
                   EIY(J)=LLIY(J)

                IF(RIBTYP(J).EQ.2) THEN

                   ERIX(J)=EIX(J)+RECES
                   ERIY(J)=EIY(J)+RECES

                ENDIF

                CALL YALPSL (EE,RR,EIY(J),ICHK,EALPHA)
                CALL THTASL (EE,RR,PI,EALPHA,ETHETA)

            IF(PHITWO.LT.PI) THEN
              PHIF=PHITWO/2.0

            ELSE IF (PHITWO.GT.PI) THEN

              PHIF=PI+(PHITWO-PI)/2.0
            ENDIF

                   MMIX(J)=XXX(T,PHIF)
                   MMIY(J)=(YYY(T,PHIF)+TRANS)*SIGN
                   MIX(J)=XXX(R,PHIF)
                   MIY(J)=(YYY(R,PHIF)+TRANS)*SIGN

                CALL PALPSL (EE,RR,PI,PHIF,TRANS,FALPHA)
                CALL THTASL (EE,RR,PI,FALPHA,FTHETA)

                   FIX(J)=FNX(FALPHA)
                   FIY(J)=FNY(FALPHA)*SIGN

                IF(RIBTYP(J).EQ.2) THEN

                   FRIX(J)=FIX(J)+RECES*COS(PHIF)
                   FRIY(J)=FIY(J)+RECES*SIN(PHIF)

                ENDIF
```

```
         ENDIF

           IF(J.EQ.1) THEN
               CHECK=TCHNL

           ELSE

                  CHECK=KKIY(J)-NNIY(J-1)
           ENDIF

C   ************************************************************
C   *                                                          *
C   *    THIS SECTION CALCUALTES THE POINT COORDINATES OF THE   *
C   *    CHANNEL FOR THE FIRST THREE OVERLAP CONDITIONS         *
C   *                                                          *
C   ************************************************************

            IIIX(J)=T*SIGN
                  IF(J.EQ.1) THEN
                       IIIY(J)=KKIY(J)-2.0*TCHNL/3.0
                       AIX(J)=XMAX*SIGN
                       AIY(J)=(KKIY(J)-TCHNL)*SIGN

             CALL YALPSL (EE,RR,AIY(J),ICHK,AALPHA)
             CALL THTASL (EE,RR,PI,AALPHA,ATHETA)

              AAAIX(J)=FINERX(AALPHA,ATHETA,CC)
              AAAIY(J)=FINERY(AALPHA,ATHETA,CC)
              AAIX(J)=FINERX(AALPHA,ATHETA,BB)
              AAIY(J)=FINERY(AALPHA,ATHETA,BB)

                  ELSE

      IIIY(J)=(ABS(KKIY(J))-ABS(2.0*(KKIY(J)-NNIY(J-1))/3.0))*SIGN

                  ENDIF

            IIX(J)=R*SIGN
            IIY(J)=IIIY(J)
            BIX(J)=XMAX*SIGN
            BIY(J)=IIIY(J)

             CALL YALPSL (EE,RR,BIY(J),ICHK,BALPHA)
             CALL THTASL (EE,RR,PI,BALPHA,BTHETA)

            JJIX(J)=T*SIGN

                  IF(J.EQ.1) THEN
                        JJIY(J)=KKIY(J)-TCHNL/3.0

                  ELSE

      JJIY(J)=(ABS(KKIY(J))-ABS((KKIY(J)-NNIY(J-1))/3.0))*SIGN

                  ENDIF

            JIX(J)=R*SIGN
            JIY(J)=JJIY(J)
            CIX(J)=XMAX*SIGN
            CIY(J)=JJIY(J)

             CALL YALPSL (EE,RR,CIY(J),ICHK,CALPHA)
             CALL THTASL (EE,RR,PI,CALPHA,CTHETA)
```

```
                HIX(J)=R*SIGN
                HIY(J)=(KKIY(J)-CHECK)*SIGN

      SIGN=ABS(SIGN)
      TRANS=ABS(TRANS)
      RECES=ABS(RECES)

C     ************************************************************
C     *                                                          *
C     *           END CHANNEL COORDINATE CALCULATION             *
C     *           OF FIRST THREE OVERLAP CONDITIONS              *
C     *                                                          *
C     ************************************************************

        ELSE IF (PHIONE.GE.0.0) THEN

     IF(SLSLVR.GT.1) THEN

            SIGN=-SIGN
            TRANS=-TRANS
            RECES=-RECES

     ENDIF

            PHIE=PHIONE+(PHITWO-PHIONE)/3.0
            PHIF=PHIONE+2.0*(PHITWO-PHIONE)/3.0

                KKIX(J)=XXX(T,PHIONE)
                KKIY(J)=YYY(T,PHIONE)+TRANS
                KIX(J)=XXX(R,PHIONE)
                KIY(J)=YYY(R,PHIONE)+TRANS
             PHID=PHIONE

              CALL PALPSL (EE,RR,PI,PHIONE,TRANS,DALPHA)
              CALL THTASL (EE,RR,PI,DALPHA,DTHETA)

                DIX(J)=FNX(DALPHA)
                DIY(J)=FNY(DALPHA)

              IF(RIBTYP(J).EQ.2) THEN

                DRIX(J)=DIX(J)+RECES*COS(PHID)
                DRIY(J)=DIY(J)+RECES*SIN(PHID)

              ENDIF

                LLIX(J)=XXX(T,PHIE)
                LLIY(J)=YYY(T,PHIE)+TRANS
                LIX(J)=XXX(R,PHIE)
                LIY(J)=YYY(R,PHIE)+TRANS

              CALL PALPSL (EE,RR,PI,PHIE,TRANS,EALPHA)
              CALL THTASL (EE,RR,PI,EALPHA,ETHETA)

                EIX(J)=FNX(EALPHA)
                EIY(J)=FNY(EALPHA)

              IF(RIBTYP(J).EQ.2) THEN

                ERIX(J)=EIX(J)+RECES*COS(PHIE)
                ERIY(J)=EIY(J)+RECES*SIN(PHIE)

              ENDIF
```

```
         MMIX(J)=XXX(T,PHIF)
         MMIY(J)=YYY(T,PHIF)+TRANS
         MIX(J)=XXX(R,PHIF)
         MIY(J)=YYY(R,PHIF)+TRANS

      CALL PALPSL (EE,RR,PI,PHIF,TRANS,FALPHA)
      CALL THTASL (EE,RR,PI,FALPHA,FTHETA)

         FIX(J)=FNX(FALPHA)
         FIY(J)=FNY(FALPHA)

      IF(RIBTYP(J).EQ.2) THEN

         FRIX(J)=FIX(J)+RECES*COS(PHIF)
         FRIY(J)=FIY(J)+RECES*SIN(PHIF)

      ENDIF

         NNIX(J)=XXX(T,PHITWO)
         NNIY(J)=YYY(T,PHITWO)+TRANS
         NIX(J)=XXX(R,PHITWO)
         NIY(J)=YYY(R,PHITWO)+TRANS
     PHIG=PHITWO

      CALL PALPSL (EE,RR,PI,PHITWO,TRANS,GALPHA)
      CALL THTASL (EE,RR,PI,GALPHA,GTHETA)

         GIX(J)=FNX(GALPHA)
         GIY(J)=FNY(GALPHA)

      IF(RIBTYP(J).EQ.2) THEN

         GRIX(J)=GIX(J)+RECES*COS(PHIG)
         GRIY(J)=GIY(J)+RECES*SIN(PHIG)

      ENDIF

  IF (J.EQ.1) THEN

    AIX(J)=XMAX*SIGN
    AIY(J)=(KKIY(J)-TCHNL)*SIGN

     CALL YALPSL (EE,RR,AIY(J),ICHK,AALPHA)
     CALL THTASL (EE,RR,PI,AALPHA,ATHETA)

      AAAIX(J)=FINERX(AALPHA,ATHETA,CC)
      AAAIY(J)=FINERY(AALPHA,ATHETA,CC)
      AAIX(J)=FINERX(AALPHA,ATHETA,BB)
      AAIY(J)=FINERY(AALPHA,ATHETA,BB)

  ENDIF

    IF(J.EQ.1) THEN
        CHECK=TCHNL

    ELSE

        CHECK=ABS(KKIY(J))-ABS(NNIY(J-1))
        CHECK=ABS(CHECK)

    ENDIF

     HIX(J)=R*SIGN
```

```
              HIY(J)=(KKIY(J)-CHECK)*SIGN

          VRTICL=ABS(KKIY(J))-ABS(TRANS)

C   ************************************************************
C   *                                                          *
C   *                 OVERLAP CONDITION FOUR                   *
C   *                                                          *
C   *        20% OF THE RIB LIES IN THE CYLINDRICAL REGION     *
C   *                (QUADRANTS ONE AND THREE)                 *
C   *                                                          *
C   ************************************************************

            IF(VRTICL.LT.CHECK/5.0) THEN
                  WRITE(NO,*) 'PART 4'
                  IIIX(J)=T*SIGN
                  IIIY(J)=(ABS(KKIY(J))-2.0*CHECK/3.0)*SIGN
                  IIX(J)=R*SIGN
                  IIY(J)=IIIY(J)
                  BIX(J)=XMAX*SIGN
                  BIY(J)=IIIY(J)

                CALL YALPSL (EE,RR,BIY(J),ICHK,BALPHA)
                CALL THTASL (EE,RR,PI,BALPHA,BTHETA)

                  JJIX(J)=T*SIGN
                  JJIY(J)=(ABS(KKIY(J))-CHECK/3.0)*SIGN
                  JIX(J)=R*SIGN
                  JIY(J)=JJIY(J)
                  CIX(J)=XMAX*SIGN
                  CIY(J)=JJIY(J)

                CALL YALPSL (EE,RR,CIY(J),ICHK,CALPHA)
                CALL THTASL (EE,RR,PI,CALPHA,CTHETA)

C   ************************************************************
C   *                                                          *
C   *                 OVERLAP CONDITION FIVE                   *
C   *                                                          *
C   *        80% OF THE RIB LIES IN THE CYLINDRICAL REGION     *
C   *                (QUADRANTS ONE AND THREE)                 *
C   *                                                          *
C   ************************************************************

            ELSE IF(VRTICL.GT.4.0*CHECK/5.0) THEN
               WRITE(NO,*) 'PART 5'
             IF(PHIONE.LT.PI) THEN
               PHIB=PHIONE/3.0
               PHIC=2.0*PHIONE/3.0

             ELSE IF(PHIONE.GT.PI) THEN

               PHIC=PHIONE-(PHITWO-PHIONE)/4.0
               PHIB=PHIONE-2.0*(PHITWO-PHIONE)/5.0
             ENDIF

                   IIIX(J)=XXX(T,PHIB)
                   IIIY(J)=YYY(T,PHIB)+TRANS
                   IIX(J)=XXX(R,PHIB)
                   IIY(J)=YYY(R,PHIB)+TRANS

                CALL PALPSL (EE,RR,PI,PH[illegible]ANS,BALPHA)
                CALL THTASL (EE,RR,PI,BA[illegible]BTHETA)
```

```
              BIX(J)=FNX(BALPHA)
              BIY(J)=FNY(BALPHA)

              JJIX(J)=XXX(T,PHIC)
              JJIY(J)=YYY(T,PHIC)+TRANS
              JIX(J)=XXX(R,PHIC)
              JIY(J)=YYY(R,PHIC)+TRANS

           CALL PALPSL (EE,RR,PI,PHIC,TRANS,CALPHA)
           CALL THTASL (EE,RR,PI,CALPHA,CTHETA)

              CIX(J)=FNX(CALPHA)
              CIY(J)=FNY(CALPHA)

C    *************************************************************
C    *                                                           *
C    *                 OVERLAP CONDITION SIX                     *
C    *                                                           *
C    *         BETWEEN 20% AND 80% OF THE RIB LIES IN THE        *
C    *                   CYLINDRICAL REGION                      *
C    *              (QUADRANTS ONE AND THREE)                    *
C    *                                                           *
C    *************************************************************

        ELSE IF((VRTICL.GT.CHECK/5.).AND.(VRTICL.LT.4.*CHECK/5.))THEN
           WRITE(NO,*) 'PART 6'
              IIIX(J)=T*SIGN
      IIIY(J)=((ABS(TRANS)-ABS(NNIY(J-1)))/2.0+ABS(NNIY(J-1)))*SIGN
              IIX(J)=R*SIGN
              IIY(J)=IIIY(J)
              BIX(J)=XMAX*SIGN
              BIY(J)=IIIY(J)

           CALL YALPSL (EE,RR,BIY(J),ICHK,BALPHA)
           CALL THTASL (EE,RR,PI,BALPHA,BTHETA)


         IF(PHIONE.LT.PI) THEN
          PHIC=PHIONE/2.0

         ELSE IF(PHIONE.GT.PI) THEN

          PHIC=PI+(PHIONE-PI)/2.0
         ENDIF

              JJIX(J)=XXX(T,PHIC)
              JJIY(J)=YYY(T,PHIC)+TRANS
              JIX(J)=XXX(R,PHIC)
              JIY(J)=YYY(R,PHIC)+TRANS

           CALL PALPSL (EE,RR,PI,PHIC,TRANS,CALPHA)
           CALL THTASL (EE,RR,PI,CALPHA,CTHETA)

              CIX(J)=FNX(CALPHA)
              CIY(J)=FNY(CALPHA)


        ENDIF
      ENDIF

        SIGN=ABS(SIGN)
        TRANS=ABS(TRANS)
        RECES=ABS(RECES)

        GO TO 110
```

```
105              IF(PHCHCK.EQ.1) THEN

            ICTA=ICTA+1
            IF(ICTA.EQ.1) ICHK=1
            IF(ICTA.GT.1) ICHK=0

      IF (SLSLVR.LT.1) THEN
         RECESX=-RECES
         RECESY=RECES
         SIGNX=-SIGN
         SIGNY=SIGN
      ELSE

         TRANS=-TRANS
         RECESX=RECES
         RECESY=-RECES
         SIGNX=SIGN
         SIGNY=-SIGN
      ENDIF

                       IF(SLSLVR.GT.1) THEN

                          PHITWI=PHITWO-2.0*PI
                          PHITW2=2.0*PI
                       ELSE
                          PHITWI=PHITWO-PI
                          PHITW2=PI

                       ENDIF

          VRTICL=ABS(T*TAN(PHITWI))

               NNIX(J)=T*SIGNX
               NNIY(J)=(ABS(TRANS)-VRTICL)*SIGNY
               NIX(J)=R*SIGNX
               NIY(J)=NNIY(J)
               GIX(J)=XMAX*SIGNX
               GIY(J)=NNIY(J)

             IF(RIBTYP(J).EQ.2) THEN

                GRIX(J)=GIX(J)+RECESX
                GRIY(J)=GIY(J)+RECESY

             ENDIF

            CALL YALPSL (EE,RR,NIY(J),ICHK,GALPHA)
            CALL THTASL (EE,RR,PI,GALPHA,GTHETA)

               KKIX(J)=XXX(T,PHIONE)
               KKIY(J)=YYY(T,PHIONE)+TRANS
               KIX(J)=XXX(R,PHIONE)
               KIY(J)=YYY(R,PHIONE)+TRANS

            CALL PALPSL (EE,RR,PI,PHIONE,TRANS,DALPHA)
            CALL THTASL (EE,RR,PI,DALPHA,DTHETA)

               DIX(J)=FNX(DALPHA)
               DIY(J)=FNY(DALPHA)

            IF(RIBTYP(J).EQ.2) THEN

               DRIX(J)=DIX(J)+RECES*COS(PHID)
```

```
                 DRIY(J)=DIY(J)+RECES*SIN(PHID)

               ENDIF

            CHECK=ABS(ABS(KKIY(J))-ABS(NNIY(J)))

C    ************************************************************
C    *                                                          *
C    *                OVERLAP CONDITION SEVEN                   *
C    *                                                          *
C    *       20% OF THE RIB LIES IN THE CYLINDRICAL REGION      *
C    *               (QUADRANTS TWO AND FOUR)                   *
C    *                                                          *
C    ************************************************************

         IF(VRTICL.LT.CHECK/5.0) THEN
            WRITE(NO,*) 'PART 7'
              PHIE=(PHITW2-PHIONE)/3.0+PHIONE
              PHIF=2.0*(PHITW2-PHIONE)/3.0+PHIONE

                 LLIX(J)=XXX(T,PHIE)
                 LLIY(J)=YYY(T,PHIE)+TRANS
                 LIX(J)=XXX(R,PHIE)
                 LIY(J)=YYY(R,PHIE)+TRANS

              CALL PALPSL (EE,RR,PI,PHIE,TRANS,EALPHA)
              CALL THTASL (EE,RR,PI,EALPHA,ETHETA)

                 EIX(J)=FNX(EALPHA)
                 EIY(J)=FNY(EALPHA)

               IF(RIBTYP(J).EQ.2) THEN

                  ERIX(J)=EIX(J)+RECES*COS(PHIE)
                  ERIY(J)=EIY(J)+RECES*SIN(PHIE)

               ENDIF

                 MMIX(J)=XXX(T,PHIF)
                 MMIY(J)=YYY(T,PHIF)+TRANS
                 MIX(J)=XXX(R,PHIF)
                 MIY(J)=YYY(R,PHIF)+TRANS

              CALL PALPSL (EE,RR,PI,PHIF,TRANS,FALPHA)
              CALL THTASL (EE,RR,PI,FALPHA,FTHETA)

                 FIX(J)=FNX(FALPHA)
                 FIY(J)=FNY(FALPHA)

               IF(RIBTYP(J).EQ.2) THEN

                  ERIX(J)=EIX(J)+RECES*COS(PHIE)
                  ERIY(J)=EIY(J)+RECES*SIN(PHIE)

               ENDIF
C    ************************************************************
C    *                                                          *
C    *                OVERLAP CONDITION EIGHT                   *
C    *                                                          *
C    *       80% OF THE RIB LIES IN THE CYLINDRICAL REGION      *
C    *               (QUADRANTS TWO AND FOUR)                   *
C    *                                                          *
C    ************************************************************
```

```
      ELSE IF (VRTICL.GT.4.0*CHECK/5.0) THEN
         WRITE(NO,*) 'PART 8'
                LLIX(J)=T*SIGNX
                LLIY(J)=(ABS(NNIY(J))+2.0*(VRTICL)/3.0)*SIGNY
                LIX(J)=R*SIGNX
                LIY(J)=LLIY(J)
                EIX(J)=XMAX*SIGNX
                EIY(J)=LLIY(J)

             IF(RIBTYP(J).EQ.2) THEN

                ERIX(J)=EIX(J)+RECESX
                ERIY(J)=EIY(J)+RECESY

             ENDIF

            CALL YALPSL (EE,RR,LIY(J),ICHK,EALPHA)
            CALL THTASL (EE,RR,PI,EALPHA,ETHETA)

               MMIX(J)=T*SIGNX
               MMIY(J)=(ABS(NNIY(J))+(VRTICL)/3.0)*SIGNY
               MIX(J)=R*SIGNX
               MIY(J)=MMIY(J)
               FIX(J)=XMAX*SIGNX
               FIY(J)=MMIY(J)

             IF(RIBTYP(J).EQ.2) THEN

                FRIX(J)=FIX(J)+RECESX
                FRIY(J)=FIY(J)+RECESY

             ENDIF

            CALL YALPSL (EE,RR,MIY(J),ICHK,FALPHA)
            CALL THTASL (EE,RR,PI,FALPHA,FTHETA)
C    *************************************************************
C    *                                                           *
C    *                  OVERLAP CONDITION NINE                   *
C    *                                                           *
C    *          BETWEEN 20% AND 80% OF THE RIB LIES IN THE       *
C    *                    CYLINDRICAL REGION                     *
C    *                (QUADRANTS TWO AND FOUR)                   *
C    *                                                           *
C    *************************************************************

      ELSE IF ((VRTICL.GT.CHECK/5.0).AND.(VRTICL.LT.4.0*CHECK/5.0)) THEN
         WRITE(NO,*) 'PART 9'
           IF (SLSLVR.GT.1) THEN
             PHIE=(2.0*PI-PHIONE)/2.0+PHIONE

           ELSE

             PHIE=(PI-PHIONE)/2.0+PHIONE
           ENDIF

                LLIX(J)=XXX(T,PHIE)
                LLIY(J)=YYY(T,PHIE)+TRANS
                LIX(J)=XXX(R,PHIE)
                LIY(J)=YYY(R,PHIE)+TRANS

             CALL PALPSL (EE,RR,PI,PHIE,TRANS,EALPHA)
             CALL THTASL (EE,RR,PI,EALPHA,ETHETA)
```

```
            EIX(J)=FNX(EALPHA)
            EIY(J)=FNY(EALPHA)

          IF(RIBTYP(J).EQ.2) THEN

             ERIX(J)=EIX(J)+RECES*COS(PHIE)
             ERIY(J)=EIY(J)+RECES*SIN(PHIE)

          ENDIF

            MMIX(J)=T*SIGNX
            MMIY(J)=(ABS(NNIY(J))+VRTICL/2.0)*SIGNY
            MIX(J)=R*SIGNX
            MIY(J)=MMIY(J)
            FIX(J)=XMAX*SIGNX
            FIY(J)=MMIY(J)

          IF(RIBTYP(J).EQ.2) THEN

             FRIX(J)=FIX(J)+RECESX
             FRIY(J)=FIY(J)+RECESY

          ENDIF

        CALL YALPSL (EE,RR,FIY(J),ICHK,FALPHA)
        CALL THTASL (EE,RR,PI,FALPHA,FTHETA)

      ENDIF

C    ************************************************************
C    *                                                          *
C    *    THIS SECTION CALCULATES THE CHANNEL COORDINATES OF    *
C    *         OVERLAP CONDITIONS SEVEN THROUGH NINE            *
C    *                                                          *
C    ************************************************************

      PHIB=PHIT+(PHIONE-PHIT)/3.0
      PHIC=PHIT+2.0*(PHIONE-PHIT)/3.0

         IIIX(J)=XXX(T,PHIB)
         IIIY(J)=YYY(T,PHIB)+TRANS
         IIX(J)=XXX(R,PHIB)
         IIY(J)=YYY(R,PHIB)+TRANS

       CALL PALPSL (EE,RR,PI,PHIB,TRANS,BALPHA)
       CALL THTASL (EE,RR,PI,BALPHA,BTHETA)

         BIX(J)=FNX(BALPHA)
         BIY(J)=FNY(BALPHA)

         JJIX(J)=XXX(T,PHIC)
         JJIY(J)=YYY(T,PHIC)+TRANS
         JIX(J)=XXX(R,PHIC)
         JIY(J)=YYY(R,PHIC)+TRANS

       CALL PALPSL (EE,RR,PI,PHIC,TRANS,CALPHA)
       CALL THTASL (EE,RR,PI,CALPHA,CTHETA)

         CIX(J)=FNX(CALPHA)
         CIY(J)=FNY(CALPHA)

         HIX(J)=NIX(J-1)
         HIY(J)=NIY(J-1)
```

```
C     ************************************************************
C     *                                                          *
C     *            END CHANNEL COORDINATE CALCULATION            *
C     *                                                          *
C     ************************************************************

        ELSE IF (PHCHCK.EQ.2) THEN

              IF(SLSLVR.GT.1) THEN
                   VRTICL=T*TAN(PHIONE-2.0*PI)
              ELSE
                   VRTICL=T*TAN(PHIONE-PI)
              ENDIF

                   KKIX(J)=T*SIGNX
                   KKIY(J)=(TRANS-VRTICL)*SIGNY
                   KIX(J)=R*SIGNX
                   KIY(J)=KKIY(J)
                   DIX(J)=XMAX*SIGNX
                   DIY(J)=KKIY(J)

                IF(RIBTYP(J).EQ.2) THEN

                    DRIX(J)=DIX(J)+RECESX
                    DRIY(J)=DIY(J)+RECESY

                ENDIF

              CALL YALPSL (EE,RR,DIY(J),ICHK,DALPHA)
              CALL THTASL (EE,RR,PI,DALPHA,DTHETA)

                   NNIX(J)=T*SIGNX
                   NNIY(J)=(TRANS-VRTICL)*SIGNY
                   NIX(J)=R*SIGNX
                   NIY(J)=NNIY(J)
                   GIX(J)=XMAX*SIGNX
                   GIY(J)=NNIY(J)

                IF(RIBTYP(J).EQ.2) THEN

                    GRIX(J)=GIX(J)+RECESX
                    GRIY(J)=GIY(J)+RECESY

                ENDIF

              CALL YALPSL (EE,RR,GIY(J),ICHK,GALPHA)
              CALL THTASL (EE,RR,PI,GALPHA,GTHETA)

             TRIB=KKIY(J)-NNIY(J)

                   LLIX(J)=T*SIGNX
                   LLIY(J)=(NNIX(J)+2.0*TRIB/3.0)*SIGNY
                   LIX(J)=R*SIGNX
                   LIY(J)=LLIY(J)
                   EIX(J)=XMAX*SIGNX
                   EIY(J)=LLIY(J)

                IF(RIBTYP(J).EQ.2) THEN

                    ERIX(J)=EIX(J)+RECESX
                    ERIY(J)=EIY(J)+RECESY

                ENDIF
```

```
            CALL YALPSL (EE,RR,EIY(J),ICHK,EALPHA)
            CALL THTASL (EE,RR,PI,EALPHA,ETHETA)

               MMIX(J)=T*SIGNX
               MMIY(J)=(NNIY(J)+TRIB/3.0)*SIGNY
               MIX(J)=R*SIGNX
               MIY(J)=MMIY(J)
               FIX(J)=XMAX*SIGNX
               FIY(J)=MMIY(J)

             IF(RIBTYP(J).EQ.2) THEN

                FRIX(J)=FIX(J)+RECESX
                FRIY(J)=FIY(J)+RECESY

             ENDIF

            CALL YALPSL (EE,RR,FIY(J),ICHK,FALPHA)
            CALL THTASL (EE,RR,PI,FALPHA,FTHETA)

                 HIX(J)=NIX(J-1)
                 HIY(J)=NIY(J-1)

          CHECK=TRANS-KKIY(J)

C    *************************************************************
C    *                                                           *
C    *                OVERLAP CONDITION TEN                      *
C    *                                                           *
C    *     20% OF THE CHANNEL LIES IN THE CYLINDRICAL REGION     *
C    *              (QUADRANTS TWO AND FOUR)                     *
C    *                                                           *
C    *************************************************************

      IF(CHECK.LT.TRIB/5.0) THEN
         WRITE(NO,*) 'PART 10'
            ANGLE=NNIY(J-1)/NNIX(J-1)
            PHIH=ATAN(ANGLE)

            PHIB=PHIH+(PHIONE-PHIH)/3.0
            PHIC=PHIH+2.0*(PHIONE-PHIH)/3.0

               IIIX(J)=XXX(T,PHIB)
               IIIY(J)=YYY(T,PHIB)+TRANS
               IIX(J)=XXX(R,PHIB)
               IIY(J)=YYY(R,PHIB)+TRANS

            CALL PALPSL (EE,RR,PI,PHIB,TRANS,BALPHA)
            CALL THTASL (EE,RR,PI,BALPHA,BTHETA)

               BIX(J)=FNX(BALPHA)
               BIY(J)=FNY(BALPHA)

               JJIX(J)=XXX(T,PHIC)
               JJIY(J)=YYY(T,PHIC)+TRANS
               JIX(J)=XXX(R,PHIC)
               JIY(J)=YYY(R,PHIC)+TRANS

            CALL PALPSL (EE,RR,PI,PHIC,TRANS,CALPHA)
            CALL THTASL (EE,RR,PI,CALPHA,CTHETA)

               CIX(J)=FNX(CALPHA)
               CIY(J)=FNY(CALPHA)
```

```
C     *************************************************************     GEN16010
C     *                                                           *     GEN16020
C     *               OVERLAP CONDITION ELEVEN                    *     GEN16030
C     *                                                           *     GEN16040
C     *     80% OF THE CHANNEL LIES IN THE CYLINDRICAL REGION     *     GEN16050
C     *               (QUADRANTS TWO AND FOUR)                    *     GEN16060
C     *                                                           *     GEN16070
C     *************************************************************     GEN16080
                                                                        GEN16090
                                                                        GEN16100
        ELSE IF(CHECK.GT.4.0*TRIB/5.0) THEN                             GEN16110
           WRITE(NO,*) 'PART 11'                                        GEN16120
                IIIX(J)=T*SIGNX                                         GEN16130
                IIIY(J)=(KKIY(J)+2.0*TRIB/3.0)*SIGNY                    GEN16140
                IIX(J)=R*SIGNX                                          GEN16150
                IIY(J)=IIIY(J)                                          GEN16160
                BIX(J)=XMAX*SIGNX                                       GEN16170
                BIY(J)=IIIY(J)                                          GEN16180
                                                                        GEN16190
             CALL YALPSL (EE,RR,BIY(J),ICHK,BALPHA)                     GEN16200
             CALL THTASL (EE,RR,PI,BALPHA,BTHETA)                       GEN16210
                                                                        GEN16220
                JJIX(J)=T*SIGNX                                         GEN16230
                JJIY(J)=(KKIY(J)+TRIB/3.0)*SIGNY                        GEN16240
                JIX(J)=R*SIGNX                                          GEN16250
                JIY(J)=JJIY(J)                                          GEN16260
                CIX(J)=XMAX*SIGNX                                       GEN16270
                CIY(J)=JJIY(J)                                          GEN16280
                                                                        GEN16290
             CALL YALPSL (EE,RR,CIY(J),ICHK,CALPHA)                     GEN16300
             CALL THTASL (EE,RR,PI,CALPHA,CTHETA)                       GEN16310
                                                                        GEN16320
C     *************************************************************     GEN16330
C     *                                                           *     GEN16340
C     *               OVERLAP CONDITION TWELVE                    *     GEN16350
C     *                                                           *     GEN16360
C     *     BETWEEN 20% AND 80% OF THE CHANNEL LIES IN THE        *     GEN16370
C     *               CYLINDRICAL REGION                          *     GEN16380
C     *               (QUADRANTS TWO AND FOUR)                    *     GEN16390
C     *                                                           *     GEN16400
C     *************************************************************     GEN16410
                                                                        GEN16420
        ELSE IF ((CHECK.GT.TRIB/5.0).AND.(CHECK.LT.4.0*TRIB/5.0)) THEN  GEN16430
           WRITE(NO,*) 'PART 12'                                        GEN16440
             ANGLE=NNIY(J-1)/NNIY(J-1)                                  GEN16450
             PHIH=ATAN(ANGLE)                                           GEN16460
                                                                        GEN16470
             PHIB=PHIH+(PHIH+PHIONE)/2.0                                GEN16480
                                                                        GEN16490
                IIIX(J)=XXX(T,PHIB)                                     GEN16500
                IIIY(J)=YYY(T,PHIB)+TRANS                               GEN16510
                IIX(J)=XXX(R,PHIB)                                      GEN16520
                IIY(J)=YYY(R,PHIB)+TRANS                                GEN16530
                                                                        GEN16540
             CALL PALPSL (EE,RR,PI,PHIB,TRANS,BALPHA)                   GEN16550
             CALL THTASL (EE,RR,PI,BALPHA,BTHETA)                       GEN16560
                                                                        GEN16570
                BIX(J)=FNX(BALPHA)                                      GEN16580
                BIY(J)=FNY(BALPHA)                                      GEN16590
                                                                        GEN16600
                JJIX(J)=T*SIGNX                                         GEN16610
                JJIY(J)=(KKIY(J)+CHECK/2.0)*SIGNY                       GEN16620
                JIX(J)=R*SIGNX                                          GEN16630
                JIY(J)=JJIY(J)                                          GEN16640
```

```
            CIX(J)=XMAX*SIGNX
            CIY(J)=JJIY(J)

         CALL YALPSL (EE,RR,CIY(J),ICHK,CALPHA)
         CALL THTASL (EE,RR,PI,CALPHA,CTHETA)

       ENDIF
     ENDIF
C   *************************************************************
C   *                                                           *
C   *    THIS SECTION CALCUALTES THE INNER SHELL COORDINATES    *
C   *             FOR THE OVERLAP CONDITIONS                    *
C   *                                                           *
C   *************************************************************

  110          BBBIX(J)=FINERX(BALPHA,BTHETA,CC)
               BBBIY(J)=FINERY(BALPHA,BTHETA,CC)
               BBIX(J)=FINERX(BALPHA,BTHETA,BB)
               BBIY(J)=FINERY(BALPHA,BTHETA,BB)

               CCCIX(J)=FINERX(CALPHA,CTHETA,CC)
               CCCIY(J)=FINERY(CALPHA,CTHETA,CC)
               CCIX(J)=FINERX(CALPHA,CTHETA,BB)
               CCIY(J)=FINERY(CALPHA,CTHETA,BB)

               DDDIX(J)=FINERX(DALPHA,DTHETA,CC)
               DDDIY(J)=FINERY(DALPHA,DTHETA,CC)
               DDIX(J)=FINERX(DALPHA,DTHETA,BB)
               DDIY(J)=FINERY(DALPHA,DTHETA,BB)

               EEEIX(J)=FINERX(EALPHA,ETHETA,CC)
               EEEIY(J)=FINERY(EALPHA,ETHETA,CC)
               EEIX(J)=FINERX(EALPHA,ETHETA,BB)
               EEIY(J)=FINERY(EALPHA,ETHETA,BB)

               FFFIX(J)=FINERX(FALPHA,FTHETA,CC)
               FFFIY(J)=FINERY(FALPHA,FTHETA,CC)
               FFIX(J)=FINERX(FALPHA,FTHETA,BB)
               FFIY(J)=FINERY(FALPHA,FTHETA,BB)

               GGGIX(J)=FINERX(GALPHA,GTHETA,CC)
               GGGIY(J)=FINERY(GALPHA,GTHETA,CC)
               GGIX(J)=FINERX(GALPHA,GTHETA,BB)
               GGIY(J)=FINERY(GALPHA,GTHETA,BB)

          TRANS=ABS(TRANS)
          GO TO 125
C   *************************************************************
C   *                                                           *
C   *     END OF OVERLAPPING RIB/CAHNNEL POINT CALCULATION      *
C   *                                                           *
C   *************************************************************

C   *************************************************************
C   *                                                           *
C   *    THIS SECTION CALCUALTES THE POINT COORDINATES OF       *
C   *    THE RIB/CHANNEL SEGMENTS THAT LIE IN THE               *
C   *    RECTANGULAR REGION ON THE RIGHT SIDE OF THE HOUSING    *
C   *                                                           *
C   *************************************************************

  115     SLSLVR=SLSLVR+2
```

```
      IF (GIX(J-1).GT.0.0) THEN
     WRITE(NO,*) 'RIGHT SIDE'
      LINK=0

              TRIB=ABS(YTWO-YONE)

     AIY(J)=GIY(J-1)
     DIY(J)=YONE
     DDIY(J)=DIY(J)
     DDDIY(J)=DIY(J)
     BIY(J)=AIY(J)+ABS(DIY(J)-AIY(J))/3.0
     BBIY(J)=BIY(J)
     BBBIY(J)=BIY(J)
     CIY(J)=AIY(J)+2.0*ABS(DIY(J)-AIY(J))/3.0
     CCIY(J)=CIY(J)
     CCCIY(J)=CIY(J)

     GIY(J)=YTWO
     GGIY(J)=GIY(J)
     GGGIY(J)=GIY(J)
     EIY(J)=DIY(J)+ABS(GIY(J)-DIY(J))/3.0
     EEIY(J)=EIY(J)
     EEEIY(J)=EIY(J)
     FIY(J)=DIY(J)+2.0*ABS(GIY(J)-DIY(J))/3.0
     FFIY(J)=FIY(J)
     FFFIY(J)=FIY(J)

C    ****************************************************************
C    *                                                              *
C    *     THIS SECTION CALCUALTES THE POINT COORDINATES OF         *
C    *     THE RIB/CHANNEL SEGMENTS IF IT LIES IN THE               *
C    *     RECTANGULAR REGION ON THE LEFT SIDE OF THE HOUSING       *
C    *                                                              *
C    ****************************************************************

      ELSE IF (GIX(J-1).LT.0.0) THEN

     WRITE(NO,*) 'LEFT SIDE'

        LINK=1

     AIY(J)=GIY(J-1)
     DIY(J)=YONE
     DDIY(J)=DIY(J)
     DDDIY(J)=DIY(J)
     BIY(J)=AIY(J)-ABS(DIY(J)-AIY(J))/3.0
     BBIY(J)=BIY(J)
     BBBIY(J)=BIY(J)
     CIY(J)=AIY(J)-2.0*ABS(DIY(J)-AIY(J))/3.0
     CCIY(J)=CIY(J)
     CCCIY(J)=CIY(J)

     GIY(J)=YTWO
     GGIY(J)=GIY(J)
     GGGIY(J)=GIY(J)
     EIY(J)=DIY(J)-ABS(GIY(J)-DIY(J))/3.0
     EEIY(J)=EIY(J)
     EEEIY(J)=EIY(J)
     FIY(J)=DIY(J)-2.0*ABS(GIY(J)-DIY(J))/3.0
     FFIY(J)=FIY(J)
     FFFIY(J)=FIY(J)
```

```
       ENDIF
             IF(LINK.EQ.1) SIGN=-SIGN

      HIY(J)=NIY(J-1)
      HHIY(J)=NNIY(J-1)
      IIY(J)=BIY(J)
      IIIY(J)=BIY(J)
      JIY(J)=CIY(J)
      JJIY(J)=CIY(J)
      KIY(J)=DIY(J)
      KKIY(J)=DIY(J)
      LIY(J)=EIY(J)
      LLIY(J)=EIY(J)
      MIY(J)=FIY(J)
      MMIY(J)=FIY(J)
      NIY(J)=GIY(J)
      NNIY(J)=GIY(J)

           BIX(J)=XMAX*SIGN
           CALL YALPSL(EE,RR,BIY(J),ICHK,ALPHA)
           CALL THTASL(EE,RR,PI,ALPHA,THETA)
           BBIX(J)=FINERX(ALPHA,THETA,BB)
           BBBIX(J)=FINERX(ALPHA,THETA,CC)

           CIX(J)=XMAX*SIGN
           CALL YALPSL(EE,RR,CIY(J),ICHK,ALPHA)
           CALL THTASL(EE,RR,PI,ALPHA,THETA)
           CCIX(J)=FINERX(ALPHA,THETA,BB)
           CCCIX(J)=FINERX(ALPHA,THETA,CC)

           DIX(J)=XMAX*SIGN
           CALL YALPSL(EE,RR,DIY(J),ICHK,ALPHA)
           CALL THTASL(EE,RR,PI,ALPHA,THETA)
           DDIX(J)=FINERX(ALPHA,THETA,BB)
           DDDIX(J)=FINERX(ALPHA,THETA,CC)

           EIX(J)=XMAX*SIGN
           CALL YALPSL(EE,RR,EIY(J),ICHK,ALPHA)
           CALL THTASL(EE,RR,PI,ALPHA,THETA)
           EEIX(J)=FINERX(ALPHA,THETA,BB)
           EEEIX(J)=FINERX(ALPHA,THETA,CC)

           FIX(J)=XMAX*SIGN
           CALL YALPSL(EE,RR,FIY(J),ICHK,ALPHA)
           CALL THTASL(EE,RR,PI,ALPHA,THETA)
           FFIX(J)=FINERX(ALPHA,THETA,BB)
           FFFIX(J)=FINERX(ALPHA,THETA,CC)

           GIX(J)=XMAX*SIGN
           CALL YALPSL(EE,RR,GIY(J),ICHK,ALPHA)
           CALL THTASL(EE,RR,PI,ALPHA,THETA)
           GGIX(J)=FINERX(ALPHA,THETA,BB)
           GGGIX(J)=FINERX(ALPHA,THETA,CC)

      HIX(J)=NIX(J-1)
      HHIX(J)=NNIX(J-1)
      IIX(J)=HIX(J)
      IIIX(J)=HHIX(J)
      JIX(J)=IIX(J)
      JJIX(J)=IIIX(J)
      KIX(J)=JIX(J)
      KKIX(J)=JJIX(J)
      LIX(J)=KIX(J)
      LLIX(J)=KKIX(J)
```

```
      MIX(J)=LIX(J)
      MMIX(J)=LLIX(J)
      NIX(J)=MIX(J)
      NNIX(J)=MMIX(J)

      IF(RIBTYP(J).EQ.2) THEN

        IF(DIX(J).LT.0.0) RECES=-RECES

           DRIX(J)=DIX(J)+RECES
           DRIY(J)=DIY(J)
           ERIX(J)=EIX(J)+RECES
           ERIY(J)=EIY(J)
           FRIX(J)=FIX(J)+RECES
           FRIY(J)=FIY(J)
           GRIX(J)=GIX(J)+RECES
           GRIY(J)=GIY(J)

      ENDIF

           SIGN=ABS(SIGN)
           TRANS=ABS(TRANS)
           GO TO 125

C   ************************************************************
C   *                                                          *
C   *          END OF RECTANGULAR REGION RIB/CHANNEL           *
C   *                     POINT CALULATION                     *
C   *                                                          *
C   ************************************************************

C   ************************************************************
C   *                                                          *
C   *    THIS SECTION CALCUALTES THE POINT COORDINATES OF      *
C   *       THE RIB/CHANNEL SEGMENTS IF IT LIES IN THE         *
C   *                   CYLINDRICAL REGION                     *
C   *                                                          *
C   ************************************************************

  120     PHIA=PHIG
          PHIA1=PHIA
          PHIB=PHIA+(PHIONE-PHIA)/3.0
          PHIB1=PHIB
          PHIC=PHIA+2.0*(PHIONE-PHIA)/3.0
          PHIC1=PHIC
          PHID=PHIONE
          PHID1=PHID
          PHIE=PHIONE+(PHITWO-PHIONE)/3.0
          PHIE1=PHIE
          PHIF=PHIONE+2.0*(PHITWO-PHIONE)/3.0
          PHIF1=PHIF
          PHIG=PHITWO
          PHIG1=PHIG
          SIGN1=1.0
              IF(PHIA.GT.PI) THEN
                   PHIA1=2.0*PI-PHIA
                   PHIB1=2.0*PI-PHIB
                   PHIC1=2.0*PI-PHIC
                   PHID1=2.0*PI-PHID
                   PHIE1=2.0*PI-PHIE
                   PHIF1=2.0*PI-PHIF
                   PHIG1=2.0*PI-PHIG
                        SIGN1=-1.0
```

```
          ENDIF

      CALL PALPSL(EE,RR,PI,PHIA1,TRANS,AALPHA)
      CALL THTASL (EE,RR,PI,AALPHA,ATHETA)
      CALL PALPSL(EE,RR,PI,PHIB1,TRANS,BALPHA)
      CALL THTASL (EE,RR,PI,BALPHA,BTHETA)
      CALL PALPSL(EE,RR,PI,PHIC1,TRANS,CALPHA)
      CALL THTASL (EE,RR,PI,CALPHA,CTHETA)
      CALL PALPSL(EE,RR,PI,PHID1,TRANS,DALPHA)
      CALL THTASL (EE,RR,PI,DALPHA,DTHETA)
      CALL PALPSL(EE,RR,PI,PHIE1,TRANS,EALPHA)
      CALL THTASL (EE,RR,PI,EALPHA,ETHETA)
      CALL PALPSL(EE,RR,PI,PHIF1,TRANS,FALPHA)
      CALL THTASL (EE,RR,PI,FALPHA,FTHETA)
      CALL PALPSL(EE,RR,PI,PHIG1,TRANS,GALPHA)
      CALL THTASL (EE,RR,PI,GALPHA,GTHETA)

         BIX(J)=FNX(BALPHA)
         BIY(J)=FNY(BALPHA)*SIGN1
         BBIX(J)=FINERX(BALPHA,BTHETA,BB)
         BBIY(J)=FINERY(BALPHA,BTHETA,BB)*SIGN1
         BBBIX(J)=FINERX(BALPHA,BTHETA,CC)
         BBBIY(J)=FINERY(BALPHA,BTHETA,CC)*SIGN1

         CIX(J)=FNX(CALPHA)
         CIY(J)=FNY(CALPHA)*SIGN1
         CCIX(J)=FINERX(CALPHA,CTHETA,BB)
         CCIY(J)=FINERY(CALPHA,CTHETA,BB)*SIGN1
         CCCIX(J)=FINERX(CALPHA,CTHETA,CC)
         CCCIY(J)=FINERY(CALPHA,CTHETA,CC)*SIGN1

         DIX(J)=FNX(DALPHA)
         DIY(J)=FNY(DALPHA)*SIGN1
         DDIX(J)=FINERX(DALPHA,DTHETA,BB)
         DDIY(J)=FINERY(DALPHA,DTHETA,BB)*SIGN1
         DDDIX(J)=FINERX(DALPHA,DTHETA,CC)
         DDDIY(J)=FINERY(DALPHA,DTHETA,CC)*SIGN1

         EIX(J)=FNX(EALPHA)
         EIY(J)=FNY(EALPHA)*SIGN1
         EEIX(J)=FINERX(EALPHA,ETHETA,BB)
         EEIY(J)=FINERY(EALPHA,ETHETA,BB)*SIGN1
         EEEIX(J)=FINERX(EALPHA,ETHETA,CC)
         EEEIY(J)=FINERY(EALPHA,ETHETA,CC)*SIGN1

         FIX(J)=FNX(FALPHA)
         FIY(J)=FNY(FALPHA)*SIGN1
         FFIX(J)=FINERX(FALPHA,FTHETA,BB)
         FFIY(J)=FINERY(FALPHA,FTHETA,BB)*SIGN1
         FFFIX(J)=FINERX(FALPHA,FTHETA,CC)
         FFFIY(J)=FINERY(FALPHA,FTHETA,CC)*SIGN1

         GIX(J)=FNX(GALPHA)
         GIY(J)=FNY(GALPHA)*SIGN1
         GGIX(J)=FINERX(GALPHA,GTHETA,BB)
         GGIY(J)=FINERY(GALPHA,GTHETA,BB)*SIGN1
         GGGIX(J)=FINERX(GALPHA,GTHETA,CC)
         GGGIY(J)=FINERY(GALPHA,GTHETA,CC)*SIGN1

         HIX(J)=NIX(J-1)
         HIY(J)=NIY(J-1)
         HHIX(J)=NNIX(J-1)
         HHIY(J)=NNIY(J-1)
```

```
            IF(PHIB.GT.PI) TRANSB=-TRANSB
            IIX(J)=XXX(R,PHIB)
            IIY(J)=YYY(R,PHIB)+TRANSB
            IIIX(J)=XXX(T,PHIB)
            IIIY(J)=YYY(T,PHIB)+TRANSB

            IF(PHIC.GT.PI) TRANSC=-TRANSC
            JIX(J)=XXX(R,PHIC)
            JIY(J)=YYY(R,PHIC)+TRANSC
            JJIX(J)=XXX(T,PHIC)
            JJIY(J)=YYY(T,PHIC)+TRANSC

            IF(PHID.GT.PI) TRANSD=-TRANSD
            KIX(J)=XXX(R,PHID)
            KIY(J)=YYY(R,PHID)+TRANSD
            KKIX(J)=XXX(T,PHID)
            KKIY(J)=YYY(T,PHID)+TRANSD

            IF(PHIE.GT.PI) TRANSE=-TRANSE
            LIX(J)=XXX(R,PHIE)
            LIY(J)=YYY(R,PHIE)+TRANSE
            LLIX(J)=XXX(T,PHIE)
            LLIY(J)=YYY(T,PHIE)+TRANSE

            IF(PHIF.GT.PI) TRANSF=-TRANSF
            MIX(J)=XXX(R,PHIF)
            MIY(J)=YYY(R,PHIF)+TRANSF
            MMIX(J)=XXX(T,PHIF)
            MMIY(J)=YYY(T,PHIF)+TRANSF

            IF(PHIG.GT.PI) TRANSG=-TRANSG
            NIX(J)=XXX(R,PHIG)
            NIY(J)=YYY(R,PHIG)+TRANSG
            NNIX(J)=XXX(T,PHIG)
            NNIY(J)=YYY(T,PHIG)+TRANSG

C     ************************************************************
C     *                                                          *
C     *    IF THE RIB IS RECESSED, CALCULATE THE EXTRA POINT     *
C     *         NEEDED TO DEFINE THE RECESS OF THE RIB           *
C     *                                                          *
C     ************************************************************

      IF(RIBTYP(J).EQ.2) THEN

          DRIX(J)=DIX(J)+RECES*COS(PHID)
          DRIY(J)=DIY(J)+RECES*SIN(PHID)

          ERIX(J)=EIX(J)+RECES*COS(PHIE)
          ERIY(J)=EIY(J)+RECES*SIN(PHIE)

          FRIX(J)=FIX(J)+RECES*COS(PHIF)
          FRIY(J)=FIY(J)+RECES*SIN(PHIF)

          GRIX(J)=GIX(J)+RECES*COS(PHIG)
          GRIY(J)=GIY(J)+RECES*SIN(PHIG)

      ENDIF

C     ************************************************************
C     *                                                          *
C     *    RESET THE VALUES OF TRANS TO THEIR ORIGINAL VALUE     *
C     *                                                          *
C     ************************************************************
```

```
 125     NN=NN+7
         TRANSB=TRANS
         TRANSC=TRANS
         TRANSD=TRANS
         TRANSE=TRANS
         TRANSF=TRANS
         TRANSG=TRANS
         PHCHCK=0
         PHIT=PHITWO

 130         CONTINUE

 135      NUMBER=J-1

C    ***********************************************************
C    *                                                         *
C    *    THE FOLLOWING LIST OF ENTITY NUMBERING IS GIVEN SO   *
C    *    THAT IT IS EASIER FOR THE USER TO MAKE ANY CHANGES   *
C    *    IN THE FEM AFTER IT IS CREATED.  THE LISTING         *
C    *    FACILLITATES EASIER LOCATION OF THE ENTITIES THAT    *
C    *    ARE TO BE CHANGED.                                   *
C    *                                                         *
C    ***********************************************************

          WRITE(NO,140) NUMBER
 140      FORMAT('0','THE NUMBER OF RIB / CHANNEL SEGMENTS =',I3)
          WRITE(20,*) NUMBER
          WRITE(NO,*) '  '
          WRITE(NO,*) '  '

      IF(REGION(J).EQ.4) THEN

          GIY(J-1)=AIY(1)
          GGIY(J-1)=AIY(1)
          GGGIY(J-1)=AIY(1)
          NIY(J-1)=AIY(1)
          NNIY(J-1)=AIY(1)

      ENDIF

C    ***********************************************************
C    *                                                         *
C    *     END OF CYLINDRICAL REGION RIB/CHANNEL POINT         *
C    *                     CALCULATION                         *
C    *     (NOTE THAT THE POINT COORDINATES OF THE PORTS ARE   *
C    *       CALCULATED IN INDIVIDUAL SUBROUTINES THAT ARE     *
C    *             LOCATED AT THE END OF THIS PROGRAM)         *
C    *                                                         *
C    ***********************************************************

C    ***********************************************************
C    *                                                         *
C    *           BEGIN GENERATION OF THE UNIVERSAL FILE        *
C    *                                                         *
C    ***********************************************************


C    ***********************************************************
C    *                                                         *
C    *           GENERATE HEADING OF UNIVERSAL FILE            *
C    *                                                         *
C    ***********************************************************
```

```
      WRITE(12,150)
 150  FORMAT(4X,'-1')
      WRITE(12,155)
 155  FORMAT(5X,'2')
      WRITE(12,160)
 160  FORMAT('IDEAS 2.2: MONITOR')
      WRITE(12,150)
      WRITE(12,150)

C    **********************************************************
C    *                                                        *
C    *     BEGIN POINT COORDINATES GENERATION OF THE INNER AND *
C    *        OUTER SHELLS, RIBS, AND STIFFENED CHANNELS IN    *
C    *                     UNIVERSAL FORMAT                   *
C    *                                                        *
C    **********************************************************

      WRITE(12,165)
 165  FORMAT(4X,'25')


      IF(IEPQUE.EQ.1) THEN

               TRANS=ABS(TRANS)

               XTRAN=BB/COS(P2ODEP)
              GGIX(IEP-1)=GIX(IEP-1)+XTRAN
              GGIY(IEP-1)=GIY(IEP-1)

               XTRAN=(CC-BB)/COS(P2ODEP)
              GGGIX(IEP-1)=GGIX(IEP-1)+XTRAN
              GGGIY(IEP-1)=GGIY(IEP-1)

           PHI3=ATAN(ABS(GGIY(IEP-1)+TRANS)/GGIX(IEP-1))
           CALL PALPSL(EE,RR,PI,PHI3,TRANS,ALPHA3)
           CALL THTASL (EE,RR,PI,ALPHA3,THETA3)

           GGIX(IEP-1)=FINERX(ALPHA3,THETA3,BB)
           GGIY(IEP-1)=-FINERY(ALPHA3,THETA3,BB)

           PHI3=ATAN(ABS(GGIY(IEP-1)+TRANS)/GGIX(IEP-1))
           PHI4=P1ODEP-PHI3
           PHI5=PHI3+PHI4/3.0
           PHI6=PHI3+2.0*PHI4/3.0

           CALL PALPSL(EE,RR,PI,PHI5,TRANS,ALPHA5)
           CALL THTASL (EE,RR,PI,ALPHA5,THETA5)
           CALL PALPSL(EE,RR,PI,PHI6,TRANS,ALPHA6)
           CALL THTASL (EE,RR,PI,ALPHA6,THETA6)

           FFIX(IEP-1)=FINERX(ALPHA5,THETA5,BB)
           FFIY(IEP-1)=-FINERY(ALPHA5,THETA5,BB)

           EEIX(IEP-1)=FINERX(ALPHA6,THETA6,BB)
           EEIY(IEP-1)=-FINERY(ALPHA6,THETA6,BB)

           PHI7=ATAN(ABS(GGGIY(IEP-1)+TRANS)/GGGIX(IEP-1))
           CALL PALPSL(EE,RR,PI,PHI7,TRANS,ALPHA7)
           CALL THTASL (EE,RR,PI,ALPHA7,THETA7)

           GGGIX(IEP-1)=FINERX(ALPHA7,THETA7,CC)
           GGGIY(IEP-1)=-FINERY(ALPHA7,THETA7,CC)

           PHI7=ATAN(ABS(GGGIY(IEP-1)+TRANS)/GGGIX(IEP-1))
```

```
          PHI8=P1ODEP-PHI7
          PHI9=PHI7+PHI8/3.0
          PHI10=PHI7+2.0*PHI8/3.0

          CALL PALPSL(EE,RR,PI,PHI9,TRANS,ALPHA9)
          CALL THTASL (EE,RR,PI,ALPHA9,THETA9)
          CALL PALPSL(EE,RR,PI,PHI10,TRANS,ALPH10)
          CALL THTASL (EE,RR,PI,ALPH10,THET10)

          FFFIX(IEP-1)=FINERX(ALPHA9,THETA9,CC)
          FFFIY(IEP-1)=-FINERY(ALPHA9,THETA9,CC)

          EEEIX(IEP-1)=FINERX(ALPH10,THET10,CC)
          EEEIY(IEP-1)=-FINERY(ALPH10,THET10,CC)

          CALL EXHST (EE,RR,R,PI,REXPT,TEXPT,AA,BB,CC,D,
     #    DEPTH,TRANS,PHI1EP,PHI2EP,P1ODEP,P2ODEP,P1NWEP,
     #    GIX(IEP-1),GIY(IEP-1),GGIX(IEP-1),GGIY(IEP-1),
     #    GGGIX(IEP-1),GGGIY(IEP-1))

      TRANS=ABS(TRANS)

      ENDIF

          DO 185 I=1,NUMBER

            IPT=0
            ZZZ(I)=0.0

 170      WRITE(12,175)  IL,ICS,COLOR,BBBIX(I),BBBIY(I),ZZZ(I)
          WRITE(12,175)  IL+1,ICS,COLOR,BBIX(I),BBIY(I),ZZZ(I)
          WRITE(12,175)  IL+2,ICS,COLOR,BIX(I),BIY(I),ZZZ(I)
          WRITE(12,175)  IL+3,ICS,COLOR,IIX(I),IIY(I),ZZZ(I)
          WRITE(12,175)  IL+4,ICS,COLOR,IIIX(I),IIIY(I),ZZZ(I)
          WRITE(12,175)  IL+5,ICS,COLOR,CCCIX(I),CCCIY(I),ZZZ(I)
          WRITE(12,175)  IL+6,ICS,COLOR,CCIX(I),CCIY(I),ZZZ(I)
          WRITE(12,175)  IL+7,ICS,COLOR,CIX(I),CIY(I),ZZZ(I)
          WRITE(12,175)  IL+8,ICS,COLOR,JIX(I),JIY(I),ZZZ(I)
          WRITE(12,175)  IL+9,ICS,COLOR,JJIX(I),JJIY(I),ZZZ(I)
          WRITE(12,175)  IL+10,ICS,COLOR,DDDIX(I),DDDIY(I),ZZZ(I)
          WRITE(12,175)  IL+11,ICS,COLOR,DDIX(I),DDIY(I),ZZZ(I)
          WRITE(12,175)  IL+12,ICS,COLOR,DIX(I),DIY(I),ZZZ(I)

        IF(RIBTYP(I).EQ.2) THEN
          WRITE(12,175)  IL+13,ICS,COLOR,DRIX(I),DRIY(I),ZZZ(I)
        IL=IL+1

        ENDIF

            WRITE(12,175)  IL+13,ICS,COLOR,KIX(I),KIY(I),ZZZ(I)
            WRITE(12,175)  IL+14,ICS,COLOR,KKIX(I),KKIY(I),ZZZ(I)
            WRITE(12,175)  IL+15,ICS,COLOR,EEEIX(I),EEEIY(I),ZZZ(I)
            WRITE(12,175)  IL+16,ICS,COLOR,EEIX(I),EEIY(I),ZZZ(I)
            WRITE(12,175)  IL+17,ICS,COLOR,EIX(I),EIY(I),ZZZ(I)

          IF(RIBTYP(I).EQ.2) THEN
            WRITE(12,175)  IL+18,ICS,COLOR,ERIX(I),ERIY(I),ZZZ(I)
            IL=IL+1

          ENDIF

            WRITE(12,175)  IL+18,ICS,COLOR,LIX(I),LIY(I),ZZZ(I)
            WRITE(12,175)  IL+19,ICS,COLOR,LLIX(I),LLIY(I),ZZZ(I)
            WRITE(12,175)  IL+20,ICS,COLOR,FFFIX(I),FFFIY(I),ZZZ(I)
```

```
          WRITE(12,175) IL+21,ICS,COLOR,FFIX(I),FFIY(I),ZZZ(I)
          WRITE(12,175) IL+22,ICS,COLOR,FIX(I),FIY(I),ZZZ(I)

      IF(RIBTYP(I).EQ.2) THEN
        WRITE(12,175) IL+23,ICS,COLOR,FRIX(I),FRIY(I),ZZZ(I)
        IL=IL+1

      ENDIF

          WRITE(12,175) IL+23,ICS,COLOR,MIX(I),MIY(I),ZZZ(I)
          WRITE(12,175) IL+24,ICS,COLOR,MMIX(I),MMIY(I),ZZZ(I)
          WRITE(12,175) IL+25,ICS,COLOR,GGGIX(I),GGGIY(I),ZZZ(I)
          WRITE(12,175) IL+26,ICS,COLOR,GGIX(I),GGIY(I),ZZZ(I)
          WRITE(12,175) IL+27,ICS,COLOR,GIX(I),GIY(I),ZZZ(I)

      IF(RIBTYP(I).EQ.2) THEN
        WRITE(12,175) IL+28,ICS,COLOR,GRIX(I),GRIY(I),ZZZ(I)
        IL=IL+1

      ENDIF

          WRITE(12,175) IL+28,ICS,COLOR,NIX(I),NIY(I),ZZZ(I)
          WRITE(12,175) IL+29,ICS,COLOR,NNIX(I),NNIY(I),ZZZ(I)

  175 FORMAT(2I10,10X,I10,3E13.5)

      IL=IL+30
      ZZZ(I)=DEPTH

      IPT=IPT+1
      IF(IPT.LT.2) GO TO 170

          IF(CNLTYP(I).EQ.2) THEN

      RZZZ(I)=DEPTH/2.0

            IF(I.EQ.1) THEN
                  WRITE(12,175) IL,ICS,COLOR,AIX(I),AIY(I),RZZZ(I)
            ELSE
                  WRITE(12,175) IL,ICS,COLOR,GIX(I-1),GIY(I-1),RZZZ(I)
            ENDIF

      WRITE(12,175) IL+1,ICS,COLOR,HIX(I),HIY(I),RZZZ(I)
      WRITE(12,175) IL+2,ICS,COLOR,BIX(I),BIY(I),RZZZ(I)
      WRITE(12,175) IL+3,ICS,COLOR,IIX(I),IIY(I),RZZZ(I)
      WRITE(12,175) IL+4,ICS,COLOR,CIX(I),CIY(I),RZZZ(I)
      WRITE(12,175) IL+5,ICS,COLOR,JIX(I),JIY(I),RZZZ(I)
      WRITE(12,175) IL+6,ICS,COLOR,DIX(I),DIY(I),RZZZ(I)
      WRITE(12,175) IL+7,ICS,COLOR,KIX(I),KIY(I),RZZZ(I)

      IL=IL+8

      ENDIF


  185 CONTINUE

      LASTPT=IL-1
      TEMP=LASTPT
      NPRC=LASTPT
      WRITE(NO,*) 'NUMER OF POINTS IN RC SEGMENTS     = ',NPRC

C     ************************************************************
C     *                                                          *
```

```
C    *      GENERATE POINT COORDINATES OF THE EXHAUST PORT              *
C    *                                                                  *
C    ********************************************************************

      IF ((IEPQUE.EQ.1).OR.(IIPQUE.EQ.1)) THEN

          IF (CCC.NE.0.0) THEN

              IEND=7
          ELSE
              IEND=6

          ENDIF

              DO 210 JJ=1,IEND

      PNTT=LASTPT
      WRITE(12,205) LASTPT+1,ICS,COLOR,X1(JJ),Y1(JJ),Z1(JJ)
      WRITE(12,205) LASTPT+2,ICS,COLOR,X2(JJ),Y2(JJ),Z2(JJ)
      WRITE(12,205) LASTPT+3,ICS,COLOR,X3(JJ),Y3(JJ),Z3(JJ)
      WRITE(12,205) LASTPT+4,ICS,COLOR,X4(JJ),Y4(JJ),Z4(JJ)
      WRITE(12,205) LASTPT+5,ICS,COLOR,X5(JJ),Y5(JJ),Z5(JJ)
      WRITE(12,205) LASTPT+6,ICS,COLOR,X6(JJ),Y6(JJ),Z6(JJ)
      WRITE(12,205) LASTPT+7,ICS,COLOR,X7(JJ),Y7(JJ),Z7(JJ)
      WRITE(12,205) LASTPT+8,ICS,COLOR,X8(JJ),Y8(JJ),Z8(JJ)
      WRITE(12,205) LASTPT+9,ICS,COLOR,X9(JJ),Y9(JJ),Z9(JJ)
      WRITE(12,205) LASTPT+10,ICS,COLOR,X10(JJ),Y10(JJ),Z10(JJ)
      WRITE(12,205) LASTPT+11,ICS,COLOR,X11(JJ),Y11(JJ),Z11(JJ)
      WRITE(12,205) LASTPT+12,ICS,COLOR,X12(JJ),Y12(JJ),Z12(JJ)
      WRITE(12,205) LASTPT+13,ICS,COLOR,X13(JJ),Y13(JJ),Z13(JJ)
      WRITE(12,205) LASTPT+14,ICS,COLOR,X14(JJ),Y14(JJ),Z14(JJ)
      WRITE(12,205) LASTPT+15,ICS,COLOR,X15(JJ),Y15(JJ),Z15(JJ)
      WRITE(12,205) LASTPT+16,ICS,COLOR,X16(JJ),Y16(JJ),Z16(JJ)
      WRITE(12,205) LASTPT+17,ICS,COLOR,X17(JJ),Y17(JJ),Z17(JJ)
      WRITE(12,205) LASTPT+18,ICS,COLOR,X18(JJ),Y18(JJ),Z18(JJ)
      WRITE(12,205) LASTPT+19,ICS,COLOR,X19(JJ),Y19(JJ),Z19(JJ)
      WRITE(12,205) LASTPT+20,ICS,COLOR,X20(JJ),Y20(JJ),Z20(JJ)
      WRITE(12,205) LASTPT+21,ICS,COLOR,X21(JJ),Y21(JJ),Z21(JJ)
      WRITE(12,205) LASTPT+22,ICS,COLOR,X22(JJ),Y22(JJ),Z22(JJ)
      WRITE(12,205) LASTPT+23,ICS,COLOR,X23(JJ),Y23(JJ),Z23(JJ)
      WRITE(12,205) LASTPT+24,ICS,COLOR,X24(JJ),Y24(JJ),Z24(JJ)

         IF((JJ.EQ.3).OR.(JJ.EQ.4)) GO TO 190

      WRITE(12,205) LASTPT+25,ICS,COLOR,X25(JJ),Y25(JJ),Z25(JJ)
      WRITE(12,205) LASTPT+26,ICS,COLOR,X26(JJ),Y26(JJ),Z26(JJ)
      WRITE(12,205) LASTPT+27,ICS,COLOR,X27(JJ),Y27(JJ),Z27(JJ)
      WRITE(12,205) LASTPT+28,ICS,COLOR,X28(JJ),Y28(JJ),Z28(JJ)
      WRITE(12,205) LASTPT+29,ICS,COLOR,X29(JJ),Y29(JJ),Z29(JJ)
      WRITE(12,205) LASTPT+30,ICS,COLOR,X30(JJ),Y30(JJ),Z30(JJ)
      LASTPT=LASTPT+6

  190 LASTPT=LASTPT-6

          IF ((JJ.EQ.3).OR.(JJ.EQ.4)) GO TO 195

      WRITE(12,205) LASTPT+31,ICS,COLOR,X31(JJ),Y31(JJ),Z31(JJ)
      WRITE(12,205) LASTPT+32,ICS,COLOR,X32(JJ),Y32(JJ),Z32(JJ)
      WRITE(12,205) LASTPT+33,ICS,COLOR,X33(JJ),Y33(JJ),Z33(JJ)
      WRITE(12,205) LASTPT+34,ICS,COLOR,X34(JJ),Y34(JJ),Z34(JJ)
      LASTPT=LASTPT+4

  195 LASTPT=LASTPT-4

```

```
          IF ((JJ.EQ.3).OR.(JJ.EQ.4).OR.(JJ.EQ.5).
     #        OR.(JJ.EQ.6).OR.(JJ.EQ.7)) GO TO 200

      WRITE(12,205)  LASTPT+35,ICS,COLOR,X35(JJ),Y35(JJ),Z35(JJ)
      WRITE(12,205)  LASTPT+36,ICS,COLOR,X36(JJ),Y36(JJ),Z36(JJ)
      WRITE(12,205)  LASTPT+37,ICS,COLOR,X37(JJ),Y37(JJ),Z37(JJ)
      WRITE(12,205)  LASTPT+38,ICS,COLOR,X38(JJ),Y38(JJ),Z38(JJ)
      WRITE(12,205)  LASTPT+39,ICS,COLOR,X39(JJ),Y39(JJ),Z39(JJ)
      WRITE(12,205)  LASTPT+40,ICS,COLOR,X40(JJ),Y40(JJ),Z40(JJ)
      WRITE(12,205)  LASTPT+41,ICS,COLOR,X41(JJ),Y41(JJ),Z41(JJ)
      WRITE(12,205)  LASTPT+42,ICS,COLOR,X42(JJ),Y42(JJ),Z42(JJ)
      WRITE(12,205)  LASTPT+43,ICS,COLOR,X43(JJ),Y43(JJ),Z43(JJ)
      WRITE(12,205)  LASTPT+44,ICS,COLOR,X44(JJ),Y44(JJ),Z44(JJ)
      WRITE(12,205)  LASTPT+45,ICS,COLOR,X45(JJ),Y45(JJ),Z45(JJ)
      WRITE(12,205)  LASTPT+46,ICS,COLOR,X46(JJ),Y46(JJ),Z46(JJ)
      WRITE(12,205)  LASTPT+47,ICS,COLOR,X47(JJ),Y47(JJ),Z47(JJ)
      WRITE(12,205)  LASTPT+48,ICS,COLOR,X48(JJ),Y48(JJ),Z48(JJ)
      WRITE(12,205)  LASTPT+49,ICS,COLOR,X49(JJ),Y49(JJ),Z49(JJ)
      LASTPT=LASTPT+15

 200  LASTPT=LASTPT-15

 205  FORMAT(2I10,10X,I10,3E13.5)
      LASTPT=LASTPT+49
      PNTB=LASTPT
      NPTEP(JJ)=PNTB-PNTT

 210          CONTINUE

      ENDIF

      NPEP=LASTPT-NPRC
      WRITE(NO,*) 'NUMBER OF POINTS IN EXHAUST PORT  = ',NPEP

C    ***********************************************************
C    *                                                         *
C    *       GENERATE POINT COORDINATES OF THE INTAKE PORT     *
C    *                                                         *
C    ***********************************************************

          IF (IIPQUE.EQ.1) THEN

      CALL INTKE (EE,RR,R,PI,RINPT,TINPT,AA,BB,CC,D,DEPTH,
     #   TRANS,PHI1IP,PHI2IP,P2ODIP,P1NWIP,LINPT,WINPT,NO)

              DO 225 JJ=1,IEND

      PNTT=LASTPT
      WRITE(12,205)  LASTPT+1,ICS,COLOR,X1(JJ),Y1(JJ),Z1(JJ)
      WRITE(12,205)  LASTPT+2,ICS,COLOR,X2(JJ),Y2(JJ),Z2(JJ)
      WRITE(12,205)  LASTPT+3,ICS,COLOR,X3(JJ),Y3(JJ),Z3(JJ)
      WRITE(12,205)  LASTPT+4,ICS,COLOR,X4(JJ),Y4(JJ),Z4(JJ)
      WRITE(12,205)  LASTPT+5,ICS,COLOR,X5(JJ),Y5(JJ),Z5(JJ)
      WRITE(12,205)  LASTPT+6,ICS,COLOR,X6(JJ),Y6(JJ),Z6(JJ)
      WRITE(12,205)  LASTPT+7,ICS,COLOR,X7(JJ),Y7(JJ),Z7(JJ)
      WRITE(12,205)  LASTPT+8,ICS,COLOR,X8(JJ),Y8(JJ),Z8(JJ)
      WRITE(12,205)  LASTPT+9,ICS,COLOR,X9(JJ),Y9(JJ),Z9(JJ)
      WRITE(12,205)  LASTPT+10,ICS,COLOR,X10(JJ),Y10(JJ),Z10(JJ)
      WRITE(12,205)  LASTPT+11,ICS,COLOR,X11(JJ),Y11(JJ),Z11(JJ)
      WRITE(12,205)  LASTPT+12,ICS,COLOR,X12(JJ),Y12(JJ),Z12(JJ)
      WRITE(12,205)  LASTPT+13,ICS,COLOR,X13(JJ),Y13(JJ),Z13(JJ)
      WRITE(12,205)  LASTPT+14,ICS,COLOR,X14(JJ),Y14(JJ),Z14(JJ)
      WRITE(12,205)  LASTPT+15,ICS,COLOR,X15(JJ),Y15(JJ),Z15(JJ)
      WRITE(12,205)  LASTPT+16,ICS,COLOR,X16(JJ),Y16(JJ),Z16(JJ)
```

```
      WRITE(12,205) LASTPT+17,ICS,COLOR,X17(JJ),Y17(JJ),Z17(JJ)
      WRITE(12,205) LASTPT+18,ICS,COLOR,X18(JJ),Y18(JJ),Z18(JJ)
      WRITE(12,205) LASTPT+19,ICS,COLOR,X19(JJ),Y19(JJ),Z19(JJ)
      WRITE(12,205) LASTPT+20,ICS,COLOR,X20(JJ),Y20(JJ),Z20(JJ)
      WRITE(12,205) LASTPT+21,ICS,COLOR,X21(JJ),Y21(JJ),Z21(JJ)
      WRITE(12,205) LASTPT+22,ICS,COLOR,X22(JJ),Y22(JJ),Z22(JJ)
      WRITE(12,205) LASTPT+23,ICS,COLOR,X23(JJ),Y23(JJ),Z23(JJ)
      WRITE(12,205) LASTPT+24,ICS,COLOR,X24(JJ),Y24(JJ),Z24(JJ)
      WRITE(12,205) LASTPT+25,ICS,COLOR,X25(JJ),Y25(JJ),Z25(JJ)
      WRITE(12,205) LASTPT+26,ICS,COLOR,X26(JJ),Y26(JJ),Z26(JJ)

         IF((JJ.EQ.3).OR.(JJ.EQ.4)) GO TO 215

      WRITE(12,205) LASTPT+27,ICS,COLOR,X27(JJ),Y27(JJ),Z27(JJ)
      WRITE(12,205) LASTPT+28,ICS,COLOR,X28(JJ),Y28(JJ),Z28(JJ)
      WRITE(12,205) LASTPT+29,ICS,COLOR,X29(JJ),Y29(JJ),Z29(JJ)
      WRITE(12,205) LASTPT+30,ICS,COLOR,X30(JJ),Y30(JJ),Z30(JJ)
      WRITE(12,205) LASTPT+31,ICS,COLOR,X31(JJ),Y31(JJ),Z31(JJ)
      WRITE(12,205) LASTPT+32,ICS,COLOR,X32(JJ),Y32(JJ),Z32(JJ)
      WRITE(12,205) LASTPT+33,ICS,COLOR,X33(JJ),Y33(JJ),Z33(JJ)
      WRITE(12,205) LASTPT+34,ICS,COLOR,X34(JJ),Y34(JJ),Z34(JJ)
      WRITE(12,205) LASTPT+35,ICS,COLOR,X35(JJ),Y35(JJ),Z35(JJ)
      WRITE(12,205) LASTPT+36,ICS,COLOR,X36(JJ),Y36(JJ),Z36(JJ)
      WRITE(12,205) LASTPT+37,ICS,COLOR,X37(JJ),Y37(JJ),Z37(JJ)
      WRITE(12,205) LASTPT+38,ICS,COLOR,X38(JJ),Y38(JJ),Z38(JJ)
      LASTPT=LASTPT+12

  215 LASTPT=LASTPT-12

          IF ((JJ.EQ.3).OR.(JJ.EQ.4)) GO TO 220

      WRITE(12,205) LASTPT+39,ICS,COLOR,X39(JJ),Y39(JJ),Z39(JJ)
      WRITE(12,205) LASTPT+40,ICS,COLOR,X40(JJ),Y40(JJ),Z40(JJ)
      WRITE(12,205) LASTPT+41,ICS,COLOR,X41(JJ),Y41(JJ),Z41(JJ)
      WRITE(12,205) LASTPT+42,ICS,COLOR,X42(JJ),Y42(JJ),Z42(JJ)
      WRITE(12,205) LASTPT+43,ICS,COLOR,X43(JJ),Y43(JJ),Z43(JJ)
      WRITE(12,205) LASTPT+44,ICS,COLOR,X44(JJ),Y44(JJ),Z44(JJ)
      WRITE(12,205) LASTPT+45,ICS,COLOR,X45(JJ),Y45(JJ),Z45(JJ)
      WRITE(12,205) LASTPT+46,ICS,COLOR,X46(JJ),Y46(JJ),Z46(JJ)
      WRITE(12,205) LASTPT+47,ICS,COLOR,X47(JJ),Y47(JJ),Z47(JJ)
      WRITE(12,205) LASTPT+48,ICS,COLOR,X48(JJ),Y48(JJ),Z48(JJ)
      WRITE(12,205) LASTPT+49,ICS,COLOR,X49(JJ),Y49(JJ),Z49(JJ)
      WRITE(12,205) LASTPT+50,ICS,COLOR,X50(JJ),Y50(JJ),Z50(JJ)
      WRITE(12,205) LASTPT+51,ICS,COLOR,X51(JJ),Y51(JJ),Z51(JJ)
      WRITE(12,205) LASTPT+52,ICS,COLOR,X52(JJ),Y52(JJ),Z52(JJ)
      WRITE(12,205) LASTPT+53,ICS,COLOR,X53(JJ),Y53(JJ),Z53(JJ)
      LASTPT=LASTPT+15

  220 LASTPT=LASTPT-15

      LASTPT=LASTPT+53
      PNTB=LASTPT
      NPTIP(JJ)=PNTB-PNTT

  225           CONTINUE

            ENDIF

      NPTTL=LASTPT

      NPIP=LASTPT-NPRC-NPEP
      WRITE(NO,*) 'NUMBER OF POINTS IN INTAKE PORT    = ',NPIP

C     ************************************************************
```

```
C    *                                                             *
C    *        GENERATE POINT COORDINATES OF THE SPARK PLUG         *
C    *                                                             *
C    ***************************************************************

        IF (ISP.GT.0) THEN

             DO 235 J=1,ISP

                  CALL SPRKPG (EE,RR,R,PI,Y1SP(J),Y2SP(J),PHI1SP(J),
     #                         PHI2SP(J),RSP(J),IEND,AA,BB,CC,D,DEPTH,
     #                         TRANS,REGION(ISPRK(J)),ICHK,NO)

                       DO 230 JJ=1,IEND

        PNTT=LASTPT
        WRITE(12,205) LASTPT+1,ICS,COLOR,X1(JJ),Y1(JJ),Z1(JJ)
        WRITE(12,205) LASTPT+2,ICS,COLOR,X2(JJ),Y2(JJ),Z2(JJ)
        WRITE(12,205) LASTPT+3,ICS,COLOR,X3(JJ),Y3(JJ),Z3(JJ)
        WRITE(12,205) LASTPT+4,ICS,COLOR,X4(JJ),Y4(JJ),Z4(JJ)
        WRITE(12,205) LASTPT+5,ICS,COLOR,X5(JJ),Y5(JJ),Z5(JJ)
        WRITE(12,205) LASTPT+6,ICS,COLOR,X6(JJ),Y6(JJ),Z6(JJ)
        WRITE(12,205) LASTPT+7,ICS,COLOR,X7(JJ),Y7(JJ),Z7(JJ)
        WRITE(12,205) LASTPT+8,ICS,COLOR,X8(JJ),Y8(JJ),Z8(JJ)
        WRITE(12,205) LASTPT+9,ICS,COLOR,X9(JJ),Y9(JJ),Z9(JJ)
        WRITE(12,205) LASTPT+10,ICS,COLOR,X10(JJ),Y10(JJ),Z10(JJ)
        WRITE(12,205) LASTPT+11,ICS,COLOR,X11(JJ),Y11(JJ),Z11(JJ)
        WRITE(12,205) LASTPT+12,ICS,COLOR,X12(JJ),Y12(JJ),Z12(JJ)
        WRITE(12,205) LASTPT+13,ICS,COLOR,X13(JJ),Y13(JJ),Z13(JJ)
        WRITE(12,205) LASTPT+14,ICS,COLOR,X14(JJ),Y14(JJ),Z14(JJ)
        WRITE(12,205) LASTPT+15,ICS,COLOR,X15(JJ),Y15(JJ),Z15(JJ)
        WRITE(12,205) LASTPT+16,ICS,COLOR,X16(JJ),Y16(JJ),Z16(JJ)
        WRITE(12,205) LASTPT+17,ICS,COLOR,X17(JJ),Y17(JJ),Z17(JJ)
        WRITE(12,205) LASTPT+18,ICS,COLOR,X18(JJ),Y18(JJ),Z18(JJ)
        WRITE(12,205) LASTPT+19,ICS,COLOR,X19(JJ),Y19(JJ),Z19(JJ)
        WRITE(12,205) LASTPT+20,ICS,COLOR,X20(JJ),Y20(JJ),Z20(JJ)
        WRITE(12,205) LASTPT+21,ICS,COLOR,X21(JJ),Y21(JJ),Z21(JJ)
        WRITE(12,205) LASTPT+22,ICS,COLOR,X22(JJ),Y22(JJ),Z22(JJ)
        WRITE(12,205) LASTPT+23,ICS,COLOR,X23(JJ),Y23(JJ),Z23(JJ)
        WRITE(12,205) LASTPT+24,ICS,COLOR,X24(JJ),Y24(JJ),Z24(JJ)
        WRITE(12,205) LASTPT+25,ICS,COLOR,X25(JJ),Y25(JJ),Z25(JJ)
        WRITE(12,205) LASTPT+26,ICS,COLOR,X26(JJ),Y26(JJ),Z26(JJ)
        WRITE(12,205) LASTPT+27,ICS,COLOR,X27(JJ),Y27(JJ),Z27(JJ)
        WRITE(12,205) LASTPT+28,ICS,COLOR,X28(JJ),Y28(JJ),Z28(JJ)
        WRITE(12,205) LASTPT+29,ICS,COLOR,X29(JJ),Y29(JJ),Z29(JJ)
        WRITE(12,205) LASTPT+30,ICS,COLOR,X30(JJ),Y30(JJ),Z30(JJ)

        LASTPT=LASTPT+30
        PNTB=LASTPT
        NPTSP(JJ)=PNTB-PNTT

 230                   CONTINUE

 235         CONTINUE

        NPSP=LASTPT-NPRC-NPEP-NPIP
        WRITE(NO,*) 'NUMBER OF POINTS IN SPARK PLUG(S) = ',NPSP

        ENDIF

        WRITE(NO,*) 'TOTAL NUMBER OF POINTS = ',LASTPT
        WRITE(NO,*) '    '
        WRITE(NO,*) '    '
```

```
      WRITE(NO,*)'POINT LABELS OF RC SEGMENTS  ARE           1 TO',NPRC
      WRITE(NO,*)'POINT LABELS OF EXHST PORT ARE',NPRC+1,'TO',NPEP+NPRC
      WRITE(NO,*)'POINT LABELS OF INTKE PORT ARE',NPRC+NPEP+1,'TO',NPIP+
     #NPRC+NPEP
      WRITE(NO,*)'POINT LABELS OF SPRK PLUGS ARE',NPRC+NPEP+NPIP+1,'TO',
     #NPSP+NPRC+NPEP+NPIP
      WRITE(NO,*) '  '
      WRITE(NO,*) '  '

            LASTPT=TEMP
            TEMP1=TEMP
      WRITE(12,150)
      WRITE(12,150)

C    *************************************************************
C    *                                                           *
C    *          END GENERATION OF THE POINT COORDINATES          *
C    *                                                           *
C    *************************************************************

C    *************************************************************
C    *                                                           *
C    *   BEGIN LINE GENERATION OF THE INNER AND OUTER SHELLS,    *
C    *   RIBS, AND STIFFENED CHANNELS IN UNIVERSAL FORMAT        *
C    *                                                           *
C    *************************************************************

      CALL COIN (NO)
      CALL FORGEN (COLOR,SOLID,DASH,GRADE,NUMBER,TEMP,TEMP1,
     #              IEND,LASTPT,IEPQUE,IIPQUE,SDC,NO,ICONTI)

      CALL MNE (NO)
      CALL GO (ICONTI)

      STOP
      END

C    *************************************************************
C    *                                                           *
C    *                BEGIN SUBROUTINE LIBRARY                   *
C    *                                                           *
C    *************************************************************

      SUBROUTINE FORGEN (COLOR,SOLID,DASH,GRADE,NUMBER,TEMP,TEMP1,
     #                   IEND,LASTPT,IEPQUE,IIPQUE,SDC,NO,ICONTI)

      DIMENSION NLNEP(7),NLNEP1(7),NLNIP(7),NLNIP1(7),NSREP(7),NSRIP(7)
     #          ,NLNIP2(7),NSIP2(7),NLNSP(7),NSRSP(7),NSIP3(7)

      COMMON / MAIN2 / CNLTYP(100),RIBTYP(100),NPTEP(7),NPTIP(7)
     #                 ,NPTSP(7),NPTTL,ISP,IVOLR(25),IVOLC(25),TCHNL,CCC

         INTEGER PT1,PT2,PT3,PT4,RIBTYP,CNLTYP,SDC
     #     ,SOLID,COLOR,START,STOP,COUNT,SPLINE
     #     ,SPENTC,LNENTC,DASH,GRADE,SURFAC,ED1,ED2,ED3,ED4
     #     ,VOLUME,SUR1,SUR2,SUR3,SUR4,SUR5,SUR6,SLSLVR
     #     ,SURT,SURB,EDGE,TEMPID,TEMPI1

      WRITE(12,240)
  240 FORMAT(4X,'26')
  245 FORMAT(4X,'-1')

      LINE=1
      START=11
```

```
      STOP=12
      WRITE(12,250) LINE,COLOR,SOLID,START,STOP
 250  FORMAT(5I10)

      DO 285 I=1,NUMBER

           COUNT=0

      IF(RIBTYP(I).EQ.1) THEN

 255       DO 260 J=1,3

                START=START+1
                STOP=STOP+1
                LINE=LINE+1
                WRITE(12,250) LINE,COLOR,SOLID,START,STOP

 260       CONTINUE

           START=START-3
           STOP=STOP+26
           LINE=LINE+1
           WRITE(12,250) LINE,COLOR,SOLID,START,STOP

           DO 265 J=1,4

                START=START+1
                STOP=STOP+1
                LINE=LINE+1
                WRITE(12,250) LINE,COLOR,SOLID,START,STOP

 265       CONTINUE

           START=START+26
           STOP=STOP-3
           LINE=LINE+1
           WRITE(12,250) LINE,COLOR,SOLID,START,STOP

           DO 270 J=1,3

                START=START+1
                STOP=STOP+1
                LINE=LINE+1
                WRITE(12,250) LINE,COLOR,SOLID,START,STOP

 270       CONTINUE

      COUNT=COUNT+1
      IF(COUNT.EQ.1) THEN

           START=START-18
           STOP=STOP-18
           LINE=LINE+1
           WRITE(12,250) LINE,COLOR,SOLID,START,STOP
           GO TO 255

      ENDIF

      ELSE IF(RIBTYP(I).EQ.2) THEN

           COUNT=0

 275       START=START+1
           STOP=STOP+1
```

```
          LINE=LINE+1
          WRITE(12,250) LINE,COLOR,SOLID,START,STOP

          START=START+2
          STOP=STOP+2
          LINE=LINE+1
          WRITE(12,250) LINE,COLOR,SOLID,START,STOP

          START=START+1
          STOP=STOP+1
          LINE=LINE+1
          WRITE(12,250) LINE,COLOR,SOLID,START,STOP

          START=START-4
          STOP=STOP+29
          LINE=LINE+1
          WRITE(12,250) LINE,COLOR,SOLID,START,STOP

          DO 280 J=1,5

               START=START+1
               STOP=STOP+1
               LINE=LINE+1
               WRITE(12,250) LINE,COLOR,SOLID,START,STOP

 280      CONTINUE

          START=START+29
          STOP=STOP-4
          LINE=LINE+1
          WRITE(12,250) LINE,COLOR,SOLID,START,STOP

          START=START+1
          STOP=STOP+1
          LINE=LINE+1
          WRITE(12,250) LINE,COLOR,SOLID,START,STOP

          START=START+2
          STOP=STOP+2
          LINE=LINE+1
          WRITE(12,250) LINE,COLOR,SOLID,START,STOP

          START=START+1
          STOP=STOP+1
          LINE=LINE+1
          WRITE(12,250) LINE,COLOR,SOLID,START,STOP

     COUNT=COUNT+1
     IF(COUNT.EQ.1) THEN

          START=START-20
          STOP=STOP-20
          LINE=LINE+1
          WRITE(12,250) LINE,COLOR,SOLID,START,STOP
          GO TO 275

     ENDIF
     ENDIF

     IF(RIBTYP(I).EQ.1) THEN

          IF(CNLTYP(I).EQ.1) THEN

               IF(I.EQ.NUMBER) GO TO 285
```

```
            START=START+12
            STOP=STOP+12
            LINE=LINE+1
            WRITE(12,250) LINE,COLOR,SOLID,START,STOP

            ELSE IF(CNLTYP(I).EQ.2) THEN

            START=START+2
            STOP=STOP+2
            LINE=LINE+1
            WRITE(12,250) LINE,COLOR,SOLID,START,STOP

            START=START+6
            STOP=STOP+6
            LINE=LINE+1
            WRITE(12,250) LINE,COLOR,SOLID,START,STOP

                 IF(I.EQ.NUMBER) GO TO 285
            START=START+12
            STOP=STOP+12
            LINE=LINE+1
            WRITE(12,250) LINE,COLOR,SOLID,START,STOP
         ENDIF

      ELSE IF(RIBTYP(I).EQ.2) THEN

                 IF(I.EQ.NUMBER) GO TO 285
            START=START+12
            STOP=STOP+12
            LINE=LINE+1
            WRITE(12,250) LINE,COLOR,SOLID,START,STOP

      ENDIF
  285 CONTINUE

      NLRC=LINE
      WRITE(NO,*) 'NUMBER OF LINES IN RC SEGMENTS   = ',NLRC
C    ****************************************************************
C    *                                                              *
C    *       BEGIN LINE GENERATION OF THE EXHAUST PORT             *
C    *                                                              *
C    ****************************************************************

      LINETP=LINE
      IF (IEPQUE.EQ.1) THEN

         DO 330 JJ=1,IEND

          LNT=LINE

             START=TEMP+1
             STOP=TEMP+2
             LINE=LINE+1
             WRITE(12,250) LINE,COLOR,SOLID,START,STOP
             ICK=1

                DO 290 J=1,22

                   START=START+1
                   STOP=STOP+1
                   LINE=LINE+1
                   WRITE(12,250) LINE,COLOR,SOLID,START,STOP
```

```
                    IF((J.EQ.10).OR.(J.EQ.22)) THEN

                       START=STOP
                       STOP=STOP-11
                       LINE=LINE+1
                       WRITE(12,250) LINE,COLOR,SOLID,START,STOP
                       START=START-1
                       STOP=STOP+11

                    ENDIF

 290             CONTINUE

             START=TEMP+1
             STOP=TEMP+14
             LINE=LINE+1
             WRITE(12,250) LINE,COLOR,SOLID,START,STOP
             ICK1=1

                 DO 295 J=1,10

                    START=START+1
                    STOP=STOP+1
                    LINE=LINE+1
                    WRITE(12,250) LINE,COLOR,SOLID,START,STOP

 295             CONTINUE

       IF((JJ.EQ.3).OR.(JJ.EQ.4)) GO TO 310

             START=TEMP+15
             STOP=TEMP+25
             LINE=LINE+1
             WRITE(12,250) LINE,COLOR,SOLID,START,STOP

                 DO 300 J=1,8

                    START=START+1
                    STOP=STOP+1
                    LINE=LINE+1
                    WRITE(12,250) LINE,COLOR,SOLID,START,STOP

                       IF (J.EQ.4) START=START+1
 300             CONTINUE

             START=TEMP+13
             STOP=TEMP+34
             LINE=LINE+1
             WRITE(12,250) LINE,COLOR,SOLID,START,STOP

             START=TEMP+25
             STOP=TEMP+26
             LINE=LINE+1
             WRITE(12,250) LINE,COLOR,SOLID,START,STOP

                 DO 305 J=1,7

                    START=START+1
                    STOP=STOP+1
                    LINE=LINE+1
                    WRITE(12,250) LINE,COLOR,SOLID,START,STOP

                       IF (J.EQ.3) THEN
```

```
                              START=START+1
                              STOP=STOP+1
                          ENDIF

 305              CONTINUE

 310        IF(JJ.EQ.IEND) GO TO 325

               START=TEMP+1
               STOP=TEMP+1+NPTEP(JJ)
               LINE=LINE+1
               WRITE(12,250) LINE,COLOR,SOLID,START,STOP

                   DO 315 J=1,23

                       START=START+1
                       STOP=STOP+1
                       LINE=LINE+1
                       WRITE(12,250) LINE,COLOR,SOLID,START,STOP

 315               CONTINUE

            IF((JJ.EQ.2).OR.(JJ.EQ.3).OR.(JJ.EQ.4)) GO TO 325

               START=TEMP+25
               STOP=TEMP+25+NPTEP(JJ)
               LINE=LINE+1
               WRITE(12,250) LINE,COLOR,SOLID,START,STOP

                   DO 320 J=1,9

                       START=START+1
                       STOP=STOP+1
                       LINE=LINE+1
                       WRITE(12,250) LINE,COLOR,SOLID,START,STOP

 320               CONTINUE

 325           TEMP=TEMP+NPTEP(JJ)
               LNB=LINE
               NLNEP(JJ)=LNB-LNT

 330        CONTINUE

            DO 355 JJ=1,2

               LNT=LINE
               START=TEMP1+35
               STOP=TEMP1+36
               LINE=LINE+1
               WRITE(12,250) LINE,COLOR,SOLID,START,STOP

                   DO 335 J=1,11

                       START=START+1
                       STOP=STOP+1
                       LINE=LINE+1
                       WRITE(12,250) LINE,COLOR,SOLID,START,STOP

                         IF((J.EQ.3).OR.(J.EQ.7)) THEN

                             START=START+1
                             STOP=STOP+1
```

```
                 ENDIF

 335          CONTINUE

          START=TEMP1+35
          STOP=TEMP1+40
          LINE=LINE+1
          WRITE(12,250) LINE,COLOR,SOLID,START,STOP

              DO 340 J=1,9

                 START=START+1
                 STOP=STOP+1
                 LINE=LINE+1
                 WRITE(12,250) LINE,COLOR,SOLID,START,STOP

 340          CONTINUE

          START=TEMP1+13
          STOP=TEMP1+36
          LINE=LINE+1
          WRITE(12,250) LINE,COLOR,SOLID,START,STOP

             DO 345 J=1,2

                  START=START+1
                  STOP=STOP+1
                  LINE=LINE+1
                  WRITE(12,250) LINE,COLOR,SOLID,START,STOP

 345         CONTINUE

          START=TEMP1+25
          STOP=TEMP1+39
          LINE=LINE+1
          WRITE(12,250) LINE,COLOR,SOLID,START,STOP

          START=TEMP1+34
          STOP=TEMP1+35
          LINE=LINE+1
          WRITE(12,250) LINE,COLOR,SOLID,START,STOP

        IF(JJ.EQ.2) GO TO 355

          START=TEMP1+35
          STOP=TEMP1+35+NPTEP(JJ)
          LINE=LINE+1
          WRITE(12,250) LINE,COLOR,SOLID,START,STOP

             DO 350 J=1,14

                  START=START+1
                  STOP=STOP+1
                  LINE=LINE+1
                  WRITE(12,250) LINE,COLOR,SOLID,START,STOP

 350         CONTINUE

          TEMP1=TEMP1+NPTEP(JJ)
          LNB=LINE
          NLNEP1(JJ)=LNB-LNT

 355    CONTINUE
```

```
      ENDIF
                TEMP1=TEMP

      NLEP=LINE-NLRC
      WRITE(NO,*) 'NUMBER OF LINES IN EXHAUST PORT  = ',NLEP

C    *************************************************************
C    *                                                           *
C    *        BEGIN LINE GENERATION OF THE INTAKE PORT           *
C    *                                                           *
C    *************************************************************

       IF (IIPQUE.EQ.1) THEN

           DO 400 JJ=1,IEND

              LNT=LINE
              START=TEMP+1
              STOP=TEMP+2
              LINE=LINE+1
              WRITE(12,250) LINE,COLOR,SOLID,START,STOP
              ICK=1

                 DO 360 J=1,25

                    START=START+1
                    STOP=STOP+1
                    LINE=LINE+1
                    WRITE(12,250) LINE,COLOR,SOLID,START,STOP

                      IF(J.EQ.10) THEN

                        START=STOP
                        STOP=STOP-11
                        LINE=LINE+1
                        WRITE(12,250) LINE,COLOR,SOLID,START,STOP
                        START=START-1
                        STOP=STOP+11

                      ENDIF

                             IF (J.EQ.24) STOP=STOP-14

 360                CONTINUE

              START=TEMP+1
              STOP=TEMP+14
              LINE=LINE+1
              WRITE(12,250) LINE,COLOR,SOLID,START,STOP
              ICK1=1

                 DO 365 J=1,12

                    START=START+1
                    STOP=STOP+1
                    LINE=LINE+1
                    WRITE(12,250) LINE,COLOR,SOLID,START,STOP

                        IF (J.EQ.1) START=START-1

 365                CONTINUE

       IF((JJ.EQ.3).OR.(JJ.EQ.4)) GO TO 380
```

```
              START=TEMP+15
              STOP=TEMP+27
              LINE=LINE+1
              WRITE(12,250) LINE,COLOR,SOLID,START,STOP

                 DO 370 J=1,11

                    START=START+1
                    STOP=STOP+1
                    LINE=LINE+1
                    WRITE(12,250) LINE,COLOR,SOLID,START,STOP

                       IF (J.EQ.5) START=START+1
                        IF (J.EQ.10) START=START-14

370              CONTINUE

              START=TEMP+27
              STOP=TEMP+28
              LINE=LINE+1
              WRITE(12,250) LINE,COLOR,SOLID,START,STOP

                 DO 375 J=1,9

                    START=START+1
                    STOP=STOP+1
                    LINE=LINE+1
                    WRITE(12,250) LINE,COLOR,SOLID,START,STOP

                       IF (J.EQ.4) START=START+1
                       IF (J.EQ.4) STOP=STOP+1

375              CONTINUE

380         IF(JJ.EQ.IEND) GO TO 395

              START=TEMP+1
              STOP=TEMP+1+NPTIP(JJ)
              LINE=LINE+1
              WRITE(12,250) LINE,COLOR,SOLID,START,STOP

                 DO 385 J=1,25

                    START=START+1
                    STOP=STOP+1
                    LINE=LINE+1
                    WRITE(12,250) LINE,COLOR,SOLID,START,STOP

385              CONTINUE

            IF((JJ.EQ.2).OR.(JJ.EQ.3).OR.(JJ.EQ.4)) GO TO 395

              START=TEMP+27
              STOP=TEMP+27+NPTIP(JJ)
              LINE=LINE+1
              WRITE(12,250) LINE,COLOR,SOLID,START,STOP

                 DO 390 J=1,11

                    START=START+1
                    STOP=STOP+1
                    LINE=LINE+1
                    WRITE(12,250) LINE,COLOR,SOLID,START,STOP
```

```
390             CONTINUE

395         TEMP=TEMP+NPTIP(JJ)
            IF (JJ.EQ.5) TEMP2=TEMP-NPTIP(JJ)
            LNB=LINE
            NLNIP(JJ)=LNB-LNT

               IF (JJ.EQ.1) I2PT21=TEMP+21
               IF (JJ.EQ.1) I2PT41=TEMP+41
               IF (JJ.EQ.1) I2PT46=TEMP+46
               IF (JJ.EQ.1) I2PT51=TEMP+51
               IF (JJ.EQ.4) I5PT21=TEMP+21
               IF (JJ.EQ.4) I5PT41=TEMP+41
               IF (JJ.EQ.4) I5PT46=TEMP+46
               IF (JJ.EQ.4) I5PT51=TEMP+51

 400       CONTINUE

          DO 430 JJ=1,2

             LNT=LINE
             START=TEMP1+39
             STOP=TEMP1+40
             LINE=LINE+1
             WRITE(12,250) LINE,COLOR,SOLID,START,STOP

                DO 405 J=1,11

                   START=START+1
                   STOP=STOP+1
                   LINE=LINE+1
                   WRITE(12,250) LINE,COLOR,SOLID,START,STOP

                     IF((J.EQ.3).OR.(J.EQ.7)) THEN

                        START=START+1
                        STOP=STOP+1

                     ENDIF

  405           CONTINUE

             START=TEMP1+39
             STOP=TEMP1+44
             LINE=LINE+1
             WRITE(12,250) LINE,COLOR,SOLID,START,STOP

                  DO 410 J=1,9

                     START=START+1
                     STOP=STOP+1
                     LINE=LINE+1
                     WRITE(12,250) LINE,COLOR,SOLID,START,STOP

   410            CONTINUE

              START=TEMP1+22
              STOP=TEMP1+40
              LINE=LINE+1
              WRITE(12,250) LINE,COLOR,SOLID,START,STOP

                 DO 415 J=1,2

                     START=START-1
```

```
                  STOP=STOP+1
                  LINE=LINE+1
                  WRITE(12,250) LINE,COLOR,SOLID,START,STOP

 415           CONTINUE

            START=TEMP1+32
            STOP=TEMP1+43
            LINE=LINE+1
            WRITE(12,250) LINE,COLOR,SOLID,START,STOP

            START=TEMP1+33
            STOP=TEMP1+39
            LINE=LINE+1
            WRITE(12,250) LINE,COLOR,SOLID,START,STOP

          IF(JJ.EQ.2) GO TO 425

            START=TEMP1+39
            STOP=TEMP1+39+NPTIP(JJ)
            LINE=LINE+1
            WRITE(12,250) LINE,COLOR,SOLID,START,STOP

               DO 420 J=1,14

                  START=START+1
                  STOP=STOP+1
                  LINE=LINE+1
                  WRITE(12,250) LINE,COLOR,SOLID,START,STOP

 420           CONTINUE

 425        TEMP1=TEMP1+NPTIP(JJ)
            LNB=LINE
            NLNIP1(JJ)=LNB-LNT

 430     CONTINUE

         DO 460 JJ=5,IEND

            LNT=LINE
            START=TEMP2+39
            STOP=TEMP2+40
            LINE=LINE+1
            WRITE(12,250) LINE,COLOR,SOLID,START,STOP

               DO 435 J=1,11

                  START=START+1
                  STOP=STOP+1
                  LINE=LINE+1
                  WRITE(12,250) LINE,COLOR,SOLID,START,STOP

                    IF((J.EQ.3).OR.(J.EQ.7)) THEN

                       START=START+1
                       STOP=STOP+1

                    ENDIF

 435           CONTINUE

            START=TEMP2+39
            STOP=TEMP2+44
```

```
            LINE=LINE+1
            WRITE(12,250) LINE,COLOR,SOLID,START,STOP

                DO 440 J=1,9

                  START=START+1
                  STOP=STOP+1
                  LINE=LINE+1
                  WRITE(12,250) LINE,COLOR,SOLID,START,STOP

 440            CONTINUE

            START=TEMP2+22
            STOP=TEMP2+40
            LINE=LINE+1
            WRITE(12,250) LINE,COLOR,SOLID,START,STOP

               DO 445 J=1,2

                   START=START-1
                   STOP=STOP+1
                   LINE=LINE+1
                   WRITE(12,250) LINE,COLOR,SOLID,START,STOP

 445           CONTINUE

            START=TEMP2+32
            STOP=TEMP2+43
            LINE=LINE+1
            WRITE(12,250) LINE,COLOR,SOLID,START,STOP

            START=TEMP2+33
            STOP=TEMP2+39
            LINE=LINE+1
            WRITE(12,250) LINE,COLOR,SOLID,START,STOP

          IF(JJ.EQ.IEND) GO TO 455

            START=TEMP2+39
            STOP=TEMP2+39+NPTIP(JJ)
            LINE=LINE+1
            WRITE(12,250) LINE,COLOR,SOLID,START,STOP

               DO 450 J=1,14

                   START=START+1
                   STOP=STOP+1
                   LINE=LINE+1
                   WRITE(12,250) LINE,COLOR,SOLID,START,STOP

 450           CONTINUE

 455        TEMP2=TEMP2+NPTIP(JJ)
            LNB=LINE
            NLNIP2(JJ)=LNB-LNT

 460     CONTINUE

      ENDIF

      NLIP=LINE-NLEP-NLRC+6
      WRITE(NO,*) 'NUMBER OF LINES IN INTAKE PORT    = ',NLIP
         NLNTL=LINE
```

```
C     ****************************************************************
C     *                                                              *
C     *          BEGIN LINE GENERATION OF THE SPARK PLUG             *
C     *                                                              *
C     ****************************************************************

        DO 490 JJJ=1,ISP

          DO 485 JJ=1,IEND

            LNT=LINE
            START=NPTTL+1
            STOP=NPTTL+2
            LINE=LINE+1
            WRITE(12,250) LINE,COLOR,SOLID,START,STOP

               DO 465 J=1,34

                  START=START+1
                  STOP=STOP+1
                  LINE=LINE+1
                  WRITE(12,250) LINE,COLOR,SOLID,START,STOP

                     IF (J.EQ.6) STOP=STOP-8

                       IF (J.EQ.7) START=START-1
                       IF (J.EQ.7) STOP=STOP+7

                         IF (J.EQ.15) STOP=STOP-8

                           IF (J.EQ.16) STOP=STOP+8

                             IF (J.EQ.23) STOP=STOP-8

                             IF (J.EQ.24) STOP=STOP+1

                           IF (J.EQ.27) STOP=STOP+1

                         IF (J.EQ.29) STOP=STOP-8

                       IF (J.EQ.30) START=START-5
                       IF (J.EQ.30) STOP=STOP+7

                     IF (J.EQ.32) START=START+1
                     IF (J.EQ.32) STOP=STOP+1
 465           CONTINUE

          START=NPTTL+1
          STOP=NPTTL+10
          LINE=LINE+1
          WRITE(12,250) LINE,COLOR,SOLID,START,STOP

             DO 470 J=1,14

                START=START+1
                STOP=STOP+1
                LINE=LINE+1
                WRITE(12,250) LINE,COLOR,SOLID,START,STOP

                   IF (J.EQ.6) START=START+1
 470         CONTINUE
```

```
          IF (JJ.EQ.IEND) GO TO 480

            START=NPTTL+1
            STOP=NPTTL+1+NPTSP(JJ)
            LINE=LINE+1
            WRITE(12,250) LINE,COLOR,SOLID,START,STOP

                 DO 475 J=1,29

                      START=START+1
                      STOP=STOP+1
                      LINE=LINE+1
                      WRITE(12,250) LINE,COLOR,SOLID,START,STOP

 475             CONTINUE

 480        NPTTL=NPTTL+NPTSP(JJ)
            LNB=LINE
            NLNSP(JJ)=LNB-LNT

 485      CONTINUE

 490   CONTINUE

       NLSP=LINE-NLRC-NLEP-NLIP+6
       WRITE(NO,*) 'NUMBER OF LINES IN SPARK PLUG(S) = ',NLSP

         IF (IIPQUE.EQ.1) THEN

            START=I2PT21
            STOP=I2PT41
            LINE=LINE+1
            WRITE(12,250) LINE,COLOR,SOLID,START,STOP
            ISTCN1=LINE

                 START=I2PT41
                 STOP=I2PT46
                 LINE=LINE+1
                 WRITE(12,250) LINE,COLOR,SOLID,START,STOP
                 ISTCN2=LINE

                      START=I2PT46
                      STOP=I2PT51
                      LINE=LINE+1
                      WRITE(12,250) LINE,COLOR,SOLID,START,STOP
                      ISTCN3=LINE

                          START=I2PT51
                          STOP=I5PT51
                          LINE=LINE+1
                          WRITE(12,250) LINE,COLOR,SOLID,START,STOP
                          ISTCN4=LINE

                           START=I5PT51
                           STOP=I5PT46
                           LINE=LINE+1
                           WRITE(12,250) LINE,COLOR,SOLID,START,STOP
                           ISTCN5=LINE

                            START=I5PT46
                            STOP=I5PT41
                            LINE=LINE+1
                            WRITE(12,250) LINE,COLOR,SOLID,START,STOP
                            ISTCN6=LINE
```

```
                          START=I5PT41
                          STOP=I5PT21
                          LINE=LINE+1
                          WRITE(12,250) LINE,COLOR,SOLID,START,STOP
                          ISTCN7=LINE

                            START=I5PT21
                            STOP=I2PT21
                            LINE=LINE+1
                            WRITE(12,250) LINE,COLOR,SOLID,START,STOP
                            ISTCN8=LINE

         ENDIF

      WRITE(NO,*) 'TOTAL NUMBER OF LINES = ',LINE
      WRITE(NO,*) '  '
      WRITE(NO,*) '  '

      WRITE(NO,*)'LINE LABELS OF RC SEGMENTS  ARE           1 TO',NLRC
      WRITE(NO,*)'LINE LABELS OF EXHST PORT ARE ',NLRC+1,'TO',NLEP+NLRC
      WRITE(NO,*)'LINE LABELS OF INTKE PORT ARE',NLRC+NLEP+1,'TO',NLIP+
     #NLRC+NLEP
      WRITE(NO,*)'LINE LABELS OF SPRK PLUG ARE',NLRC+NLEP+NLIP+1,'TO',
     #NLSP+NLRC+NLEP+NLIP
      WRITE(NO,*) '  '
      WRITE(NO,*) '  '

      WRITE(12,245)
      WRITE(12,245)

C   *************************************************************
C   *                                                           *
C   *    END GENERATION OF THE LINES IN UNIVERSAL FORMAT        *
C   *                                                           *
C   *************************************************************

C   *************************************************************
C   *                                                           *
C   *    BEGIN SPLINE GENERATION OF THE INNER AND OUTER SHELLS, *
C   *    RIBS, AND STIFFENED CHANNELS IN UNIVERSAL FORMAT       *
C   *    (NOTE THAT THERE ARE NO SPLINES IN ANY OF THE PORTS)   *
C   *                                                           *
C   *************************************************************

      WRITE(12,495)
 495  FORMAT(4X,'28')
      SPLINE=1
      NUMB=4

      PT1=LASTPT-34
      PT2=1
      PT3=6
      PT4=11
      WRITE(12,520) SPLINE,SDC,COLOR,SOLID,NUMB
 520  FORMAT(5I10)
      WRITE(12,525) PT1,PT2,PT3,PT4
 525  FORMAT(4I10)

      DO 560 I=1,NUMBER

           IF(RIBTYP(I).EQ.1) THEN

               DO 530 J=1,4
```

```
                 PT1=PT1+1
             IF((RIBTYP(I-1).EQ.2).AND.(J.EQ.3)) PT1=PT1+1
                 PT2=PT2+1
                 PT3=PT3+1
                 PT4=PT4+1
                 SPLINE=SPLINE+1
                 WRITE(12,520) SPLINE,SDC,COLOR,SOLID,NUMB
                 WRITE(12,525) PT1,PT2,PT3,PT4

 530          CONTINUE

              PT1=PT1+26
             IF(RIBTYP(I-1).EQ.2) PT1=PT1+3
              PT2=PT2+26
              PT3=PT3+26
              PT4=PT4+26
              SPLINE=SPLINE+1
              WRITE(12,520) SPLINE,SDC,COLOR,SOLID,NUMB
              WRITE(12,525) PT1,PT2,PT3,PT4

              DO 535 J=1,4

                 PT1=PT1+1
             IF((RIBTYP(I-1).EQ.2).AND.(J.EQ.3)) PT1=PT1+1
                 PT2=PT2+1
                 PT3=PT3+1
                 PT4=PT4+1
                 SPLINE=SPLINE+1
                 WRITE(12,520) SPLINE,SDC,COLOR,SOLID,NUMB
                 WRITE(12,525) PT1,PT2,PT3,PT4

 535          CONTINUE

          IF(I.EQ.1) THEN

              PT1=11

          ELSE

              PT1=PT1+11
            IF(CNLTYP(I-1).EQ.2) PT1=PT1+8

          ENDIF

              PT2=PT2-19
              PT3=PT3-19
              PT4=PT4-19
              SPLINE=SPLINE+1
              WRITE(12,520) SPLINE,SDC,COLOR,SOLID,NUMB
              WRITE(12,525) PT1,PT2,PT3,PT4

              DO 540 J=1,4

                PT1=PT1+1
                PT2=PT2+1
                PT3=PT3+1
                PT4=PT4+1
                SPLINE=SPLINE+1
                WRITE(12,520) SPLINE,SDC,COLOR,SOLID,NUMB
                WRITE(12,525) PT1,PT2,PT3,PT4

 540          CONTINUE
```

```
              PT1=PT1+26
              PT2=PT2+26
              PT3=PT3+26
              PT4=PT4+26
              SPLINE=SPLINE+1
              WRITE(12,520) SPLINE,SDC,COLOR,SOLID,NUMB
              WRITE(12,525) PT1,PT2,PT3,PT4

              DO 545 J=1,4

                PT1=PT1+1
                PT2=PT2+1
                PT3=PT3+1
                PT4=PT4+1
                SPLINE=SPLINE+1
                WRITE(12,520) SPLINE,SDC,COLOR,SOLID,NUMB
                WRITE(12,525) PT1,PT2,PT3,PT4
 545          CONTINUE

        ELSE IF(RIBTYP(I).EQ.2) THEN

        DO 547 J=1,2
             PT1=PT1+1
             PT2=PT2+1
             PT3=PT3+1
             PT4=PT4+1
             SPLINE=SPLINE+1
             WRITE(12,520) SPLINE,SDC,COLOR,SOLID,NUMB
             WRITE(12,525) PT1,PT2,PT3,PT4
 547    CONTINUE

             PT1=PT1+2
            IF(RIBTYP(I-1).EQ.1) PT1=PT1-1
             PT2=PT2+1
             PT3=PT3+1
             PT4=PT4+2
             SPLINE=SPLINE+1
             WRITE(12,520) SPLINE,SDC,COLOR,SOLID,NUMB
             WRITE(12,525) PT1,PT2,PT3,PT4

             PT1=PT1+1
             PT2=PT2+1
             PT3=PT3+1
             PT4=PT4+1
             SPLINE=SPLINE+1
             WRITE(12,520) SPLINE,SDC,COLOR,SOLID,NUMB
             WRITE(12,525) PT1,PT2,PT3,PT4

             PT1=PT1+29
            IF(RIBTYP(I-1).EQ.1) PT1=PT1-3
             PT2=PT2+30
             PT3=PT3+30
             PT4=PT4+29
             SPLINE=SPLINE+1
             WRITE(12,520) SPLINE,SDC,COLOR,SOLID,NUMB
             WRITE(12,525) PT1,PT2,PT3,PT4

        DO 549 J=1,2
             PT1=PT1+1
             PT2=PT2+1
             PT3=PT3+1
             PT4=PT4+1
```

```
            SPLINE=SPLINE+1
            WRITE(12,520) SPLINE,SDC,COLOR,SOLID,NUMB
            WRITE(12,525) PT1,PT2,PT3,PT4

 549      CONTINUE

            PT1=PT1+2
           IF(RIBTYP(I-1).EQ.1) PT1=PT1-1
            PT2=PT2+1
            PT3=PT3+1
            PT4=PT4+2
            SPLINE=SPLINE+1
            WRITE(12,520) SPLINE,SDC,COLOR,SOLID,NUMB
            WRITE(12,525) PT1,PT2,PT3,PT4

            PT1=PT1+1
            PT2=PT2+1
            PT3=PT3+1
            PT4=PT4+1
            SPLINE=SPLINE+1
            WRITE(12,520) SPLINE,SDC,COLOR,SOLID,NUMB
            WRITE(12,525) PT1,PT2,PT3,PT4

            PT1=PT1+11
           IF(CNLTYP(I-1).EQ.2) PT1=PT1+8
            PT2=PT2-22
            PT3=PT3-21
            PT4=PT4-21
            SPLINE=SPLINE+1
            WRITE(12,520) SPLINE,SDC,COLOR,SOLID,NUMB
            WRITE(12,525) PT1,PT2,PT3,PT4

            DO 550 J=1,5

              PT1=PT1+1
              PT2=PT2+1
              PT3=PT3+1
              PT4=PT4+1
              SPLINE=SPLINE+1
              WRITE(12,520) SPLINE,SDC,COLOR,SOLID,NUMB
              WRITE(12,525) PT1,PT2,PT3,PT4

 550        CONTINUE

            PT1=PT1+29
            PT2=PT2+29
            PT3=PT3+29
            PT4=PT4+29
            SPLINE=SPLINE+1
            WRITE(12,520) SPLINE,SDC,COLOR,SOLID,NUMB
            WRITE(12,525) PT1,PT2,PT3,PT4

            DO 555 J=1,5

              PT1=PT1+1
              PT2=PT2+1
              PT3=PT3+1
              PT4=PT4+1
              SPLINE=SPLINE+1
              WRITE(12,520) SPLINE,SDC,COLOR,SOLID,NUMB
              WRITE(12,525) PT1,PT2,PT3,PT4

 555        CONTINUE
```

```
          ENDIF

      IF(RIBTYP(I).EQ.1) THEN

          IF(CNLTYP(I).EQ.2) THEN

            PT1=PT1+16
            PT2=PT2+13
            PT3=PT3+10
            PT4=PT4+7
            SPLINE=SPLINE+1
            WRITE(12,520) SPLINE,SDC,COLOR,SOLID,NUMB
            WRITE(12,525) PT1,PT2,PT3,PT4

            PT1=PT1+1
            PT2=PT2+1
            PT3=PT3+1
            PT4=PT4+1
            SPLINE=SPLINE+1
            WRITE(12,520) SPLINE,SDC,COLOR,SOLID,NUMB
            WRITE(12,525) PT1,PT2,PT3,PT4

               IF(I.EQ.NUMBER) GO TO 560

            PT1=PT1-36
            PT2=PT2+5
            PT3=PT3+8
            PT4=PT4+11
            SPLINE=SPLINE+1
            WRITE(12,520) SPLINE,SDC,COLOR,SOLID,NUMB
            WRITE(12,525) PT1,PT2,PT3,PT4

          ELSE IF(CNLTYP(I).EQ.1) THEN

               IF(I.EQ.NUMBER) GO TO 560

            PT1=PT1-19
            PT2=PT2+11
            PT3=PT3+11
            PT4=PT4+11
            SPLINE=SPLINE+1
            WRITE(12,520) SPLINE,SDC,COLOR,SOLID,NUMB
            WRITE(12,525) PT1,PT2,PT3,PT4

          ENDIF

      ELSE IF(RIBTYP(I).EQ.2) THEN

               IF(I.EQ.NUMBER) GO TO 560

          PT1=PT1-21
          PT2=PT2+13
          PT3=PT3+12
          PT4=PT4+11
          SPLINE=SPLINE+1
          WRITE(12,520) SPLINE,SDC,COLOR,SOLID,NUMB
          WRITE(12,525) PT1,PT2,PT3,PT4

      ENDIF

 560  CONTINUE

      NLNTL=NLNTL+SPLINE
```

```
      WRITE(NO,*)'NO. IN RC SEGMENT & TOTAL NUMBER OF SPLINES = ',SPLINE
      WRITE(NO,*) '  '
      WRITE(NO,*) '  '

              WRITE(12,245)

C    ************************************************************
C    *                                                          *
C    *    END GENERATION OF THE SPLINES IN UNIVERSAL FORMAT     *
C    *                                                          *
C    ************************************************************

       WRITE(NO,565)
  565  FORMAT('0','THE POINTS, LINES AND SPLINES HAVE BEEN GENERATED.')
       WRITE(NO,566)
  566  FORMAT('0','THE PROGRAM CAN BE EXITED AT THIS POINT TO VERIFY')
       WRITE(NO,567)
  567  FORMAT('0','THAT THE INTENDED GEOMETRY IS CREATED.')
       WRITE(NO,570)
  570  FORMAT('0','ENTER A ...')
       WRITE(NO,575)
  575  FORMAT('0','     1    TO   CONTINUE   MODEL GENERATION OR ...')
       WRITE(NO,580)
  580  FORMAT('0','     2    TO      STOP     MODEL GENERATION.')

      READ(5,*) ICONTI
      IF (STATUS.NE.0) WRITE(8,*) ICONTI
      WRITE(NO,*) ICONTI
      WRITE(20,*) ICONTI

      IF (ICONTI.EQ.2) THEN

       PRINT*,  ' '
       PRINT*,  ' '
       PRINT*,  '        THE UNIVERSAL HAS BEEN CREATED.'
       PRINT*,  ' '
       PRINT*,  '      BEFORE THE UNIVERSAL FILE CAN BE READ,'
       PRINT*,  '      THE USER MUST ENTER THE SDRC SOFTWARE.'

      PRINT*,  ' '
      PRINT*,  ' NOTE THAT A FILE CALLED "MODEL DATA" CAN NOT'
      PRINT*,  'EXIST ON YOUR DISK.  IF IT DOES, AN ERROR WILL'
      PRINT*,  ' RESULT WHEN YOU ENTER THE SDRC SOFTWARE.  TO'
      PRINT*,  '  CORRECT THE ERROR, CHANGE THE NAME OF THE'
      PRINT*,  ' "MODEL DATA" FILE THAT EXISTS ON YOUR DISK.'
      PRINT*,  '    (BEFORE YOU ENTER THE SDRC SOFTWARE)'
      PRINT*,  ' '
      PRINT*,  'AFTER ENTERING THE SDRC SOFTWARE IN THE PROGRAM MODE,'
      PRINT*,  '    RESPOND "R" (FOR RUN) TO THE FIRST QUESTION,'
      PRINT*,  '       RESPOND "GO" TO THE SECOND QUESTION AND'
      PRINT*,  '          IN RESPONSE TO THE THIRD QUESTION,'
      PRINT*,  '    ENTER THE TERMINAL TYPE THAT YOU ARE USING.'

      CALL GO (ICONTI)

      STOP
      ENDIF

              WRITE(12,245)

C    ************************************************************
C    *                                                          *
C    *    BEGIN GENERATION OF THE EDGES IN UNIVERSAL FORMAT     *
C    *                                                          *
```

```
C    *****************************************************************
C    *****************************************************************

      WRITE(12,585)
  585 FORMAT(4X,'29')

      SPENTC=3
      LNENTC=1
      NUMB=1
      ISPENL=1
      ILNENL=1
      EDGE=1

         DO 625 III=1,NUMBER

              DO 610 II=1,2

      IF((RIBTYP(III).EQ.2).AND.(II.EQ.2)) THEN

           NMSPS=12
      ELSE
           NMSPS=10

      ENDIF

                   DO 600 I=1,NMSPS

      WRITE(12,590) EDGE,COLOR,SOLID,NUMB
  590 FORMAT(4I10)
      WRITE(12,595) SPENTC,ISPENL
  595 FORMAT(6I10)

      EDGE=EDGE+1
      ISPENL=ISPENL+1

  600              CONTINUE

      IF(RIBTYP(III).EQ.2) THEN

         NMLNS=14

      ELSE IF(RIBTYP(III).EQ.1) THEN

         NMLNS=13

      ENDIF

         DO 605 I=1,NMLNS

      WRITE(12,590) EDGE,COLOR,SOLID,NUMB
      WRITE(12,595) LNENTC,ILNENL

      EDGE=EDGE+1
      ILNENL=ILNENL+1

  605    CONTINUE
  610    CONTINUE

      IF(CNLTYP(III).EQ.2) THEN

            DO 615 J=1,2

         WRITE(12,590) EDGE,COLOR,SOLID,NUMB
         WRITE(12,595) SPENTC,ISPENL
         EDGE=EDGE+1
```

```
          ISPENL=ISPENL+1

 615          CONTINUE

        DO 620 J=1,2

          WRITE(12,590) EDGE,COLOR,SOLID,NUMB
          WRITE(12,595) LNENTC,ILNENL
          EDGE=EDGE+1
          ILNENL=ILNENL+1

 620    CONTINUE

        ENDIF

 625        CONTINUE

       TEMPED=EDGE-1

        IF (IEPQUE.EQ.1) THEN

          DO 630 J=EDGE,LINE+SPLINE-6

            WRITE(12,590) EDGE,COLOR,SOLID,NUMB
            WRITE(12,595) LNENTC,ILNENL
            ILNENL=ILNENL+1
            EDGE=EDGE+1

          IF ((EDGE.EQ.LINE+SPLINE-5).OR.(EDGE.EQ.LINE+SPLINE-7)) THEN

              NUMB=3
              WRITE(12,590) EDGE,COLOR,SOLID,NUMB
              WRITE(12,595) LNENTC,ILNENL,LNENTC,ILNENL+1,LNENTC,ILNENL+2
              ILNENL=ILNENL+3
              EDGE=EDGE+1
              NUMB=1

          ENDIF

 630      CONTINUE

         ENDIF

        WRITE(12,245)
        WRITE(12,245)

        NERC=EDGE-1-NLEP-NLIP-NLSP
        NEEP=EDGE-1-SPLINE-NLRC-NLIP-NLSP
        NEIP=EDGE-1-SPLINE-NLRC-NLEP-NLSP
        NESP=EDGE-1-SPLINE-NLRC-NLEP-NLIP+6

        WRITE(NO,*) '   '
        WRITE(NO,*) '   '
        WRITE(NO,*) 'NUMBER OF EDGES IN RC SEGMENTS    = ',NERC
        WRITE(NO,*) 'NUMBER OF EDGES IN EXHAUST PORT   = ',NEEP
        WRITE(NO,*) 'NUMBER OF EDGES IN INTAKE PORT    = ',NEIP
        WRITE(NO,*) 'NUMBER OF EDGES IN SPARK PLUG(S)  = ',NESP
        WRITE(NO,*) 'TOTAL NUMBER OF EDGES = ',EDGE-1
        WRITE(NO,*) '  '
        WRITE(NO,*) '  '

        WRITE(NO,*)'EDGE LABELS OF RC SEGMENTS  ARE            1 TO',NERC
        WRITE(NO,*)'EDGE LABELS OF EXHST PORT ARE ',NERC+1,'TO',NEEP+NERC
        WRITE(NO,*)'EDGE LABELS OF INTKE PORT ARE ',NERC+NEEP+1,'TO',NEIP+
```

```
     #NERC+NEEP
      WRITE(NO,*)'EDGE LABELS OF SPRK PLUGS ARE ',NERC+NEEP+NEIP+1,'TO',
     #NESP+NERC+NEEP+NEIP
      WRITE(NO,*) '  '
      WRITE(NO,*) '  '

C  ***************************************************************
C  *                                                             *
C  *       END GENERATION OF THE EDGES IN UNIVERSAL FORMAT       *
C  *                                                             *
C  ***************************************************************

C  ***************************************************************
C  *                                                             *
C  *    BEGIN GENERATION OF THE SURFACES OF THE INNER AND        *
C  *    OUTER SHELLS, AND RIBS IN UNIVERSAL FORMAT               *
C  *                                                             *
C  ***************************************************************

      ISRCT=0
      WRITE(12,635)
 635  FORMAT(4X,'30')

      EDGE=TEMPED
      SURFAC=1
      DASH=2
      NUMB=4
      GRADE=2

      ED1=1
      IF(RIBTYP(NUMBER).EQ.2) THEN

      EDGE=EDGE-2

      ENDIF

      ED2=EDGE-8
      ED3=6
      ED4=15
      WRITE(12,640) SURFAC,COLOR,DASH,GRADE,NUMB
 640  FORMAT(5I10)
      WRITE(12,645) ED1,ED2,ED3,ED4
 645  FORMAT(4I10)

           DO 690 I=1,NUMBER

      IF(RIBTYP(I).EQ.1) THEN

                DO 675 JJ=1,2

                     DO 650 J=1,4

                        ED1=ED1+1
                        ED2=ED2+1
       IF (((J.EQ.3).AND.(JJ.EQ.1)).AND.(RIBTYP(I-1).EQ.2))
     #       ED2=ED2+1
                        ED3=ED3+1
                        ED4=ED4+1
                        SURFAC=SURFAC+1
                        WRITE(12,640) SURFAC,COLOR,DASH,GRADE,NUMB
                        WRITE(12,645) ED1,ED2,ED3,ED4

 650                 CONTINUE
```

```
               ED1=ED1-4
               ED2=ED2-8
                  IF((RIBTYP(I-1).EQ.2).AND.(JJ.EQ.1)) ED2=ED2-1
               ED3=ED3-8
               ED4=ED4-8
               SURFAC=SURFAC+1
               WRITE(12,640) SURFAC,COLOR,DASH,GRADE,NUMB
               WRITE(12,645) ED1,ED2,ED3,ED4

      IF(JJ.EQ.1) THEN

          ED1=ED1+1
          ED2=ED2+1
          ED3=ED3+1
          ED4=ED4+1
          SURFAC=SURFAC+1
          WRITE(12,640) SURFAC,COLOR,DASH,GRADE,NUMB
          WRITE(12,645) ED1,ED2,ED3,ED4

          ED1=ED1+2
          ED2=ED2+2
          ED3=ED3+2
          ED4=ED4+2
          SURFAC=SURFAC+1
          WRITE(12,640) SURFAC,COLOR,DASH,GRADE,NUMB
          WRITE(12,645) ED1,ED2,ED3,ED4

          ED1=ED1+2
          ED2=ED2+6
             IF(RIBTYP(I-1).EQ.2) ED2=ED2+1
          ED3=ED3+2
          ED4=ED4+6
          SURFAC=SURFAC+1
          WRITE(12,640) SURFAC,COLOR,DASH,GRADE,NUMB
          WRITE(12,645) ED1,ED2,ED3,ED4

          ED1=ED1+1
          ED2=ED2+1
          ED3=ED3+1
          ED4=ED4+1
          SURFAC=SURFAC+1
          WRITE(12,640) SURFAC,COLOR,DASH,GRADE,NUMB
          WRITE(12,645) ED1,ED2,ED3,ED4

          ED1=ED1+2
          ED2=ED2+2
          ED3=ED3+2
          ED4=ED4+2
          SURFAC=SURFAC+1
          WRITE(12,640) SURFAC,COLOR,DASH,GRADE,NUMB
          WRITE(12,645) ED1,ED2,ED3,ED4

      ELSE IF(JJ.EQ.2) THEN

                  DO 655 JJJ=1,3

                     ED1=ED1+1
                     ED2=ED2+1
                     ED3=ED3+1
                     ED4=ED4+1
                     SURFAC=SURFAC+1
                     WRITE(12,640) SURFAC,COLOR,DASH,GRADE,NUMB
                     WRITE(12,645) ED1,ED2,ED3,ED4
```

```
 655                 CONTINUE

                          ED1=ED1+2
                          ED2=ED2+6
                          ED3=ED3+2
                          ED4=ED4+6
                          SURFAC=SURFAC+1
                          WRITE(12,640) SURFAC,COLOR,DASH,GRADE,NUMB
                          WRITE(12,645) ED1,ED2,ED3,ED4

                     DO 660 JJJ=1,3

                          ED1=ED1+1
                          ED2=ED2+1
                          ED3=ED3+1
                          ED4=ED4+1
                          SURFAC=SURFAC+1
                          WRITE(12,640) SURFAC,COLOR,DASH,GRADE,NUMB
                          WRITE(12,645) ED1,ED2,ED3,ED4

 660                 CONTINUE

       ENDIF
           ED1=ED1+6
             IF(I.GT.1) GO TO 665

                IF((I.EQ.1).AND.(JJ.EQ.1)) THEN
                     ED2=ED2+20-EDGE
                ELSE IF(JJ.EQ.2) THEN
                     ED2=ED2+20
                ENDIF

 665               IF((CNLTYP(I-1).EQ.2).AND.(JJ.EQ.2)) THEN
                     ED2=ED2+20
                   ELSE IF((CNLTYP(I-1).EQ.2).AND.(JJ.EQ.1)) THEN
                     ED2=ED2+24
                   ELSE IF(CNLTYP(I-1).EQ.1) THEN
                     ED2=ED2+20
                   ENDIF

          ED3=ED3+6
          ED4=ED4-12
          SURFAC=SURFAC+1
          WRITE(12,640) SURFAC,COLOR,DASH,GRADE,NUMB
          WRITE(12,645) ED1,ED2,ED3,ED4

      DO 670 J=1,3

         ED1=ED1+1
         ED2=ED2+1
         ED3=ED3+1
         ED4=ED4+1
         SURFAC=SURFAC+1
         WRITE(12,640) SURFAC,COLOR,DASH,GRADE,NUMB
         WRITE(12,645) ED1,ED2,ED3,ED4

 670  CONTINUE

      IF(JJ.EQ.1) THEN
           ED1=ED1+6
           ED2=ED2-8
           ED3=ED3+10
           ED4=ED4+24
           SURFAC=SURFAC+1
```

```
          WRITE(12,640) SURFAC,COLOR,DASH,GRADE,NUMB
          WRITE(12,645) ED1,ED2,ED3,ED4

      ENDIF

  675           CONTINUE

      ELSE IF(RIBTYP(I).EQ.2) THEN

         DO 680 J=1,2

              ED1=ED1+1
              ED2=ED2+1
              ED3=ED3+1
              ED4=ED4+1
              SURFAC=SURFAC+1
              WRITE(12,640) SURFAC,COLOR,DASH,GRADE,NUMB
              WRITE(12,645) ED1,ED2,ED3,ED4

  680    CONTINUE

                ED1=ED1+1
                ED2=ED2+2
                   IF(RIBTYP(I-1).EQ.1) ED2=ED2-1
                ED3=ED3+1
                ED4=ED4+2
                SURFAC=SURFAC+1
                WRITE(12,640) SURFAC,COLOR,DASH,GRADE,NUMB
                WRITE(12,645) ED1,ED2,ED3,ED4

                ED1=ED1+1
                ED2=ED2+1
                ED3=ED3+1
                ED4=ED4+1
                SURFAC=SURFAC+1
                WRITE(12,640) SURFAC,COLOR,DASH,GRADE,NUMB
                WRITE(12,645) ED1,ED2,ED3,ED4

                ED1=ED1-4
                ED2=ED2-9
                   IF(RIBTYP(I-1).EQ.1) ED2=ED2+1
                ED3=ED3-8
                ED4=ED4-9
                SURFAC=SURFAC+1
                WRITE(12,640) SURFAC,COLOR,DASH,GRADE,NUMB
                WRITE(12,645) ED1,ED2,ED3,ED4

                ED1=ED1+1
                ED2=ED2+1
                ED3=ED3+1
                ED4=ED4+1
                SURFAC=SURFAC+1
                WRITE(12,640) SURFAC,COLOR,DASH,GRADE,NUMB
                WRITE(12,645) ED1,ED2,ED3,ED4

                ED1=ED1+2
                ED2=ED2+2
                ED3=ED3+2
                ED4=ED4+2
                SURFAC=SURFAC+1
                WRITE(12,640) SURFAC,COLOR,DASH,GRADE,NUMB
                WRITE(12,645) ED1,ED2,ED3,ED4

                ED1=ED1+2
```

```
      ED2=ED2+7
         IF(RIBTYP(I-1).EQ.1) ED2=ED2-1
      ED3=ED3+2
      ED4=ED4+7
      SURFAC=SURFAC+1
      WRITE(12,640) SURFAC,COLOR,DASH,GRADE,NUMB
      WRITE(12,645) ED1,ED2,ED3,ED4

      ED1=ED1+1
      ED2=ED2+1
      ED3=ED3+1
      ED4=ED4+1
      SURFAC=SURFAC+1
      WRITE(12,640) SURFAC,COLOR,DASH,GRADE,NUMB
      WRITE(12,645) ED1,ED2,ED3,ED4

      ED1=ED1+2
      ED2=ED2+2
      ED3=ED3+2
      ED4=ED4+2
      SURFAC=SURFAC+1
      WRITE(12,640) SURFAC,COLOR,DASH,GRADE,NUMB
      WRITE(12,645) ED1,ED2,ED3,ED4

      ED1=ED1+6
      ED2=ED2+23
         IF(RIBTYP(I-1).EQ.1) ED2=ED2-2
         IF(RIBTYP(I-1).EQ.2) ED2=ED2-2
         IF(CNLTYP(I-1).EQ.2) ED2=ED2+4
      ED3=ED3+6
      ED4=ED4-13
      SURFAC=SURFAC+1
      WRITE(12,640) SURFAC,COLOR,DASH,GRADE,NUMB
      WRITE(12,645) ED1,ED2,ED3,ED4

      ED1=ED1+1
      ED2=ED2+1
      ED3=ED3+1
      ED4=ED4+1
      SURFAC=SURFAC+1
      WRITE(12,640) SURFAC,COLOR,DASH,GRADE,NUMB
      WRITE(12,645) ED1,ED2,ED3,ED4

      ED1=ED1+2
      ED2=ED2+1
      ED3=ED3+2
      ED4=ED4+1
      SURFAC=SURFAC+1
      WRITE(12,640) SURFAC,COLOR,DASH,GRADE,NUMB
      WRITE(12,645) ED1,ED2,ED3,ED4

      ED1=ED1+1
      ED2=ED2+1
      ED3=ED3+1
      ED4=ED4+1
      SURFAC=SURFAC+1
      WRITE(12,640) SURFAC,COLOR,DASH,GRADE,NUMB
      WRITE(12,645) ED1,ED2,ED3,ED4

      ED1=ED1+6
      ED2=ED2-9
      ED3=ED3+11
      ED4=ED4+27
      SURFAC=SURFAC+1
```

```
          WRITE(12,640) SURFAC,COLOR,DASH,GRADE,NUMB
          WRITE(12,645) ED1,ED2,ED3,ED4

        DO 685 J=1,5

             ED1=ED1+1
             ED2=ED2+1
             ED3=ED3+1
             ED4=ED4+1
             SURFAC=SURFAC+1
             WRITE(12,640) SURFAC,COLOR,DASH,GRADE,NUMB
             WRITE(12,645) ED1,ED2,ED3,ED4

685     CONTINUE

          ED1=ED1-5
          ED2=ED2-9
          ED3=ED3-10
          ED4=ED4-9
          SURFAC=SURFAC+1
          WRITE(12,640) SURFAC,COLOR,DASH,GRADE,NUMB
          WRITE(12,645) ED1,ED2,ED3,ED4

          ED1=ED1+1
          ED2=ED2+1
          ED3=ED3+1
          ED4=ED4+1
          SURFAC=SURFAC+1
          WRITE(12,640) SURFAC,COLOR,DASH,GRADE,NUMB
          WRITE(12,645) ED1,ED2,ED3,ED4

          ED1=ED1+2
          ED2=ED2+1
          ED3=ED3+2
          ED4=ED4+1
          SURFAC=SURFAC+1
          WRITE(12,640) SURFAC,COLOR,DASH,GRADE,NUMB
          WRITE(12,645) ED1,ED2,ED3,ED4

          ED1=ED1+1
          ED2=ED2+1
          ED3=ED3+1
          ED4=ED4+1
          SURFAC=SURFAC+1
          WRITE(12,640) SURFAC,COLOR,DASH,GRADE,NUMB
          WRITE(12,645) ED1,ED2,ED3,ED4

          ED1=ED1+2
          ED2=ED2+7
          ED3=ED3+2
          ED4=ED4+7
          SURFAC=SURFAC+1
          WRITE(12,640) SURFAC,COLOR,DASH,GRADE,NUMB
          WRITE(12,645) ED1,ED2,ED3,ED4

          ED1=ED1+1
          ED2=ED2+1
          ED3=ED3+1
          ED4=ED4+1
          SURFAC=SURFAC+1
          WRITE(12,640) SURFAC,COLOR,DASH,GRADE,NUMB
          WRITE(12,645) ED1,ED2,ED3,ED4

          ED1=ED1+2
```

```
              ED2=ED2+1
              ED3=ED3+2
              ED4=ED4+1
              SURFAC=SURFAC+1
              WRITE(12,640) SURFAC,COLOR,DASH,GRADE,NUMB
              WRITE(12,645) ED1,ED2,ED3,ED4

              ED1=ED1+1
              ED2=ED2+1
              ED3=ED3+1
              ED4=ED4+1
              SURFAC=SURFAC+1
              WRITE(12,640) SURFAC,COLOR,DASH,GRADE,NUMB
              WRITE(12,645) ED1,ED2,ED3,ED4

              ED1=ED1+6
              ED2=ED2+23
              ED3=ED3+6
              ED4=ED4-13
              SURFAC=SURFAC+1
              WRITE(12,640) SURFAC,COLOR,DASH,GRADE,NUMB
              WRITE(12,645) ED1,ED2,ED3,ED4

              ED1=ED1+1
              ED2=ED2+1
              ED3=ED3+1
              ED4=ED4+1
              SURFAC=SURFAC+1
              WRITE(12,640) SURFAC,COLOR,DASH,GRADE,NUMB
              WRITE(12,645) ED1,ED2,ED3,ED4


              ED1=ED1+2
              ED2=ED2+1
              ED3=ED3+2
              ED4=ED4+1
              SURFAC=SURFAC+1
              WRITE(12,640) SURFAC,COLOR,DASH,GRADE,NUMB
              WRITE(12,645) ED1,ED2,ED3,ED4

              ED1=ED1+1
              ED2=ED2+1
              ED3=ED3+1
              ED4=ED4+1
              SURFAC=SURFAC+1
              WRITE(12,640) SURFAC,COLOR,DASH,GRADE,NUMB
              WRITE(12,645) ED1,ED2,ED3,ED4

      ENDIF

         IF(RIBTYP(I).EQ.1) THEN

             IF(CNLTYP(I).EQ.1) THEN

               IF(I.EQ.NUMBER) GO TO 690
                 ED1=ED1+6
                 ED2=ED2-8
                 ED3=ED3+10
                 ED4=ED4+24
                 SURFAC=SURFAC+1
                 WRITE(12,640) SURFAC,COLOR,DASH,GRADE,NUMB
                 WRITE(12,645) ED1,ED2,ED3,ED4

             ELSE IF(CNLTYP(I).EQ.2) THEN
```

```
                  ED1=ED1+6
                  ED2=ED2+3
                  ED3=ED3+6
                  ED4=ED4+13
                  SURFAC=SURFAC+1

                    ISRCT=ISRCT+1
                    WRITE(20,*) SURFAC,I

                  WRITE(12,640) SURFAC,COLOR,DASH,GRADE,NUMB
                  WRITE(12,645) ED1,ED2,ED3,ED4

                IF(I.EQ.NUMBER) GO TO 690

                  ED1=ED1+4
                  ED2=ED2-11
                  ED3=ED3+8
                  ED4=ED4+15
                  SURFAC=SURFAC+1
                  WRITE(12,640) SURFAC,COLOR,DASH,GRADE,NUMB
                  WRITE(12,645) ED1,ED2,ED3,ED4

             ENDIF

        ELSE IF(RIBTYP(I).EQ.2) THEN

               IF(I.EQ.NUMBER) GO TO 690
             ED1=ED1+6
             ED2=ED2-9
             ED3=ED3+10
             ED4=ED4+25
             SURFAC=SURFAC+1
             WRITE(12,640) SURFAC,COLOR,DASH,GRADE,NUMB
             WRITE(12,645) ED1,ED2,ED3,ED4

        ENDIF

 690  CONTINUE

      TEMPSU=SURFAC
      TEMSU=SURFAC
      EDGE=TEMPED

      NSRC=SURFAC
      WRITE(NO,*) 'NUMBER OF SURFACES IN RC SEGMENTS   = ',NSRC

C    ************************************************************
C    *                                                          *
C    *        BEGIN GENERATION OF THE SURFACES OF THE           *
C    *                    EXHAUST PORT                          *
C    *                                                          *
C    ************************************************************

      IF (IEPQUE.EQ.1) THEN

          DO 715 JJ=1,IEND

             SURT=SURFAC
             ED1=EDGE+1
             ED2=EDGE+26
             ED3=EDGE+15
             ED4=EDGE+27
             SURFAC=SURFAC+1
```

```
           WRITE(12,640) SURFAC,COLOR,DASH,GRADE,NUMB
           WRITE(12,645) ED1,ED2,ED3,ED4

             DO 695 J=1,19

                  ED1=ED1+1
                  ED2=ED2+1
                  ED3=ED3+1
                  ED4=ED4+1
                  SURFAC=SURFAC+1
                  WRITE(12,640) SURFAC,COLOR,DASH,GRADE,NUMB
                  WRITE(12,645) ED1,ED2,ED3,ED4

                     IF (J.EQ.9) ED4=ED4-24

                       IF (J.EQ.10) THEN

                            ED2=ED2-24
                            ED3=ED3-12
                            ED4=ED4+12

                       ENDIF

                         IF (J.EQ.11) THEN

                              ED1=ED1+3
                              ED2=ED2+23
                              ED3=ED3+32
                              ED4=ED4+11

                          ENDIF

         IF (((JJ.EQ.3).OR.(JJ.EQ.4)).AND.(J.EQ.11)) GO TO 700

                             IF (J.EQ.15) THEN

                                 ED1=ED1+2
                                 ED2=ED2+1
                                 ED4=ED4+1

                             ENDIF

 695             CONTINUE

 700  IF (JJ.EQ.IEND) GO TO 710

         ED1=EDGE+1
         ED2=EDGE+55
         ED3=EDGE+1+NLNEP(JJ)
         ED4=EDGE+56

              IF((JJ.EQ.3).OR.(JJ.EQ.4)) THEN

                   ED2=ED2-18
                   ED4=ED4-18

              ENDIF

         SURFAC=SURFAC+1
         WRITE(12,640) SURFAC,COLOR,DASH,GRADE,NUMB
         WRITE(12,645) ED1,ED2,ED3,ED4

            DO 705 J=1,53
```

```
              ED1=ED1+1
              ED2=ED2+1
              ED3=ED3+1
              ED4=ED4+1
              SURFAC=SURFAC+1
              WRITE(12,640) SURFAC,COLOR,DASH,GRADE,NUMB
              WRITE(12,645) ED1,ED2,ED3,ED4

                   IF (J.EQ.10) ED4=ED4-12
                   IF (J.EQ.11) ED4=ED4+10
                   IF (J.EQ.12) ED2=ED2-1
                   IF (J.EQ.12) ED4=ED4+1

                   IF (J.EQ.23) ED4=ED4-12
                   IF (J.EQ.24) ED2=ED2-24

      IF (((JJ.EQ.2).OR.(JJ.EQ.3).OR.(JJ.EQ.4)).AND.
     #        (J.EQ.35)) GO TO 710

                   IF (J.EQ.35) ED2=ED2+3
                   IF (J.EQ.40) ED2=ED2+1

                   IF (J.EQ.44) ED2=ED2-12
                   IF (J.EQ.45) ED2=ED2+11
                   IF (J.EQ.45) ED4=ED4-9

                   IF (J.EQ.49) ED2=ED2+1
                   IF (J.EQ.49) ED4=ED4+1

705             CONTINUE

710           EDGE=EDGE+NLNEP(JJ)
              SURB=SURFAC
              NSREP(JJ)=SURB-SURT

715         CONTINUE

              ITOTAL=0

            DO 720 J=1,IEND

720         ITOTAL=ITOTAL+NLNEP(J)

            NEDEP=ITOTAL+TEMPED
            TEMSU1=SURFAC

            DO 755 JJ=1,2

                 ED1=NEDEP+1
                 ED2=NEDEP+13
                 ED3=NEDEP+5
                 ED4=NEDEP+14
                 SURFAC=SURFAC+1
                 WRITE(12,640) SURFAC,COLOR,DASH,GRADE,NUMB
                 WRITE(12,645) ED1,ED2,ED3,ED4

                      DO 725 J=1,7

                           ED1=ED1+1
                           ED2=ED2+1
                           ED3=ED3+1
                           ED4=ED4+1
                           SURFAC=SURFAC+1
                           WRITE(12,640) SURFAC,COLOR,DASH,GRADE,NUMB
```

```
                  WRITE(12,645) ED1,ED2,ED3,ED4

                       IF (J.EQ.3) THEN

                            ED2=ED2+1
                            ED4=ED4+1

                       ENDIF

 725              CONTINUE

        ED1=TEMPED+46
        ED2=NEDEP+27
        ED3=NEDEP+1
        ED4=NEDEP+23
        SURFAC=SURFAC+1
        WRITE(12,640) SURFAC,COLOR,DASH,GRADE,NUMB
        WRITE(12,645) ED1,ED2,ED3,ED4

        ED1=ED1-32
        ED2=ED2-4
        ED3=ED3+1
        ED4=ED4+1
        SURFAC=SURFAC+1
        WRITE(12,640) SURFAC,COLOR,DASH,GRADE,NUMB
        WRITE(12,645) ED1,ED2,ED3,ED4

             DO 730 J=1,2

                  ED1=ED1+1
                  ED2=ED2+1
                  ED3=ED3+1
                  ED4=ED4+1
                  SURFAC=SURFAC+1
                  WRITE(12,640) SURFAC,COLOR,DASH,GRADE,NUMB
                  WRITE(12,645) ED1,ED2,ED3,ED4

                       IF (J.EQ.1) ED1=ED1+21

 730         CONTINUE

        IF (JJ.EQ.2) GO TO 750

             ED1=NEDEP+1
             ED2=NEDEP+28
             ED3=NEDEP+1+NLNEP1(JJ)
             ED4=NEDEP+29
             SURFAC=SURFAC+1
             WRITE(12,640) SURFAC,COLOR,DASH,GRADE,NUMB
             WRITE(12,645) ED1,ED2,ED3,ED4

                  DO 735 J=1,11

                       ED1=ED1+1
                       ED2=ED2+1
                       ED3=ED3+1
                       ED4=ED4+1
                       SURFAC=SURFAC+1
                       WRITE(12,640) SURFAC,COLOR,DASH,GRADE,NUMB
                       WRITE(12,645) ED1,ED2,ED3,ED4

                            IF ((J.EQ.3).OR.(J.EQ.7)) THEN

                                 ED2=ED2+1
```

```
                              ED4=ED4+1

                         ENDIF

 735              CONTINUE

             ED1=NEDEP+13
             ED2=NEDEP+28
             ED3=NEDEP+13+NLNEP1(JJ)
             ED4=NEDEP+33
             SURFAC=SURFAC+1
             WRITE(12,640) SURFAC,COLOR,DASH,GRADE,NUMB
             WRITE(12,645) ED1,ED2,ED3,ED4

                  DO 740 J=1,9

                       ED1=ED1+1
                       ED2=ED2+1
                       ED3=ED3+1
                       ED4=ED4+1
                       SURFAC=SURFAC+1
                       WRITE(12,640) SURFAC,COLOR,DASH,GRADE,NUMB
                       WRITE(12,645) ED1,ED2,ED3,ED4

 740              CONTINUE

             ED1=NEDEP+27
             ED2=TEMPED+88
             ED3=NEDEP+27+NLNEP1(JJ)
             ED4=NEDEP+28
             SURFAC=SURFAC+1
             WRITE(12,640) SURFAC,COLOR,DASH,GRADE,NUMB
             WRITE(12,645) ED1,ED2,ED3,ED4

                       ED1=ED1-4
                       ED2=ED2-21
                       ED3=ED3-4
                       ED4=ED4+1
                       SURFAC=SURFAC+1
                       WRITE(12,640) SURFAC,COLOR,DASH,GRADE,NUMB
                       WRITE(12,645) ED1,ED2,ED3,ED4

                         DO 745 J=1,3

                           ED1=ED1+1
                           ED2=ED2+1
                           ED3=ED3+1
                           ED4=ED4+1
                           SURFAC=SURFAC+1
                           WRITE(12,640) SURFAC,COLOR,DASH,GRADE,NUMB
                           WRITE(12,645) ED1,ED2,ED3,ED4

                                IF (J.EQ.2) ED2=ED2+9

 745                     CONTINUE

 750         NEDEP=NEDEP+NLNEP1(JJ)
             TEMPED=TEMPED+NLNEP(JJ)

 755        CONTINUE

      ENDIF

      NEDEP=NLEP+NLRC+SPLINE
```

```
      TEMPID=NEDEP
      TEMPI1=NEDEP+NLNIP(1)+NLNIP(2)+NLNIP(3)+NLNIP(4)

              NSEP=SURFAC
      NSEP1=SURFAC-NSRC
      WRITE(NO,*) 'NUMBER OF SURFACES IN EXHAUST PORT  = ',NSEP1
              NSEPX=NSEP

C    ************************************************************
C    *                                                          *
C    *         BEGIN GENERATION OF THE SURFACES OF THE          *
C    *                     INTAKE PORT                          *
C    *                                                          *
C    ************************************************************

      IF (IIPQUE.EQ.1) THEN

          DO 780 JJ=1,IEND

             SURT=SURFAC
             ED1=NEDEP+1
             ED2=NEDEP+28
             ED3=NEDEP+15
             ED4=NEDEP+29
             SURFAC=SURFAC+1
             WRITE(12,640) SURFAC,COLOR,DASH,GRADE,NUMB
             WRITE(12,645) ED1,ED2,ED3,ED4

               DO 760 J=1,21

                 IF (J.EQ.1) THEN

                  NUMB=3
                  ED2=ED2+1
                  ED3=ED3+1
                  ED4=ED4+1
                  SURFAC=SURFAC+1
                  WRITE(12,640) SURFAC,COLOR,DASH,GRADE,NUMB
                  WRITE(12,645) ED2,ED3,ED4
                  NUMB=4

                 ENDIF

                    ED1=ED1+1
                    ED2=ED2+1
                    ED3=ED3+1
                    ED4=ED4+1
                    SURFAC=SURFAC+1
                    WRITE(12,640) SURFAC,COLOR,DASH,GRADE,NUMB
                    WRITE(12,645) ED1,ED2,ED3,ED4

                         IF (J.EQ.10) THEN

                             NUMB=3
                             ED2=ED2+1
                             ED3=ED3+1
                             ED4=ED4-27
                             SURFAC=SURFAC+1
                             WRITE(12,640) SURFAC,COLOR,DASH,GRADE,NUMB
                             WRITE(12,645) ED2,ED3,ED4
                              ED2=ED2-28
                              ED3=ED3-14
                              ED4=ED4+14
                             NUMB=4
```

```
                                   ENDIF

                                      IF (J.EQ.11) THEN

                                             ED1=ED1+3
                                             ED2=ED2+27
                                             ED3=ED3+38
                                             ED4=ED4+13

                                       ENDIF

          IF (((JJ.EQ.3).OR.(JJ.EQ.4)).AND.(J.EQ.11)) GO TO 765

                                           IF (J.EQ.16) THEN

                                                ED1=ED1+2
                                                ED2=ED2+1
                                                ED4=ED4+1

                                           ENDIF

760              CONTINUE

765   IF (JJ.EQ.IEND) GO TO 775

          ED1=NEDEP+1
          ED2=NEDEP+63
          ED3=NEDEP+1+NLNIP(JJ)
          ED4=NEDEP+64

               IF((JJ.EQ.3).OR.(JJ.EQ.4)) THEN

                    ED2=ED2-22
                    ED4=ED4-22

               ENDIF

          SURFAC=SURFAC+1
          WRITE(12,640) SURFAC,COLOR,DASH,GRADE,NUMB
          WRITE(12,645) ED1,ED2,ED3,ED4

             DO 770 J=1,61

               ED1=ED1+1
               ED2=ED2+1
               ED3=ED3+1
               ED4=ED4+1
               SURFAC=SURFAC+1
               WRITE(12,640) SURFAC,COLOR,DASH,GRADE,NUMB
               WRITE(12,645) ED1,ED2,ED3,ED4

                    IF (J.EQ.10) ED4=ED4-12
                    IF (J.EQ.11) ED4=ED4+11
                    IF (J.EQ.11) ED2=ED2-1

                    IF (J.EQ.25) ED4=ED4-14
                    IF (J.EQ.26) ED2=ED2-26
                    IF (J.EQ.28) ED2=ED2-1

      IF (((JJ.EQ.2).OR.(JJ.EQ.3).OR.(JJ.EQ.4)).AND.
     #       (J.EQ.39)) GO TO 775

                    IF (J.EQ.39) ED2=ED2+2
```

```
                IF (J.EQ.45) ED2=ED2+1
                IF (J.EQ.50) ED2=ED2-14

                IF (J.EQ.51) ED2=ED2+13
                IF (J.EQ.51) ED4=ED4-11

                IF (J.EQ.56) ED2=ED2+1
                IF (J.EQ.56) ED4=ED4+1

 770        CONTINUE

 775      NEDEP=NEDEP+NLNIP(JJ)
          SURB=SURFAC
          NSRIP(JJ)=SURB-SURT
          IF (JJ.EQ.4) NSIP2(JJ)=SURFAC
 780      CONTINUE
              NSIP1=SURFAC

         DO 815 JJ=1,2

         ED1=TEMPID+47
         ED2=NEDEP+27
         ED3=NEDEP+1
         ED4=NEDEP+23
         SURFAC=SURFAC+1
         WRITE(12,640) SURFAC,COLOR,DASH,GRADE,NUMB
         WRITE(12,645) ED1,ED2,ED3,ED4

         ED1=ED1-25
         ED2=ED2-4
         ED3=ED3+1
         ED4=ED4+1
         SURFAC=SURFAC+1
         WRITE(12,640) SURFAC,COLOR,DASH,GRADE,NUMB
         WRITE(12,645) ED1,ED2,ED3,ED4

              DO 785 J=1,2

                   ED1=ED1-1
                   ED2=ED2+1
                   ED3=ED3+1
                   ED4=ED4+1
                   SURFAC=SURFAC+1
                   WRITE(12,640) SURFAC,COLOR,DASH,GRADE,NUMB
                   WRITE(12,645) ED1,ED2,ED3,ED4

                        IF (J.EQ.1) ED1=ED1+26

 785          CONTINUE

              ED1=NEDEP+1
              ED2=NEDEP+13
              ED3=NEDEP+5
              ED4=NEDEP+14
              SURFAC=SURFAC+1
              WRITE(12,640) SURFAC,COLOR,DASH,GRADE,NUMB
              WRITE(12,645) ED1,ED2,ED3,ED4

                   DO 790 J=1,7

                        ED1=ED1+1
                        ED2=ED2+1
                        ED3=ED3+1
                        ED4=ED4+1
```

```
                    SURFAC=SURFAC+1
                    WRITE(12,640) SURFAC,COLOR,DASH,GRADE,NUMB
                    WRITE(12,645) ED1,ED2,ED3,ED4

                         IF (J.EQ.3) THEN

                              ED2=ED2+1
                              ED4=ED4+1

                         ENDIF

 790                CONTINUE

       IF (JJ.EQ.2) GO TO 810

            ED1=NEDEP+1
            ED2=NEDEP+28
            ED3=NEDEP+1+NLNIP1(JJ)
            ED4=NEDEP+29
            SURFAC=SURFAC+1
            WRITE(12,640) SURFAC,COLOR,DASH,GRADE,NUMB
            WRITE(12,645) ED1,ED2,ED3,ED4

                 DO 795 J=1,11

                      ED1=ED1+1
                      ED2=ED2+1
                      ED3=ED3+1
                      ED4=ED4+1
                      SURFAC=SURFAC+1
                      WRITE(12,640) SURFAC,COLOR,DASH,GRADE,NUMB
                      WRITE(12,645) ED1,ED2,ED3,ED4

                           IF ((J.EQ.3).OR.(J.EQ.7)) THEN

                                ED2=ED2+1
                                ED4=ED4+1

                           ENDIF

 795             CONTINUE

            ED1=NEDEP+27
            ED2=TEMPID+95
            ED3=NEDEP+27+NLNIP1(JJ)
            ED4=NEDEP+28
            SURFAC=SURFAC+1
            WRITE(12,640) SURFAC,COLOR,DASH,GRADE,NUMB
            WRITE(12,645) ED1,ED2,ED3,ED4

                      ED1=ED1-4
                      ED2=ED2-11
                      ED3=ED3-4
                      ED4=ED4+1
                      SURFAC=SURFAC+1
                      WRITE(12,640) SURFAC,COLOR,DASH,GRADE,NUMB
                      WRITE(12,645) ED1,ED2,ED3,ED4

                        DO 800 J=1,3

                          ED1=ED1+1
                          ED2=ED2-1
                          ED3=ED3+1
                          ED4=ED4+1
```

```
                          SURFAC=SURFAC+1
                          WRITE(12,640) SURFAC,COLOR,DASH,GRADE,NUMB
                          WRITE(12,645) ED1,ED2,ED3,ED4

                              IF (J.EQ.2) ED2=ED2+13

 800                    CONTINUE

                ED1=NEDEP+13
                ED2=NEDEP+28
                ED3=NEDEP+13+NLNIP1(JJ)
                ED4=NEDEP+33
                SURFAC=SURFAC+1
                WRITE(12,640) SURFAC,COLOR,DASH,GRADE,NUMB
                WRITE(12,645) ED1,ED2,ED3,ED4

                    DO 805 J=1,9

                        ED1=ED1+1
                        ED2=ED2+1
                        ED3=ED3+1
                        ED4=ED4+1
                        SURFAC=SURFAC+1
                        WRITE(12,640) SURFAC,COLOR,DASH,GRADE,NUMB
                        WRITE(12,645) ED1,ED2,ED3,ED4

 805                CONTINUE

 810            NEDEP=NEDEP+NLNIP1(JJ)
                TEMPID=TEMPID+NLNIP(JJ)

 815        CONTINUE

            DO  850 JJ=5,IEND

             SURT=SURFAC
             ED1=TEMPI1+47
             ED2=NEDEP+27
             ED3=NEDEP+1
             ED4=NEDEP+23
             SURFAC=SURFAC+1
             WRITE(12,640) SURFAC,COLOR,DASH,GRADE,NUMB
             WRITE(12,645) ED1,ED2,ED3,ED4

             ED1=ED1-25
             ED2=ED2-4
             ED3=ED3+1
             ED4=ED4+1
             SURFAC=SURFAC+1
             WRITE(12,640) SURFAC,COLOR,DASH,GRADE,NUMB
             WRITE(12,645) ED1,ED2,ED3,ED4

                 DO 820 J=1,2

                     ED1=ED1-1
                     ED2=ED2+1
                     ED3=ED3+1
                     ED4=ED4+1
                     SURFAC=SURFAC+1
                     WRITE(12,640) SURFAC,COLOR,DASH,GRADE,NUMB
                     WRITE(12,645) ED1,ED2,ED3,ED4

                         IF (J.EQ.1) ED1=ED1+26
```

```
820           CONTINUE

              ED1=NEDEP+1
              ED2=NEDEP+13
              ED3=NEDEP+5
              ED4=NEDEP+14
              SURFAC=SURFAC+1
              WRITE(12,640) SURFAC,COLOR,DASH,GRADE,NUMB
              WRITE(12,645) ED1,ED2,ED3,ED4

                   DO 825 J=1,7

                        ED1=ED1+1
                        ED2=ED2+1
                        ED3=ED3+1
                        ED4=ED4+1
                        SURFAC=SURFAC+1
                        WRITE(12,640) SURFAC,COLOR,DASH,GRADE,NUMB
                        WRITE(12,645) ED1,ED2,ED3,ED4

                             IF (J.EQ.3) THEN

                                  ED2=ED2+1
                                  ED4=ED4+1

                             ENDIF

825                CONTINUE

         IF (JJ.EQ.IEND) GO TO 845

              ED1=NEDEP+1
              ED2=NEDEP+28
              ED3=NEDEP+1+NLNIP2(JJ)
              ED4=NEDEP+29
              SURFAC=SURFAC+1
              WRITE(12,640) SURFAC,COLOR,DASH,GRADE,NUMB
              WRITE(12,645) ED1,ED2,ED3,ED4

                   DO 830 J=1,11

                        ED1=ED1+1
                        ED2=ED2+1
                        ED3=ED3+1
                        ED4=ED4+1
                        SURFAC=SURFAC+1
                        WRITE(12,640) SURFAC,COLOR,DASH,GRADE,NUMB
                        WRITE(12,645) ED1,ED2,ED3,ED4

                             IF ((J.EQ.3).OR.(J.EQ.7)) THEN

                                  ED2=ED2+1
                                  ED4=ED4+1

                             ENDIF

830                CONTINUE

              ED1=NEDEP+27
              ED2=TEMPI1+95
              ED3=NEDEP+27+NLNIP2(JJ)
              ED4=NEDEP+28
              SURFAC=SURFAC+1
              WRITE(12,640) SURFAC,COLOR,DASH,GRADE,NUMB
```

```
                  WRITE(12,645) ED1,ED2,ED3,ED4

                          ED1=ED1-4
                          ED2=ED2-11
                          ED3=ED3-4
                          ED4=ED4+1
                          SURFAC=SURFAC+1
                          WRITE(12,640) SURFAC,COLOR,DASH,GRADE,NUMB
                          WRITE(12,645) ED1,ED2,ED3,ED4

                            DO 835 J=1,3

                              ED1=ED1+1
                              ED2=ED2-1
                              ED3=ED3+1
                              ED4=ED4+1
                              SURFAC=SURFAC+1
                              WRITE(12,640) SURFAC,COLOR,DASH,GRADE,NUMB
                              WRITE(12,645) ED1,ED2,ED3,ED4

                                 IF (J.EQ.2) ED2=ED2+13

 835                        CONTINUE

                  ED1=NEDEP+13
                  ED2=NEDEP+28
                  ED3=NEDEP+13+NLNIP2(JJ)
                  ED4=NEDEP+33
                  SURFAC=SURFAC+1
                  WRITE(12,640) SURFAC,COLOR,DASH,GRADE,NUMB
                  WRITE(12,645) ED1,ED2,ED3,ED4

                      DO 840 J=1,9

                          ED1=ED1+1
                          ED2=ED2+1
                          ED3=ED3+1
                          ED4=ED4+1
                          SURFAC=SURFAC+1
                          WRITE(12,640) SURFAC,COLOR,DASH,GRADE,NUMB
                          WRITE(12,645) ED1,ED2,ED3,ED4

 840                  CONTINUE

 845              NEDEP=NEDEP+NLNIP2(JJ)
                  TEMPI1=TEMPI1+NLNIP(JJ)
                  SURB=SURFAC
                  NSIP3(JJ)=SURB-SURT

 850          CONTINUE

        ENDIF

        NSRTL=SURFAC

        NSIP5=SURFAC+1-NSRC-NSEP1
        WRITE(NO,*) 'NUMBER OF SURFACES IN INTAKE PORT   = ',NSIP5

C    **************************************************************
C    *                                                            *
C    *           BEGIN GENERATION OF THE SURFACES OF THE          *
C    *                     SPARK PLUG PORT                        *
C    *                                                            *
C    **************************************************************
```

```
      DO 875 JJJ=1,ISP

           DO 870 JJ=1,IEND

                SURT=SURFAC
                ED1=NLNTL+1
                ED2=NLNTL+36
                ED3=NLNTL+11
                ED4=NLNTL+37
                SURFAC=SURFAC+1
                WRITE(12,640) SURFAC,COLOR,DASH,GRADE,NUMB
                WRITE(12,645) ED1,ED2,ED3,ED4

                     DO 855 J=1,19

                          ED1=ED1+1
                          ED2=ED2+1
                          ED3=ED3+1
                          ED4=ED4+1
                          SURFAC=SURFAC+1
                          WRITE(12,640) SURFAC,COLOR,DASH,GRADE,NUMB
                          WRITE(12,645) ED1,ED2,ED3,ED4

                               IF (J.EQ.5) ED4=ED4-34

                                 IF (J.EQ.6) THEN

                                    ED2=ED2-34
                                    ED3=ED3-8
                                    ED4=ED4+26

                                  ENDIF

                                 IF (J.EQ.7) THEN

                                    ED1=ED1+1
                                    ED2=ED2+33
                                    ED3=ED3+7
                                    ED4=ED4+7

                                  ENDIF

                                    IF (J.EQ.14) ED4=ED4-8

                                      IF (J.EQ.15) THEN

                                         ED1=ED1+2
                                         ED2=ED2-25
                                         ED3=ED3+6
                                         ED4=ED4-17

                                       ENDIF

                                          IF (J.EQ.17) ED1=ED1+2
                                          IF (J.EQ.17) ED2=ED2+1
                                          IF (J.EQ.17) ED4=ED4+1

855             CONTINUE

           IF (JJ.EQ.IEND) GO TO 865

                ED1=NLNTL+1
                ED2=NLNTL+51
```

```
                ED3=NLNTL+1+NLNSP(JJ)
                ED4=NLNTL+52
                SURFAC=SURFAC+1
                WRITE(12,640) SURFAC,COLOR,DASH,GRADE,NUMB
                WRITE(12,645) ED1,ED2,ED3,ED4

                     DO 860 J=1,49

                          ED1=ED1+1
                          ED2=ED2+1
                          ED3=ED3+1                                    %%[ PrinterEr
                          ED4=ED4+1
                          SURFAC=SURFAC+1
                          WRITE(12,640) SURFAC,COLOR,DASH,GRADE,NUMB
                          WRITE(12,645) ED1,ED2,ED3,ED4

                               IF (J.EQ.6) ED4=ED4-8

                                 IF (J.EQ.7) ED2=ED2-1
                                 IF (J.EQ.7) ED4=ED4+7

                                    IF (J.EQ.15) ED4=ED4-8

                                       IF (J.EQ.16) ED4=ED4+8

                                          IF (J.EQ.23) ED4=ED4-8

                                             IF (J.EQ.24) ED2=ED2-6
                                             IF (J.EQ.24) ED4=ED4+7

                                              IF (J.EQ.27) ED2=ED2+1

                                               IF (J.EQ.29) ED2=ED2-8

                                              IF (J.EQ.30) ED2=ED2+7
                                              IF (J.EQ.30) ED4=ED4-5

                                             IF (J.EQ.32) ED2=ED2+1
                                             IF (J.EQ.32) ED4=ED4+1

                                           IF (J.EQ.34) ED2=ED2-29
                                           IF (J.EQ.34) ED4=ED4-21

                                         IF (J.EQ.41) ED2=ED2+1

 860            CONTINUE

 865       NLNTL=NLNTL+NLNSP(JJ)
           SURB=SURFAC
           NSRSP(JJ)=SURB-SURT

 870     CONTINUE

 875  CONTINUE

      NSSP=SURFAC-NSRC-NSEP1-NSIP5+1
      WRITE(NO,*) 'NUMBER OF SURFACES IN SPARK PLUG(S) = ',NSSP

                ED1=ISTCN1+SPLINE
                ED2=ISTCN2+SPLINE
                ED3=ISTCN3+SPLINE
                ED4=ISTCN4+SPLINE
                SURFAC=SURFAC+1
                WRITE(20,*) SURFAC,NUMBER
```

```
                WRITE(12,640) SURFAC,COLOR,DASH,GRADE,NUMB
                WRITE(12,645) ED1,ED2,ED3,ED4

      CLOSE (20)
      WRITE(12,245)
      WRITE(12,245)

      WRITE(NO,*) 'TOTAL NUMBER OF SURFACES = ',SURFAC
      WRITE(NO,*) '  '
      WRITE(NO,*) '  '

      WRITE(NO,*)'SURFACE LABELS OF RC SEGMENTS  ARE         1  TO',NSRC
      WRITE(NO,*)'SURFACE LBLS OF EXHST PORT ARE',NSRC+1,'TO',NSEP1+NSRC
      WRITE(NO,*)'SURFACE LBLS OF INTKE PORT ARE',NSRC+NSEP1+1,'TO',NSIP
     #5+NSRC+NSEP1
      WRITE(NO,*)'SURFACE LBLS OF SPRK PLUGS ARE',NSRC+NSEP1+NSIP5+1,'TO
     #',NSSP+NSRC+NSEP1+NSIP5
      WRITE(NO,*) '  '
      WRITE(NO,*) '  '

C   ***********************************************************
C   *                                                         *
C   *    END GENERATION OF THE SURFACES IN UNIVERSAL FORMAT   *
C   *                                                         *
C   ***********************************************************

C   ***********************************************************
C   *                                                         *
C   *   BEGIN GENERATION OF THE VOLUMES OF THE INNER AND      *
C   *           OUTER SHELLS IN UNIVERSAL FORMAT              *
C   *                                                         *
C   ***********************************************************

      WRITE(12,880)
  880 FORMAT(4X,'39')

      SURFAC=TEMPSU
      NUMB=6
      VOLUME=1
      SUR1=1
      SUR2=6
      SUR3=SURFAC-3
           IF(CNLTYP(NUMBER).EQ.2) SUR3=SUR3-1
      SUR4=9
      SUR5=12
      SUR6=2

      WRITE(12,885) VOLUME,COLOR,DASH,GRADE,NUMB
  885 FORMAT(5I10)
      WRITE(12,890) SUR1,SUR2,SUR3,SUR4,SUR5,SUR6
  890 FORMAT(6I10)

        DO 900 I=1,NUMBER

            IF(RIBTYP(I).EQ.1) THEN

                SUR1=SUR1+1
                SUR2=SUR2+1
                SUR3=SUR3+1
                SUR4=SUR4+1
                SUR5=SUR5+1
                SUR6=SUR6+1
                VOLUME=VOLUME+1
```

```
          IF (I.NE.IVOLC(I)) THEN

             WRITE(12,885) VOLUME,COLOR,DASH,GRADE,NUMB
             WRITE(12,890) SUR1,SUR2,SUR3,SUR4,SUR5,SUR6

          ENDIF

      SUR1=SUR1+2
      SUR2=SUR2+1
      SUR3=SUR3+2
      SUR4=SUR4+1
      SUR5=SUR5+2
      SUR6=SUR6+2
      VOLUME=VOLUME+1

          IF (I.NE.IVOLC(I)) THEN

             WRITE(12,885) VOLUME,COLOR,DASH,GRADE,NUMB
             WRITE(12,890) SUR1,SUR2,SUR3,SUR4,SUR5,SUR6

          ENDIF

      SUR1=SUR1+12
      SUR2=SUR2+13
          IF(I.EQ.1) THEN

             SUR3=SUR3+12-SURFAC

          ELSE IF(CNLTYP(I-1).EQ.1) THEN
             SUR3=SUR3+12

          ELSE IF(CNLTYP(I-1).EQ.2) THEN

             SUR3=SUR3+13

          ENDIF

      SUR4=SUR4+14
      SUR5=SUR5+14
      SUR6=SUR6+12
      VOLUME=VOLUME+1

          IF (I.NE.IVOLR(I)) THEN

             WRITE(12,885) VOLUME,COLOR,DASH,GRADE,NUMB
             WRITE(12,890) SUR1,SUR2,SUR3,SUR4,SUR5,SUR6

          ENDIF

        DO 895 J=1,3

           SUR1=SUR1+1
           SUR2=SUR2+1
           SUR3=SUR3+1
           SUR4=SUR4+1
           SUR5=SUR5+1
           SUR6=SUR6+1
           VOLUME=VOLUME+1

          IF (I.NE.IVOLR(I)) THEN

             WRITE(12,885) VOLUME,COLOR,DASH,GRADE,NUMB
             WRITE(12,890) SUR1,SUR2,SUR3,SUR4,SUR5,SUR6
```

```
            ENDIF

 895        CONTINUE

      ELSE IF(RIBTYP(I).EQ.2) THEN

              SUR1=SUR1+1
              SUR2=SUR2+1
              SUR3=SUR3+1
              SUR4=SUR4+1
              SUR5=SUR5+1
              SUR6=SUR6+1
              VOLUME=VOLUME+1
              WRITE(12,885) VOLUME,COLOR,DASH,GRADE,NUMB
              WRITE(12,890) SUR1,SUR2,SUR3,SUR4,SUR5,SUR6

               SUR1=SUR1+2
               SUR2=SUR2+1
               SUR3=SUR3+1
       IF ((CNLTYP(I-1).EQ.2).OR.(RIBTYP(I-1).EQ.1)) SUR3=SUR3+1
       IF (RIBTYP(I-1).EQ.2) SUR3=SUR3+1
               SUR4=SUR4+1
               SUR5=SUR5+2
               SUR6=SUR6+2
               VOLUME=VOLUME+1
               WRITE(12,885) VOLUME,COLOR,DASH,GRADE,NUMB
               WRITE(12,890) SUR1,SUR2,SUR3,SUR4,SUR5,SUR6

                SUR1=SUR1+12
                SUR2=SUR2+14
                SUR3=SUR3+12
                IF(CNLTYP(I-1).EQ.2) SUR3=SUR3+1
                SUR4=SUR4+15
                SUR5=SUR5+15
                SUR6=SUR6+12
                VOLUME=VOLUME+1
                WRITE(12,885) VOLUME,COLOR,DASH,GRADE,NUMB
                WRITE(12,890) SUR1,SUR2,SUR3,SUR4,SUR5,SUR6

                 SUR1=SUR1+1
                 SUR2=SUR2+1
                 SUR3=SUR3+1
                 SUR4=SUR4+1
                 SUR5=SUR5+1
                 SUR6=SUR6+1
                 VOLUME=VOLUME+1
                 WRITE(12,885) VOLUME,COLOR,DASH,GRADE,NUMB
                 WRITE(12,890) SUR1,SUR2,SUR3,SUR4,SUR5,SUR6

                  SUR1=SUR1+2
                  SUR2=SUR2+1
                  SUR3=SUR3+1
                  SUR4=SUR4+1
                  SUR5=SUR5+1
                  SUR6=SUR6+2
                  VOLUME=VOLUME+1
                  WRITE(12,885) VOLUME,COLOR,DASH,GRADE,NUMB
                  WRITE(12,890) SUR1,SUR2,SUR3,SUR4,SUR5,SUR6


                   SUR1=SUR1+1
                   SUR2=SUR2+1
                   SUR3=SUR3+1
                   SUR4=SUR4+1
```

```
                      SUR5=SUR5+1
                      SUR6=SUR6+1
                      VOLUME=VOLUME+1
                      WRITE(12,885) VOLUME,COLOR,DASH,GRADE,NUMB
                      WRITE(12,890) SUR1,SUR2,SUR3,SUR4,SUR5,SUR6


      ENDIF

           IF(RIBTYP(I).EQ.1) THEN

                IF(CNLTYP(I).EQ.1) THEN

                   IF(I.EQ.NUMBER) GO TO 900

                     SUR1=SUR1+14
                     SUR2=SUR2+14
                     SUR3=SUR3+14
                     SUR4=SUR4+13
                     SUR5=SUR5+12
                     SUR6=SUR6+14
                     VOLUME=VOLUME+1

                  IF (I.NE.IVOLC(I+1)-1) THEN

                        WRITE(12,885) VOLUME,COLOR,DASH,GRADE,NUMB
                        WRITE(12,890) SUR1,SUR2,SUR3,SUR4,SUR5,SUR6

                  ENDIF

                ELSE IF(CNLTYP(I).EQ.2) THEN

                   IF(I.EQ.NUMBER) GO TO 900

                     SUR1=SUR1+15
                     SUR2=SUR2+15
                     SUR3=SUR3+14
                     SUR4=SUR4+14
                     SUR5=SUR5+13
                     SUR6=SUR6+15
                     VOLUME=VOLUME+1

                  IF (I.NE.IVOLC(I+1)-1) THEN

                       WRITE(12,885) VOLUME,COLOR,DASH,GRADE,NUMB
                       WRITE(12,890) SUR1,SUR2,SUR3,SUR4,SUR5,SUR6

                  ENDIF

                ENDIF

           ELSE IF(RIBTYP(I).EQ.2) THEN

                   IF(I.EQ.NUMBER) GO TO 900

                     SUR1=SUR1+14
                     SUR2=SUR2+14
                     SUR3=SUR3+15
                     SUR4=SUR4+13
                     SUR5=SUR5+12
                     SUR6=SUR6+14
                     VOLUME=VOLUME+1
                     WRITE(12,885) VOLUME,COLOR,DASH,GRADE,NUMB
                     WRITE(12,890) SUR1,SUR2,SUR3,SUR4,SUR5,SUR6
```

```
                ENDIF

 900  CONTINUE

      NVRC=VOLUME
      WRITE(NO,*) 'NUMBER OF VOLUMES IN RC SEGMENTS   = ',NVRC-14

C    *************************************************************
C    *                                                           *
C    *            BEGIN GENERATION OF THE VOLUMES OF THE         *
C    *               EXHAUST PORT IN UNIVERSAL FORMAT            *
C    *                                                           *
C    *************************************************************

      IF (IEPQUE.EQ.1) THEN

           DO 915 JJ=1,IEND-1

                SUR1=TEMPSU+1
                SUR2=TEMPSU+21
                SUR3=TEMPSU+46
                SUR4=TEMPSU+35
                SUR5=TEMPSU+47
                SUR6=TEMPSU+1+NSREP(JJ)

                     IF ((JJ.EQ.3).OR.(JJ.EQ.4)) THEN

                          SUR2=SUR2-8
                          SUR3=SUR3-8
                          SUR4=SUR4-8
                          SUR5=SUR5-8

                     ENDIF

                VOLUME=VOLUME+1
                WRITE(12,885) VOLUME,COLOR,DASH,GRADE,NUMB
                WRITE(12,890) SUR1,SUR2,SUR3,SUR4,SUR5,SUR6

                     DO 905 J=1,19

                          SUR1=SUR1+1
                          SUR2=SUR2+1
                          SUR3=SUR3+1
                          SUR4=SUR4+1
                          SUR5=SUR5+1
                          SUR6=SUR6+1
                          VOLUME=VOLUME+1
                          WRITE(12,885) VOLUME,COLOR,DASH,GRADE,NUMB
                          WRITE(12,890) SUR1,SUR2,SUR3,SUR4,SUR5,SUR6

                               IF (J.EQ.9) SUR5=SUR5-24

                               IF (J.EQ.10) THEN

                                    SUR3=SUR3-24
                                    SUR4=SUR4-12
                                    SUR5=SUR5+12
                               ENDIF

     IF(((JJ.EQ.2).OR.(JJ.EQ.3).OR.(JJ.EQ.4)).AND.(J.EQ.11)) GO TO 910

                                    IF (J.EQ.11) THEN

```

```
                            SUR2=SUR2+3
                            SUR3=SUR3+23
                            SUR4=SUR4+32
                            SUR5=SUR5+11

                         ENDIF

                            IF (J.EQ.15) THEN

                               SUR2=SUR2+2
                               SUR3=SUR3+1
                               SUR5=SUR5+1

                            ENDIF

 905                CONTINUE

 910             TEMPSU=TEMPSU+NSREP(JJ)

 915          CONTINUE

            NIMVEP=VOLUME+1

              SUR1=TEMSU1+1
              SUR2=TEMSU1+13
              SUR3=TEMSU1+25
              SUR4=TEMSU1+17
              SUR5=TEMSU1+26
              SUR6=TEMSU1+40
              VOLUME=VOLUME+1
              WRITE(12,885) VOLUME,COLOR,DASH,GRADE,NUMB
              WRITE(12,890) SUR1,SUR2,SUR3,SUR4,SUR5,SUR6

                 DO 920 J=1,11

                    SUR1=SUR1+1
                    SUR2=SUR2+1
                    SUR3=SUR3+1
                    SUR4=SUR4+1
                    SUR5=SUR5+1
                    SUR6=SUR6+1
                    VOLUME=VOLUME+1
                    WRITE(12,885) VOLUME,COLOR,DASH,GRADE,NUMB
                    WRITE(12,890) SUR1,SUR2,SUR3,SUR4,SUR5,SUR6

                       IF (J.EQ.3) SUR3=SUR3+1
                       IF (J.EQ.3) SUR5=SUR5+1

                          IF (J.EQ.7) THEN

                             SUR2=TEMSU+65
                             SUR3=SUR3+1
                             SUR4=SUR4-12
                             SUR5=SUR5+1

                          ENDIF

                           IF (J.EQ.8) SUR2=SUR2-33

                             IF (J.EQ.10) SUR2=SUR2+21

 920                CONTINUE

       ENDIF
```

```
      NVEP=VOLUME-NVRC
      WRITE(NO,*) 'NUMBER OF VOLUMES IN EXHAUST PORT  = ',NVEP

C  *************************************************************
C  *                                                           *
C  *           BEGIN GENERATION OF THE VOLUMES OF THE          *
C  *              INTAKE PORT IN UNIVERSAL FORMAT              *
C  *                                                           *
C  *************************************************************

      IF (IIPQUE.EQ.1) THEN

           DO 935 JJ=1,IEND-1

                SUR1=NSEP+1
                SUR2=NSEP+25
                SUR3=NSEP+52
                SUR4=NSEP+39
                SUR5=NSEP+53
                SUR6=NSEP+1+NSRIP(JJ)

                     IF ((JJ.EQ.3).OR.(JJ.EQ.4)) THEN

                          SUR2=SUR2-10
                          SUR3=SUR3-10
                          SUR4=SUR4-10
                          SUR5=SUR5-10

                     ENDIF

                VOLUME=VOLUME+1
                WRITE(12,885) VOLUME,COLOR,DASH,GRADE,NUMB
                WRITE(12,890) SUR1,SUR2,SUR3,SUR4,SUR5,SUR6

                     DO 925 J=1,21

                       IF (J.EQ.1) THEN

                        NUMB=5
                        SUR1=SUR1+1
                        SUR3=SUR3+1
                        SUR4=SUR4+1
                        SUR5=SUR5+1
                        SUR6=SUR6+1
                        VOLUME=VOLUME+1
                        WRITE(12,885) VOLUME,COLOR,DASH,GRADE,NUMB
                        WRITE(12,890) SUR1,SUR3,SUR4,SUR5,SUR6
                        NUMB=6

                      ENDIF

                          SUR1=SUR1+1
                          SUR2=SUR2+1
                          SUR3=SUR3+1
                          SUR4=SUR4+1
                          SUR5=SUR5+1
                          SUR6=SUR6+1
                          VOLUME=VOLUME+1
                          WRITE(12,885) VOLUME,COLOR,DASH,GRADE,NUMB
                          WRITE(12,890) SUR1,SUR2,SUR3,SUR4,SUR5,SUR6

                IF (J.EQ.10) THEN
```

```
                   NUMB=5
                   SUR1=SUR1+1
                   SUR3=SUR3+1
                   SUR4=SUR4+1
                   SUR5=SUR5-27
                   SUR6=SUR6+1
                   VOLUME=VOLUME+1
                   WRITE(12,885) VOLUME,COLOR,DASH,GRADE,NUMB
                   WRITE(12,890) SUR1,SUR3,SUR4,SUR5,SUR6
                   NUMB=6

                        SUR3=SUR3-28
                        SUR4=SUR4-14
                        SUR5=SUR5+14

              ENDIF

      IF(((JJ.EQ.2).OR.(JJ.EQ.3).OR.(JJ.EQ.4)).AND.(J.EQ.11)) GO TO 930

                                   IF (J.EQ.11) THEN

                                        SUR2=SUR2+3
                                        SUR3=SUR3+27
                                        SUR4=SUR4+38
                                        SUR5=SUR5+13
                                   ENDIF

                                        IF (J.EQ.16) THEN

                                             SUR2=SUR2+2
                                             SUR3=SUR3+1
                                             SUR5=SUR5+1

                                        ENDIF

925                  CONTINUE

930             NSEP=NSEP+NSRIP(JJ)

935       CONTINUE

        NSEP=NSEPX
        NIMVIP=VOLUME+1

               SUR1=NSIP1+1
               SUR2=NSIP1+13
               SUR3=NSIP1+25
               SUR4=NSEP+71
               SUR5=NSIP1+26
               SUR6=NSIP1+40
               VOLUME=VOLUME+1
               WRITE(12,885) VOLUME,COLOR,DASH,GRADE,NUMB
               WRITE(12,890) SUR1,SUR2,SUR3,SUR4,SUR5,SUR6

                SUR1=SUR1+1
                SUR2=SUR2+1
                SUR3=SUR3+1
                SUR4=SUR4-25
                SUR5=SUR5+1
                SUR6=SUR6+1
                VOLUME=VOLUME+1
                WRITE(12,885) VOLUME,COLOR,DASH,GRADE,NUMB
                WRITE(12,890) SUR1,SUR2,SUR3,SUR4,SUR5,SUR6
```

```
                    SUR4=SUR4-2

                      DO 940 J=1,10

                           SUR1=SUR1+1
                           SUR2=SUR2+1
                           SUR3=SUR3+1
                           SUR4=SUR4+1
                           SUR5=SUR5+1
                           SUR6=SUR6+1
                           VOLUME=VOLUME+1
                           WRITE(12,885) VOLUME,COLOR,DASH,GRADE,NUMB
                           WRITE(12,890) SUR1,SUR2,SUR3,SUR4,SUR5,SUR6

                              IF (J.EQ.1) SUR4=SUR4+24

                                IF (J.EQ.2) SUR3=SUR3+1
                                IF (J.EQ.2) SUR4=NSIP1+12
                                IF (J.EQ.2) SUR5=SUR5+1

                                   IF (J.EQ.6) SUR3=SUR3+1
                                   IF (J.EQ.6) SUR5=SUR5+1

940                   CONTINUE

                 NSIP1=NSIP1+51

            DO 950 JJ=5,IEND-1

                 SUR1=NSIP1+1
                 SUR2=NSIP1+13
                 SUR3=NSIP1+25
                 SUR4=NSIP2(JJ-1)+71
                 SUR5=NSIP1+26
                 SUR6=NSIP1+40
                 VOLUME=VOLUME+1
                 WRITE(12,885) VOLUME,COLOR,DASH,GRADE,NUMB
                 WRITE(12,890) SUR1,SUR2,SUR3,SUR4,SUR5,SUR6

                  SUR1=SUR1+1
                  SUR2=SUR2+1
                  SUR3=SUR3+1
                  SUR4=SUR4-25
                  SUR5=SUR5+1
                  SUR6=SUR6+1
                  VOLUME=VOLUME+1
                  WRITE(12,885) VOLUME,COLOR,DASH,GRADE,NUMB
                  WRITE(12,890) SUR1,SUR2,SUR3,SUR4,SUR5,SUR6

                    SUR4=SUR4-2

                      DO 945 J=1,10

                           SUR1=SUR1+1
                           SUR2=SUR2+1
                           SUR3=SUR3+1
                           SUR4=SUR4+1
                           SUR5=SUR5+1
                           SUR6=SUR6+1
                           VOLUME=VOLUME+1
                           WRITE(12,885) VOLUME,COLOR,DASH,GRADE,NUMB
                           WRITE(12,890) SUR1,SUR2,SUR3,SUR4,SUR5,SUR6

                              IF (J.EQ.1) SUR4=SUR4+24
```

```
                        IF (J.EQ.2) SUR3=SUR3+1
                        IF (J.EQ.2) SUR4=NSIP1+12
                        IF (J.EQ.2) SUR5=SUR5+1

                          IF (J.EQ.6) SUR3=SUR3+1
                          IF (J.EQ.6) SUR5=SUR5+1

 945                CONTINUE

              NSIP1=NSIP1+NSIP3(JJ)
              NSIP2(JJ)=NSIP2(JJ-1)+NSRIP(JJ)

 950          CONTINUE

        ENDIF

        NVIP=VOLUME-NVRC-NVEP
        WRITE(NO,*) 'NUMBER OF VOLUMES IN INTAKE PORT   = ',NVIP

C   ************************************************************
C   *                                                          *
C   *         BEGIN GENERATION OF THE VOLUMES OF THE           *
C   *             SPARK PLUG IN UNIVERSAL FORMAT               *
C   *                                                          *
C   ************************************************************

        DO 965 JJJ=1,ISP

          NIVSP2=VOLUME+1

             DO 960 JJ=1,IEND-1

                  SUR1=NSRTL+1
                  SUR2=NSRTL+21
                  SUR3=NSRTL+56
                  SUR4=NSRTL+31
                  SUR5=NSRTL+57
                  SUR6=NSRTL+1+NSRSP(JJ)
                  VOLUME=VOLUME+1
                  WRITE(12,885) VOLUME,COLOR,DASH,GRADE,NUMB
                  WRITE(12,890) SUR1,SUR2,SUR3,SUR4,SUR5,SUR6


                      DO 955 J=1,19

                          SUR1=SUR1+1
                          SUR2=SUR2+1
                          SUR3=SUR3+1
                          SUR4=SUR4+1
                          SUR5=SUR5+1
                          SUR6=SUR6+1
                          VOLUME=VOLUME+1
                          WRITE(12,885) VOLUME,COLOR,DASH,GRADE,NUMB
                          WRITE(12,890) SUR1,SUR2,SUR3,SUR4,SUR5,SUR6

                           IF (J.EQ.5) SUR5=SUR5-34

                              IF (J.EQ.6) THEN

                                 SUR3=SUR3-34
                                 SUR4=SUR4-8
                                 SUR5=SUR5+26
```

```
                              ENDIF

                              IF (J.EQ.7) THEN

                                  SUR2=SUR2+1
                                  SUR3=SUR3+33
                                  SUR4=SUR4+7
                                  SUR5=SUR5+7

                              ENDIF

                               IF (J.EQ.14) SUR5=SUR5-8

                                  IF (J.EQ.15) THEN

                                      SUR2=SUR2+2
                                      SUR3=SUR3-25
                                      SUR4=SUR4+6
                                      SUR5=SUR5-17

                                  ENDIF

                                     IF (J.EQ.17) THEN

                                        SUR2=SUR2+2
                                        SUR3=SUR3+1
                                        SUR5=SUR5+1

                                     ENDIF

 955                  CONTINUE

                   NSRTL=NSRTL+NSRSP(JJ)

 960          CONTINUE

         IF (JJ.EQ.IEND) NSRTL=NSRTL+NSRSP(IEND)

 965     CONTINUE

         NVSP=VOLUME-NVRC-NVEP-NVIP
         WRITE(NO,*) 'NUMBER OF VOLUMES IN SPARK PLUG(S) = ',NVSP

         WRITE(12,245)

         WRITE(NO,*) 'TOTAL NUMBER OF VOLUMES = ',VOLUME-14
         WRITE(NO,*) '  '
         WRITE(NO,*) '  '

         WRITE(NO,*)'VOLUME LABELS OF RC SEGMENTS  ARE           1 TO',NVRC
         WRITE(NO,*)'VOLUME LBLS OF EXHST PORT ARE ',NVRC+1,'TO',NVEP+NVRC
         WRITE(NO,*)'VOLUME LBLS OF INTKE PORT ARE ',NVRC+NVEP+1,'TO',NVIP
        #+NVRC+NVEP
         WRITE(NO,*)'VOLUME LBLS OF SPRK PLUGS ARE ',NVRC+NVEP+NVIP+1,'TO'
        #,NVSP+NVRC+NVEP+NVIP
         WRITE(NO,*) '  '
         WRITE(NO,*) '  '

         WRITE(13,970) NVRC+1,NIMVEP,NVRC+NVEP+1,NIMVIP
 970     FORMAT(4I4)

         WRITE(13,970) NVRC+NVEP+NVIP+1,NIVSP2,NVSP+NVRC+NVEP+NVIP,IEND
         WRITE(13,*) TCHNL,CCC,ISRCT,ISP
         CLOSE (13)
```

```
      WRITE(NO,975)
  975 FORMAT('0','ALL GEOMETRIC ENTITIES (POINTS THROUGH VOLUMES)')
      WRITE(NO,980)
  980 FORMAT('0','HAVE BEEN GENERATED.  THE PROGRAM CAN BE EXITED AT')
      WRITE(NO,985)
  985 FORMAT('0','THIS POINT IF MANUAL ELEMENT GENERATION IS DESIRED.')
      WRITE(NO,990)
  990 FORMAT('0','ENTER A ...')
      WRITE(NO,995)
  995 FORMAT('0','    1   TO   CONTINUE   MODEL GENERATION OR ...')
      WRITE(NO,999)
  999 FORMAT('0','    2   TO     STOP     MODEL GENERATION.')

      READ(5,*) ICONTI
      IF (STATUS.NE.0) WRITE(8,*) ICONTI
      WRITE(NO,*) ICONTI

      IF(ICONTI.EQ.2) THEN

       PRINT*,  ' '
       PRINT*,  '           THE UNIVERSAL HAS BEEN CREATED.'
       PRINT*,  ' '
       PRINT*,  '        BEFORE THE UNIVERSAL FILE CAN BE READ,'
       PRINT*,  '        THE USER MUST ENTER THE SDRC SOFTWARE.'

      PRINT*,  ' '
      PRINT*,  '    NOTE THAT A FILE CALLED "MODEL DATA" CAN NOT'
      PRINT*,  '   EXIST ON YOUR DISK.  IF IT DOES, AN ERROR WILL'
      PRINT*,  '    RESULT WHEN YOU ENTER THE SDRC SOFTWARE.  TO'
      PRINT*,  '     CORRECT THE ERROR, CHANGE THE NAME OF THE'
      PRINT*,  '    "MODEL DATA" FILE THAT EXISTS ON YOUR DISK'
      PRINT*,  '              TO A DIFFERENT NAME.'
      PRINT*,  ' '
      PRINT*,  ' AFTER ENTERING THE SDRC SOFTWARE IN THE PROGRAM MODE,'
      PRINT*,  '      RESPOND "R" (FOR RUN) TO THE FIRST QUESTION,'
      PRINT*,  '        RESPOND "GO" TO THE SECOND QUESTION AND'
      PRINT*,  '          IN RESPONSE TO THE THIRD QUESTION,'
      PRINT*,  '      ENTER THE TERMINAL TYPE THAT YOU ARE USING.'

      CALL GO (ICONTI)

      STOP
      ENDIF

      RETURN
      END

      SUBROUTINE MNE (NO)

      DIMENSION PHIONE(50),PHITWO(0:50),JJJ(50),JJJC(50),JJJR(50),
     #          IRIB(50),JJJSC(50),NUMES(20),JSUR(20)

      CHARACTER *10 NOAL,SLASH,GENERL,YES

           NOAL='4 -1NOAL'
           SLASH='1 -1/'
           GENERL='10 -1'
           YES='3 -1YES'

C    ***************************************************************
C    *                                                             *
C    *    READ VALUES THAT WERE CALCULATED IN THE MAIN PROGRAM     *
C    *                                                             *
C    ***************************************************************
```

```
            OPEN (13)
            OPEN (20)

            READ(20,*) NUMBER
            READ(13,*) PI,R,TRANS

               DO 15 J=1,NUMBER

                    READ(13,*) JJJ(J)
                    READ(13,5) PHIONE(J),PHITWO(J)
  5                 FORMAT(F13.5,5X,F13.5)
                    READ(13,*) IRIB(J)
                    READ(13,10) JJJC(J),JJJR(J)
 10                 FORMAT(F13.5,5X,F13.5)

 15            CONTINUE

                    READ(13,*) NIVEP,NIMVEP,NIVIP,NIMVIP
                    READ(13,*) NIVSP1,NIVSP2,NFVSP,IEND
                    READ(13,*) TCHNL,CCC,NSTCN,ISP
                    ISUR=1
                    MN=0
                    NTW=2

                         DO 20 III=1,NSTCN+1

                              READ(20,*) JJJSC(III),JSUR(III)

 20                      CONTINUE
C    ************************************************************
C    *                                                          *
C    *                       END READ                           *
C    *                                                          *
C    ************************************************************

      IF (CCC.GT.0.0) THEN

           I=1
           ISTOP=6
           ISTOP1=36
           ISTOP2=12
           ISTOP3=14

      ELSE

           I=2
           ISTOP=4
           ISTOP1=24
           ISTOP2=10
           ISTOP3=11

      ENDIF
C    ************************************************************
C    *                                                          *
C    *     SET ELEMENT TYPE FOR GENERATION OF THICK SHELLS      *
C    *                                                          *
C    ************************************************************

                         PHITWO(0)=ATAN((TRANS-TCHNL)/R)

           WRITE(15,25) GENERL
```

```
 25         FORMAT(1X,A5,'AU')

            WRITE(15,30) SLASH
 30         FORMAT(2X,A5)

            WRITE(15,35) GENERL
 35         FORMAT(1X,A5,'T')

            WRITE(15,40) GENERL
 40         FORMAT(1X,A5,'MA')

            WRITE(15,42) GENERL
 42         FORMAT(1X,A5,'EL')

            WRITE(15,44) GENERL
 44         FORMAT(1X,A5,'DEF')

            WRITE(15,46) GENERL
 46         FORMAT(1X,A5,'TY')

            WRITE(15,48) GENERL
 48         FORMAT(1X,A5,'TK')

            WRITE(15,50) GENERL
 50         FORMAT(1X,A5,'PB')

            WRITE(15,52) SLASH
 52         FORMAT(2X,A5)

            WRITE(15,54) GENERL
 54         FORMAT(1X,A5,'MD')

            WRITE(15,56) GENERL
 56         FORMAT(1X,A5,'CR')

            WRITE(15,58)
 58         FORMAT(2X,'4 27 K')

        DO 195 J=1,NUMBER

            DO 190 JJ=1,ISTOP
C    *************************************************************
C    *                                                           *
C    *       CALCUALTE THE LENGTH OF THE RIBS AND CHANNELS       *
C    *                                                           *
C    *************************************************************

                IF ((JJJ(J-1).EQ.3).AND.(JJJ(J).EQ.2)) THEN

                    IF ((JJ.LT.4).AND.(CCC.GT.0.0)) THEN

                        PHIT=PHITWO(J)*PI/180.0
                        Y=R*SIN(PHIT)+TRANS
                        PHI=ABS(ABS(Y)-ABS(PHIONE(J)))

                    ELSE IF ((JJ.GT.3).AND.(CCC.GT.0.0)) THEN

                        PHI=ABS(ABS(PHIONE(J))-ABS(PHITWO(J)))

                    ENDIF

                    IF ((JJ.LT.3).AND.(CCC.EQ.0.0)) THEN
```

```
                PHIT=PHITWO(J)*PI/180.0
                Y=R*SIN(PHIT)+TRANS
                PHI=ABS(ABS(Y)-ABS(PHIONE(J)))

           ELSE IF ((JJ.GT.2).AND.(CCC.EQ.0.0)) THEN

                PHI=ABS(ABS(PHIONE(J))-ABS(PHITWO(J)))

           ENDIF

      ELSE IF ((JJJ(J).EQ.3).AND.(JJJ(J-1).EQ.2)) THEN

           IF ((JJ.LT.4).AND.(CCC.GT.0.0)) THEN

                PHIT=PHIONE(J)*PI/180.0
                Y=R*SIN(PHIT)+TRANS
                PHI=ABS(ABS(Y)-ABS(PHITWO(J-1)))

           ELSE IF ((JJ.GT.3).AND.(CCC.GT.0.0)) THEN

                PHI=ABS(PHITWO(J)-PHIONE(J))

           ENDIF

           IF ((JJ.LT.3).AND.(CCC.EQ.0.0)) THEN

                PHIT=PHIONE(J)*PI/180.0
                Y=R*SIN(PHIT)+TRANS
                PHI=ABS(ABS(Y)-ABS(PHITWO(J-1)))

           ELSE IF ((JJ.GT.2).AND.(CCC.EQ.0.0)) THEN

                PHI=ABS(PHITWO(J)-PHIONE(J))

           ENDIF

      ELSE IF (JJJ(J).EQ.2) THEN

           IF ((JJ.LT.4).AND.(CCC.GT.0.0)) THEN

                PHI=ABS(ABS(PHIONE(J))-ABS(PHITWO(J-1)))

           ELSE IF ((JJ.GT.3).AND.(CCC.GT.0.0)) THEN

                PHI=ABS(ABS(PHIONE(J))-ABS(PHITWO(J)))

           ENDIF

           IF ((JJ.LT.3).AND.(CCC.EQ.0.0)) THEN

                PHI=ABS(ABS(PHIONE(J))-ABS(PHITWO(J-1)))

           ELSE IF ((JJ.GT.2).AND.(CCC.EQ.0.0)) THEN

                PHI=ABS(ABS(PHIONE(J))-ABS(PHITWO(J)))

           ENDIF

      ELSE IF ((JJJ(J).EQ.1).OR.(JJJ(J).EQ.3)) THEN

           IF ((JJ.LT.4).AND.(CCC.GT.0.0)) THEN

                PHI=ABS(PHIONE(J)-PHITWO(J-1))
```

```
                   ELSE IF ((JJ.GT.3).AND.(CCC.GT.0.0)) THEN

                       PHI=ABS(PHITWO(J)-PHIONE(J))

                   ENDIF

                   IF ((JJ.LT.3).AND.(CCC.EQ.0.0)) THEN

                       PHI=ABS(PHIONE(J)-PHITWO(J-1))

                   ELSE IF ((JJ.GT.2).AND.(CCC.EQ.0.0)) THEN

                       PHI=ABS(PHITWO(J)-PHIONE(J))

                   ENDIF

               ENDIF

C   ************************************************************
C   *                                                          *
C   *            SET THE NUMBER OF ELEMENTS ON AN EDGE.        *
C   *        THE NUMBER IS DEPENDENT UPON THE LENGTH OF THE    *
C   *                     RIB OR CHANNEL                       *
C   *                                                          *
C   ************************************************************

                  IF (JJJ(J).EQ.2) THEN

                       IF (PHI.GT.0.80) THEN

                           NUMEA=3

                       ELSE IF ((PHI.GT.0.40).AND.(PHI.LE.0.80)) THEN

                           NUMEA=2

                       ELSE

                           NUMEA=2

                       ENDIF

                  ELSE IF ((JJJ(J).EQ.1).OR.(JJJ(J).EQ.3)) THEN

                       IF (PHI.GT.5.0) THEN

                           NUMEA=3

                       ELSE IF ((PHI.GT.3.0).AND.(PHI.LE.5.0)) THEN

                           NUMEA=2

                       ELSE

                           NUMEA=2

                       ENDIF
                  ENDIF

         IF ((J.EQ.JSUR(ISUR)).AND.(JJ.EQ.1)) THEN

             NUMES(ISUR)=NUMEA
             ISUR=ISUR+1

```

```
      ENDIF

      IF (CCC.GT.0.0) THEN

         IF((JJJC(J).NE.0).AND.(JJ.LT.4)) GO TO 190

          IF ((JJJC(J).NE.0).AND.(JJ.EQ.4)) I=I+3

            IF((JJJR(J).NE.0).AND.(JJ.GT.3)) GO TO 190

             IF ((J.NE.1).AND.(JJJR(J-1).NE.0).AND.(JJ.EQ.1)) I=I+4

      ELSE IF (CCC.EQ.0.0) THEN

         IF((JJJC(J).NE.0).AND.(JJ.LT.3)) GO TO 190

          IF ((JJJC(J).NE.0).AND.(JJ.EQ.3)) I=I+3

            IF((JJJR(J).NE.0).AND.(JJ.GT.2)) GO TO 190

             IF ((J.NE.1).AND.(JJJR(J-1).NE.0).AND.(JJ.EQ.1)) I=I+4

      ENDIF
C    ************************************************************
C    *                                                          *
C    *      BEGIN THE GENERATION OF THE COMMANDS THAT WILL      *
C    *      CREATE THE MESHES OF THE INNER AND OUTER SHELLS     *
C    *                                                          *
C    ************************************************************

            IA=I/100
            IB=(I-IA*100)/10
            IC=I-IA*100-IB*10

                 IF (J.EQ.1) NUMEA=3

                      IF (IA.NE.0) THEN

                           WRITE(15,65) GENERL,IA,IB,IC
 65                        FORMAT(1X,A5,'V',3I1)

                      ELSE IF (IB.NE.0) THEN

                           WRITE(15,70) GENERL,IB,IC
 70                        FORMAT( 1X,A5,'V',2I1)

                      ELSE

                           WRITE(15,75) GENERL,IC
 75                        FORMAT( 1X,A5,'V',I1)

                      ENDIF

            MN=MN+1

       IF (CCC.GT.0.0) THEN

            IF ((JJ.EQ.2).OR.(JJ.EQ.5)) GO TO 85

                 WRITE(15,80) GENERL,NUMEA
 80              FORMAT(1X,A5,I1)

 85         IF ((J.EQ.1).AND.(JJ.EQ.2)) THEN
```

```
            WRITE(15,90) GENERL
 90         FORMAT(1X,A5,'1')

         ENDIF

         IF ((JJJC(J).NE.0).AND.(JJ.EQ.5)) THEN

            WRITE(15,95) GENERL
 95         FORMAT(1X,A5,'1')

         ENDIF

         IF ((JJJR(J-1).NE.0).AND.(JJ.EQ.2)) THEN

            WRITE(15,100) GENERL
 100        FORMAT(1X,A5,'1')

         ENDIF

                WRITE(15,105) NOAL
 105             FORMAT(2X,A8)

            IF ((J.EQ.1).AND.((JJ.EQ.1).OR.(JJ.EQ.3))) THEN

                WRITE(15,110) GENERL
 110            FORMAT(1X,A5,'4')

                WRITE(15,115) GENERL
 115            FORMAT(1X,A5,'1')

            ENDIF

            IF ((JJJC(J).NE.0).AND.((JJ.EQ.4).OR.(JJ.EQ.6))) THEN

              IF (J.EQ.NUMBER) GO TO 130

                WRITE(15,120) GENERL
 120            FORMAT(1X,A5,'4')

                WRITE(15,125) GENERL
 125            FORMAT(1X,A5,'1')

              ENDIF

 130        IF (((JJJR(J-1).NE.0).AND.(J.NE.1)).AND.((JJ.EQ.1)
     #         .OR.(JJ.EQ.3))) THEN

                WRITE(15,135) GENERL
 135            FORMAT(1X,A5,'4')

                WRITE(15,140) GENERL
 140            FORMAT(1X,A5,'1')

            ENDIF

             ELSE IF (CCC.EQ.0.0) THEN

              WRITE(15,145) GENERL,NUMEA
 145          FORMAT(1X,A5,I1)

              WRITE(15,150) NOAL
 150          FORMAT(2X,A8)
```

```
                IF ((J.EQ.1).AND.((JJ.EQ.1).OR.(JJ.EQ.2))) THEN

                      WRITE(15,155) GENERL
  155                 FORMAT(1X,A5,'4')

                      WRITE(15,160) GENERL
  160                 FORMAT(1X,A5,'1')

                ENDIF

                IF ((JJJC(J).NE.0).AND.((JJ.EQ.3).OR.(JJ.EQ.4))) THEN

                  IF (J.EQ.NUMBER) GO TO 175

                     WRITE(15,165) GENERL
  165                FORMAT(1X,A5,'4')

                     WRITE(15,170) GENERL
  170                FORMAT(1X,A5,'1')

                ENDIF

  175            IF (((JJJR(J-1).NE.0).AND.(J.NE.1)).AND.((JJ.EQ.1)
     #              .OR.(JJ.EQ.2))) THEN

                   WRITE(15,180) GENERL
  180              FORMAT(1X,A5,'4')

                   WRITE(15,185) GENERL
  185              FORMAT(1X,A5,'1')

                ENDIF
      ENDIF

      IF (CCC.GT.0.0) THEN

           IF (JJ.EQ.5) THEN

                I=I+2
           ELSE
                I=I+1

           ENDIF

      ELSE IF (CCC.EQ.0.0) THEN

           IF (JJ.EQ.1) THEN

                I=I+1
           ELSE
                I=I+2

           ENDIF
      ENDIF

  190 CONTINUE

  195 CONTINUE

        NTKMRC=MN

           WRITE(15,200)
  200      FORMAT(2X,'0 -1')
```

```
C    ************************************************************
C    *                                                          *
C    *      BEGIN THE GENERATION OF THE COMMANDS THAT WILL      *
C    *           CREATE THE MESHES OF THE EXHAUST PORT          *
C    *                                                          *
C    ************************************************************

          WRITE(15,30) SLASH
          WRITE(15,35) GENERL
          WRITE(15,40) GENERL
          WRITE(15,54) GENERL
          WRITE(15,56) GENERL
          WRITE(15,58)
          I=NIVEP

        DO 240 JJ=1,IEND-1

             DO 235 IJJ=1,20

                    IF (((JJ.EQ.2).OR.(JJ.EQ.3).OR.(JJ.EQ.4)).AND.
     #                 (IJJ.EQ.13)) GO TO 240

          IA=I/100
          IB=(I-IA*100)/10
          IC=I-IA*100-IB*10

                    IF (IA.NE.0) THEN

                         WRITE(15,205) GENERL,IA,IB,IC
 205                     FORMAT(1X,A5,'V',3I1)

                    ELSE IF (IB.NE.0) THEN

                         WRITE(15,210) GENERL,IB,IC
 210                     FORMAT( 1X,A5,'V',2I1)

                    ELSE

                         WRITE(15,215) GENERL,IC
 215                     FORMAT( 1X,A5,'V',I1)

                    ENDIF

          MN=MN+1
          N=1
          IF (I.EQ.NIVEP) N=3

     IF ((JJ.EQ.1).OR.(IJJ.EQ.1).OR.(IJJ.EQ.13).OR.(IJJ.EQ.17)) THEN

       IF ((JJ.EQ.6).AND.((IJJ.EQ.13).OR.(IJJ.EQ.17))) GO TO 230

          IF ((IJJ.EQ.14).OR.(IJJ.EQ.15).OR.(IJJ.EQ.16).OR.
     #       (IJJ.EQ.18).OR.(IJJ.EQ.19).OR.(IJJ.EQ.20)) GO TO 230

               DO 225 JJJJ=1,N

                    WRITE(15,220) GENERL
 220                FORMAT(1X,A5,'1')

 225           CONTINUE

          ENDIF

 230      I=I+1
```

```
           IF (I.EQ.NIMVEP) GO TO 245

 235            CONTINUE

 240       CONTINUE

C     INTEMEDIATE EXHAUST   ***********************************

 245  DO 280 L=1,12

           IA=I/100
           IB=(I-IA*100)/10
           IC=I-IA*100-IB*10

                     IF (IA.NE.0) THEN

                        WRITE(15,250) GENERL,IA,IB,IC
 250                    FORMAT(1X,A5,'V',3I1)

                     ELSE IF (IB.NE.0) THEN

                        WRITE(15,255) GENERL,IB,IC
 255                    FORMAT( 1X,A5,'V',2I1)

                     ELSE

                        WRITE(15,260) GENERL,IC
 260                    FORMAT( 1X,A5,'V',I1)

                     ENDIF

                  MN=MN+1

                IF (L.EQ.1) THEN

                     N=3
                ELSE
                     N=1

                ENDIF

           IF ((L.EQ.6).OR.(L.EQ.7).OR.(L.EQ.8).OR.(L.EQ.10).OR.
     #     (L.EQ.11).OR.(L.EQ.12)) GO TO 275

                DO 270 JJJJ=1,N

                     WRITE(15,265) GENERL
 265                 FORMAT(1X,A5,'1')

 270            CONTINUE

 275         I=I+1

 280       CONTINUE

           WRITE(15,200)

          NTKMEX=MN-NTKMRC

C   *************************************************************
C   *                                                           *
C   *      BEGIN THE GENERATION OF THE COMMANDS THAT WILL       *
C   *           CREATE THE MESHES OF THE INTAKE PORT            *
C   *                                                           *
```

```
C     ************************************************************

          WRITE(15,30) SLASH
          WRITE(15,35) GENERL
          WRITE(15,40) GENERL
          WRITE(15,54) GENERL
          WRITE(15,56) GENERL
          WRITE(15,58)
          I=NIVIP

        DO 320 JJ=1,IEND-1

              DO 315 IJJ=1,24

                  IF (((JJ.EQ.2).OR.(JJ.EQ.3).OR.(JJ.EQ.4)).AND.
     #                (IJJ.EQ.15)) GO TO 320

          IA=I/100
          IB=(I-IA*100)/10
          IC=I-IA*100-IB*10

                  IF (IA.NE.0) THEN

                      WRITE(15,285) GENERL,IA,IB,IC
 285                  FORMAT(1X,A5,'V',3I1)

                  ELSE IF (IB.NE.0) THEN

                      WRITE(15,290) GENERL,IB,IC
 290                  FORMAT( 1X,A5,'V',2I1)

                  ELSE

                      WRITE(15,295) GENERL,IC
 295                  FORMAT( 1X,A5,'V',I1)

                  ENDIF

          MN=MN+1
          N=1
          IF (I.EQ.NIVIP) N=3
     IF ((JJ.EQ.1).OR.(IJJ.EQ.1).OR.(IJJ.EQ.15).OR.(IJJ.EQ.20)) THEN

       IF ((JJ.EQ.6).AND.((IJJ.EQ.15).OR.(IJJ.EQ.20))) GO TO 310

          IF ((IJJ.EQ.16).OR.(IJJ.EQ.17).OR.(IJJ.EQ.18).OR.
     #       (IJJ.EQ.19).OR.(IJJ.EQ.21).OR.(IJJ.EQ.22).OR.
     #       (IJJ.EQ.23).OR.(IJJ.EQ.24)) GO TO 310

              DO 305 JJJJ=1,N

                  WRITE(15,300) GENERL
 300              FORMAT(1X,A5,'1')

 305          CONTINUE

     ENDIF

 310      I=I+1
          IF (I.EQ.NIMVIP) GO TO 325

 315          CONTINUE

 320       CONTINUE
```

```
C     INTERMEDIATE INTAKE    ************************************

 325      DO 350 L=1,ISTOP1

            IA=I/100
            IB=(I-IA*100)/10
            IC=I-IA*100-IB*10

                      IF (IA.NE.0) THEN

                         WRITE(15,330) GENERL,IA,IB,IC
 330                     FORMAT(1X,A5,'V',3I1)

                      ELSE IF (IB.NE.0) THEN

                         WRITE(15,335) GENERL,IB,IC
 335                     FORMAT( 1X,A5,'V',2I1)

                      ELSE

                         WRITE(15,340) GENERL,IC
 340                     FORMAT( 1X,A5,'V',I1)

                      ENDIF

               MN=MN+1

            IF ((L.EQ.1).OR.(L.EQ.5).OR.(L.EQ.9).OR.(L.EQ.13)
     #         .OR.(L.EQ.17).OR.(L.EQ.21)) THEN

                      WRITE(15,345) GENERL
 345                  FORMAT(1X,A5,'1')

            ENDIF

              I=I+1

 350        CONTINUE

               NTKMIN=MN-NTKMRC-NTKMEX

            WRITE(15,200)

C     ************************************************************
C     *                                                          *
C     *      BEGIN THE GENERATION OF THE COMMANDS THAT WILL      *
C     *        CREATE THE MESHES OF THE SPARK PLUG PORT          *
C     *                                                          *
C     ************************************************************

            WRITE(15,30) SLASH
            WRITE(15,35) GENERL
            WRITE(15,40) GENERL
            WRITE(15,54) GENERL
            WRITE(15,56) GENERL
            WRITE(15,58)
            I=NIVSP1

        DO 405 IN=1,ISP

           DO 400 JJ=1,IEND-1

                 DO 395 IJJ=1,20
```

```
          IA=I/100
          IB=(I-IA*100)/10
          IC=I-IA*100-IB*10

                        IF (IA.NE.0) THEN

                           WRITE(15,355) GENERL,IA,IB,IC
 355                       FORMAT(1X,A5,'V',3I1)

                        ELSE IF (IB.NE.0) THEN

                           WRITE(15,360) GENERL,IB,IC
 360                       FORMAT( 1X,A5,'V',2I1)

                        ELSE

                           WRITE(15,365) GENERL,IC
 365                       FORMAT( 1X,A5,'V',I1)

                        ENDIF

          MN=MN+1
          N=1
          IF ((I.EQ.NIVSP1).OR.(I.EQ.NIVSP2)) N=3

      IF (JJ.EQ.1) THEN

        IF ((IJJ.EQ.1).OR.(IJJ.EQ.2).OR.(IJJ.EQ.3).OR.(IJJ.EQ.4)
     #  .OR.(IJJ.EQ.5).OR.(IJJ.EQ.6).OR.(IJJ.EQ.7).OR.(IJJ.EQ.8)
     #  .OR.(IJJ.EQ.9).OR.(IJJ.EQ.17.).OR.(IJJ.EQ.19)) THEN

                  DO 375 JJJJ=1,N

                        WRITE(15,370) GENERL
 370                    FORMAT(1X,A5,'1')

 375              CONTINUE

      ENDIF

          ELSE IF ((JJ.GT.1).AND.(IJJ.EQ.1)) THEN

                  DO 385 JJJJ=1,N

                        WRITE(15,380) GENERL
 380                    FORMAT(1X,A5,'1')

 385              CONTINUE

      ENDIF

                        I=I+1

 395              CONTINUE

 400        CONTINUE

 405  CONTINUE

                  NTKMSP=MN-NTKMRC-NTKMEX-NTKMIN

C    ************************************************************
C    *                                                          *
```

```
C    *      BEGIN THE GENERATION OF THE COMMANDS THAT WILL        *
C    *              CREATE THE MESHES OF THE RIBS                 *
C    *                                                            *
C    **************************************************************

C    **************************************************************
C    *                                                            *
C    *     SET ELEMENT TYPE FOR GENERATION OF SOLID ELEMENTS      *
C    *                                                            *
C    **************************************************************

             WRITE(15,200)

             WRITE(15,30) SLASH
             WRITE(15,35) GENERL
             WRITE(15,40) GENERL

             WRITE(15,420) GENERL
  420        FORMAT(1X,A5,'EL')

             WRITE(15,425) GENERL
  425        FORMAT(1X,A5,'DEF')

             WRITE(15,430) GENERL
  430        FORMAT(1X,A5,'TY')

             WRITE(15,435) GENERL
  435        FORMAT(1X,A5,'SOL')

             WRITE(15,440) GENERL
  440        FORMAT(1X,A5,'PB')

             WRITE(15,30) SLASH
             WRITE(15,54) GENERL
             WRITE(15,56) GENERL
             WRITE(15,58)

               I=6
               NUMEA=3

                  DO 505 J=1,NUMBER

                    IF (JJJR(J).NE.0) GO TO 500

                     IA=I/100
                     IB=(I-IA*100)/10
                     IC=I-IA*100-IB*10

                        IF (IA.NE.0) THEN

                          WRITE(15,465) GENERL,IA,IB,IC
  465                     FORMAT(1X,A5,'V',3I1)

                        ELSE IF (IB.NE.0) THEN

                          WRITE(15,470) GENERL,IB,IC
  470                     FORMAT( 1X,A5,'V',2I1)

                        ELSE

                          WRITE(15,475) GENERL,IC
  475                     FORMAT( 1X,A5,'V',I1)

                        ENDIF
```

```
                 MN=MN+1

            IF (IRIB(J).EQ.1) THEN

               WRITE(15,480) GENERL,NUMEA
 480           FORMAT(1X,A5,I1)

               WRITE(15,485) NOAL
 485           FORMAT(2X,A8)

            ELSE

               WRITE(15,490) NOAL
 490           FORMAT(2X,A8)

               WRITE(15,495) GENERL,NUMEA
 495           FORMAT(1X,A5,I1)

            ENDIF

 500         I=I+7

 505        CONTINUE

                        NSLMRB=MN-NTKMRC-NTKMEX-NTKMIN-NTKMSP

C    ***************************************************************
C    *                                                             *
C    *      BEGIN THE GENERATION OF THE COMMANDS THAT WILL         *
C    *       CREATE THE MESHES OF THE STIFFENED CHANNELS           *
C    *                                                             *
C    ***************************************************************

C    ***************************************************************
C    *                                                             *
C    *     SET ELEMENT TYPE FOR GENERATION OF THIN SHELLS          *
C    *                                                             *
C    ***************************************************************

            WRITE(15,200)
            WRITE(15,30) SLASH
            WRITE(15,35) GENERL
            WRITE(15,40) GENERL

            WRITE(15,420) GENERL
            WRITE(15,425) GENERL
            WRITE(15,430) GENERL

            WRITE(15,520) GENERL
 520        FORMAT(1X,A5,'TN')

            WRITE(15,525) GENERL
 525        FORMAT(1X,A5,'PQ')

            WRITE(15,30) SLASH
            WRITE(15,54) GENERL
            WRITE(15,56) GENERL
            WRITE(15,58)

                 DO 575 JJ=1,NSTCN+1

                      I=JJJSC(JJ)
                      IA=I/1000
```

```
                  IB=(I-IA*1000)/100
                  IC=(I-IA*1000-IB*100)/10
                  ID=I-IA*1000-IB*100-IC*10

                   IF (IA.NE.0) THEN

                       WRITE(15,545) GENERL,IA,IB,IC,ID
  545                  FORMAT(1X,A5,'SU',4I1)

                  ELSE IF (IB.NE.0) THEN

                       WRITE(15,550) GENERL,IB,IC,ID
  550                  FORMAT( 1X,A5,'SU',3I1)

                  ELSEIF (IC.NE.0) THEN

                       WRITE(15,555) GENERL,IC,ID
  555                  FORMAT( 1X,A5,'SU',2I1)

                  ELSE

                       WRITE(15,560) GENERL,ID
  560                  FORMAT( 1X,A5,'SU',I1)

                  ENDIF

                       MN=MN+1

                       WRITE(15,565) GENERL,NUMES(JJ)
  565                  FORMAT(1X,A5,I1)

                       WRITE(15,485) NOAL

                       WRITE(15,570) GENERL
  570                  FORMAT(1X,A5,'3')

  575      CONTINUE

               NTNMSC=MN-NTKMRC-NTKMEX-NTKMIN-NTKMSP-NSLMRB
C    *************************************************************
C    *                                                           *
C    *                  END MESH GENERATION                      *
C    *                                                           *
C    *************************************************************

C    *************************************************************
C    *                                                           *
C    *      BEGIN THE GENERATION OF THE COMMANDS THAT WILL       *
C    *             CREATE THE NODES AND ELEMENTS                 *
C    *                                                           *
C    *************************************************************

C    *************************************************************
C    *                                                           *
C    *           BEGIN NODE AND ELEMENT GENERATION               *
C    *              OF THE INNER AND OUTER SHELLS                *
C    *                                                           *
C    *************************************************************

C    *************************************************************
C    *                                                           *
C    *        SET THE TWO DIFFERENT MATERIAL PROPERTIES          *
C    *                                                           *
```

```
C     **************************************************************

            WRITE(15,200)
            WRITE(15,30) SLASH
            WRITE(15,35) GENERL
            WRITE(15,40) GENERL
            WRITE(15,420) GENERL

            WRITE(15,580) GENERL
  580       FORMAT(1X,A5,'MAT')

            WRITE(15,585) GENERL
  585       FORMAT(1X,A5,'EN')

            WRITE(15,590) GENERL
  590       FORMAT(1X,A5,'1')

            WRITE(15,595) GENERL
  595       FORMAT(1X,A5,'10E6')

            WRITE(15,600) GENERL
  600       FORMAT(1X,A5,'0.30')

                 DO 610 J=1,9

                      WRITE(15,605)
  605                 FORMAT(2X,'0 -1')

  610            CONTINUE

            WRITE(15,585) GENERL

            WRITE(15,615) GENERL
  615       FORMAT(1X,A5,'2')

            WRITE(15,620) GENERL
  620       FORMAT(1X,A5,'20E6')

            WRITE(15,625) GENERL
  625       FORMAT(1X,A5,'0.25')

                 DO 630 J=1,9

                      WRITE(15,605)

  630            CONTINUE

            WRITE(15,30) SLASH
            WRITE(15,35) GENERL
            WRITE(15,40) GENERL
            WRITE(15,30) SLASH
            WRITE(15,420) GENERL
            WRITE(15,425) GENERL
            WRITE(15,430) GENERL
            WRITE(15,48) GENERL
            WRITE(15,50) GENERL
            WRITE(15,30) SLASH

C     **************************************************************
C     *                                                            *
C     *                  END MATERIAL PROPERTY SET                 *
C     *                                                            *
C     **************************************************************
```

```
          WRITE(15,635) GENERL
 635      FORMAT(1X,A5,'NE')

          WRITE(15,58)

          WRITE(15,640) GENERL,NTKMRC
 640      FORMAT(1X,A5,'1',1X,I3,1X,'1')

            I=0

               DO 665 JJ=1,NUMBER

                    DO 660 IJJ=1,ISTOP

                      I=I+1
                      IF (I.GT.NTKMRC) GO TO 670

                         WRITE(15,645) GENERL
 645                     FORMAT(1X,A5,'1')

                              IF ((JJ.EQ.1).AND.(IJJ.EQ.1)) THEN

                                   WRITE(15,650) YES
 650                               FORMAT(2X,A8)

                              ENDIF

               IF ((CCC.GT.0.0).AND.((IJJ.EQ.1).OR.(IJJ.EQ.4))) THEN

                     WRITE(15,655) GENERL
 655                 FORMAT(1X,A5,'2')

               ELSE

                     WRITE(15,645) GENERL

               ENDIF

                      IF ((JJ.EQ.1).AND.(IJJ.EQ.1)) THEN

                           WRITE(15,650) YES

                      ENDIF

 660                CONTINUE
 665          CONTINUE

C    ***********************************************************
C    *                                                         *
C    *           BEGIN NODE AND ELEMENT GENERATION             *
C    *                  OF THE EXHAUST PORT                    *
C    *                                                         *
C    ***********************************************************

 670      WRITE(15,30) SLASH
          WRITE(15,35) GENERL
          WRITE(15,40) GENERL
          WRITE(15,635) GENERL
          WRITE(15,58)

          WRITE(15,675) GENERL,NTKMRC+1,NTKMRC+NTKMEX
 675      FORMAT(1X,A5,I3,1X,I3,1X,'1')

               DO 680 JJ=1,NTKMEX
```

```

                    WRITE(15,645) GENERL

                        IF (JJ.EQ.1) THEN

                            WRITE(15,650) YES

                        ENDIF

              IF ((CCC.GT.0.0).AND.((IJJ.EQ.1).OR.(IJJ.EQ.4))) THEN

                    WRITE(15,655) GENERL

              ELSE

                    WRITE(15,645) GENERL

              ENDIF

                        IF (JJ.EQ.1) THEN

                            WRITE(15,650) YES

                        ENDIF

 680          CONTINUE

          WRITE(15,30) SLASH

C   ************************************************************
C   *                                                          *
C   *            BEGIN NODE AND ELEMENT GENERATION             *
C   *                  OF THE INTAKE PORT                      *
C   *                                                          *
C   ************************************************************

          WRITE(15,30) SLASH
          WRITE(15,35) GENERL
          WRITE(15,40) GENERL
          NTW1=1

        DO 695 JJ=1,ISTOP3

          WRITE(15,635) GENERL

          WRITE(15,58)

         IF ((JJ.EQ.1).OR.(JJ.EQ.12)) THEN

       WRITE(15,685) GENERL,NTKMRC+NTKMEX+NTW1
 685   FORMAT(1X,A5,I3)
       NTW2=NTW1

        ELSE

       WRITE(15,675) GENERL,NTKMRC+NTKMEX+NTW1,NTKMRC+NTKMEX+NTW2

        ENDIF
             III=1

        IF ((JJ.NE.1).OR.(JJ.NE.12)) THEN

          DO 690 K=NTKMRC+NTKMEX+NTW1,NTKMRC+NTKMEX+NTW2

```

```
                         WRITE(15,645) GENERL

                              IF (III.EQ.1) THEN

                                   WRITE(15,650) YES

                              ENDIF

                  IF ((CCC.GT.0.0).AND.((IJJ.EQ.1).OR.(IJJ.EQ.4))) THEN

                         WRITE(15,655) GENERL

                  ELSE

                         WRITE(15,645) GENERL

                  ENDIF

                              IF (III.EQ.1) THEN

                                   WRITE(15,650) YES

                              ENDIF
                    III=III+1

690               CONTINUE

      ELSE IF ((JJ.EQ.1).OR.(JJ.EQ.12)) THEN

           WRITE(15,645) GENERL
           WRITE(15,645) GENERL
           WRITE(15,650) YES

      ENDIF

                IF (JJ.EQ.1) NTW1=3
                IF (JJ.EQ.1) NTW2=12
                IF (JJ.EQ.2) NTW1=14
                IF (JJ.EQ.2) NTW2=25
                IF (JJ.EQ.3) NTW1=27
                IF (JJ.EQ.3) NTW2=36
                IF (JJ.EQ.4) NTW1=38
                IF (JJ.EQ.4) NTW2=39
                IF (JJ.EQ.5) NTW1=41
                IF (JJ.EQ.5) NTW2=50
                IF (JJ.EQ.6) NTW1=52
                IF (JJ.EQ.6) NTW2=53
                IF (JJ.EQ.7) NTW1=55
                IF (JJ.EQ.7) NTW2=64
                IF (JJ.EQ.8) NTW1=66
                IF (JJ.EQ.8) NTW2=67
                IF (JJ.EQ.9) NTW1=69
                IF (JJ.EQ.9) NTW2=78
                IF (JJ.EQ.10) NTW1=80
                IF (JJ.EQ.10) NTW2=90
                   IF ((CCC.EQ.0.0).AND.(JJ.EQ.10)) NTW2=NTW2+24
                IF (JJ.EQ.11) NTW1=91
                IF (JJ.EQ.12) NTW1=93
                IF (JJ.EQ.12) NTW2=102
                IF (JJ.EQ.13) NTW1=104
                IF (JJ.EQ.13) NTW2=114+36

695      CONTINUE
```

```
      WRITE(15,30) SLASH

C     ************************************************************
C     *                                                          *
C     *          BEGIN NODE AND ELEMENT GENERATION               *
C     *               OF THE SPARK PLUG PORT                     *
C     *                                                          *
C     ************************************************************

      WRITE(15,30) SLASH
      WRITE(15,35) GENERL
      WRITE(15,40) GENERL
      WRITE(15,30) SLASH
      WRITE(15,420) GENERL
      WRITE(15,425) GENERL
      WRITE(15,430) GENERL
      WRITE(15,48) GENERL
      WRITE(15,50) GENERL
      WRITE(15,30) SLASH

      WRITE(15,635) GENERL
      WRITE(15,58)

      WRITE(15,675) GENERL,NTKMRC+NTKMEX+NTKMIN+1
     #                    ,NTKMRC+NTKMEX+NTKMIN+NTKMSP

            DO 700 JJ=1,NTKMSP

                  WRITE(15,645) GENERL

                        IF (JJ.EQ.1) THEN

                              WRITE(15,650) YES

                        ENDIF

               IF ((CCC.GT.0.0).AND.((IJJ.EQ.1).OR.(IJJ.EQ.4))) THEN

                  WRITE(15,655) GENERL

               ELSE

                  WRITE(15,645) GENERL

               ENDIF

                        IF (JJ.EQ.1) THEN

                              WRITE(15,650) YES

                        ENDIF

 700        CONTINUE

      WRITE(15,30) SLASH

C     ************************************************************
C     *                                                          *
C     *          BEGIN NODE AND ELEMENT GENERATION               *
C     *        TRIANGULAR WEDGES IN THE INTAKE PORT              *
C     *                                                          *
C     ************************************************************

C     ************************************************************
```

```
C     *                                                              *
C     *         SET THE ELEMENT TYPE FOR PARABOLIC WEDGES            *
C     *                                                              *
C     ****************************************************************

            WRITE(15,30) SLASH
            WRITE(15,35) GENERL
            WRITE(15,40) GENERL
            WRITE(15,420) GENERL
            WRITE(15,425) GENERL
            WRITE(15,430) GENERL

            WRITE(15,705) GENERL
  705       FORMAT(1X,A5,'TK')

            WRITE(15,710) GENERL
  710       FORMAT(1X,A5,'PB')

            WRITE(15,30) SLASH

                 DO 720 JJ=1,ISTOP2

                      WRITE(15,635) GENERL

                      WRITE(15,58)

                      WRITE(15,715) GENERL,NTKMRC+NTKMEX+NTW
  715                 FORMAT(1X,A5,I3)

                           WRITE(15,645) GENERL

                           WRITE(15,650) YES

                   IF ((CCC.GT.0.0).AND.((IJJ.EQ.1).OR.(IJJ.EQ.4))) THEN

                           WRITE(15,655) GENERL

                   ELSE

                           WRITE(15,645) GENERL

                   ENDIF

                           WRITE(15,650) YES

                                IF (JJ.EQ.1) NTW=13
                                IF (JJ.EQ.2) NTW=26
                                IF (JJ.EQ.3) NTW=37
                                IF (JJ.EQ.4) NTW=40
                                IF (JJ.EQ.5) NTW=51
                                IF (JJ.EQ.6) NTW=54
                                IF (JJ.EQ.7) NTW=65
                                IF (JJ.EQ.8) NTW=68
                                IF (JJ.EQ.9) NTW=79
                                IF (JJ.EQ.10) NTW=92
                                IF (JJ.EQ.11) NTW=103

  720            CONTINUE

            WRITE(15,30) SLASH

C     ****************************************************************
C     *                                                              *
C     *            BEGIN NODE AND ELEMENT GENERATION                 *
```

```
C     *                     FOR THE RIBS                              *
C     *                                                               *
C     *****************************************************************

C     *****************************************************************
C     *                                                               *
C     *          SET THE ELEMENT TYPE FOR SOLID ELEMENTS              *
C     *                                                               *
C     *****************************************************************

            WRITE(15,30) SLASH
            WRITE(15,35) GENERL
            WRITE(15,40) GENERL
            WRITE(15,420) GENERL
            WRITE(15,425) GENERL
            WRITE(15,430) GENERL

            WRITE(15,725) GENERL
 725         FORMAT(1X,A5,'SOL')

            WRITE(15,50) GENERL
            WRITE(15,30) SLASH
            WRITE(15,420) GENERL

C     *****************************************************************
C     *                                                               *
C     *         SET THE PHYSICAL PROPERTY FOR THIN SHELLS             *
C     *                                                               *
C     *****************************************************************

            WRITE(15,800) GENERL
 800        FORMAT(1X,A5,'PHY')

            WRITE(15,805) GENERL
 805        FORMAT(1X,A5,'EN')

            WRITE(15,810) GENERL
 810        FORMAT(1X,A5,'3')

            WRITE(15,815) GENERL
 815        FORMAT(1X,A5)

            WRITE(15,815) GENERL
            WRITE(15,815) GENERL
            WRITE(15,815) GENERL

            WRITE(15,30) SLASH
            WRITE(15,635) GENERL
            WRITE(15,58)

            WRITE(15,675) GENERL,NTKMRC+NTKMEX+NTKMIN+NTKMSP+1
     #                        ,NTKMRC+NTKMEX+NTKMIN+NTKMSP+NSLMRB

                 DO 820 JJ=1,NSLMRB

                           WRITE(15,810) GENERL


                           WRITE(15,645) GENERL

                               IF (JJ.EQ.1) THEN

                                  WRITE(15,650) YES
```

```
                                 ENDIF

 820            CONTINUE

          WRITE(15,30) SLASH

C    ************************************************************
C    *                                                          *
C    *            BEGIN NODE AND ELEMENT GENERATION             *
C    *               FOR THE STIFFENED CHANNELS                 *
C    *                                                          *
C    ************************************************************

C    ************************************************************
C    *                                                          *
C    *          SET THE ELEMENT TYPE FOR THIN SHELLS            *
C    *                                                          *
C    ************************************************************

          WRITE(15,30) SLASH
          WRITE(15,35) GENERL
          WRITE(15,40) GENERL
          WRITE(15,420) GENERL
          WRITE(15,425) GENERL
          WRITE(15,430) GENERL

          WRITE(15,825) GENERL
 825      FORMAT(1X,A5,'TN')

          WRITE(15,830) GENERL
 830      FORMAT(1X,A5,'PQ')

          WRITE(15,30) SLASH
          WRITE(15,420) GENERL

C    ************************************************************
C    *                                                          *
C    *        SET THE PHYSICAL PROPERTY FOR THIN SHELLS         *
C    *                                                          *
C    ************************************************************

          WRITE(15,835) GENERL
 835      FORMAT(1X,A5,'PHY')

          WRITE(15,840) GENERL
 840      FORMAT(1X,A5,'EN')

          WRITE(15,845) GENERL
 845      FORMAT(1X,A5,'2')

          WRITE(15,850) GENERL
 850      FORMAT(1X,A5,'0.25,0,0,0')

          WRITE(15,855) GENERL
 855      FORMAT(1X,A5)

          WRITE(15,855) GENERL
          WRITE(15,855) GENERL
          WRITE(15,855) GENERL
          WRITE(15,855) GENERL
          WRITE(15,855) GENERL
          WRITE(15,855) GENERL

          WRITE(15,30) SLASH
```

```
      WRITE(15,635) GENERL
      WRITE(15,58)

      WRITE(15,675) GENERL,NTKMRC+NTKMEX+NTKMIN+NTKMSP+NSLMRB+1
     #              ,NTKMRC+NTKMEX+NTKMIN+NTKMSP+NSLMRB+NTNMSC

           DO 865 JJ=1,NTNMSC

                    WRITE(15,860) GENERL
860                 FORMAT(1X,A5,'2')

                    WRITE(15,645) GENERL

                         IF (JJ.EQ.1) THEN

                              WRITE(15,650) YES

                         ENDIF

865        CONTINUE

      WRITE(15,30) SLASH

      PRINT*,  ' '
      PRINT*,  '  THE UNIVERSAL AND PROGRAM FILES HAVE BEEN CREATED.'
      PRINT*,  ' '
      PRINT*,  '   BEFORE THE UNIVERSAL FILE CAN BE READ AND THE'
      PRINT*,  '   PROGRAM FILE EXECUTED, THE USER MUST ENTER THE'
      PRINT*,  '                SDRC SOFTWARE.'

     PRINT*,  ' '
     PRINT*,  '    NOTE THAT A FILE CALLED "MODEL DATA" CAN NOT'
     PRINT*,  '   EXIST ON YOUR DISK.  IF IT DOES, AN ERROR WILL'
     PRINT*,  '    RESULT WHEN YOU ENTER THE SDRC SOFTWARE.  TO'
     PRINT*,  '     CORRECT THE ERROR, CHANGE THE NAME OF THE'
     PRINT*,  '    "MODEL DATA" FILE THAT EXISTS ON YOUR DISK'
     PRINT*,  '              TO A DIFFERENT NAME.'
     PRINT*,  ' '
     PRINT*,  ' AFTER ENTERING THE SDRC SOFTWARE IN THE PROGRAM MODE,'
     PRINT*,  '       RESPOND "R" (FOR RUN) TO THE FIRST QUESTION,'
     PRINT*,  '         RESPOND "GO" TO THE SECOND QUESTION AND'
     PRINT*,  '           IN RESPONSE TO THE THIRD QUESTION,'
     PRINT*,  '       ENTER THE TERMINAL TYPE THAT YOU ARE USING.'

C    ************************************************************
C    *                                                          *
C    *              END NODE AND ELEMENT GENERATION             *
C    *                                                          *
C    ************************************************************

     RETURN
     END

C    ************************************************************
C    *                                                          *
C    *   THIS PROGRAM GENERATES A PROGRAM FILE THAT, IN         *
C    *   TURN CALLS A PROGRAM FILE.  THE PROGRAM FILE           *
C    *   FILE GENERATED IS CALLED RUN.  THE RUN PROGRAM FILE    *
C    *   READS TEH UNIVERSAL FILE - GEOMETRY- AND RUNS THE      *
C    *               PROGRAM FILE - MSNDEL.                     *
C    *                                                          *
C    ************************************************************
```

```
      SUBROUTINE GO (ICONTI)                                           GEN76170
                                                                       GEN76180
      CHARACTER*10 GENERL,YES                                          GEN76190
                                                                       GEN76200
           GENERL='10 -1'                                              GEN76210
           YES='3 -1YES'                                               GEN76220
                                                                       GEN76230
           WRITE(22,25)                                                GEN76240
   25      FORMAT(2X,'4 -1%L')                                         GEN76250
                                                                       GEN76260
           WRITE(22,30)                                                GEN76270
   30      FORMAT(2X,'5 -1MODEL')                                      GEN76280
                                                                       GEN76290
           WRITE(22,35) YES                                            GEN76300
   35      FORMAT(2X,A8)                                               GEN76310
                                                                       GEN76320
            WRITE(22,40)                                               GEN76330
   40       FORMAT(1X,'65 -1FINITE ELEMENT MODEL OF A ROTARY COMBUSTION GEN76340
     #ENGINE CENTER HOUSING')                                          GEN76350
                                                                       GEN76360
           WRITE(22,45)                                                GEN76370
   45      FORMAT(2X,'2 -1IN')                                         GEN76380
                                                                       GEN76390
           WRITE(22,50)                                                GEN76400
   50      FORMAT(2X,'3 -1FEM')                                        GEN76410
                                                                       GEN76420
           WRITE(22,55)                                                GEN76430
   55      FORMAT(2X,'2 -1FT')                                         GEN76440
                                                                       GEN76450
           WRITE(22,60)                                                GEN76460
   60      FORMAT(2X,'2 -1RF')                                         GEN76470
                                                                       GEN76480
           WRITE(22,65)                                                GEN76490
   65      FORMAT(2X,'2 -1UN')                                         GEN76500
                                                                       GEN76510
           WRITE(22,70)                                                GEN76520
   70      FORMAT(2X,'8 -1GEOMETRY')                                   GEN76530
                                                                       GEN76540
           WRITE(22,71)                                                GEN76550
   71      FORMAT(2X,'2 -1AU')                                         GEN76560
                                                                       GEN76570
           WRITE(22,72)                                                GEN76580
   72      FORMAT(1X,'12 -1/ DO E V OFF')                              GEN76590
                                                                       GEN76600
           WRITE(22,73)                                                GEN76610
   73      FORMAT(2X,'6 -1SU OFF')                                     GEN76620
                                                                       GEN76630
           WRITE(22,74)                                                GEN76640
   74      FORMAT(2X,'2 -1DR')                                         GEN76650
                                                                       GEN76660
      IF (ICONTI.EQ.2) GO TO 110                                       GEN76670
                                                                       GEN76680
           WRITE(22,75)                                                GEN76690
   75      FORMAT(2X,'2 -1NM')                                         GEN76700
                                                                       GEN76710
           WRITE(22,80)                                                GEN76720
   80      FORMAT(2X,'2 -1MC')                                         GEN76730
                                                                       GEN76740
           WRITE(22,85)                                                GEN76750
   85      FORMAT(2X,'1 -1T')                                          GEN76760
                                                                       GEN76770
           WRITE(22,90)                                                GEN76780
   90      FORMAT(2X,'2 -1MA')                                         GEN76790
                                                                       GEN76800
```

```
          WRITE(22,95)
 95       FORMAT(2X,'2 -1MO')

          WRITE(22,100)
 100      FORMAT(2X,'2 -1PR')

          WRITE(22,105)
 105      FORMAT(2X,'8 -1R MSNDEL')

 110  RETURN
      END

C    ***************************************************************
C    *                                                             *
C    *        THIS SUBROUTINE SOLVES FOR ALPHA GIVEN PHI           *
C    *                                                             *
C    ***************************************************************

        SUBROUTINE PALPSL(EE,RR,PI,PHI,TRANS,ALPHA,NO)

      FA(GAMMA)=TAN(PHI)*(EE*COS(3.0*GAMMA)+RR*COS(GAMMA))-
     #          (EE*SIN(3.0*GAMMA)+RR*SIN(GAMMA))+TRANS

        PHI=PHI*180.0/PI
             DO 5 II=1,11,2
                  TEST=PHI*II
                       IF(TEST.EQ.90.0*II) THEN

                  PHI=PHI*PI/180.0
                  ALPHA=PHI+0.10*PHI
                  GO TO 25

         ENDIF

C    ***************************************************************
C    *                                                             *
C    *               INCREMENTAL SEARCH METHOD                     *
C    *                                                             *
C    ***************************************************************

  5          CONTINUE

        EPSI=0.000001
         PHI=PHI*PI/180.0

      IT=0
      IL=0
      AAVG1=5.0

      DELTA=0.3*PI/180.0
         ALPHA=PHI

      AONE=FA(ALPHA)

 10   ALPHA1=ALPHA+DELTA
      ATWO=FA(ALPHA1)

         IF(AONE*ATWO.EQ.0.0) THEN

        ALPHA=ATWO
        GO TO 25
```

```
           ELSE IF(ATWO*AONE.GT.0.0) THEN

      IF(ATWO.GT.25.0) THEN
      WRITE(NO,*)'ALPHA IS GOING TO INFINITY'
      STOP

       ELSE
         IT=IT+1

         IF((ATWO.GT.0.0).AND.(AONE.GT.0.0)) THEN

            IF(AONE-ATWO.LT.0.0) THEN
             IL=IL+1
              IF(IL.GT.1) GO TO 15

                   DELTA=-DELTA

            ENDIF
       ELSE IF((ATWO.LT.0.0).AND.(AONE.LT.0.0)) THEN

            IF(AONE-ATWO.GT.0.0) THEN
             IL=IL+1
              IF(IL.GT.1) GO TO 15

                   DELTA=-DELTA

            ENDIF

         ENDIF
 15     AONE=ATWO
        ALPHA=ALPHA1
        GO TO 10

      ENDIF

           ELSE IF(ATWO*AONE.LT.0.0) THEN

C   *************************************************************
C   *                                                           *
C   *                   BISECTION METHOD                        *
C   *                                                           *
C   *************************************************************

  20       AAVG=(ALPHA+ALPHA1)/2.0
           ATHREE=FA(AAVG)

                IF(ATHREE.GT.10.0) THEN
                     STOP

                ENDIF

      IF(ABS(AAVG1-AAVG).GT.EPSI) THEN

           IF(AONE*ATHREE.GT.0.0) THEN

                AAVG1=AAVG
                ALPHA=AAVG
                AONE=ATHREE
                GO TO 20

           ELSE IF(AONE*ATHREE.LT.0.0) THEN

                AAVG1=AAVG
                ALPHA1=AAVG
```

```
                ATWO=ATHREE
                GO TO 20
           ENDIF

      ELSE IF(ABS(AAVG1-AAVG).LT.EPSI) THEN

         ALPHA=AAVG
      ENDIF
      ENDIF

 25   RETURN
      END

C   **************************************************************
C   *                                                            *
C   *    THIS SUBROUTINE SOLVES FOR ALPHA GIVEN A Y-COORDINATE   *
C   *                                                            *
C   **************************************************************

           SUBROUTINE YALPSL(EE,RR,Y,ICHK,ALPHA,NO)

      FA(GAMMA)=EE*SIN(3.0*GAMMA)+RR*SIN(GAMMA)-Y

      IT=0
      IL=0
      AAVG1=5.0
      EPSI=0.0001

      DELTA=1.0*3.141592654/180.0

         IF(ICHK.EQ.0) THEN
              ALPHA=0.1

         ELSE IF(ICHK.EQ.1) THEN

              ALPHA=2.75
         ENDIF


      AONE=FA(ALPHA)

C   **************************************************************
C   *                                                            *
C   *              INCREMENTAL SEARCH METHOD                     *
C   *                                                            *
C   **************************************************************

 5    ALPHA1=ALPHA+DELTA
      ATWO=FA(ALPHA1)

          IF(AONE*ATWO.EQ.0.0) THEN

         ALPHA=ATWO
         GO TO 20


           ELSE IF(ATWO*AONE.GT.0.0) THEN

      IF(ATWO.GT.25.0) THEN
      WRITE(NO,*)'ALPHA IS GOING TO INFINITY'
      STOP

       ELSE
         IT=IT+1
```

```fortran

         IF((ATWO.GT.0.0).AND.(AONE.GT.0.0)) THEN

            IF(AONE-ATWO.LT.0.0) THEN
             IL=IL+1
              IF(IL.GT.1) GO TO 10

                  DELTA=-DELTA
            ENDIF
        ELSE IF((ATWO.LT.0.0).AND.(AONE.LT.0.0)) THEN

            IF(AONE-ATWO.GT.0.0) THEN
             IL=IL+1
              IF(IL.GT.1) GO TO 10

                  DELTA=-DELTA
            ENDIF

         ENDIF
 10     AONE=ATWO
        ALPHA=ALPHA1
        GO TO 5

      ENDIF

           ELSE IF(ATWO*AONE.LT.0.0) THEN

C   ***************************************************************
C   *                                                             *
C   *                   BISECTION METHOD                          *
C   *                                                             *
C   ***************************************************************

  15       AAVG=(ALPHA+ALPHA1)/2.0
           ATHREE=FA(AAVG)

                IF(ATHREE.GT.10.0) THEN
                     STOP

                ENDIF

      IF(ABS(AAVG1-AAVG).GT.EPSI) THEN

           IF(AONE*ATHREE.GT.0.0) THEN

                AAVG1=AAVG
                ALPHA=AAVG
                AONE=ATHREE
                GO TO 15

           ELSE IF(AONE*ATHREE.LT.0.0) THEN

                AAVG1=AAVG
                ALPHA1=AAVG
                ATWO=ATHREE
                GO TO 15
           ENDIF

      ELSE IF(ABS(AAVG1-AAVG).LT.EPSI) THEN

         ALPHA=AAVG
      ENDIF
      ENDIF

```

```
 20   RETURN
      END

C    *************************************************************
C    *                                                           *
C    *       THIS SUBROUTINE SOLVES FOR THETA GIVEN ALPHA        *
C    *                                                           *
C    *************************************************************

          SUBROUTINE THTASL (EE,RR,PI,ALPHA,THETA,NO)

           FA(GAMMA)=(RR+3.0*EE*COS(2.0*ALPHA))/(SQRT
     #     (9.0*EE**2+RR**2+6.0*EE*RR*COS(2.0*ALPHA)))-COS(GAMMA)


      IT=0
      IL=0
      TAVG1=5.0
      EPSI=0.0001

      DELTA=1.0*PI/180.0
         THETA=0.25

      TONE=FA(THETA)

C    *************************************************************
C    *                                                           *
C    *               INCREMENTAL SEARCH METHOD                   *
C    *                                                           *
C    *************************************************************

  5   THETA1=THETA+DELTA
      TTWO=FA(THETA1)

          IF(TONE*TTWO.EQ.0.0) THEN

         THETA=TTWO
         GO TO 20


          ELSE IF(TTWO*TONE.GT.0.0) THEN

      IF(TTWO.GT.25.0) THEN
      WRITE(NO,*)'THETA IS GOING TO INFINITY'
      STOP

       ELSE
         IT=IT+1

         IF((TTWO.GT.0.0).AND.(TONE.GT.0.0)) THEN

            IF(TONE-TTWO.LT.0.0) THEN
             IL=IL+1
              IF(IL.GT.1) GO TO 10

                   DELTA=-DELTA
            ENDIF
       ELSE IF((TTWO.LT.0.0).AND.(TONE.LT.0.0)) THEN

            IF(TONE-TTWO.GT.0.0) THEN
             IL=IL+1
              IF(IL.GT.1) GO TO 10

                   DELTA=-DELTA
```

```
              ENDIF

           ENDIF
  10       TONE=TTWO
           THETA=THETA1
           GO TO 5

        ENDIF

             ELSE IF(TTWO*TONE.LT.0.0) THEN

C    *************************************************************
C    *                                                           *
C    *                   BISECTION METHOD                        *
C    *                                                           *
C    *************************************************************

  15         TAVG=(THETA+THETA1)/2.0
             TTHREE=FA(TAVG)

                 IF(TTHREE.GT.10.0) THEN
                     STOP

                 ENDIF

        IF(ABS(TAVG1-TAVG).GT.EPSI) THEN

             IF(TONE*TTHREE.GT.0.0) THEN

                 TAVG1=TAVG
                 THETA=TAVG
                 TONE=TTHREE
                 GO TO 15

             ELSE IF(TONE*TTHREE.LT.0.0) THEN

                 TAVG1=TAVG
                 THETA1=TAVG
                 TTWO=TTHREE
                 GO TO 15
             ENDIF

        ELSE IF(ABS(TAVG1-TAVG).LT.EPSI) THEN

           THETA=TAVG
        ENDIF
        ENDIF

 20     IF(((ALPHA.GT.0.0).AND.(ALPHA.LT.PI/2.0)).OR.((ALPHA.GT.PI).AND.
     #    (ALPHA.LT.3.0*PI/2.0)))  THETA=-THETA

        RETURN
        END

C    *************************************************************
C    *                                                           *
C    *    THIS SUBROUTINE CALCULATES THE POINT COORDINATES       *
C    *                OF THE EXHAUST PORT                        *
C    *                                                           *
C    *************************************************************

        SUBROUTINE EXHST (EE,RR,R,PI,REXPT,TEXPT,AA,BB,CC,D,DEPTH,
     #    TRANS,PHI1EP,PHI2EP,PHI1OD,PHI2OD,PHI1NW,
     #    GIX,GIY,GGIX,GGIY,GGGIX,GGGIY)
```

```
      COMMON / PORT / X1(7),Y1(7),Z1(7),X2(7),Y2(7),Z2(7),X3(7),Y3(7),
     #   Z3(7),X4(7),Y4(7),Z4(7),X5(7),Y5(7),Z5(7),X6(7),Y6(7),Z6(7),
     #   X7(7),Y7(7),Z7(7),X8(7),Y8(7),Z8(7),X9(7),Y9(7),Z9(7),
     #   X10(7),Y10(7),Z10(7),X11(7),Y11(7),Z11(7),X12(7),Y12(7),Z12(7),
     #   X13(7),Y13(7),Z13(7),X14(7),Y14(7),Z14(7),X15(7),Y15(7),Z15(7),
     #   X16(7),Y16(7),Z16(7),X17(7),Y17(7),Z17(7),X18(7),Y18(7),Z18(7),
     #   X19(7),Y19(7),Z19(7),X20(7),Y20(7),Z20(7),X21(7),Y21(7),Z21(7),
     #   X22(7),Y22(7),Z22(7),X23(7),Y23(7),Z23(7),X24(7),Y24(7),Z24(7),
     #   X25(7),Y25(7),Z25(7),X26(7),Y26(7),Z26(7),X27(7),Y27(7),Z27(7),
     #   X28(7),Y28(7),Z28(7),X29(7),Y29(7),Z29(7),X30(7),Y30(7),Z30(7),
     #   X31(7),Y31(7),Z31(7),X32(7),Y32(7),Z32(7),X33(7),Y33(7),Z33(7),
     #   X34(7),Y34(7),Z34(7),X35(7),Y35(7),Z35(7),X36(7),Y36(7),Z36(7),
     #   X37(7),Y37(7),Z37(7),X38(7),Y38(7),Z38(7),X39(7),Y39(7),Z39(7),
     #   X40(7),Y40(7),Z40(7),X41(7),Y41(7),Z41(7),X42(7),Y42(7),Z42(7),
     #   X43(7),Y43(7),Z43(7),X44(7),Y44(7),Z44(7),X45(7),Y45(7),Z45(7),
     #   X46(7),Y46(7),Z46(7),X47(7),Y47(7),Z47(7),X48(7),Y48(7),Z48(7),
     #   X49(7),Y49(7),Z49(7),X50(7),Y50(7),Z50(7),X51(7),Y51(7),Z51(7),
     #   X52(7),Y52(7),Z52(7),X53(7),Y53(7),Z53(7)


          XXX(Z,BETA)=Z*COS(BETA)

          YYY(ZZ,BETA)=ZZ*SIN(BETA)

          FNX(GAMMA)=EE*COS(3.0*GAMMA)+RR*COS(GAMMA)

          FNY(GAMMA)=EE*SIN(3.0*GAMMA)+RR*SIN(GAMMA)

          FINERX(GAMMA,RLAMB,FF)=EE*COS(3.0*GAMMA)+RR*COS(GAMMA)+
     #              FF*COS(GAMMA+RLAMB)

          FINERY(GAMMA,RLAMB,FF)=EE*SIN(3.0*GAMMA)+RR*SIN(GAMMA)+
     #              FF*SIN(GAMMA+RLAMB)

      R=RR-EE+D-TRANS+AA
      TRANS=-TRANS
      PHI=(PHI1EP+PHI2EP)/2.0
      DROP=R*SIN(PHI)
      YCNTR=TRANS-DROP
      YYONE=R*SIN(PHI1EP)
      YYTWO=R*SIN(PHI2EP)
      YYTRE=(YYONE-YYTWO)/2.0

      DO 15 JJ=1,7

            Y1(JJ)=YCNTR-REXPT
            Z1(JJ)=DEPTH/2.0
            X1(JJ)=SQRT(R**2-(Y1(JJ)-TRANS)**2)

            Y2(JJ)=YCNTR-REXPT*SIN(60.0*PI/180.0)
            Z2(JJ)=DEPTH/2.0+REXPT*COS(60.0*PI/180.0)
            X2(JJ)=SQRT(R**2-(Y2(JJ)-TRANS)**2)

            Y3(JJ)=YCNTR-REXPT*SIN(30.0*PI/180.0)
            Z3(JJ)=DEPTH/2.0+REXPT*COS(30.0*PI/180.0)
            X3(JJ)=SQRT(R**2-(Y3(JJ)-TRANS)**2)

            Y4(JJ)=YCNTR
            Z4(JJ)=DEPTH/2.0+REXPT
            X4(JJ)=SQRT(R**2-(Y4(JJ)-TRANS)**2)

            Y5(JJ)=YCNTR+REXPT*SIN(30.0*PI/180.0)
            Z5(JJ)=DEPTH/2.0+REXPT*COS(30.0*PI/180.0)
```

```
      X5(JJ)=SQRT(R**2-(Y5(JJ)-TRANS)**2)

      Y6(JJ)=YCNTR+REXPT*SIN(60.0*PI/180.0)
      Z6(JJ)=DEPTH/2.0+REXPT*COS(60.0*PI/180.0)
      X6(JJ)=SQRT(R**2-(Y6(JJ)-TRANS)**2)

      Y7(JJ)=YCNTR+REXPT
      Z7(JJ)=DEPTH/2.0
      X7(JJ)=SQRT(R**2-(Y7(JJ)-TRANS)**2)

      Y8(JJ)=YCNTR+REXPT*SIN(60.0*PI/180.0)
      Z8(JJ)=DEPTH/2.0-REXPT*COS(60.0*PI/180.0)
      X8(JJ)=SQRT(R**2-(Y8(JJ)-TRANS)**2)

      Y9(JJ)=YCNTR+REXPT*SIN(30.0*PI/180.0)
      Z9(JJ)=DEPTH/2.0-REXPT*COS(30.0*PI/180.0)
      X9(JJ)=SQRT(R**2-(Y9(JJ)-TRANS)**2)

      Y10(JJ)=YCNTR
      Z10(JJ)=DEPTH/2.0-REXPT
      X10(JJ)=SQRT(R**2-(Y10(JJ)-TRANS)**2)

      Y11(JJ)=YCNTR-REXPT*SIN(30.0*PI/180.0)
      Z11(JJ)=DEPTH/2.0-REXPT*COS(30.0*PI/180.0)
      X11(JJ)=SQRT(R**2-(Y11(JJ)-TRANS)**2)

      Y12(JJ)=YCNTR-REXPT*SIN(60.0*PI/180.0)
      Z12(JJ)=DEPTH/2.0-REXPT*COS(60.0*PI/180.0)
      X12(JJ)=SQRT(R**2-(Y12(JJ)-TRANS)**2)


      Y13(JJ)=YCNTR-REXPT-TEXPT
      X13(JJ)=SQRT(R**2-(Y13(JJ)-TRANS)**2)
      Z13(JJ)=3.0*DEPTH/4.0

            IF(JJ.EQ.5) THEN

                  Y13(JJ)=GIY
                  X13(JJ)=GIX
            ELSE IF (JJ.EQ.6) THEN

                  Y13(JJ)=GGIY
                  X13(JJ)=GGIX
            ELSE IF (JJ.EQ.7) THEN

                  Y13(JJ)=GGGIY
                  X13(JJ)=GGGIX

            ENDIF

      X14(JJ)=X13(JJ)
      Y14(JJ)=Y13(JJ)
      Z14(JJ)=DEPTH/2.0

      X15(JJ)=X13(JJ)
      Y15(JJ)=Y13(JJ)
      Z15(JJ)=DEPTH/4.0

      Y16(JJ)=Y3(JJ)-TEXPT*SIN(30.0*PI/180.0)
      Z16(JJ)=Z3(JJ)+TEXPT*COS(30.0*PI/180.0)
      X16(JJ)=SQRT(R**2-(Y16(JJ)-TRANS)**2)

      Y17(JJ)=YCNTR
      Z17(JJ)=Z4(JJ)+TEXPT
```

```
      X17(JJ)=SQRT(R**2-(Y17(JJ)-TRANS)**2)

      Y18(JJ)=Y5(JJ)+TEXPT*SIN(30.0*PI/180.0)
      Z18(JJ)=Z5(JJ)+TEXPT*COS(30.0*PI/180.0)
      X18(JJ)=SQRT(R**2-(Y18(JJ)-TRANS)**2)

      X19(JJ)=XXX(R,PHI1NW)
      Y19(JJ)=-YYY(R,PHI1NW)+TRANS
      Z19(JJ)=DEPTH/4.0

      X20(JJ)=X19(JJ)
      Y20(JJ)=Y19(JJ)
      Z20(JJ)=DEPTH/2.0

      X21(JJ)=X19(JJ)
      Y21(JJ)=Y19(JJ)
      Z21(JJ)=3.0*DEPTH/4.0

      Y22(JJ)=Y9(JJ)+TEXPT*SIN(30.0*PI/180.0)
      Z22(JJ)=Z9(JJ)-TEXPT*COS(30.0*PI/180.0)
      X22(JJ)=SQRT(R**2-(Y22(JJ)-TRANS)**2)

      Y23(JJ)=YCNTR
      Z23(JJ)=Z10(JJ)-TEXPT
      X23(JJ)=SQRT(R**2-(Y23(JJ)-TRANS)**2)

      Y24(JJ)=Y11(JJ)-TEXPT*SIN(30.0*PI/180.0)
      Z24(JJ)=Z11(JJ)-TEXPT*COS(30.0*PI/180.0)
      X24(JJ)=SQRT(R**2-(Y24(JJ)-TRANS)**2)

      Z25(JJ)=0.0
      Z26(JJ)=0.0
      Z27(JJ)=0.0
      Z28(JJ)=0.0
      Z31(JJ)=DEPTH
      Z32(JJ)=DEPTH
      Z33(JJ)=DEPTH
      Z34(JJ)=DEPTH

        IF ((JJ.EQ.1).OR.(JJ.EQ.2).OR.(JJ.EQ.5).OR.(JJ.EQ.6).
     #      OR.(JJ.EQ.7)) THEN

      Y26(JJ)=Y16(JJ)
      X26(JJ)=X16(JJ)

      Y27(JJ)=Y17(JJ)
      X27(JJ)=X17(JJ)

      Y28(JJ)=Y18(JJ)
      X28(JJ)=X18(JJ)

      Y31(JJ)=Y22(JJ)
      X31(JJ)=X22(JJ)

      Y32(JJ)=Y23(JJ)
      X32(JJ)=X23(JJ)

      X33(JJ)=X24(JJ)
      Y33(JJ)=Y24(JJ)

      X34(JJ)=X13(JJ)
      Y34(JJ)=Y13(JJ)

      X25(JJ)=X13(JJ)
```

```
      Y25(JJ)=Y13(JJ)

        ENDIF

      X29(JJ)=X19(JJ)
      Y29(JJ)=Y19(JJ)
      Z29(JJ)=0.0

      X30(JJ)=X29(JJ)
      Y30(JJ)=Y29(JJ)
      Z30(JJ)=DEPTH

      Z35(JJ)=DEPTH
      Z36(JJ)=3.0*DEPTH/4.0
      Z37(JJ)=DEPTH/2.0
      Z38(JJ)=DEPTH/4.0
      Z39(JJ)=0.0
      Z40(JJ)=DEPTH
      Z41(JJ)=3.0*DEPTH/4.0
      Z42(JJ)=DEPTH/2.0
      Z43(JJ)=DEPTH/4.0
      Z44(JJ)=0.0
      Z45(JJ)=DEPTH
      Z46(JJ)=3.0*DEPTH/4.0
      Z47(JJ)=DEPTH/2.0
      Z48(JJ)=DEPTH/4.0
      Z49(JJ)=0.0

        IF((JJ.EQ.1).OR.(JJ.EQ.2)) THEN

      PHI11=ATAN(ABS(Y34(1)-TRANS)/X34(1))
      PHI12=PHI2OD-PHI11
      PHI13=PHI11+2.0*PHI12/5.0
      PHI14=PHI11+3.0*PHI12/4.0

      X35(JJ)=XXX(R,PHI13)
      Y35(JJ)=-YYY(R,PHI13)+TRANS

      X40(JJ)=XXX(R,PHI14)
      Y40(JJ)=-YYY(R,PHI14)+TRANS

      X45(JJ)=XXX(R,PHI2OD)
      Y45(JJ)=-YYY(R,PHI2OD)+TRANS

      X36(JJ)=X35(JJ)
      Y36(JJ)=Y35(JJ)
      X37(JJ)=X35(JJ)
      Y37(JJ)=Y35(JJ)
      X38(JJ)=X35(JJ)
      Y38(JJ)=Y35(JJ)
      X39(JJ)=X35(JJ)
      Y39(JJ)=Y35(JJ)

      X41(JJ)=X40(JJ)
      Y41(JJ)=Y40(JJ)
      X42(JJ)=X40(JJ)
      Y42(JJ)=Y40(JJ)
      X43(JJ)=X40(JJ)
      Y43(JJ)=Y40(JJ)
      X44(JJ)=X40(JJ)
      Y44(JJ)=Y40(JJ)

      X46(JJ)=X45(JJ)
      Y46(JJ)=Y45(JJ)
```

```
            X47(JJ)=X45(JJ)
            Y47(JJ)=Y45(JJ)
            X48(JJ)=X45(JJ)
            Y48(JJ)=Y45(JJ)
            X49(JJ)=X45(JJ)
            Y49(JJ)=Y45(JJ)

          ENDIF

          IF(JJ.EQ.1) R=R-AA
          IF(JJ.EQ.2) R=R-D/3.0
          IF(JJ.EQ.3) R=R-D/3.0
          IF(JJ.EQ.4) R=R-D/3.0
          IF(JJ.EQ.5) R=R+BB
          IF(JJ.EQ.6) R=R-BB+CC

 15   CONTINUE


      RETURN
      END

C     ****************************************************************
C     *                                                              *
C     *    THIS SUBROUTINE CALCULATES THE POINT COORDINATES          *
C     *                 OF THE INTAKE PORT                           *
C     *                                                              *
C     ****************************************************************

      SUBROUTINE INTKE (EE,RR,R,PI,RINPT,TINPT,AA,BB,CC,D,DEPTH,
     #   TRANS,PHI1IP,PHI2IP,PHI2OD,PHI1NW,LINPT,WINPT)


      COMMON / PORT / X1(7),Y1(7),Z1(7),X2(7),Y2(7),Z2(7),X3(7),Y3(7),
     #   Z3(7),X4(7),Y4(7),Z4(7),X5(7),Y5(7),Z5(7),X6(7),Y6(7),Z6(7),
     #   X7(7),Y7(7),Z7(7),X8(7),Y8(7),Z8(7),X9(7),Y9(7),Z9(7),
     #   X10(7),Y10(7),Z10(7),X11(7),Y11(7),Z11(7),X12(7),Y12(7),Z12(7),
     #   X13(7),Y13(7),Z13(7),X14(7),Y14(7),Z14(7),X15(7),Y15(7),Z15(7),
     #   X16(7),Y16(7),Z16(7),X17(7),Y17(7),Z17(7),X18(7),Y18(7),Z18(7),
     #   X19(7),Y19(7),Z19(7),X20(7),Y20(7),Z20(7),X21(7),Y21(7),Z21(7),
     #   X22(7),Y22(7),Z22(7),X23(7),Y23(7),Z23(7),X24(7),Y24(7),Z24(7),
     #   X25(7),Y25(7),Z25(7),X26(7),Y26(7),Z26(7),X27(7),Y27(7),Z27(7),
     #   X28(7),Y28(7),Z28(7),X29(7),Y29(7),Z29(7),X30(7),Y30(7),Z30(7),
     #   X31(7),Y31(7),Z31(7),X32(7),Y32(7),Z32(7),X33(7),Y33(7),Z33(7),
     #   X34(7),Y34(7),Z34(7),X35(7),Y35(7),Z35(7),X36(7),Y36(7),Z36(7),
     #   X37(7),Y37(7),Z37(7),X38(7),Y38(7),Z38(7),X39(7),Y39(7),Z39(7),
     #   X40(7),Y40(7),Z40(7),X41(7),Y41(7),Z41(7),X42(7),Y42(7),Z42(7),
     #   X43(7),Y43(7),Z43(7),X44(7),Y44(7),Z44(7),X45(7),Y45(7),Z45(7),
     #   X46(7),Y46(7),Z46(7),X47(7),Y47(7),Z47(7),X48(7),Y48(7),Z48(7),
     #   X49(7),Y49(7),Z49(7),X50(7),Y50(7),Z50(7),X51(7),Y51(7),Z51(7),
     #   X52(7),Y52(7),Z52(7),X53(7),Y53(7),Z53(7)


        REAL LINPT

        XXX(Z,BETA)=Z*COS(BETA)

        YYY(ZZ,BETA)=ZZ*SIN(BETA)

      TRANS=ABS(TRANS)
      R=RR-EE+D+AA-TRANS
      PHI=(PHI1IP+PHI2IP)/2.0
      PHIZ=PHI*180.0/PI
      DROP=R*SIN(PHI)
```

```
      YCNTR=TRANS+DROP
      YYONE=R*SIN(PHI1IP)
      YYTWO=R*SIN(PHI2IP)
      YYTRE=YYTWO-YYONE

      DO 15 JJ=1,7

        IF ((JJ.EQ.1).OR.(JJ.EQ.2)) THEN

           Y1(JJ)=YCNTR-RINPT
           Z1(JJ)=DEPTH/2.0
           X1(JJ)=SQRT(R**2-(Y1(JJ)-TRANS)**2)

           Y2(JJ)=YCNTR-RINPT*SIN(60.0*PI/180.0)
           Z2(JJ)=DEPTH/2.0+RINPT*COS(60.0*PI/180.0)
           X2(JJ)=SQRT(R**2-(Y2(JJ)-TRANS)**2)

           Y3(JJ)=YCNTR-RINPT*SIN(30.0*PI/180.0)
           Z3(JJ)=DEPTH/2.0+RINPT*COS(30.0*PI/180.0)
           X3(JJ)=SQRT(R**2-(Y3(JJ)-TRANS)**2)

           Y4(JJ)=YCNTR
           Z4(JJ)=DEPTH/2.0+RINPT
           X4(JJ)=SQRT(R**2-(Y4(JJ)-TRANS)**2)

           Y5(JJ)=YCNTR+RINPT*SIN(30.0*PI/180.0)
           Z5(JJ)=DEPTH/2.0+RINPT*COS(30.0*PI/180.0)
           X5(JJ)=SQRT(R**2-(Y5(JJ)-TRANS)**2)

           Y6(JJ)=YCNTR+RINPT*SIN(60.0*PI/180.0)
           Z6(JJ)=DEPTH/2.0+RINPT*COS(60.0*PI/180.0)
           X6(JJ)=SQRT(R**2-(Y6(JJ)-TRANS)**2)

           Y7(JJ)=YCNTR+RINPT
           Z7(JJ)=DEPTH/2.0
           X7(JJ)=SQRT(R**2-(Y7(JJ)-TRANS)**2)

           Y8(JJ)=YCNTR+RINPT*SIN(60.0*PI/180.0)
           Z8(JJ)=DEPTH/2.0-RINPT*COS(60.0*PI/180.0)
           X8(JJ)=SQRT(R**2-(Y8(JJ)-TRANS)**2)

           Y9(JJ)=YCNTR+RINPT*SIN(30.0*PI/180.0)
           Z9(JJ)=DEPTH/2.0-RINPT*COS(30.0*PI/180.0)
           X9(JJ)=SQRT(R**2-(Y9(JJ)-TRANS)**2)

           Y10(JJ)=YCNTR
           Z10(JJ)=DEPTH/2.0-RINPT
           X10(JJ)=SQRT(R**2-(Y10(JJ)-TRANS)**2)

           Y11(JJ)=YCNTR-RINPT*SIN(30.0*PI/180.0)
           Z11(JJ)=DEPTH/2.0-RINPT*COS(30.0*PI/180.0)
           X11(JJ)=SQRT(R**2-(Y11(JJ)-TRANS)**2)

           Y12(JJ)=YCNTR-RINPT*SIN(60.0*PI/180.0)
           Z12(JJ)=DEPTH/2.0-RINPT*COS(60.0*PI/180.0)
           X12(JJ)=SQRT(R**2-(Y12(JJ)-TRANS)**2)

       ELSE IF (JJ.GT.2) THEN

          Y1(JJ)=YCNTR-LINPT/2.0
          Z1(JJ)=DEPTH/2.0
          X1(JJ)=SQRT(R**2-(Y1(JJ)-TRANS)**2)

          Y2(JJ)=YCNTR-LINPT/2.0
```

```
            Z2(JJ)=DEPTH/2.0+WINPT/2.0
            X2(JJ)=SQRT(R**2-(Y2(JJ)-TRANS)**2)

            Y3(JJ)=YCNTR-LINPT/4.0
            Z3(JJ)=DEPTH/2.0+WINPT/2.0
            X3(JJ)=SQRT(R**2-(Y3(JJ)-TRANS)**2)

            Y4(JJ)=YCNTR
            Z4(JJ)=DEPTH/2.0+WINPT/2.0
            X4(JJ)=SQRT(R**2-(Y4(JJ)-TRANS)**2)

            Y5(JJ)=YCNTR+LINPT/4.0
            Z5(JJ)=DEPTH/2.0+WINPT/2.0
            X5(JJ)=SQRT(R**2-(Y5(JJ)-TRANS)**2)

            Y6(JJ)=YCNTR+LINPT/2.0
            Z6(JJ)=DEPTH/2.0+WINPT/2.0
            X6(JJ)=SQRT(R**2-(Y6(JJ)-TRANS)**2)

            Y7(JJ)=YCNTR+LINPT/2.0
            Z7(JJ)=DEPTH/2.0
            X7(JJ)=SQRT(R**2-(Y7(JJ)-TRANS)**2)

            Y8(JJ)=YCNTR+LINPT/2.0
            Z8(JJ)=DEPTH/2.0-WINPT/2.0
            X8(JJ)=SQRT(R**2-(Y8(JJ)-TRANS)**2)

            Y9(JJ)=YCNTR+LINPT/4.0
            Z9(JJ)=DEPTH/2.0-WINPT/2.0
            X9(JJ)=SQRT(R**2-(Y9(JJ)-TRANS)**2)

            Y10(JJ)=YCNTR
            Z10(JJ)=DEPTH/2.0-WINPT/2.0
            X10(JJ)=SQRT(R**2-(Y10(JJ)-TRANS)**2)

            Y11(JJ)=YCNTR-LINPT/4.0
            Z11(JJ)=DEPTH/2.0-WINPT/2.0
            X11(JJ)=SQRT(R**2-(Y11(JJ)-TRANS)**2)

            Y12(JJ)=YCNTR-LINPT/2.0
            Z12(JJ)=DEPTH/2.0-WINPT/2.0
            X12(JJ)=SQRT(R**2-(Y12(JJ)-TRANS)**2)

         ENDIF

            X13(JJ)=XXX(R,PHI1IP)
            Y13(JJ)=YYY(R,PHI1IP)+TRANS
            Z13(JJ)=3.0*DEPTH/4.0

            X14(JJ)=X13(JJ)
            Y14(JJ)=Y13(JJ)
            Z14(JJ)=DEPTH/2.0

            X15(JJ)=X13(JJ)
            Y15(JJ)=Y13(JJ)
            Z15(JJ)=DEPTH/4.0

         IF ((JJ.EQ.1).OR.(JJ.EQ.2)) THEN

            Y17(JJ)=Y3(JJ)-TINPT*SIN(30.0*PI/180.0)
            Z17(JJ)=Z3(JJ)+TINPT*COS(30.0*PI/180.0)
            X17(JJ)=SQRT(R**2-(Y17(JJ)-TRANS)**2)

            Y18(JJ)=YCNTR
```

```
            Z18(JJ)=Z4(JJ)+TINPT
            X18(JJ)=SQRT(R**2-(Y18(JJ)-TRANS)**2)

            Y19(JJ)=Y5(JJ)+TINPT*SIN(30.0*PI/180.0)
            Z19(JJ)=Z5(JJ)+TINPT*COS(30.0*PI/180.0)
            X19(JJ)=SQRT(R**2-(Y19(JJ)-TRANS)**2)

         ELSE IF (JJ.GT.2) THEN

            Y17(JJ)=Y3(JJ)
            Z17(JJ)=Z3(JJ)+TINPT
            X17(JJ)=X3(JJ)

            Y18(JJ)=YCNTR
            Z18(JJ)=Z4(JJ)+TINPT
            X18(JJ)=X4(JJ)

            Y19(JJ)=Y5(JJ)
            Z19(JJ)=Z5(JJ)+TINPT
            X19(JJ)=X5(JJ)

         ENDIF

            X16(JJ)=X2(JJ)
            Y16(JJ)=Y2(JJ)
            Z16(JJ)=Z17(JJ)

            Y20(JJ)=YCNTR+RINPT+TINPT
            X20(JJ)=SQRT(R**2-(Y20(JJ)-TRANS)**2)
            Z20(JJ)=Z19(JJ)

            X21(JJ)=X20(JJ)
            Y21(JJ)=Y20(JJ)
            Z21(JJ)=DEPTH/2.0

         IF ((JJ.EQ.1).OR.(JJ.EQ.2)) THEN

            Y23(JJ)=Y9(JJ)+TINPT*SIN(30.0*PI/180.0)
            Z23(JJ)=Z9(JJ)-TINPT*COS(30.0*PI/180.0)
            X23(JJ)=SQRT(R**2-(Y23(JJ)-TRANS)**2)

            Y24(JJ)=YCNTR
            Z24(JJ)=Z10(JJ)-TINPT
            X24(JJ)=SQRT(R**2-(Y24(JJ)-TRANS)**2)

            Y25(JJ)=Y11(JJ)-TINPT*SIN(30.0*PI/180.0)
            Z25(JJ)=Z11(JJ)-TINPT*COS(30.0*PI/180.0)
            X25(JJ)=SQRT(R**2-(Y25(JJ)-TRANS)**2)

         ELSE IF (JJ.GT.2) THEN

            Y23(JJ)=Y9(JJ)
            Z23(JJ)=Z9(JJ)-TINPT
            X23(JJ)=X9(JJ)

            Y24(JJ)=YCNTR
            Z24(JJ)=Z10(JJ)-TINPT
            X24(JJ)=X10(JJ)

            Y25(JJ)=Y11(JJ)
            Z25(JJ)=Z11(JJ)-TINPT
            X25(JJ)=X11(JJ)

         ENDIF
```

```
            X22(JJ)=X20(JJ)
            Y22(JJ)=Y20(JJ)
            Z22(JJ)=Z23(JJ)

            X26(JJ)=X12(JJ)
            Y26(JJ)=Y12(JJ)
            Z26(JJ)=Z25(JJ)

              IF ((JJ.EQ.1).OR.(JJ.EQ.2).OR.(JJ.EQ.5).OR.(JJ.EQ.6).
     #            OR.(JJ.EQ.7)) THEN

                 X27(JJ)=X13(JJ)
                 Y27(JJ)=Y13(JJ)
                 Z27(JJ)=0.0

                 X28(JJ)=X16(JJ)
                 Y28(JJ)=Y16(JJ)
                 Z28(JJ)=0.0

                 Y29(JJ)=Y17(JJ)
                 X29(JJ)=X17(JJ)
                 Z29(JJ)=0.0

                 Y30(JJ)=Y18(JJ)
                 X30(JJ)=X18(JJ)
                 Z30(JJ)=0.0

                 Y31(JJ)=Y19(JJ)
                 X31(JJ)=X19(JJ)
                 Z31(JJ)=0.0

                 X32(JJ)=X20(JJ)
                 Y32(JJ)=Y20(JJ)
                 Z32(JJ)=0.0

                 X33(JJ)=X22(JJ)
                 Y33(JJ)=Y22(JJ)
                 Z33(JJ)=DEPTH

                 Y34(JJ)=Y23(JJ)
                 X34(JJ)=X23(JJ)
                 Z34(JJ)=DEPTH

                 Y35(JJ)=Y24(JJ)
                 X35(JJ)=X24(JJ)
                 Z35(JJ)=DEPTH

                 X36(JJ)=X25(JJ)
                 Y36(JJ)=Y25(JJ)
                 Z36(JJ)=DEPTH

                 X37(JJ)=X26(JJ)
                 Y37(JJ)=Y26(JJ)
                 Z37(JJ)=DEPTH

                 X38(JJ)=X13(JJ)
                 Y38(JJ)=Y13(JJ)
                 Z38(JJ)=DEPTH

             ENDIF

                 IF((JJ.NE.3).OR.(JJ.NE.4)) THEN
```

```
                X49(JJ)=XXX(R,PHI1NW)
                Y49(JJ)=YYY(R,PHI1NW)+TRANS
                Z49(JJ)=DEPTH

      IF (JJ.EQ.1) THEN

          PHI12=PHI1NW-PHI2IP
          PHI13=PHI2IP+1.0*PHI12/3.0
          PHI14=PHI2IP+2.0*PHI12/3.0

      ELSE IF (JJ.GT.1) THEN

          PHI12=PHI1NW-ATAN((Y21(JJ)-TRANS)/X21(JJ))
          PHI13=PHI1NW-2.0*PHI12/3.0
          PHI14=PHI1NW-1.0*PHI12/3.0

      ENDIF

            X39(JJ)=XXX(R,PHI13)
            Y39(JJ)=YYY(R,PHI13)+TRANS
            Z39(JJ)=DEPTH

             X40(JJ)=X39(JJ)
             Y40(JJ)=Y39(JJ)
             Z40(JJ)=3.0*DEPTH/4.0

              X41(JJ)=X39(JJ)
              Y41(JJ)=Y39(JJ)
              Z41(JJ)=DEPTH/2.0

               X42(JJ)=X39(JJ)
               Y42(JJ)=Y39(JJ)
               Z42(JJ)=DEPTH/4.0

                X43(JJ)=X39(JJ)
                Y43(JJ)=Y39(JJ)
                Z43(JJ)=0.0

                 X44(JJ)=XXX(R,PHI14)
                 Y44(JJ)=YYY(R,PHI14)+TRANS
                 Z44(JJ)=DEPTH

                  X45(JJ)=X44(JJ)
                  Y45(JJ)=Y44(JJ)
                  Z45(JJ)=3.0*DEPTH/4.0

                   X46(JJ)=X44(JJ)
                   Y46(JJ)=Y44(JJ)
                   Z46(JJ)=DEPTH/2.0

                  X47(JJ)=X44(JJ)
                  Y47(JJ)=Y44(JJ)
                  Z47(JJ)=DEPTH/4.0

                 X48(JJ)=X44(JJ)
                 Y48(JJ)=Y44(JJ)
                 Z48(JJ)=0.0

               X50(JJ)=X49(JJ)
               Y50(JJ)=Y49(JJ)
               Z50(JJ)=3.0*DEPTH/4.0

              X51(JJ)=X49(JJ)
              Y51(JJ)=Y49(JJ)
```

```
                       Z51(JJ)=DEPTH/2.0

                      X52(JJ)=X49(JJ)
                      Y52(JJ)=Y49(JJ)
                      Z52(JJ)=DEPTH/4.0

                     X53(JJ)=X49(JJ)
                     Y53(JJ)=Y49(JJ)
                     Z53(JJ)=0.0

                  ENDIF

              IF(JJ.EQ.1) R=R-AA
              IF(JJ.EQ.2) R=R-D/3.0
              IF(JJ.EQ.3) R=R-D/3.0
              IF(JJ.EQ.4) R=R-D/3.0
              IF(JJ.EQ.5) R=R+BB
              IF(JJ.EQ.6) R=R-BB+CC

 15   CONTINUE

      RETURN
      END
C     ***************************************************************
C     *                                                             *
C     *   THIS SUBROUTINE CALCULATES THE POINT COORDINATES          *
C     *             OF THE SPARK PLUG PORT                          *
C     *                                                             *
C     ***************************************************************

      SUBROUTINE SPRKPG (EE,RR,R,PI,YONE,YTWO,PHIONE,PHITWO,RSP,IEND,
     #   AA,BBB,CC,D,DEPTH,TRANS,REGION,ICHK)

      COMMON / PORT / X1(7),Y1(7),Z1(7),X2(7),Y2(7),Z2(7),X3(7),Y3(7),
     #   Z3(7),X4(7),Y4(7),Z4(7),X5(7),Y5(7),Z5(7),X6(7),Y6(7),Z6(7),
     #   X7(7),Y7(7),Z7(7),X8(7),Y8(7),Z8(7),X9(7),Y9(7),Z9(7),
     #   X10(7),Y10(7),Z10(7),X11(7),Y11(7),Z11(7),X12(7),Y12(7),Z12(7),
     #   X13(7),Y13(7),Z13(7),X14(7),Y14(7),Z14(7),X15(7),Y15(7),Z15(7),
     #   X16(7),Y16(7),Z16(7),X17(7),Y17(7),Z17(7),X18(7),Y18(7),Z18(7),
     #   X19(7),Y19(7),Z19(7),X20(7),Y20(7),Z20(7),X21(7),Y21(7),Z21(7),
     #   X22(7),Y22(7),Z22(7),X23(7),Y23(7),Z23(7),X24(7),Y24(7),Z24(7),
     #   X25(7),Y25(7),Z25(7),X26(7),Y26(7),Z26(7),X27(7),Y27(7),Z27(7),
     #   X28(7),Y28(7),Z28(7),X29(7),Y29(7),Z29(7),X30(7),Y30(7),Z30(7),
     #   X31(7),Y31(7),Z31(7),X32(7),Y32(7),Z32(7),X33(7),Y33(7),Z33(7),
     #   X34(7),Y34(7),Z34(7),X35(7),Y35(7),Z35(7),X36(7),Y36(7),Z36(7),
     #   X37(7),Y37(7),Z37(7),X38(7),Y38(7),Z38(7),X39(7),Y39(7),Z39(7),
     #   X40(7),Y40(7),Z40(7),X41(7),Y41(7),Z41(7),X42(7),Y42(7),Z42(7),
     #   X43(7),Y43(7),Z43(7),X44(7),Y44(7),Z44(7),X45(7),Y45(7),Z45(7),
     #   X46(7),Y46(7),Z46(7),X47(7),Y47(7),Z47(7),X48(7),Y48(7),Z48(7),
     #   X49(7),Y49(7),Z49(7),X50(7),Y50(7),Z50(7),X51(7),Y51(7),Z51(7),
     #   X52(7),Y52(7),Z52(7),X53(7),Y53(7),Z53(7)

      INTEGER REGION

         FINERX(GAMMA,RLAMB,FF)=EE*COS(3.0*GAMMA)+RR*COS(GAMMA)+
     #             FF*COS(GAMMA+RLAMB)

         FINERY(GAMMA,RLAMB,FF)=EE*SIN(3.0*GAMMA)+RR*SIN(GAMMA)+
     #             FF*SIN(GAMMA+RLAMB)

         XXX(Z,BETA)=Z*COS(BETA)

         YYY(ZZ,BETA)=ZZ*SIN(BETA)
```

```
         BB=BBB
         TRANS=ABS(TRANS)
         R=RR-EE+D+AA-TRANS

          IF (REGION.EQ.3) THEN

               PHI1=PI-PHIONE
               PHI2=PI-PHITWO
               YONE=ABS(R*SIN(PHI1))
               YTWO=ABS(R*SIN(PHI2))

                 WIDTH=YONE+YTWO
                 TCKSP=(-DEPTH/4.0-RSP)/2.0

                    YONE=YONE+TRANS
                    YTWO=TRANS-YTWO
                    YCNTR=(YONE+YTWO)/2.0

          ELSE IF (REGION.EQ.2) THEN

               WIDTH=ABS(YTWO-YONE)
               TCKSP=(-DEPTH/4.0-RSP)/2.0

                    YCNTR=(YONE+YTWO)/2.0

          ENDIF

       DO 15 JJ=1,IEND

         IF (JJ.LT.6) THEN

           X1(JJ)=-R
           Y1(JJ)=YCNTR-RSP

            X2(JJ)=-R
            Y2(JJ)=YCNTR-RSP*COS(PI/4.0)

             X3(JJ)=-R
             Y3(JJ)=YCNTR

              X4(JJ)=-R
              Y4(JJ)=YCNTR+RSP*COS(PI/4.0)

               X5(JJ)=-R
               Y5(JJ)=YCNTR+RSP

                X6(JJ)=-R
                Y6(JJ)=YCNTR+RSP*COS(PI/4.0)

                 X7(JJ)=-R
                 Y7(JJ)=YCNTR

                  X8(JJ)=-R
                  Y8(JJ)=YCNTR-RSP*COS(PI/4.0)

                 X9(JJ)=-R
                 Y9(JJ)=Y8(JJ)-TCKSP*COS(PI/4.0)

                X10(JJ)=-R
                Y10(JJ)=Y1(JJ)-TCKSP

               X11(JJ)=-R
               Y11(JJ)=Y2(JJ)-TCKSP*COS(PI/4.0)
```

```
          X12(JJ)=-R
          Y12(JJ)=Y3(JJ)

         X13(JJ)=-R
         Y13(JJ)=Y4(JJ)+TCKSP*COS(PI/4.0)

        IF (REGION.EQ.2) THEN

         X14(JJ)=-R
         Y14(JJ)=Y5(JJ)+TCKSP

        ENDIF

       X15(JJ)=-R
       Y15(JJ)=Y6(JJ)+TCKSP*COS(PI/4.0)

      X16(JJ)=-R
      Y16(JJ)=Y7(JJ)

       X17(JJ)=-R
       Y17(JJ)=YCNTR-WIDTH/2.0

        X18(JJ)=-R
        Y18(JJ)=Y17(JJ)

         X19(JJ)=-R
         Y19(JJ)=Y17(JJ)

          X20(JJ)=-R
          Y20(JJ)=Y12(JJ)

       IF (REGION.EQ.3) THEN

           X21(JJ)=XXX(R,PHIONE)
           Y21(JJ)=YYY(R,PHIONE)+TRANS

            X22(JJ)=XXX(R,PHIONE)
            Y22(JJ)=YYY(R,PHIONE)+TRANS

             X23(JJ)=XXX(R,PHIONE)
             Y23(JJ)=YYY(R,PHIONE)+TRANS

              X14(JJ)=-R
              Y14(JJ)=(Y5(JJ)+Y22(JJ))/2.0

       ELSE IF (REGION.EQ.2) THEN

           X21(JJ)=-R
           Y21(JJ)=YCNTR+WIDTH/2.0

            X22(JJ)=-R
            Y22(JJ)=Y21(JJ)

             X23(JJ)=-R
             Y23(JJ)=Y21(JJ)

       ENDIF

              X24(JJ)=-R
              Y24(JJ)=Y16(JJ)

      ELSE IF (JJ.GT.5) THEN
```

```
      CALL YALPSL (EE,RR,Y1(JJ-1),ICHK,ALPHA1)
      CALL THTASL (EE,RR,PI,ALPHA1,THETA1)
      CALL YALPSL (EE,RR,Y2(JJ-1),ICHK,ALPHA2)
      CALL THTASL (EE,RR,PI,ALPHA2,THETA2)
      CALL YALPSL (EE,RR,Y3(JJ-1),ICHK,ALPHA3)
      CALL THTASL (EE,RR,PI,ALPHA3,THETA3)
      CALL YALPSL (EE,RR,Y4(JJ-1),ICHK,ALPHA4)
      CALL THTASL (EE,RR,PI,ALPHA4,THETA4)

      CALL YALPSL (EE,RR,Y5(JJ-1),ICHK,ALPHA5)
      CALL THTASL (EE,RR,PI,ALPHA5,THETA5)
      CALL YALPSL (EE,RR,Y6(JJ-1),ICHK,ALPHA6)
      CALL THTASL (EE,RR,PI,ALPHA6,THETA6)
      CALL YALPSL (EE,RR,Y8(JJ-1),ICHK,ALPHA8)
      CALL THTASL (EE,RR,PI,ALPHA8,THETA8)
      CALL YALPSL (EE,RR,Y9(JJ-1),ICHK,ALPHA9)
      CALL THTASL (EE,RR,PI,ALPHA9,THETA9)

      CALL YALPSL (EE,RR,Y10(JJ-1),ICHK,ALPH10)
      CALL THTASL (EE,RR,PI,ALPH10,THET10)
      CALL YALPSL (EE,RR,Y13(JJ-1),ICHK,ALPH13)
      CALL THTASL (EE,RR,PI,ALPH13,THET13)
      CALL YALPSL (EE,RR,Y14(JJ-1),ICHK,ALPH14)
      CALL THTASL (EE,RR,PI,ALPH14,THET14)
      CALL YALPSL (EE,RR,Y17(JJ-1),ICHK,ALPH17)
      CALL THTASL (EE,RR,PI,ALPH17,THET17)
      CALL YALPSL (EE,RR,Y21(JJ-1),ICHK,ALPH21)
      CALL THTASL (EE,RR,PI,ALPH21,THET21)

           X1(JJ)=-FINERX(ALPHA1,THETA1,BBB)
           Y1(JJ)=Y1(JJ-1)

            X2(JJ)=-FINERX(ALPHA2,THETA2,BBB)
            Y2(JJ)=Y2(JJ-1)

             X3(JJ)=-FINERX(ALPHA3,THETA3,BBB)
             Y3(JJ)=Y3(JJ-1)

              X4(JJ)=-FINERX(ALPHA4,THETA4,BBB)
              Y4(JJ)=Y4(JJ-1)

               X5(JJ)=-FINERX(ALPHA5,THETA5,BBB)
               Y5(JJ)=Y5(JJ-1)

                X6(JJ)=-FINERX(ALPHA6,THETA6,BBB)
                Y6(JJ)=Y6(JJ-1)

                 X7(JJ)=-FINERX(ALPHA3,THETA3,BBB)
                 Y7(JJ)=Y7(JJ-1)

                  X8(JJ)=-FINERX(ALPHA8,THETA8,BBB)
                  Y8(JJ)=Y8(JJ-1)

                 X9(JJ)=-FINERX(ALPHA9,THETA9,BBB)
                 Y9(JJ)=Y9(JJ-1)

                X10(JJ)=-FINERX(ALPH10,THET10,BBB)
                Y10(JJ)=Y10(JJ-1)

               X11(JJ)=-FINERX(ALPHA9,THETA9,BBB)
               Y11(JJ)=Y9(JJ)

              X12(JJ)=-FINERX(ALPHA3,THETA3,BBB)
              Y12(JJ)=Y3(JJ)
```

```
                        X13(JJ)=-FINERX(ALPH13,THET13,BBB)
                        Y13(JJ)=Y13(JJ-1)

                       X14(JJ)=-FINERX(ALPH14,THET14,BBB)
                       Y14(JJ)=Y14(JJ-1)

                      X15(JJ)=-FINERX(ALPH13,THET13,BBB)
                      Y15(JJ)=Y13(JJ)

                     X16(JJ)=-FINERX(ALPHA3,THETA3,BBB)
                     Y16(JJ)=Y7(JJ)

                      X17(JJ)=-FINERX(ALPH17,THET17,BBB)
                      Y17(JJ)=Y17(JJ-1)

                       X18(JJ)=-FINERX(ALPH17,THET17,BBB)
                       Y18(JJ)=Y17(JJ)

                        X19(JJ)=-FINERX(ALPH17,THET17,BBB)
                        Y19(JJ)=Y17(JJ)

                         X20(JJ)=-FINERX(ALPHA3,THETA3,BBB)
                         Y20(JJ)=Y3(JJ)

              IF (JJ.EQ.6) DD=BB
              IF (JJ.EQ.7) DD=CC-BB

        IF (REGION.EQ.3) THEN

             X21(JJ)=-FINERX(ALPH21,THET21,BBB)
             Y21(JJ)=Y21(JJ-1)+DD*SIN(PI-PHIONE)

        ELSE IF (REGION.EQ.2) THEN

              X21(JJ)=-FINERX(ALPH21,THET21,BBB)
              Y21(JJ)=Y21(JJ-1)

        ENDIF

                          X22(JJ)=-FINERX(ALPH21,THET21,BBB)
                          Y22(JJ)=Y21(JJ)

                           X23(JJ)=-FINERX(ALPH21,THET21,BBB)
                           Y23(JJ)=Y21(JJ)

                            X24(JJ)=-FINERX(ALPHA3,THETA3,BBB)
                            Y24(JJ)=Y3(JJ)

              IF (JJ.EQ.6) BBB=CC
              IF (JJ.EQ.7) BBB=BB

        ENDIF

          X25(JJ)=X19(JJ)
          Y25(JJ)=Y19(JJ)
          X26(JJ)=X20(JJ)
          Y26(JJ)=Y20(JJ)
          X27(JJ)=X21(JJ)
          Y27(JJ)=Y21(JJ)
          X28(JJ)=X23(JJ)
          Y28(JJ)=Y23(JJ)
          X29(JJ)=X24(JJ)
          Y29(JJ)=Y24(JJ)
```

```
          X30(JJ)=X17(JJ)
          Y30(JJ)=Y17(JJ)

          Z1(JJ)=DEPTH/2.0
           Z2(JJ)=DEPTH/2.0-RSP*SIN(PI/4.0)
            Z3(JJ)=DEPTH/2.0-RSP
             Z4(JJ)=DEPTH/2.0-RSP*SIN(PI/4.0)
              Z5(JJ)=DEPTH/2.0
               Z6(JJ)=DEPTH/2.0+RSP*SIN(PI/4.0)
                Z7(JJ)=DEPTH/2.0+RSP
                 Z8(JJ)=DEPTH/2.0+RSP*SIN(PI/4.0)
                Z9(JJ)=Z8(JJ)+TCKSP*SIN(PI/4.0)
               Z10(JJ)=Z1(JJ)
              Z11(JJ)=Z2(JJ)-TCKSP*SIN(PI/4.0)
             Z12(JJ)=Z3(JJ)-TCKSP
            Z13(JJ)=Z4(JJ)-TCKSP*SIN(PI/4.0)
           Z14(JJ)=Z5(JJ)
          Z15(JJ)=Z6(JJ)+TCKSP*SIN(PI/4.0)
         Z16(JJ)=Z7(JJ)+TCKSP
          Z17(JJ)=DEPTH/4.0
           Z18(JJ)=DEPTH/2.0
            Z19(JJ)=3.0*DEPTH/4.0
             Z20(JJ)=3.0*DEPTH/4.0
              Z21(JJ)=3.0*DEPTH/4.0
               Z22(JJ)=DEPTH/2.0
                Z23(JJ)=DEPTH/4.0
                 Z24(JJ)=DEPTH/4.0
                 Z25(JJ)=DEPTH
                Z26(JJ)=DEPTH
               Z27(JJ)=DEPTH
              Z28(JJ)=0.0
             Z29(JJ)=0.0
            Z30(JJ)=0.0

               IF(JJ.EQ.1) R=R-AA
               IF(JJ.EQ.2) R=R-D/3.0
               IF(JJ.EQ.3) R=R-D/3.0
               IF(JJ.EQ.4) R=R-D/3.0
               IF(JJ.EQ.5) R=R-BB
               IF(JJ.EQ.6) R=R-CC

 15     CONTINUE

      RETURN
      END

      SUBROUTINE NEWFIL (STATUS,NO)

C    ************************************************************
C    *                                                          *
C    *   THIS SUBROUTINE CHECKS TO SEE IF THE FILE NAMED        *
C    *   "GENERATE DATA A" EXISTS ON THE USER'S DISK.  IF IT    *
C    *   DOES NOT EXIST, IT IS CREATED AND THE USER IS          *
C    *   PROMPTED FOR THE INPUTS.                               *
C    *                                                          *
C    *   IF IT DOES EXIST, THEN THE USER IS                     *
C    *   NOT PROMPTED FOR THE INPUTS.  THE INPUTS ARE READ      *
C    *   FROM THE EXISTING DATA FILE.                           *
C    *                                                          *
C    ************************************************************


      CHARACTER FILE*20,STRING*38
      INTEGER STATUS,CMSCMD
```

```
C  SET STRING TO BLANKS

      CLOSE(5)
      STRING = '                                                  '

      FILE='GENERATE DATA A'

C  CHECK STATUS OF INPUT FILENAME

      STRING='STATE '//FILE
      STATUS=CMSCMD(STRING)

      IF (STATUS.EQ.28) THEN
       WRITE(NO,*) ' '
       WRITE(NO,*)' THE "GENERATE DATA" FILE WAS NOT FOUND ON DISK.'
       WRITE(NO,*) 'THEREFORE, YOU WILL BE PROMPTED FOR THE INPUTS.'
       WRITE(NO,*) 'A "GENERATE DATA" FILE WILL BE CREATED FROM YOUR'
       WRITE(NO,*) 'RESPONSES SO THAT CHANGES IN THE FILE CAN EASILY'
       WRITE(NO,*) '           BE MADE AT A LATER TIME.'
       WRITE(NO,*) ' '
        GO TO 40
      ELSE IF (STATUS.EQ.36) THEN
        WRITE(NO,*)' DISK NOT ACCESSED'
        WRITE(NO,*)' RETURN CODE = 36'
        GO TO 50
      ELSE IF (STATUS.NE.0) THEN
15      WRITE(NO,*)' FILE STATUS ERROR'
        WRITE(NO,*)' RETURN CODE = ',STATUS
        GO TO 50
      ENDIF

      IF (STATUS.EQ.0) GO TO 50
C
C  IF FILE DOES NOT EXIST, THEN CREATE IT ON UNIT #8
C
      IF (STATUS.EQ.0) GO TO 50

40    STRING='FILEDEF 8 DISK '//FILE
      STATUS=CMSCMD(STRING)
      IF(STATUS.NE.0)THEN
       WRITE(NO,*)' FILEDEF ERROR - DEFINITION OF FILE AS UNIT 8 FAILED'
       WRITE(NO,*)' RETURN CODE = ',STATUS
      ENDIF

       STATUS=1

50    RETURN
      END

      FUNCTION CMS CMD (COMAND)
C
      INTEGER        MAX CMD
      PARAMETER     (MAX CMD = 20)
      INTEGER        MAX DBL
      PARAMETER     (MAX DBL = MAX CMD+1)
C
      INTEGER        $END IT
      REAL*8         BUFFER   (MAX DBL)
      CHARACTER*8    CMD STR  (MAX CMD)
      INTEGER        CMD SUB
      INTEGER        CMS CMD
      CHARACTER*(*)  COMAND
      INTEGER        FIRST
```

```
      INTRINSIC     INDEX
      INTRINSIC     LEN
      INTEGER       LONG
      INTEGER       LST
      INTEGER       LST 1
      INTEGER       LST 2
      INTEGER       LST 3
      INTRINSIC     MIN
C
C     get the length of the command and initialize the start
      LONG = LEN(COMAND)
C     initialize the starting position and number of parameters
      FIRST = 1
      CMD SUB = 0
C     find the next blank in the command
   10 LST 1 = INDEX(COMAND(FIRST:),' ')
      IF (LST 1 .EQ. 1)
     > THEN
C         first column was blank, move to next column
          FIRST  = FIRST + 1
          GO TO 100
        ENDIF
C     look for a left or right paren
      LST 2 = INDEX(COMAND(FIRST:),'(')
      LST 3 = INDEX(COMAND(FIRST:),')')
      LST = LONG-FIRST+2
C     set LST to the location of the first "(", ")", or " "
      IF (LST 1 .GT. 0) LST = MIN(LST,LST 1)
      IF (LST 2 .GT. 0) LST = MIN(LST,LST 2)
      IF (LST 3 .GT. 0) LST = MIN(LST,LST 3)
      LST = LST + FIRST - 2
C     pick up next parameter, if it is there
      IF (LST .GE. FIRST)
     > THEN
          CMD SUB = CMD SUB + 1
C         make sure we have room for this parameter
          IF (CMD SUB .GT. MAX CMD) GO TO 200
          CMD STR(CMD SUB)=COMAND(FIRST:LST)
        ENDIF
C     skip over trailing blank, if any
      IF (LST+1 .LE. LONG)
     > THEN
          IF (COMAND(LST+1:LST+1) .NE. ' ')
     >      THEN
              CMD SUB = CMD SUB + 1
              IF (CMD SUB .GT. MAX CMD) GO TO 200
              CMD STR(CMD SUB)=COMAND(LST+1:LST+1)
            ENDIF
        ENDIF
      FIRST = LST+2
C     loop back if there is more to process
  100 IF (FIRST .LE. LONG) GO TO 10
C     no more to process, call $ENDIT to set it up for CM$CMD
      CMSCMD = $ENDIT (CMD STR,BUFFER,CMD SUB)
      RETURN
  200 CMSCMD = -1
      RETURN
      END
      FUNCTION $ENDIT (CMD STR,BUFFER,BUF SIZ)
C
C     this function will make sure that the command is properly
C     aligned for the SVC in CM$CMD
C
      INTEGER      $END IT
```

```
      INTEGER       BUF SIZ
      REAL*8        BUFFER    (BUF SIZ)
      INTEGER       CM$CMD
      LOGICAL*1     CMD STR   (8,BUF SIZ)
      REAL*8        D TEMP
      REAL*8        EOM FLG
      INTEGER       I
      INTEGER       J
      LOGICAL*1     L TEMP    (8)
C
      DATA EOM FLG /ZFFFFFFFFFFFFFFF/
C
      EQUIVALENCE (L TEMP,D TEMP)
C
        DO 20 I = 1,BUF SIZ
            DO 10 J = 1,8
              L TEMP(J)=CMD STR(J,I)
   10       CONTINUE
          BUFFER(I) = D TEMP
   20   CONTINUE
      BUFFER(BUF SIZ+1) = EOM FLG
      $END IT = CM$CMD(BUFFER)
      RETURN
      END

C   ***************************************************************
C   *                                                             *
C   *      THIS SUBROUTINE WRITES A PROGRAM FILE THAT             *
C   *      CONTAINS THE COMMANDS THAT WILL MERGE THE              *
C   *      COINCIDENT NODES THAT EXIST IN THE MODEL.              *
C   *        THE TOLERAENCE IS SET AT 0.01 INCHES.                *
C   *                                                             *
C   ***************************************************************

      SUBROUTINE COIN (NO)

      CHARACTER *10 NOAL,SLASH,GENERL,YES

           NOAL='4 -1NOAL'
           SLASH='1 -1/'
           GENERL='10 -1'
           YES='3 -1YES'

                WRITE(15,30) SLASH
 30             FORMAT(2X,A5)

                WRITE(15,35) GENERL
 35             FORMAT(1X,A5,'T')

                WRITE(15,40) GENERL
 40             FORMAT(1X,A5,'MC')

                WRITE(15,45) GENERL
 45             FORMAT(1X,A5,'CN')

                WRITE(15,50) GENERL
 50             FORMAT(1X,A5,'ALL')

                WRITE(15,55) GENERL
 55             FORMAT(1X,A5,'0.01')

                WRITE(15,60) YES
 60             FORMAT(2X,A8)
```

```
      WRITE(15,60) YES
      WRITE(15,60) YES
      WRITE(15,60) YES
      WRITE(15,60) YES
      WRITE(15,60) YES
      WRITE(15,60) YES
      WRITE(15,60) YES
      WRITE(15,60) YES
      WRITE(15,60) YES
      WRITE(15,60) YES
      WRITE(15,60) YES
      WRITE(15,60) YES
      WRITE(15,60) YES
      WRITE(15,60) YES
      WRITE(15,60) YES
      WRITE(15,60) YES
      WRITE(15,60) YES
      WRITE(15,60) YES
      WRITE(15,60) YES
      WRITE(15,60) YES
      WRITE(15,60) YES
      WRITE(15,60) YES
      WRITE(15,60) YES
      WRITE(15,60) YES
      WRITE(15,60) YES
      WRITE(15,60) YES
      WRITE(15,60) YES
      WRITE(15,60) YES
      WRITE(15,60) YES
      WRITE(15,60) YES
      WRITE(15,60) YES
      WRITE(15,60) YES
      WRITE(15,60) YES
      WRITE(15,60) YES
      WRITE(15,60) YES
      WRITE(15,60) YES
      WRITE(15,60) YES
      WRITE(15,60) YES
      WRITE(15,60) YES
      WRITE(15,60) YES
      WRITE(15,60) YES
      WRITE(15,60) YES
      WRITE(15,60) YES

      WRITE(15,30) SLASH
      WRITE(15,35) GENERL

      WRITE(15,65) GENERL
   65 FORMAT(1X,A5,'MA')

      WRITE(15,70) GENERL
   70 FORMAT(1X,A5,'NO')

      WRITE(15,75) GENERL
   75 FORMAT(1X,A5,'DEL')

      WRITE(15,80)
   80 FORMAT(2X,'4 27 K')

      WRITE(15,85) GENERL
   85 FORMAT(1X,A5,'LABEL')

      WRITE(15,90) GENERL
   90 FORMAT(1X,A5,'1 50000 1')
```

```
                 WRITE(15,60) YES

      RETURN
      END
```