GENENG — A PROGRAM FOR CALCULATING
DESIGN AND OFF-DESIGN PERFORMANCE
FOR TURBOJET AND TURBOFAN ENGINES

by Robert W. Koenig and Laurence H. Fishbach

Lewis Research Center
Cleveland, Ohio 44135
A computer program titled GENENG is described. The program uses component performance maps to enable the user to do analytical steady-state engine cycle calculations. Through a scaling procedure, each of the component maps can be used to represent a family of maps (different design values of pressure ratios, efficiency, weight flow, etc.). Either convergent or convergent-divergent nozzles may be used. Included is a complete FORTRAN IV listing of the program. Sample results and input explanations are shown for one-spool and two-spool turbojets and two-spool separate- and mixed-flow turbofans operating at design and off-design conditions. The computer program is available from the authors.
## CONTENTS

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>SUMMARY</td>
<td>1</td>
</tr>
<tr>
<td>INTRODUCTION</td>
<td>1</td>
</tr>
<tr>
<td>THERMODYNAMIC ANALYSIS OF ENGINE TYPES</td>
<td>2</td>
</tr>
<tr>
<td>Two-Spool Afterburning Turbofan</td>
<td>3</td>
</tr>
<tr>
<td>Two-Spool Turbojet</td>
<td>6</td>
</tr>
<tr>
<td>One-Spool Turbojet</td>
<td>6</td>
</tr>
<tr>
<td>BALANCING TECHNIQUE</td>
<td>6</td>
</tr>
<tr>
<td>SUBROUTINES AND INPUTS FOR ENGINE PERFORMANCE CALCULATIONS</td>
<td>9</td>
</tr>
<tr>
<td>GENENG Subroutine Functions and Descriptions</td>
<td>9</td>
</tr>
<tr>
<td>Entering the Data</td>
<td>12</td>
</tr>
<tr>
<td>Choice of component maps - scaling laws</td>
<td>12</td>
</tr>
<tr>
<td>BLOCK DATA input</td>
<td>12</td>
</tr>
<tr>
<td>Inputs required at execution time</td>
<td>17</td>
</tr>
<tr>
<td>Inputs required for additional options to basic cycles</td>
<td>19</td>
</tr>
<tr>
<td>Means of Specifying Mode of Engine Operations</td>
<td>20</td>
</tr>
<tr>
<td>SAMPLE CALCULATIONS</td>
<td>21</td>
</tr>
<tr>
<td>Single-Spool Turbojet</td>
<td>21</td>
</tr>
<tr>
<td>Design point</td>
<td>21</td>
</tr>
<tr>
<td>Off-design</td>
<td>24</td>
</tr>
<tr>
<td>Two-Spool Turbojet</td>
<td>32</td>
</tr>
<tr>
<td>Design point</td>
<td>32</td>
</tr>
<tr>
<td>Off-design</td>
<td>33</td>
</tr>
<tr>
<td>Separate-Flow Turbofan</td>
<td>36</td>
</tr>
<tr>
<td>Design point</td>
<td>36</td>
</tr>
<tr>
<td>Off-design</td>
<td>36</td>
</tr>
<tr>
<td>Mixed-Flow Turbofan</td>
<td>38</td>
</tr>
<tr>
<td>Design point</td>
<td>38</td>
</tr>
<tr>
<td>Off-design</td>
<td>38</td>
</tr>
<tr>
<td>Design Case Without Pressure Ratio Rematch</td>
<td>40</td>
</tr>
<tr>
<td>CONCLUDING REMARKS</td>
<td>44</td>
</tr>
</tbody>
</table>
APPENDIXES

A - GENENG PROGRAM LISTING .................................. 45
B - HUFF INPUT ROUTINE ....................................... 137
C - SYMBOLS ..................................................... 141

REFERENCES ..................................................... 150
GENENG - A PROGRAM FOR CALCULATING DESIGN AND OFF-DESIGN PERFORMANCE FOR TURBOJET AND TURBOFAN ENGINES

by Robert W. Koenig and Laurence H. Fishbach

Lewis Research Center

SUMMARY

A digital computer program titled GENENG is described. The original version of the computer program is titled SMOTE (SiMulation Of Turbofan Engine) and was developed by the Turbine Engine Division of the Air Force Aero Propulsion Laboratory, Wright-Patterson Air Force Base, Ohio. SMOTE is capable of calculating only turbofan design and off-design performance using specific component performance maps. GENENG calculates steady-state design and off-design turbofan and one- and two-spool turbojet engine performance. Discussed in this report are changes to SMOTE which enable the user to do jet engine calculations of a general nature, thereby allowing the mission analyst a greater freedom in selection of engine design characteristics. Typical of these changes is an automatic redesign of fan and compressor pressure ratios when the static pressures do not balance at the mix point for mixed-flow turbofans. A convergent-divergent nozzle performance map addition is shown. Other changes that permit greater flexibility for generalized cycle studies for GENENG are described.

Included as an appendix to the report is a complete FORTRAN IV listing of GENENG. Sample results and input explanations are included for two-spool turbojet, one-spool turbojet, and two-spool turbofan engines operating at design and off-design conditions.

INTRODUCTION

For preliminary as well as in-depth studies it is often necessary to study a broad range of engines operating at both design and off-design conditions in order to find an efficient airframe/engine combination. The spectrum of flight conditions through which an engine must operate will strongly affect the optimum design parameter for that engine.

The SMOTE code (SiMulation Of Turbofan Engines), discussed in references 1 and 2,
provided a computer program having off-design-point calculation capability for either existing engines or theoretical ones - a major advance. Theoretical engines are simulated by scaling component performances from existing engines to the design conditions of the theoretical engines.

GENENG (GENerized ENGine), a computer code derived from SMOTE, was written to improve the versatility of SMOTE. Among the changes made are as follows:

1. One- and two-spool turbojets can be calculated, as well as turbofans.
2. Afterburner performance maps can be used.
3. Nozzle performance maps can be used.
4. Fan and compressor pressure ratios are automatically redesigned for mixed-flow turbofans if the static pressures at the mix point do not match.
5. Duct combustor pressure losses are calculated.
6. A new method of entering data into the program is used.

A derivative program from GENENG, called GENENG II, is reported in a companion report to this one (ref. 3). GENENG II calculates performance of two- or three-spool front- or aft-fan turbofan engines with as many as three nozzles (or airstreams).

These programs have proven to be very versatile, and minor changes to them can greatly increase the number of engine configurations that can be studied. As an example, GENENG was used to study nuclear-powered turbofan engines. This was accomplished by adding a heat-exchanger subroutine as a substitute for the combustor subroutine to simulate the use of nuclear power to raise the temperature of the air entering the first turbine.

In this report are included illustrative examples of the use of GENENG to study various one- and two-spool turbojets and separate- and mixed-flow turbofans.

GENENG is available from the authors upon request. This FORTRAN IV program can be used by computing centers having an IBM 7094 Mod 2 computer. With modifications, the program can be used on all machines that have a FORTRAN compiler.

**THERMODYNAMIC ANALYSIS OF ENGINE TYPES**

All thermodynamic properties of air and gas are calculated by considering variable specific heats and no dissociation. The air and gas property tables of reference 4 were curve fit and are used herein.

The following discussion presents the thermodynamic analysis of the engine cycles that can be studied using GENENG.
Two-Spool Afterburning Turbofan

The basic engine, a two-spool turbofan, is shown in figure 1. All other engine types are treated as variations of this basic engine. Free-stream conditions exist at station 1 and are determined by using the U.S. Standard Atmosphere Table of 1962 (ref. 5). The conditions at station 2 are determined by flight speed and inlet recovery.

GENENG compressor maps work with corrected values of airflow. At the entrance to the fan, the corrected airflow $W_{AF, c}$ is

$$W_{AF, c} = \frac{W_{AF} \sqrt{T_2/518.668}}{P_2/P_{SLS}}$$  \hspace{1cm} (1)

where $P_2$ and $P_{SLS}$ are in atmospheres and $P_{SLS}$ equals 1.0. All symbols are defined in appendix C. Some symbols are formed as the combination of other symbols; thus $WA$ is airflow, $F$ is for fan, and $c$ when following a compound symbol means corrected. Station numbers are defined on the appropriate figure.

All the fan air $W_{AF}$ is compressed by the fan giving rise to conditions at station 21. The power required to do this is

$$\text{Fan power} = W_{AF} \times (H_{21} - H_2)$$  \hspace{1cm} (2)

Some fan air may be lost to the cycle as fan bleed $B_{LF}$, which is expressed as a fraction of the fan airflow

$$B_{LF} = P_{C_{Bl, F}} \times W_{AF}$$  \hspace{1cm} (3)

The corrected airflow into the core compressor is

$$W_{AC, c} = \frac{W_{AC} \sqrt{T_{21}/518.668}}{P_{21}/1.0}$$  \hspace{1cm} (4)

The remaining air goes through the fan duct where some leakage from the core air may also enter (see eq. (11)).

$$W_{AD} = W_{AF} - B_{LF} - W_{AC} + B_{LDU}$$  \hspace{1cm} (5)

The air which may be heated by a duct burner to a temperature $T_{24}$ undergoes a pressure drop.
\[ P_{25} = P_{24} \times \left[ 1 - \left( \frac{\Delta P}{P} \right)_{DUCT} \right] \]  

(6)

The air would have been heated by the addition of fuel, which can be expressed as a fuel-air ratio so that

\[ W_{G24} = W_{A23} \times \left[ 1 + (f/a)_{23} \right] \]  

(7)

The gas is then either expanded through a nozzle (station 29) to produce thrust or is mixed with the core air as shown in figure 2 (mixed-flow turbofan). The bypass ratio of the engine is defined by

\[ \text{BYPASS} = \frac{W_{A_D}}{W_{A_C}} \]  

(8)

The air continuing into the core is compressed to conditions at station 3. The power required is

\[ \text{Compressor power} = W_{A_C} \times (H_3 - H_{21}) = W_{A3} \times (H_3 - H_{21}) \]  

(9)

Some core bleed air \( B_{IC} \) may be used for turbine cooling. Some of the air is put back into the cycle into each of the two turbines, and some is lost to the cycle as overboard bleed or leakage into the fan duct.

\[ B_{IC} = PC_{B1, C} \times W_{A3} \]  

(10)

\[ B_{IDU} = PC_{B1, DU} \times B_{IC} \]  

(11)

\[ B_{IOB} = PC_{B1, OB} \times B_{IC} \]  

(12)

\[ B_{IHP} = PC_{B1, HP} \times B_{IC} \]  

(13)

\[ B_{ILP} = PC_{B1, LP} \times B_{IC} \]  

(14)

Since \( B_{IDU} + B_{IOB} + B_{IHP} + B_{ILP} = B_{IC} \), the sum of \( PC_{B1, DU} + PC_{B1, OB} + PC_{B1, HP} + PC_{B1, LP} \) must be equal to 1.0.

The remaining air is

\[ W_{A4} = W_{A3} - B_{IC} \]  

(15)
and is heated to a turbine inlet temperature $T_4$ while undergoing a combustor pressure drop $(\Delta P/P)_{\text{COMP}}$. The fuel required to do this is expressed as a fuel-air ratio $(f/a)_4$ so that the weight of the gas entering the first (high pressure) turbine $W_{G4}$ can be expressed as

$$W_{G4} = W_{A4} \times \left[1 + (f/a)_4\right]$$

This gas is then expanded through the turbine to conditions at station 5. The enthalpy at station 5 is first calculated by making a power balance, since this turbine drives the compressor and supplies any work extracted ($H_{\text{PEXT}}$). By using equation (9),

$$W_{G4} \times (H_4 - H_5) = W_{A3} \times (H_3 - H_{21}) + H_{\text{PEXT}}$$

In addition, the physical speeds must match

$$N_{\text{HP, TURBINE}} = N_{\text{COMP}}$$

(18)

If high-pressure-turbine bleed air $B_{I1\text{HP}}$ is added back into the cycle at this point, $H_5$ must be readjusted.

$$H_5 = \frac{(B_{I1\text{HP}} \times H_3) + W_{G4} H_5}{W_{G4} + B_{I1\text{HP}}} = \frac{(B_{I1\text{HP}} \times H_3) + W_{G4} H_5}{W_{G5}}$$

(19)

Similarly,

$$W_{G5} \times (H_5 - H_{55}) = W_{A5} \times (H_{21} - H_2)$$

(20)

$$N_{\text{LP, TURBINE}} = N_{\text{FAN}}$$

(21)

$$H_{55} = \frac{(B_{I1\text{LP}} \times H_3) + W_{G5} H_{55}}{W_{G5} + B_{I1\text{LP}}} = \frac{(B_{I1\text{LP}} \times H_3) + W_{G5} H_{55}}{W_{G55}}$$

(22)

For non-mixed-flow turbofans, the gas flow at station 6, $W_{G6}$, is identical with that at station 55, $W_{G55}$. For mixed-flow turbofans, the air in the fan duct is added.

$$W_{G6} = W_{G55} + W_{A5}$$

(23)

Mixed-flow turbofans additionally require that the static pressures at station 25 and at station 55 (fig. 2) match.

$$P_{S55} = P_{S25}$$

(24)
The gas flow $WG_6$ then may be heated by an afterburner to a gas temperature $T_7$ and may undergo a pressure drop.

$$P_7 = P_6 \left[ 1 - \frac{\Delta P}{P} \right]_{AFTERBURNER}$$

(25)

And the gas flow rate would be increased by any fuel burned.

$$WG_7 = WG_5 + WFA$$

The gas is then expanded through the nozzle (station 9) to produce the remainder of the engine thrust.

Two-Spool Turbojet

The two-spool turbojet is equivalent to a two-spool turbofan with a BYPASS of zero. This engine is shown in figure 3. In calculating this type of engine, there is no fan duct and the air entering the inner compressor is the same as the air entering the inlet less any bleed.

$$WA_C = WA_F - B_1 F$$

(26)

The thermodynamic calculations proceed identically to the previous case, the two-spool turbofan case, except that any equations referring to the fan duct are eliminated.

One-Spool Turbojet

The one-spool turbojet is shown in figure 4. As can be seen, to simulate this engine the inner compressor and its driving turbine are eliminated. That is, stations 21 and 3 become identical and stations 4 and 5 become identical.

The only calculation changes required therefore are (1) to eliminate any thermodynamic equations relating to the fan duct and the inner spool of the two-spool turbofan engine and (2) to add the horsepower extracted to the power requirements of the outer turbine.

BALANCING TECHNIQUE

An off-design engine cycle calculation requires satisfying various matching con-
straints (rotational speeds, airflows, compressor and turbine work functions, and nozzle flow functions) at each specified operating condition. GENENG internally searches for compressor and turbine operating points that will satisfy the constraints. It does this by generating differential errors caused by small changes in the independent variables. The program then uses a matrix that is loaded with the differential errors to solve for the zero-error condition. This procedure is known as the Newton-Raphson iteration technique.

For the two-spool turbofan or turbojet engines a solution for a set of six simultaneous linear algebraic equations is obtained; for the one-spool turbojet a set of three simultaneous linear equations is solved. The six independent variables selected are

\[ Z_F = \frac{\text{Pressure ratio along speed line}}{\text{High pressure ratio on speed line}} - \frac{\text{Low pressure ratio on speed line}}{\text{Low pressure ratio on speed line}} \]

Percent fan speed or turbine inlet temperature

\[ Z_C = \frac{\text{Pressure ratio along speed line}}{\text{High pressure ratio on speed line}} - \frac{\text{Low pressure ratio on speed line}}{\text{Low pressure ratio on speed line}} \]

Percent compressor speed or turbine inlet temperature

Inner (high pressure) turbine flow function, \( \text{WG}_4 \sqrt{\frac{T_4}{P_4}} \)

Outer (low pressure) turbine flow function, \( \text{WG}_5 \sqrt{\frac{T_5}{P_5}} \)

ZC, PCNC, and TFFHP are not used for the one-spool turbojet.

The program initially selects new (perturbed) values for the variables, based on the design values. It is then possible to proceed through the entire engine cycle, where six (or three) errors are generated. The initial values of the six (or three) variables and six (or three) errors are base values.

As per reference 1, the partial differential equations for \( E = f(V) \) are

\[ \frac{dE_i}{dV_j} = \sum_{j=1}^{j_{\max}} \frac{\partial E_{ij}}{\partial V_j} dV_j \] (27)

for \( i \) going from 1 to \( j_{\max} \) where \( j_{\max} \) is 6 for two-spool engines or 3 for one-spool turbojets; \( E \) is an error; \( V \) is a variable; and \( \partial E_{ij} \) is the change in \( E_i \) caused by a change in \( V_j \).

The assumption of a small change in the variable results in the following approxima-
tions (B refers to a base value):

\[ \begin{align*}
    dE &= E - EB \\
    dV &= V - VB \\
    \frac{\partial E}{\partial V} &= \frac{E}{V}
\end{align*} \tag{28} \tag{29} \tag{30} \]

With these approximations and the knowledge that \( E \) should equal zero for the balanced engine, the set of partial differential equations (eq. (27)) reduces to

\[ E_i - EB_i = \sum_{j=1}^{j_{\text{max}}} \frac{\Delta E_{ij}}{\Delta V_j} dV_j = -EB_i \tag{31} \]

for \( i \) going from 1 to \( j_{\text{max}} \).

Thus the calculations made with the perturbed variables are used to compute \( \Delta E/\Delta V \), and equation (31) is solved for \( dV_j \). The variables \( V \) are then given new values from

\[ V_j = V_j B + dV_j \tag{32} \]

If the engine cycle calculations were linear functions, the engine would balance (errors within some allowable limits) with the new values of the variables. However, this is not the case, and it is usually necessary to repeat the process of changing each variable by a small amount for each pass. A change in each error because of the small change in the variable is calculated for each pass, where the new values become base values. This process occurs several times before a balance is obtained.

A subroutine (MATRIX) to determine the solution of a matrix is used to solve the set of differential equations. After each pass through the engines, a matrix array is loaded with the appropriate values; after a number of passes equal to \( 1 + j_{\text{max}} \) (base value plus perturbation on each of the independent variables), the matrix subroutine is used to solve the matrix. This solution (\( E \) within some allowable limit) yields the correct values of the independent variable and satisfies all the component matching constraints.

The most-often-used independent variable and the differential errors for four types of engines that can be run on GENENG are listed in table I.
SUBROUTINES AND INPUTS FOR ENGINE PERFORMANCE CALCULATIONS

Two forms of data are supplied to GENENG. Some data, such as all the constants and component map data, are in the form of a BLOCK DATA subprogram. The varying data are supplied at execution time by the use of input data cards.

The FORTRAN listings of GENENG are presented in appendix A. The function and description of the subroutines follows in the next section.

GENENG Subroutine Functions and Descriptions

A flow chart of the computer program with the subroutines is shown in figure 5. The functions of the subroutines are listed here and the purpose of each described.

GENENG Dummy main program to initiate the calculations and cause the input of the controlled output variables. Because of the looping between subroutines, control is never transferred back to this routine.

ENGBAL Main routine. Controls all engine balancing loops; checks tolerances and number of loops and loads matrix; calls input.

GUESS Determines initial values of independent variable (see table I) at each point.

MATRIX Solves error matrix.

PUTIN Calls input subroutine package. Controls loop on static pressures for mixed-flow turbofan.

ZERO Zeros nearly all of common and certain controls.

COINLT Determines ram recovery and performs inlet calculations.

ATMOS 1962 U. S. Standard Atmosphere Table.

RAM Calculates ram recovery defined by MIL-E-5008B specifications.

RAM2 Calculates special cases of input ram recovery as a function of flight Mach number.

COFAN Uses BLOCK DATA to perform outer-compressor (fan) calculations.

COCOMP Uses BLOCK DATA to perform inner-compressor calculations (two spools only).
COCOMB Uses BLOCK DATA to perform combustor calculations. May use either T4 or WFB as the main parameter.

COHPTB Uses BLOCK DATA to perform inner-turbine calculations (two spools only).

COLPTB Uses BLOCK DATA to perform outer-turbine calculations.

CODUCT Performs duct and duct-burning calculations for turbofans. May use either T24 or WFD as main parameters.

COMIX Performs gas mixing calculations if in mixed-flow mode. At design points it calculates areas either from an input static pressure PS55 or from an input Mach number AM55 if PS55 = 0. At off-design points it calculates static pressures and Mach numbers from the design areas. Calculates ERR (5). Rescales pressure ratios for mixed-flow turbofans to match duct and core static pressures just prior to mixing. COMIX also calculates afterburner entrance area A6 as a function of afterburner entrance Mach number AM6.

COAFBN Performs afterburning calculations. May use either T7 or WFA as the main parameters.

FRTOSD Dummy routine to transfer values from common FRONT to common SIDE.

FASTBK Dummy routine to transfer values from common SIDE to common BACK.

COMNOZ Controls the main nozzle.

ERROR Controls all printouts if an error occurs. Prints names of subroutine where error occurred and also prints the values of all variables in the main commons.

SYG Controls printing from UNIT08. Throughout the program and particularly in ENGBAL, certain messages, variables, and matrix values are written on UNIT08 as an aid in determining why an error occurred or why a point did not balance. These values are printed out if subroutine ERROR is called and IDUMP is greater than zero, or after a good point if IDUMP = 2.

PERF Calculates performance after the engine is balanced.

OUTPUT Prints output except for controlled output. Prints the main commons after the design point.

CONOUT Controls and prints the controlled output variables.
THCOMP Performs isentropic calculations for compressors.

PROCOM Calculates thermodynamic gas properties for either air or a fuel-air mixture based on JP-4 using curve fits of the tables of reference 4.

SEARCH General table lookup and interpolation routine to obtain data from the BLOCK DATA subroutines.

MAPBAC Used when calculations result in values not on the turbine maps. Changes the map value and an independent variable (PCNF, PCNC, or T4) in an attempt to rectify the situation.

CONVRG Performs nozzle calculations for a convergent nozzle.

CONDIV Performs nozzle calculations for a convergent-divergent (C-D) nozzle.

THTURB Performs isentropic calculations for turbines.

THERMO Provides thermodynamic conditions using PROCOM.

AFQUIR General quadratic interpolation routine.

PARABO Parabolic curve-fit routine.

BLKFAN Performance data for outer compressor (fan) map (BLOCK DATA).

BLKOMP Performance data for inner compressor map (BLOCK DATA; two-spool engines).

CMBDAT BLOCK DATA for combustor.

HPTDAT Performance data for inner turbine map (BLOCK DATA; two-spool engines).

LPTDAT Performance data for outer turbine map (BLOCK DATA).

ETAAB Generalized afterburner performance BLOCK DATA as a function of fuel-air ratio with correction factors for off-design afterburner entrance pressure and Mach number.

FRATIO Convergent-divergent nozzle velocity coefficient (BLOCK DATA input as a function of nozzle pressure ratio and area expansion ratio).

INPUT Package of Huff input subroutines. (The Huff Input Routine is a very versatile input mechanism further detailed in appendix B.)
Entering the Data

The Huff Input Routine, used to enter input data into the program at execution time, is discussed in appendix B. Appendix C presents the individual symbols internal to the program, station numbers, etc., from which compound names such as WAFCDs (WA + F + C + DS) are formed. Table II and appendix C present the names of the variables, the values of which are supplied on data cards.

Choice of component maps - scaling laws. - Many of the engines that are studied using GENENG are theoretical. Therefore, actual component maps for these engines will be nonexistent. The program, however, does require component maps to do off-design calculations. To alleviate this problem, GENENG uses scaling laws to change data from one component map into a new component map. Hopefully, a component map can be found which could be expected to perform in a similar manner to the actual map for the engine type being studied. In fact, several maps that the authors have obtained are identified as to the range of pressure ratio and the engine component design type for which they are valid (i.e., pressure ratio range, 4 to 8; subsonic compressor or inner compressor). Thus, for example, a high-bypass-ratio, subsonic-flight-speed, low-pressure-ratio fan map for a CF6 engine would not properly simulate a low-bypass-ratio, high-pressure-ratio, supersonic multistage fan.

The scaling equations used for the compressor maps are

\[
PR = \frac{PR_{design} - 1}{PR_{map, design} - 1} (PR_{map} - 1) + 1
\]

\[
WA = \frac{WA_{design}}{WA_{map, design}} \times WA_{map}
\]

\[
ETA = \frac{ETA_{design}}{ETA_{map, design}} \times ETA_{map}
\]

Similar equations are used for combustor and turbine map scaling. These equations are found in the appropriate subroutines. In the output are printed the correction factors used in scaling the maps. The closer these values are to 1.0 (especially pressure ratio, a primary characteristic of a given compressor map), the more reasonable are the simulated maps of the engine. Conversely, however, not being close to 1.0 does not necessarily mean that the simulation is poor since many maps have been shown to be typical over quite a large range of variables.

BLOCK DATA input. - The two compressor performance maps are entered into the
code as the BLOCK DATA subprograms BLKFAN and BLKCMP. The subprograms supplied by the authors with the code and shown in appendix A are not to be taken as realistic maps. These maps are only of an illustrative nature and are the ones used to run the sample calculations.

Using subprogram BLKFAN as an example (the first nine cards of which are printed here) and referring to a typical compressor map (fig. 6), the data are programmed as follows:

```
$IBFTC BLKFAN DECK
C THIS IS A GENERALIZED FAN MAP FOR UNREALISTIC SUPersonic ENGINE

BLOCK DATA
COMDCN / FAN/CM(15),PR(15,15),WAC(15,15),ETA(15,15),N,N,N
DATA 4,NP/10,6,3*,7,5*,10,8,5*/0
DATA CP0,3*,0.4,0.5,0.6,0.7,0.8,0.9,1.0,1.1,1.2,5*/0
DATA CP0,2*,10.6,3.6,7.5,10.8,5*/0
DATA CP0,1.0,120.0,229.800,0.76120,7
1 1.03000, 243.600, 0.75592, 1.01200, 229.800, 0.76120, 7
2 1.02800, 199.800, 0.76648, 1.03480, 166.800, 0.75592, 8
3 1.04800, 133.200, 0.72652, 1.04800, 86.400, 0.64152, 9

Card 1 reminds the reader that these maps are fictitious. Card 2 identifies subprogram as BLOCK DATA. Card 3 identifies common block FAN into which data are to be stored and dimensions the program variables. Card 4 indicates that there are 10 speed lines (N) and the number of points NF on each line - six on the lowest speed, seven on the next three lines, etc. Card 5 assigns the value of speed to each of the 10 lines (low to high). Cards 6 to 9 along speed line CN=0.3 set pressure ratio PR, corrected airflow WAC, and efficiency ETA in sets of three going from low pressure (PR = 1.0) to the surge line (PR = 1.048). Note there are six sets of three values (NP(1) = 6). The rest of the cards (appendix A) set the values for each speed line.

The combustor map is also a BLOCK DATA subprogram (CMBDAT). It is a plot of temperature rise across the combustor against efficiency for constant input pressure. Entry to the map is through temperature rise and input pressure with efficiency being output. The cards in the subprogram CMBDAT are reproduced here; a typical combustor map is shown in figure 7.

```
$IBFTC CMBDAT DECK

BLOCK DATA
COMDCN / COMB/PSI(15),DELT(15,15),ETA(15,15),N,N,N
DATA 51F4,911B,9.8232,14.735,19.640,24.558,29.470,34.381,
139.273,44.207,17.674,100.0*200.0*300.0*500.0*/
DATA CP0,15*200.0,15*300.0,15*500.0,15*600.0,15*700.0,15*800.0,
115*900.0,15*1000.0,15*1100.0,15*1200.0,15*1300.0,15*1400.0,15*1500.0,
215*1500.0,
DATA ETA/
1.600, 7.260, 7.770, 805.0, 826.0, 843.0, 855.0, 865.0, 7.870,
2.750, 928.0, 885.0, 898.0, 906.0, 912.0, 914.0, 915.0
3.860, 933.0, 911.0, 925.0, 935.0, 942.0, 947.0, 951.0, 953.0,
4.925, 972.0, 977.0, 982.0, 985.0, 990.0, 992.0, 993.0, 995.0
5.960, 966.0, 972.0, 977.0, 982.0, 985.0, 990.0, 992.0, 993.0, 995.0
6.988, 991.0, 992.0, 994.0, 995.0, 997.0, 998.0, 999.0, 1000.0
78*1.030, 7.899, 120.0, 100.0/
END
```
Card 1 identifies subprogram as BLOCK DATA. Card 2 identifies common block COMB into which data are to be stored and the dimension of each variable. Card 3 indicates that there are 15 lines of constant PSI(P3) by the value of N, and that there are 15 values of DELT(DT) and ETA(ETAB) along each line of constant PSI(P3). Cards 4 and 5 assign values to each of the P3 lines from low to high pressure. Cards 6 to 8 assign values of ΔT to each of the P3 lines starting at low ΔT. The lowest value of ΔT on each of the P3 lines is given starting with the lowest value of ΔT on the lowest value of P3. Next comes the second lowest value of ΔT on each P3, etc. Again this map is unrealistic, being used for illustrative purposes only. Cards 9 to 16 assign the value of ηB in a one-to-one correspondence with the ΔT values just assigned. The order is the same. The combustor pressure loss is input as a design value (i.e., DPCODS = 0.05) for a combustor type being considered. During off-design operation, the pressure loss is related to the design value by the following equation: DPCOM = DPCODS \times (WA3C/WA3CDS). This equation is found in the COCOMB routine.

Also entered as BLOCK DATA subprograms are the turbine maps (HPTDAT and LPTDAT). To illustrate the entering of turbine data, LPTDAT is used. A typical turbine map is shown in figure 8; the data are programmed as follows:

```
$IBFTC LPTDAT DECK
BLOCK DATA
COMMON /LTURB/TFF(15),CN(15,15),DH(15,15),ETA(15,15),N,NP(15)
DATA &NP/11,9,H15,12,9,4*0/  
DATA TFF / 88.470, 102.795, 116.835, 129.330, 141.045,  
1 145.725, 150.000, 153.345, 156.405, 159.780, 163.170, 4*0,  
DATA CN 1 1.3502, 0.5336, 0.0026, 0.0026, 0.7300,  
1 0.3502, 0.5336, 0.0026, 0.0026, 0.7300,  
2 0.0365, 0.0039, 0.7472, 0.9754, 0.0044, 0.2300,  
3 1.2166, 0.0051, 0.7140, 0.9173, 0.0056, 0.7000,  
4 1.5304, 0.0059, 0.6850, 0.7520, 0.0067, 0.6730,  
5 2.0247, 0.0062, 0.6452, 2.2827, 0.0061, 0.6200,  
6 2.4555, 0.0057, 0.6000, 2.6137, 0.0053, 0.5750,  
7 2.8166, 0.0044, 0.5310, 2.9456, 0.0053, 0.5000,  
8 3.3138, 0.0001, 0.3850/  
```

Card 1 identifies the subprogram as BLOCK DATA. Card 2 identifies the common block into which data are to be loaded and dimensions the program variables. Card 3 indicates the number of constant turbine flow function lines TFF as 11(N) and the number of points on each line from low to high TFF. Cards 4 and 5 set values of TFF from low to high. Cards 6 to 14 set values of corrected speed CN, work function DH, and efficiency ETA along TFF(1) starting from low CN(0.3682) and ending at high CN(3.3138). The rest of the cards set values along higher TFF lines.

In many cases, turbine maps for high-performance engines operate at a choked condition (constant TFF). Thus, a turbine map to be represented could possibly have no lines representing constant TFF for a significant portion of the map. For complete map representation, lines of constant TFF may be estimated on the map up to the limit load-
ing line by inputing slight changes for the values of TFF (example: one line for TFF is 62.105, the next may be input as equal to 62.108). This will eliminate computational difficulties which would arise if constant values for TFF lines were input.

Afterburner performance has been programmed in a generalized form in subroutine COAFBN. The afterburner performance map included in the program is shown in figure 9(a). The performance map shows afterburner combustion efficiency ratio as a function of fuel-air ratio. The value of afterburner combustion efficiency correction factor \( \Delta \text{ETAA} \) during off-design operation is shown against design afterburner inlet Mach number ratio (fig. 9(b)) and design afterburner inlet total-pressure ratio (fig. 9(c)). Other correction factors or performance maps could be added as desired. The afterburner efficiency, fuel-air ratio, inlet total pressure, and Mach number are generalized external to the program.

A specific afterburner performance is generalized by dividing the specific off-design values by the design values, as shown here. The symbols shown are the symbols used in the ABETTA subroutine, where the generalized and specific values are input. The generalized afterburner values are obtained as follows:

\[
\text{Efficiency (ETABRT)} = \frac{\text{ETAA}}{\text{ETAADS}}
\]

\[
\text{Fuel-air ratio (FART)} = \frac{\text{FART}}{\text{FARTDS}}
\]

\[
\text{Entrance total pressure (P6T)} = \frac{P6}{P6DS}
\]

\[
\text{Entrance Mach number (EM6T)} = \frac{AM6}{AM6DS}
\]

However, the correction factor for efficiency \( \Delta \text{ETAA} \) is not a generalized value. The value of \( \Delta \text{ETAA} \) is an efficiency degradation. These degradations are input in ABETTA as discrete points on a curve. The values of \( \Delta \text{ETAA} \) are input as functions of the following:

1. The change in efficiency as a function of EM6T is input as DELM6 (which is really \( \Delta \text{ETAA} = f(AM6) \)).
2. The change in efficiency as a function of P6T is input as DELP6 (which is really \( \Delta \text{ETAA} = f(P6) \)).

At execution time for the design point, values of afterburner combustion efficiency ETAADS, exit total temperature T7DS, and entrance Mach number AM6DS are input. Design fuel-air ratio and entrance pressure ratio are calculated from the input values.
and the other design engine characteristics.

The afterburner pressure loss is input as a design value (DPAFDS) during the design case. During afterburning the pressure loss due to combustion is calculated and the regular pressure drop is included for a total afterburner pressure loss. The equations for off-design and combustion pressure losses are contained in the COAFBN subroutines.

To achieve a reasonable accuracy in cycle calculations when using any generalized component map, the usage of the map should be limited within a certain range of the original design values and configuration changes. Therefore if, for example, an afterburner has a design task that differs significantly from an example used, a new generalized performance map should be used in order to simulate the component more accurately.

SMOTE, the original code, uses a single-point input for nozzle velocity coefficients (CVMNOZ and CVDNOZ) when calculating engine performance. GENENG, however, uses a convergent-divergent nozzle velocity coefficient which is input in map form in the FRATIO subroutine. The velocity coefficient is input as a function of nozzle total-pressure ratio (P8/P1 or P28/P1). The data for the nozzle are programmed as shown below for a performance map similar to that shown in figure 10.

```
$10FTC FRATIO DECK
SUBROJINE FRASMO (PRATIO,ARATIO,CF)
DIMENSION PR(301),AR(101),CFR(30,101),XX(31),YY(31),ZZ(2)
DATA PR/N,1.0,1.25,1.5,1.75,2.0,2.25,2.5,2.75,3.0,3.25,3.5,3.75,4.0/
1.4,2.5,3.6,4.7,5.8,6.25,7.5,8.7,9.25,11.13,15.17,19.21,23.25,7
DATA A*/1.0,1.05,1.025,1.228,1.318,1.423,1.81,9.11,9.72,1.15/
DATA CFR/1.1,1.1,1.1803/
1.971, 775, 978, 980, 981, 980, 9784, 9765, 9762, 9715, 7
2.988, 9658, 9632, 960, 957, 9538, 9506, 9465, 938, 932, 927, 918, 8
3.919, 937, 8855, 875, 867, 859, 8515, 844, .
4.956, 766, 970, 9743, 978, 9805, 982, 9805, 9786, 9762, 974, 9718, 10
5.983, 967, 9645, 962, 9595, 9564, 949, 9499, 9395, 9308, 9238, 11
6.917, 901, 892, 884, 8702, 869, 862, 12
7.956, 7555, 7636, 967, 970, 9731, 9754, 9775, 9795, 981, 980, 978, 13
8.975, 7972, 9723, 9712, 968, 9637, 9594, 9554, 9517, 9497, 9387, 14
9.926, 9177, 9082, 8948, 8931, 883, 8752, 15
1.947, 349, 9517, 9545, 9601, 9632, 9668, 9711, 9737, 9767, 9777, 15
1.978, 7772, 976, 9752, 974, 970, 9655, 9637, 9603, 9555, 950, 9405, 17
1.9317, 9232, 919, 9071, 8996, 892, 18
1.944, 9502, 9508, 947, 9430, 9411, 939, 940, 950, 966, 9605, 966, 19
1.967, 968, 9692, 970, 9703, 9698, 9696, 9684, 966, 960, 9444, 20
1.9436, 9347, 9265, 9193, 9128, 9067, 901, 21
1.948, 9485, 9487, 9486, 947, 9435, 9316, 9275, 9185, 9155, 9155, 22
1.929, 341, 9455, 9535, 9531, 9558, 9591, 9608, 9609, 9601, 957, 23
1.931, 9448, 9362, 9279, 9188, 910, 901, 8921/
```

Card 1 identifies the subroutine. Card 2 identifies the subroutine from its calling name used in the subprogram in addition to the variables that are desired to be transferred. Card 3 dimensions the program variables. Cards 4 and 5 set the value of PR(P9/P1 or P29/P1) from low to high. Card 6 sets the values of AR(A9/A8 or A29/A28) from low to high. Card 7 fills the first 30x6 array of CFR(CVMNOZ or CVDNOZ) and is followed by another data statement which fills the remaining 30x4 values. Cards 8 to 24, etc., are the first values of CFR being loaded for constant values of pressure ratio for varying
ratios going from low to high.

After the data input, the equations required to interpolate between points of the input values are included in the FRATIO subroutine.

Usually, a turbojet or a mixed-flow turbofan has a single exhaust nozzle; therefore the velocity coefficient is determined as a function of \( A_9/A_8 \) and \( P_9/P_1 \). A separate-flow turbofan is considered to have two separate nozzles; thus in addition to using \( A_9/A_8 \) and \( P_9/P_1 \), it will use the duct \( A_{29}/A_{28} \) and \( P_{29}/P_{28} \) to determine a velocity coefficient to correct each stream separately. When desired, however, a different nozzle map (i.e., convergent nozzle) or single-point inputs can be readily incorporated.

Inputs required at execution time. - Basically, what must be supplied are a list of the desired output variables, design values of any component existing in the engine (compressors, combustors, turbines, etc.), and engine operational controls. The variables that are to be output are selected by the first section of data cards. Any variables that are in one of the main commons (DESIGN, FRONT, SIDE, BACK, or DUMMYS) may be selected for output by punching, in columns 1 to 6, the name of the variable as it appears in the common. Up to 150 variables (25 lines of six variables) may be chosen for a particular run. During the output phase, the name of the variable is printed out, with its value printed immediately below the name.

Another feature of the controlled output is the ability to change the name of the variable to be output; for example, it may be desired to change a station designation to one more common to a particular programmer. In this case, the variable name would be punched in columns 1 to 6 as previously described, but, in addition, the described name would be punched in columns 13 to 18. Special symbols, such as \(/\), may be used in the new name. The last card of the controlled output must be a card with "THEEND" punched in columns 1 to 6.

Table II summarizes the design inputs for the four basic engines discussed in this report.

The following control variables should always be supplied at the design point. The value used is dependent on how the user wants the engine to operate. The symbols and their purpose are listed in subroutine PUTIN but are shown here for the reader's convenience. The superscripts (1) to (4) on the symbols have the following meanings: (1) means "automatically returned to zero after each point is calculated, must be re-input if option is again desired"; (2) means "option can be used for design or off-design", whereas the other two modes can only be used at off-design; (3) means "these input values remain as input unless changed by a new input"; (4) means "before using these options which are not equal to zero, a setup case must be run where all the components are first matched, and then the identical case may be repeated exercising these options."
IDES = 1\(^{(1)}\)  
For calculating design points.

(2)MODE = 0\(^{(3)}\)  
Specify T4.

MODE = 1\(^{(3)}\)  
Specify PCNC.

(2)MODE = 2\(^{(3)}\)  
Specify WFB.

MODE = 3\(^{(3)}\)  
Specify PCNF.

INIT = 0\(^{(3)}\)  
Initializes point.

INIT = 1\(^{(3)}\)  
Will not initialize point.

IDUMP = 0\(^{(3)}\)  
Will not dump looping write-outs.

IDUMP = 1\(^{(3)}\)  
Will dump looping write-outs if error occurs.

IDUMP = 2\(^{(3)}\)  
Will dump looping write-outs after every point.

IAMTP = 0\(^{(3)}\)  
Will use input AM and military specification ETAR.

IAMTP = 1\(^{(3)}\)  
Will use input AM and input ETAR.

IAMTP = 2\(^{(3)}\)  
Will use input T2 to determine T1 = 518.668 + T2 and standard P1. (T2 value needs to be input at every point or an error will occur whenever used.)

IAMTP = 3\(^{(3)}\)  
Will use inputs P2 and standard T1.

IAMTP = 4\(^{(3)}\)  
Will use input T2 and input P2.

IAMTP = 5\(^{(3)}\)  
Will use specific schedule of ETAR located in subroutine RAMTWO.

IGASMX = -1\(^{(3)}\)  
Separate flow, input A6.

IGASMX = 0\(^{(3)}\)  
Separate flow, A6 = A55.

IGASMX = 1\(^{(3)}\)  
Will mix fan duct and main streams, A6 = A25 + A55.

IGASMX = 2\(^{(3)}\)  
Will mix fan duct and main streams, input A6.

IDBURN = 1\(^{(1)}\)  
For duct burning (fan stream only), input T24. \(^{(4)}\)

IDBURN = 2\(^{(1)}\)  
For duct burning (fan stream only), input WFD. \(^{(4)}\)
IAFTBN = 1
For afterburning (main stream or mixed stream of fan and main stream), input T7.

IAFTBN = 2
For afterburning (main stream or mixed stream of fan and main stream), input WFA.

IDCD = 1
Fan duct nozzle will be convergent-divergent.

IMCD = 1
Main nozzle will be convergent-divergent.

NOZFLT = 1
For floating main nozzle exit area.

NOZFLT = 2
For floating fan duct nozzle exit area.

NOZFLT = 3
For floating fan duct and main nozzle exit area.

ITRYS = N
Number of passes through engine before quitting.

TOLALL = X
Tolerance which the errors must satisfy before engine is matched.

The following are other input variables for which some value, depending on the engine design, should be input at the discretion of the user:

DELFN, DELFN, DELSFC
Normally input as 1.0 unless correction is desired.

A6, AM55, AM23, AM6, HPEXT, AM, ALTP
See appendix C, Input Symbols.

PCBLF, PCBLC, PCBLDU, PCBLOB
Value for bleed out of the cycle; decimal equivalent of percent compressor flow.

PCBLHP, PCBLLP
Value of total bleed returned to turbines for cycle; fractional equivalent of flow. The sum of these variables plus PCBLDU and PCBLOB should equal 1.

Inputs required for additional options to basic cycles. - To run duct-burning (available only in fan stream duct) cases load ETAD, T24 or WFDDS, and DPDUDS. To run afterburning (mixed-flow fan or unmixed fan - available only for core and fan stream) cases, load T7DS or WFADS, ETAADS, and DPAFDS. Afterburner operation is the same as in reference 2 with the exception of a generalized afterburner performance map addition. For changing the generalized map to a specific map for a specific engine design, the preceding design values are needed at the design point.
Means of Specifying Mode of Engine Operations

Shown in the section SAMPLE CALCULATIONS (pp. 21 to 46) are the methods of specifying off-design operation points. The most common one and that used exclusively herein is to select a Mach number, altitude, and turbine inlet temperature other than design values. There are, however, several other possibilities which the user may employ. For example, changing the following controls:

- **MODE = 0** Specify a new turbine inlet temperature $T_4$.
- **MODE = 1** Specify a compressor rotational speed $PCNC$.
- **MODE = 2** Specify a fuel flow rate $WFB$.
- **MODE = 3** Specify a fan rotational speed $PCNF$.

If the engine has all its nozzles fixed, an input such as turbine inlet temperature, fuel flow, or speed will set the thrust level. But other means of changing engine operation can be accomplished by varying such nozzle thrust areas as

- **A8** Main nozzle thrust area
- **A28** Fan nozzle thrust area

For example, an off-design condition may exist where, in an attempt to satisfy continuity of mass flow (one of the component matching requirements), the fan operating point may lie outside the limits of the data map that was input for the component map. A fan nozzle thrust area change could be used to return the fan operating condition on the map such that a match would occur. This would indicate a possibility exists that variable fan nozzle would be required on this engine for operation at the desired condition. The area is changed by inputing (example: $A28 = 1.2 \times A28$). Since the design areas are not known prior to running the design point, the Huff Input Routine provides the versatility in which $A28$ is increased by 20 percent, as was shown. It would be noted that any area changed remains changed until it is recalculated by a new design case or altered by a new input. The preceding example and statements would also apply if changes were made instead to $A8$.

The nozzle exit area (such as $A9$ or $A29$) may then be fully expanded (if $A8$ and $A28$ are sonic) after component matching is achieved by using the control variables $NOZFLT = 1, 2, \text{ or } 3$. The significance of these values was explained in the previous section.
SAMPLE CALCULATIONS

Shown in this section are complete listings of sample calculations for a single-spool turbojet, a two-spool turbojet, a separate-flow turbofan, and a mixed-flow turbofan. Detailed instructions for operating the program are disclosed through means of these sample calculations plus comments in the program listing (appendix A) and the PUTIN routine (pp. 50 to 52). The input and output data associated with one complete design and off-design computer run are shown.

### Single-Spool Turbojet

**Design point.** - This case is for design-point operation of a single-spool turbojet engine and is represented by three groups of data printout: (1) card images of input data, (2) results, and (3) the common variables printed out only at the design point.

For this point, the component maps were loaded as BLOCK DATA and scaled to maps for the desired engine design. The common variables appearing in the printout are defined by the relative locations of the common blocks of DESIGN, FRONT, SIDE, and BACK. The code for the common variables shown on pages 23 and 24 is as follows:

<table>
<thead>
<tr>
<th>ZF, PCNF, ZC, PCNC, T4, MODE</th>
</tr>
</thead>
<tbody>
<tr>
<td>COMMON / DESIGN/</td>
</tr>
<tr>
<td>1PCNFGJ,PCNCGU,T4GU,DUM1,DUM2,DELF,DELFN,DELSFC,</td>
</tr>
<tr>
<td>2ZGDS,PCNCDS,PRCGS,ETACDS,WACS,PRCCF,ETACCF,WACC,</td>
</tr>
<tr>
<td>34T4DS,WFTDS,DTFTDS,W4TDS,TA3WDS,DPFDSDS,DTDUCF,ETAF,</td>
</tr>
<tr>
<td>5TFHPDS,CNHPDS,ETHPDS,THPFD,CNHPDF,ETHPCF,DHPCF,T2DS,</td>
</tr>
<tr>
<td>6TFLPDS,CNLPDS,ETLPDS,TLFPCF,CNLPDF,ETLPDF,DHLPF,T21DS,</td>
</tr>
<tr>
<td>7T240DS,WFTDDS,DTUDDS,TA23DDS,DPDDUDS,DTTDUCF,ETADCF,</td>
</tr>
<tr>
<td>8T7DS,WFTDS,DTFADS,ETAADS,WG6CDS,DPADDS,DTAFCD,ETAACF,</td>
</tr>
<tr>
<td>9A55,A25,A6,A7,A9,A28,A29,</td>
</tr>
<tr>
<td>$P55,AM55,CVDNOZ,CVNOZ,A29SAV,A29SAV,A28SAV,A29SAV</td>
</tr>
<tr>
<td>COMMON / FRONT/</td>
</tr>
<tr>
<td>1T1,P1,H1,S1,T2,P2,H2,S2,</td>
</tr>
<tr>
<td>2T21,P21,H21,S21,T3,P3,H3,S3,</td>
</tr>
<tr>
<td>3T4,P4,H4,S4,T5,P5,H5,S5,</td>
</tr>
<tr>
<td>4T55,P55,H55,S55,BLF,BLC,BLDW,BLDW,</td>
</tr>
<tr>
<td>5CNF,PRF,ETAF,WAFC,WAF,W4A,W44,FDF,</td>
</tr>
<tr>
<td>6CNC,PRC,ETAC,WACC,WAC,ETAB,DPCGM,DUMP,</td>
</tr>
<tr>
<td>7THNP,ETATHP,DHTHP,DHTHP,BLHP,GP5,FAR5,CS,</td>
</tr>
<tr>
<td>8CNLP,ETATLP,DHTCLP,DHTF,BLLP,WG55,FAR55,HPFXT,</td>
</tr>
<tr>
<td>9AM,ALTNP,ETAR,IF,PCNF,ZC,PCNC,WFB,</td>
</tr>
<tr>
<td>$F7TFHP,F7TFPL,PCBFLC,PCBLC,PCBLDU,PCBLB,PCBLHP,PCBLLP,PCBFLP,PCBLFLP</td>
</tr>
<tr>
<td>COMMON / SIDE/</td>
</tr>
<tr>
<td>1XP1,XMAF,XWAC,XBLF,XBLDU,XH3,DUM51,DUM52,</td>
</tr>
<tr>
<td>2XP21,XH21,X521,T23,P23,H23,S23,</td>
</tr>
<tr>
<td>3T24,P24,H24,S24,T25,P25,H25,S25,</td>
</tr>
<tr>
<td>4T28,P28,H28,S28,T29,P29,H29,S29,</td>
</tr>
<tr>
<td>5WAD,WFD,WG4,FAR4,ETAD,DPDFUC,BYPASS,DUM53,</td>
</tr>
<tr>
<td>6T25,P25,V28,AM28,T29,P29,V29,AM29</td>
</tr>
<tr>
<td>COMMON / BACK/</td>
</tr>
<tr>
<td>1XT55,XP55,XH55,X555,XT25,XP25,XH25,X255,</td>
</tr>
<tr>
<td>2XF56,XW565,XTAR555,XTMF5,XW6G4,XFAR24,XXP1,DUMB,</td>
</tr>
<tr>
<td>3T6,P6,H6,S6,T7,P7,H7,S7,</td>
</tr>
<tr>
<td>4T8,P8,H8,S8,T9,P9,H9,S9,</td>
</tr>
<tr>
<td>5W66,WFA,WG7,FART,ETAA,DPART,V55,V25,</td>
</tr>
<tr>
<td>6AM6,P57,T57,P57,V7,AM7,V7,</td>
</tr>
<tr>
<td>7T58,P59,S59,T59,P59,V9,AM9,</td>
</tr>
<tr>
<td>8V9,FRD,VJD,FGMD,VJM,FGMM,FGPD,FGPD,FGPD,FGPD,FGPD</td>
</tr>
<tr>
<td>9FGM,FGP,WFT,WGT,FART,FG,FGN,FGN,FGN,FGN,FGN</td>
</tr>
</tbody>
</table>

21
It should be noted that the program uses fixed effective areas calculated at the design point. The design case which calculates the area uses a convergent nozzle. Thus, as seen on the listing (pp. 23 and 24), \( A_8 = A_9 \). Before an afterburning case or before the nozzle is allowed to float (become fully expanded to a convergent-divergent nozzle), the engine components must first be matched in a setup run. Then, the next case may be run with IAFTBN or NOZFLT control cards unequal to zero and may be used as shown in the off-design cases 4 and 5 or 6 and 7. Either a convergent or a convergent-divergent subroutine may be used, depending on the input controls. If afterburning has been selected, the nozzle area is allowed to float to satisfy continuity. However, the areas are returned to their original design values after the afterburning or nozzle float points are completed.

Because the first example is a single-spool turbojet, values for the inner compressor and turbine, as well as duct parameters, are either zero or 1. The actual or corrected values of compressor exit flow (WAC or WACC, respectively) are printed out prior to any bleed extraction. The value of WA3 is the combustor entrance airflow after bleed extraction.

The first 11 lines of the following example show the storage locations reserved for the input data. The locations are relative to the common block locations shown on page 21. Only locations for which values of data will be input are required. Following the end of the storage locations are the input data which are typical for running a single-spool turbojet engine.

The following is an example of the normal design-case printout. The fan, compressor, combustor, high-pressure- and low-pressure-turbine correction factors are printed out, as are the various designated design areas. Following these are the internal engine thermodynamic characteristics and thrust and specific fuel consumption.

```plaintext
*DATA(11) I: INPUT DATA CARDS

TABLE: REAL 22=TOTAL, 34=DELFG, 35=DELFL, 36=DELSFC, 37=ZFDS, 38=PCNFDS, 39=PRFDS,
40=ETFADS, 41=WACDS, 46=PCNODS, 47=T4ODS, 54=T4DS, 56=TFODS, 58=ETFADS, 66=PCNFDS, 69=TFODS,
70=CNLDFS, 71=ETLDFS, 82=OPUDS, 95=TTDS, 96=ETFADS, 97=DAFDS, 99=ZNAP, 100=PSAP,
102=AM55, 103=CVN0U, 146=CVNODS, 113=TW4, 114=T4, 115=WACDS, 116=PCNODS, 117=ETAFDS,
172=HPFD, 173=AM1, 174=ALTP, 175=ETAR, 177=PCNF, 179=PCNC, 183=PCCLF, 184=PCWLC,
185=PCBLDF, 186=PCBLHF, 187=PCBLHP, 205=T24, 225=ETAD, 257=TFAP, 273=ETAD, 279=AM6,
313=AM23, 399=AM28, 180=WFB, 222=WFD, INTEGER 12=IDES, 3=MODE,
7=ITAP, 8=IAHP, 9=IGASM, 10=IDBURN, 11=IAFTBN, 12=IDCO, 13=IMCD, 16=NOZFLT, 17=
ITRS, 314=ISPOOL, .LOGICAL .319=FAN) END OF TABLE GIVING NAMES AND LOCATIONS

*) I INPUT DATA TO RUN ONE SPOOL TURBOJET

FAN=.FALSE.,ISPOOL=1, ZFDS=.03333333, PRFDS=0, WACDS=.663, ETFADS=.829, PCNFDS=100,
PCCLF=0, PCCLH=0.97, PCCLPH=1, PCCLBP=0, T4ODS=3260, DPCUDS=.156, ETADS=.99,
ETFADS=.903, TFLDFS=1.30, CNLDFS=2.30, HPFD=.385, WFD=.238, AM6=.238, ETFADS=.90,
TTDS=.368, DPAFDS=.49, DELFG=1, DELFL=1, TDLALL=.005, TRYS=210, IDCO=0,
IMCD=0, IDES=1, IAKTP=0, MODE=0, IDUMP=0, AM=0, ALTP=0,

The following is an example of the normal design-case printout. The fan, compressor, combustor, high-pressure- and low-pressure-turbine correction factors are printed out, as are the various designated design areas. Following these are the internal engine thermodynamic characteristics and thrust and specific fuel consumption.
The following shows the common variables which can be read, if desired, by referring to the common designation (p. 21).

<table>
<thead>
<tr>
<th>COMMON</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.333333E+00</td>
</tr>
<tr>
<td>0.11000E+03</td>
</tr>
<tr>
<td>0.033333E+00</td>
</tr>
<tr>
<td>0.0</td>
</tr>
<tr>
<td>0.0</td>
</tr>
<tr>
<td>0.0</td>
</tr>
</tbody>
</table>

**PAIN**
**SCSIC**
**CONVERGENT**
**NOZZLE**

**DUET**
**SHOCK**
**OUTSIDE**
**C-D**
**NOZZLE**

**CONVERGED**
**AFTER**
**1**
**LOOPS**

---

The complete list of variables is available on the next page.
Off-design. - In the following cases, only the input data that are changed are printed out as input. The other data (normal design-case printout, p. 23) are not affected. The common variables are not repeated because IDES was set to zero internally in the program. That printout is eliminated and the design printout converts to the normal off-design printout format.

Case 1: The engine is operating at a reduced turbine inlet temperature at sea-level-static condition. The only input change required is T4.

Case 2: The engine is operating at design turbine inlet temperature at sea-level-static condition. The engine components must always be matched at the desired turbine inlet temperature prior to running the engine at afterburning conditions.
greater than the turbine exit temperature may be calculated. Therefore, TAFTBN must be used for each afterburning case; A8 is computed when afterburning to account for the temperature, pressure, and mass flow changes. A9 is allowed to float to provide for complete expansion to PS9 = PS0 at station 9. The new values of A8 and A9 are printed at the top of the normal off-design output page. The value of IAFTBN is set to zero after each case; therefore, IAFTBN = 1 must be used for each afterburning case.
Case 4: The engine is operating at a subsonic Mach number (0.7) at 25,000 feet altitude with a turbine inlet temperature less than design. The performance shown is for A8 = A9 (i.e., the engine is using a convergent nozzle). If desired, a convergent-divergent nozzle could have been simulated by using this case as a setup and having an additional case with NOZFLT = 1 or IMCD = 1 for complete expansion to occur.

Case 5: The engine is operating at the same subsonic condition as in case 4, but at the design turbine inlet temperature. Afterburner operation may be added at this point. If a different turbine inlet temperature is desired, a new setup case, as in case 4, must be calculated for component matching purposes immediately prior to running the afterburner case.
Case 6: The engine is now operating at a supersonic flight Mach number (2.5) at an altitude of 50 000 feet. The engine components are matched with an engine nozzle throat area A8 equal to the sea-level-static design value. This case is a setup case for matching the components only.
Case 7: The performance data from this case reflect an expansion to the maximum attainable area ratio for A9/A8, thus simulating a convergent-divergent nozzle. Performance is calculated such that the nozzle static pressure PS9 is equal to the free-stream static pressure PS0. This is accomplished by setting NOZFLT equal to 1. The improvement in thrust and specific fuel consumption from case 6 should be noted.

Although engine performance is calculated as if complete expansion occurred (A9/A8 = 3.449), a performance penalty from the fully expanded case resulted because this particular nozzle has a physical expansion limit A9/A8 of 2.15. This means that the nozzle has a lower-than-optimum thrust coefficient, resulting in a performance penalty.

**IO(l~YOZFLT=I~* PERFORMANCE DATA WITH MAXIMUM ATTAINABLE NOZZLE RATIO**

<table>
<thead>
<tr>
<th>NOZFLT DESIGN</th>
<th>A8= 0.45921256E+01</th>
<th>AM8= 0.10000000E+01</th>
<th>A9= 0.29598868E+02</th>
<th>AM9= 0.27030654E+01</th>
</tr>
</thead>
<tbody>
<tr>
<td>ARATIO=</td>
<td>3.449</td>
<td>OUT OF RANGE, USE DATA FOR 2.150</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**OUTPUT**

<table>
<thead>
<tr>
<th>AM= 2.500</th>
<th>ALT= 50000.00</th>
<th>T= 3280.00</th>
<th>WAF= 0.8073</th>
</tr>
</thead>
<tbody>
<tr>
<td>P9 = 0.32188E+02</td>
<td>P6 = 0.640647E+00</td>
<td>P5 = 0.665972E+00</td>
<td>P4 = 0.36153E+01</td>
</tr>
<tr>
<td>PC6 = 0.114102E+03</td>
<td>P5 = 0.1000000E+01</td>
<td>P4 = 0.1000000E+01</td>
<td>P3 = 0.1000000E+01</td>
</tr>
<tr>
<td>T2 = 0.47561E+03</td>
<td>P4 = 0.170275E+01</td>
<td>P3 = 0.132684E+04</td>
<td>P2 = 0.415444E+01</td>
</tr>
<tr>
<td>PCE = BLF</td>
<td>PCBL = BLFP</td>
<td>PCBF = BLHF</td>
<td>PCBL = BLOU</td>
</tr>
<tr>
<td>0.0 = 0.0</td>
<td>0.0 = 0.0</td>
<td>0.0 = 0.0</td>
<td>0.0 = 0.0</td>
</tr>
<tr>
<td>WAF = 0.0</td>
<td>WAF = 0.0</td>
<td>WAF = 0.0</td>
<td>WAF = 0.0</td>
</tr>
<tr>
<td>TA = 0.0</td>
<td>TA = 0.0</td>
<td>TA = 0.0</td>
<td>TA = 0.0</td>
</tr>
<tr>
<td>T2 = 0.0</td>
<td>T2 = 0.0</td>
<td>T2 = 0.0</td>
<td>T2 = 0.0</td>
</tr>
<tr>
<td>P9 = 0.0</td>
<td>P9 = 0.0</td>
<td>P9 = 0.0</td>
<td>P9 = 0.0</td>
</tr>
<tr>
<td>WAD = 0.0</td>
<td>WAD = 0.0</td>
<td>WAD = 0.0</td>
<td>WAD = 0.0</td>
</tr>
<tr>
<td>ETAF = 0.0</td>
<td>ETAF = 0.0</td>
<td>ETAF = 0.0</td>
<td>ETAF = 0.0</td>
</tr>
<tr>
<td>Ta3 = 0.0</td>
<td>Ta3 = 0.0</td>
<td>Ta3 = 0.0</td>
<td>Ta3 = 0.0</td>
</tr>
<tr>
<td>T8 = 0.0</td>
<td>T8 = 0.0</td>
<td>T8 = 0.0</td>
<td>T8 = 0.0</td>
</tr>
<tr>
<td>W9 = 0.0</td>
<td>W9 = 0.0</td>
<td>W9 = 0.0</td>
<td>W9 = 0.0</td>
</tr>
<tr>
<td>Bypass = 0.0</td>
<td>Bypass = 0.0</td>
<td>Bypass = 0.0</td>
<td>Bypass = 0.0</td>
</tr>
</tbody>
</table>

Case 8: The engine is operating in the supersonic flight condition with partial afterburning. This case illustrates that for an afterburning condition (IAFTBN = 1), the program automatically expands the nozzle so PS9 is equal to PS0. There is no need to input NOZFLT = 1 as in the previous case.
Case 9: The engine is operating in the supersonic condition at the maximum design afterburner temperature.
Case 10: Ability to fix the mechanical shaft speed PCNF is a revision to reference 2 which was supplied by its author. This engine operation method is accomplished by setting the control MODE equal to 3. MODE = 3 allows the engine to run at some desired value of shaft speed while the primary stream area A8 is varied. The value of turbine inlet temperature necessary to properly match the components is calculated in the program.

The supersonic flight condition is used for illustrative purposes. Any condition desired could have been used. In case 9, the outer compressor shaft speed was considered.

Case II: This case indicates that if MODE = 3 and the area is changed, T4 will change. The nozzle throat area has been opened to 120 percent of its design value (A8 = 8.586 x 1.20). The engine components match at a point where the calculated T4 is less than the design value and the compressor shaft speed is 100 percent of design. Area
variation A8 and compressor speed variations are the options that are available when MODE = 3. The engine components are matched in all setup cases with the engine operating with the throat area A8 equal to A9 as if the engine has a convergent nozzle. In order to return to a convergent nozzle, an input of the control variable NOZFLT = 0 is required when a new setup case is run because NOZFLT is not reset to zero internally. This case is a setup case and illustrates only one of the many options available for operation of the GENENG program.

Case 12: The final example case is the performance for the previous case (1%) when the maximum attainable area expansion is used (PS9 = PS0).
Two-Spool Turbojet

Design point. - An example is given for design-point operation of a two-spool turbojet. For the design point the data are printed out in three groups: (1) card images of the input data that were changed from the original input data, (2) results, and (3) common variables at the design point. The card images show two cards having $D(1), \text{IDES} = 1$, imprinted. These two cards in sequence recall the original design inputs. This method can be used as many times as desired in any computer run to design a new engine. A typical use for this may be a study of several engines where design-point performance only is sufficient. Thus, only changes in engine design characteristics need be input and a new engine is run. Or, as in this case, the engine was changed from a single-spool turbojet to a two-spool turbojet by changing ISPOOL = 1 to ISPOOL = 2 and by inputing the design pressure ratio and efficiency of the inner compressor and the design efficiency speed and flow function of the inner turbine. Also a change to the front compressor pressure ratio was made to have an overall pressure ratio of 8:1. The MODE was reset to zero (MODE = 0) from the previous cases of the single-spool turbojet engine.
Off-design. - The following cases are off-design cases for the two-spool turbojet.

Case 1: The engine is operating at a reduced turbine inlet temperature at sea-level-static condition. The input change of T4 was required. The compressor shaft speed of both compressors is reduced as are the pressure ratios, airflows, etc., all of which are representative of part-power operation.
Case 2: The engine is now operating at a supersonic flight Mach number (2.5) at an altitude of 50,000 feet. The engine components are matched with an engine nozzle throat area A8 equal to the sea-level-static design value. This case is a setup case for matching components only.
Case 3: The performance data for this case reflect an expansion to the maximum attainable area ratio for A9/A8, thus simulating a convergent-divergent nozzle. Performance is calculated such that the nozzle static pressure PS9 is equal to the free-stream static pressure PS0. This is accomplished by setting NOZFLT = 1.

Case 4: The engine is operating in the supersonic flight condition with partial afterburning. No new setup case was required because the components are still matched from the previous two cases (2 and 3). IAFBN = 1 automatically expands the nozzle so PS9 = PS0. It also adjusts A8 for the new temperature and pressure condition such that the nozzle throat is sonic. Both A8 and A9 are printed out at the top of the listing.
Case 5: The engine is operating in the supersonic flight condition at design afterburner temperature.

SEPARATE-FLOW TURBOFAN

Design point. - This is the design case for the separate-flow turbofan. For the design point, the data that were changed are shown on the top of the following example.
The control card FAN = . TRUE. designates that there is now a bypass ratio. Thus the front compressor (fan) flow does not equal the inner compressor flow. Consequently, a value for the inner compressor flow must be input. This value determines the engine bypass ratio. Also, a duct Mach number AM23 must be input. If the design is to be a duct-burning turbofan engine, AM23 should remain low (i.e., similar to an afterburner inlet Mach number - between 0.15 and 0.25) to keep friction losses at a minimum in the burner. Also a computational problem could arise when the duct burner is operating if AM23 is too high. The large temperature increase due to fuel addition will cause choking at engine duct station 24, consequently a printout error would occur.

The values of AM and ALTP are input to zero which are the conditions designated for the engine design point. A duct pressure loss DPDUDS is also input.

In the output of the results, the duct parameters now have values associated with the engine design, which includes area (A28 and A29) and duct Mach numbers.
Off-design. - The following are the off-design cases for the separate-flow turbofan engine.

Case 1: The engine is operating at Mach 2.5 at an altitude of 50 000 feet. The engine components are matched, with the duct and primary nozzle throat areas (A28 and A8) equal to the sea-level-static design value. This is a setup case for matching components only. However, note the very low thrust of the turbofan engine at this flight condition because the ram drag FRD is nearly equal to the gross thrust of the two streams FG.
Case 2: The same flight condition and engine operation is shown in this case as in case 1 because it is a continuation of the previous setup case. However, the engine is now duct burning. Therefore T24, IDBURN, and ETAD must be input. The duct burner does not have a performance map included in the subprogram as was the situation with the afterburner. Consequently, every efficiency (ETAD) and the control card and mass flow of the gas in order that the nozzle is sonic. The value of $A_8$ remains at the design value unless it is altered with a separate input. A permanent change in $A_8$ would require a new setup case prior to running a duct-burning case.
### Mixed-Flow Turbofan

**Design point.** - A mixed-flow turbofan engine is designed by using the input $\text{IGASMX} = 1$ and resetting the AM and ALTP to the desired value (sea-level-static in the example). This output listing shows the rebalancing of the fan and inner compressor pressure ratio to a value such that the static pressures at stations 25 and 55 will match. The overall compressor pressure ratio remains fixed. The values of the combination are not necessarily the optimum values; but it does give an indication, for the selected design values, as to what combination can be considered. Once the static-pressure values are satisfied, the program operates in a normal design mode. The other input cards are those required for the afterburner design. These values of design efficiency and temperature must be input for the design case in order to scale the generalized afterburner map to the proper design conditions. The afterburning takes place in the completely mixed streams - similar to an afterburning turbojet. The mixed-flow turbofan engine uses a single exhaust nozzle. The design areas are then printed out for this type of operation.

```
<table>
<thead>
<tr>
<th>Design Case</th>
<th>PRFCF</th>
<th>ETAFCF</th>
<th>W&amp;FCF</th>
<th>T2DS</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>FAN</strong></td>
<td>0.37500000E+01</td>
<td>0.94204545E+00</td>
<td>0.83333334E+00</td>
<td>0.5186820E+03</td>
</tr>
<tr>
<td><strong>COMPRESSOR</strong></td>
<td>0.12857143E+01</td>
<td>0.98837209E+00</td>
<td>0.10000000E+01</td>
<td>0.17057432E+03</td>
</tr>
<tr>
<td><strong>COMBUSTERS</strong></td>
<td>0.19974841E+02</td>
<td>0.98500000E+00</td>
<td>0.10000000E+01</td>
<td></td>
</tr>
<tr>
<td><strong>H.P. TURBINE</strong></td>
<td>0.11419282E+01</td>
<td>0.10000000E+01</td>
<td>0.89256619E+00</td>
<td></td>
</tr>
<tr>
<td><strong>L.P. TURBINE</strong></td>
<td>0.11419282E+01</td>
<td>0.10000000E+01</td>
<td>0.84956619E+00</td>
<td></td>
</tr>
<tr>
<td><strong>L.P. TURBINE</strong></td>
<td>0.11419282E+01</td>
<td>0.10000000E+01</td>
<td>0.84956619E+00</td>
<td></td>
</tr>
</tbody>
</table>

**CHANGE PRFS FROM** 2.500 TO 3.700 AND PRCS FROM 10.000 TO 6.754

<table>
<thead>
<tr>
<th>Design Case</th>
<th>PRFCF</th>
<th>ETAFCF</th>
<th>W&amp;FCF</th>
<th>T2DS</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>FAN</strong></td>
<td>0.67532611E+01</td>
<td>0.94204545E+00</td>
<td>0.83333334E+00</td>
<td>0.5186820E+03</td>
</tr>
<tr>
<td><strong>COMPRESSOR</strong></td>
<td>0.82003565E+00</td>
<td>0.98837209E+00</td>
<td>0.10000000E+01</td>
<td>0.17057432E+03</td>
</tr>
<tr>
<td><strong>COMBUSTERS</strong></td>
<td>0.27928302E+02</td>
<td>0.98500000E+00</td>
<td>0.10000000E+01</td>
<td></td>
</tr>
<tr>
<td><strong>H.P. TURBINE</strong></td>
<td>0.11419282E+01</td>
<td>0.10000000E+01</td>
<td>0.89256619E+00</td>
<td></td>
</tr>
<tr>
<td><strong>L.P. TURBINE</strong></td>
<td>0.11419282E+01</td>
<td>0.10000000E+01</td>
<td>0.84956619E+00</td>
<td></td>
</tr>
</tbody>
</table>

**CHANGE PRFS FROM** 2.500 TO 4.057 AND PRCS FROM 10.000 TO 6.163

<table>
<thead>
<tr>
<th>Design Case</th>
<th>PRFCF</th>
<th>ETAFCF</th>
<th>W&amp;FCF</th>
<th>T2DS</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>FAN</strong></td>
<td>0.76415902E+01</td>
<td>0.94204545E+00</td>
<td>0.83333334E+00</td>
<td>0.5186820E+03</td>
</tr>
<tr>
<td><strong>COMPRESSOR</strong></td>
<td>0.73753452E+00</td>
<td>0.98837209E+00</td>
<td>0.10000000E+01</td>
<td>0.17057432E+03</td>
</tr>
<tr>
<td><strong>COMBUSTERS</strong></td>
<td>0.40403457E+02</td>
<td>0.98500000E+00</td>
<td>0.10000000E+01</td>
<td></td>
</tr>
<tr>
<td><strong>H.P. TURBINE</strong></td>
<td>0.11419282E+01</td>
<td>0.10000000E+01</td>
<td>0.89256619E+00</td>
<td></td>
</tr>
<tr>
<td><strong>L.P. TURBINE</strong></td>
<td>0.11419282E+01</td>
<td>0.10000000E+01</td>
<td>0.84956619E+00</td>
<td></td>
</tr>
</tbody>
</table>

**CHANGE PRFS FROM** 2.500 TO 4.215 AND PRCS FROM 10.000 TO 9.931

<table>
<thead>
<tr>
<th>Design Case</th>
<th>PRFCF</th>
<th>ETAFCF</th>
<th>W&amp;FCF</th>
<th>T2DS</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>FAN</strong></td>
<td>0.80373937E+01</td>
<td>0.94204545E+00</td>
<td>0.83333334E+00</td>
<td>0.5186820E+03</td>
</tr>
<tr>
<td><strong>COMPRESSOR</strong></td>
<td>0.70466542E+00</td>
<td>0.98837209E+00</td>
<td>0.10000000E+01</td>
<td>0.17057432E+03</td>
</tr>
<tr>
<td><strong>COMBUSTERS</strong></td>
<td>0.31056852E+02</td>
<td>0.98500000E+00</td>
<td>0.10000000E+01</td>
<td></td>
</tr>
<tr>
<td><strong>H.P. TURBINE</strong></td>
<td>0.11419282E+01</td>
<td>0.10000000E+01</td>
<td>0.89256619E+00</td>
<td></td>
</tr>
<tr>
<td><strong>L.P. TURBINE</strong></td>
<td>0.11419282E+01</td>
<td>0.10000000E+01</td>
<td>0.84956619E+00</td>
<td></td>
</tr>
</tbody>
</table>

**TURBINE/FAN AREA DESIGN**

- $A_{55} = 0.87631926E+01$
- $A_{25} = 0.10496306E+02$
- $A_{25} = 0.5761657E+01$

**AFTERTURBAN DESIGN**

- $A_{25} = 0.48660238E+01$
- $A_{25} = 0.48660238E+01$
- $A_{25} = 0.10000000E+01$

**PLANE DESIGN**

- $A_{25} = 0.48660238E+01$
- $A_{25} = 0.48660238E+01$
- $A_{25} = 0.10000000E+01$
```
Off-design. - The following are off-design cases for the mixed-flow turbofan.

Case 1: The off-design case was selected to be supersonic Mach number (2.5) at 50 000 feet altitude. Case 1 illustrates a setup case for the afterburning case (case 2) which follows immediately after. Since the engine is operating at the design T4, the only input cards required are the AM and ALTP cards.
Case 2: The previous setup case is used here for an afterburning example. Values of afterburner efficiency ETAA, pressure loss DPAFT, and Mach number AM6 indicate values selected from the afterburner performance map at off-design. The expanded nozzle areas are shown; a printout indicating that a nozzle having a greater expansion ratio than the convergent-divergent nozzle map used would have been more optimum (instead of 2.150). Consequently, a slight loss in velocity coefficient due to an underexpanded nozzle occurred.
Design case without pressure ratio rematch. - The mixed-flow turbofan engine is shown again. However, the design fan and compressor pressure ratios were picked such that no rematch was required to balance the static pressures. This is another design case. As many design cases as desired can be run in succession. Care must always be taken to include the variables in input that need to be changed from the previous design case.

**Design Case Details**

**Main Shock Outside C-D Nozzle**

<table>
<thead>
<tr>
<th>T6</th>
<th>P6</th>
<th>P6S</th>
<th>AM6</th>
<th>V6</th>
<th>WGA</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.14345E+04</td>
<td>0.23060E+01</td>
<td>0.22748E+01</td>
<td>0.143137E+00</td>
<td>0.276665E+03</td>
<td>0.294650E+03</td>
</tr>
</tbody>
</table>

**Engine Design**

- **Main Engine Designs**

  - **Fan Design**
    - PRCF = 0.100000E+00
    - ETAFCF = 0.942045E+00
  - **Compressor Design**
    - PRCCF = 0.500000E+00
    - ETAFCF = 0.945000E+00
  - **Combustor Design**
    - WACOS = 0.358186E+02
    - ETAFCF = 0.985000E+00
  - **Turbine Design**
    - DHRPCF = 0.116221E+01
    - ETAFCF = 0.833333E+00
  - **Afterburner Design**
    - WACOS = 0.486049E+00

**Output Parameters**

- **Flow Rate**
  - AM = 0.460326E+01
  - AM# = 0.100000E+01
  - AM# = 0.100000E+01

- **Engine Performance**
  - T6 = 3260.00
  - ETA# = 1.0000

**Converged After 1 loops**
The computer code (GENENG) presented herein has proven to be an indispensable tool for steady-state cycle analysis of various types of jet engines, namely two-spool mixed- or unmixed-flow turbofans and one- or two-spool turbojets.

This program has been found to be valuable for many applications because it has the capability of studying a broad range of engine types having different design characteristics, while it also has low-execution-time requirements. The laborious task of matching components manually has been eliminated.

The program has proven itself to be easy to use, especially in terms of input requirements. It is felt that with a minimum of effort the reader can become proficient in using the computer code.

The code is available to be reproduced on the requestor's tape upon application to the authors.

Lewis Research Center,
National Aeronautics and Space Administration,
Cleveland, Ohio, October 1, 1971,
APPENDIX A

GENENG PROGRAM LISTING

$IBFCC GENENG DECK
COMMV/POINT/IDATPT
COMMV/LOOPPR/KKGO,PRFNEW,PRCNEW
DATAIII/0/
DIMENSION XLO(5),XHI(5)
EQUIVALENCE (XLO,DLDSV),(XHI,DDHISV)
COMMV/ALL/X(28)/DESIGN/Y(80)/FRONT/Z(80)/SIDE/W(48)/BACK/V(72)
COMMV/DUMMY/DUMMY(100)
EQUIVALENCE (P6DSAV,DUMMY(7)),(AM6DSV,DUMMY(8)),( ETAASV,DUMMY(9)),
l(FAR75V,DUMMY(10))
LOGICAL ERROR,CLEAR
DATA CLEAR/,TRUE,
COMMV/ERROR/ERROR
ERROR=.FALSE.
IF (*.VOT,CLEAR) CALL ENGBAL
CLEAR=.FALSE.
DO 1 J=1,408
1 X(J)=0.
IF (III,.EQ.0) KKGO=0
III=1
IDATPT=0
CALL CONDOUT (1)
P6DSAV=1.
AM6DSV=1.
ETAASV=1.
FAR75V=1.
DO 2 I=1,5
XLO(I)=100.
2 XHI(I)=100.
CALL ENGBAL
STOP
END

$IBFCC ENGBAL DECK
SUBROJTIME ENGBAL
COMMV/ALL/
1WORD,IDES,JDES,KDES,MODE,INIT,TIME,IBMTP,
2IGASMX,IBURM,IAFTBN,ICDCL,IMCD,IDSHNC,IMSHDC,NOZFLT,
3ITRYS,FLOPER,NOMAP,NUMMAP,MAPEDE,TOLALL,ERR(6)
COMMV/DESIGN/
1PCNFEG,PCNCGU,T4GU,DUMDI,DUMD2,DELFG,DELFA,DELSFC,
2ZFDS,PCNFDZ,PRFDZ,ETAFDZ,WAFTDZ,PRCFZ,ETACFZ,WAFCFZ,
3ZCDS,PCNCDS,PRCDS,ETACDS,WAACDS,PRCCFZ,ETACCFZ,WAFCFZ,
4T4DS,WFDOS,TDCDS,ETABD,W3CDS,DPDCDS,TDCGCF,ETABCF,
5TFHPDS,CNHPDS,ETHHPDS,TFHPDCF,CNHPCF,ETHPCF,DDHPCF,T2DS,
6TLPDS,CNLPCF,TFLPCF,CNLPCF,ETLPCF,DLHPCF,T21DS,
7T24DS,WFDOS,TDUDS,ETADD,W23DS,DPDUDS,TDUCF,ETADCF,
8T2DS,WFDOS,DTAFDS,ETAADS,WF6CDS,DPFDS,DAFDCF,ETACFZ,
9A55,A25,A6,A7A8A9,A28,A29
$PS55,AM55,CVDNOZ,CVMN2Z,ABSAV,A9SAV,A28SAV,A29SAV
COMMON / FRONT/

1T1  P1  H1  S1  T2  P2  H2  S2  
2T21  P21  H21  S21  T3  P3  H3  S3  
3T4  P4  H4  S4  T5  P5  H5  S5  
4T55  P55  H55  S55  BLF  BLA  BLDU  BLDB  
5CNF  PRF  ETA  WAFC  WA  W4  FAR4  
6CN  PRC  ETAC  WACC  WAC  ETAB  DPCOM  DUMF  
7CNH  ETAHLP  DHTCHP  DHTC  BLHP  #W5  FAR5  CS  
8CNLP  ETAI  DHTCLP  DHTF  BLP  W5G  FAR55  HPX  
9AM  ALTP  ETA  ZF  PCNF  ZC  PCNC  HFB  
$TFFHP  TFILP  PCBLF  PCBLG  PCBLDU  PCBLDG  PCBLHP  PCBLLP  
COMON / SIDE/

1XP1  XWAF  XWAC  XBLF  XBLDU  XH3  DUM31  DUM32  
2XT21  XP21  XH21  XS21  T23  P23  H23  S23  
4T28  P28  H28  S28  T29  P29  H29  S29  
5WAD  WFD  W624  FAR24  ETAD  JPDUC  BYPASS  DUMS3  
6TS29  PS29  V28  AM28  TS29  PS29  V29  AM29  
COMON / BACK/

1XT55  XP55  XH55  XS55  XT25  XP25  XH25  XS25  
2WMBJ  XW55  XFAR55  XWFD  XWG4  XFR24  XXP1  DUM8  
3T6  P6  H6  S6  T7  P7  H7  S7  
4T8  P8  H8  S8  T9  P9  H9  S9  
5W66  WFA  WG7  FAR7  ETA  EPAFT  V55  V25  
6PS6  V6  AM6  TS7  PS7  V7  AM7  AM25  
7PS8  V8  AM8  TS9  PS9  V9  AM9  
8VA  FRD  VJD  FGMD  VJM  FGMM  FPD  FSPM  
9FGM  FGP  WFT  WGT  FART  FG  FN  SFC  
COMMON/DUMY/DUMMYS/DUMMY(100)  
LOGICAL ERRER,FAN  
EQUIVALENCE (FAN,DUMY(1)),(ISPOOL,DUMMY(6))  
COMMON/ERRER/ERRER  
DIMENSION VAR(6),DEL(6),ERRB(6),DELVAR(6),EMAT(6,6),VMAT(6),  
1AMAT(5)  
DATA AWORD/6HENGAL/  
DATA VDELTA,VLIM,VCHNGE,NOMISX/  
1 0.001,0.100,0.850,4/  
CALL PUTIN  
IF (INIT.EQ.1) GO TO 1  
1 LOOPX=0  
2 NUMMAP=0  
3 NOMISS=0  
2 LOOP=0  
3 MISMAT=0  
4 NQMAP=0  
IGO=2  
DO 3 I=1,6  
5 MAT(I)=0.  
6 AMAT(I)=0.  
7 DELVAR(I)=0.  
8 DO 3 L=1,6  
9 EMAT(I,L)=0.  
4 LOOP=LOOP+1  
CALL CDFAN  
WORD=AWORD  
IF (LLOOP.GT.ITRYS) ERRER=.TRUE.  
IF (LLOOP.GT.ITRYS) GO TO 24  
IF (NOMAP.GT.0) GO TO 2  
NUMMAP=0  

VAR(1)=ZF=100.
IF (M0DE.LT.3) VAR(2)=PCNF
IF (M0DE.EQ.3) VAR(2)=T4/10.0
VAR(3)=ZC=100.
IF (M0DE.LT.1) VAR(4)=PCNC
IF (M0DE.EQ.1) VAR(4)=T4/10.0
VAR(5)=TFFHP
VAR(6)=TFFLP
NMAX=5
IF (FAVOR.ISPOOL.EQ.2) GO TO 6
NMAX=3
VAR(3)=TFFLP
DO 7 I=1,NMAX
7 IF (ABS(ERR(I)).GT.TOLALL) GO TO 8
CALL PERF
CALL ERROR
8 IF (LC0P.GT.0) GO TO 10
MAPEDG=0
MAPSET=0
DO 9 I=1,NMAX
9 ERRB(I)=ERR(I)
DEL(I)=VDELTAT+VAR(I)
GO TO 13
10 IF (MISMAT.GT.0) GO TO 27
IF (MAPEDG.EQ.0) GO TO 11
MAPEDG=0
MAPSET=1
VAR(L00P)=VAR(L00P)+2.*DEL(L00P)
GO TO 14
11 IF (MAPSET.EQ.0) VAR(L00P)=VAR(L00P)+DEL(L00P)
IF (MAPSET.EQ.1) VAR(L00P)=VAR(L00P)-DEL(L00P)
MAPSET=0
DO 12 I=1,NMAX
12 EMAT(I,L00P)=(ERRB(I)-ERR(I))/DEL(L00P)
13 LOOP=LOOP+1
IF (LC0P.GT.NMAX) GO TO 15
VAR(L00P)=VAR(L00P)-DEL(L00P)
14 ZF=VAR(1)/100.
IF (M0DE.LT.3) PCNF=VAR(2)
IF (M0DE.EQ.3) T4=VAR(2)*10.0
ZC=VAR(3)/100.
IF (M0DE.LT.1) PCNC=VAR(4)
IF (M0DE.EQ.1) T4=VAR(4)*10.0
TFFHP=VAR(5)
TFFLP=VAR(6)
IF (ISPOOL.EQ.1) TFFLP=VAR(3)
IF (ZF.LT.0.0) ZF=0.05
IF (ZC.LT.0.0) ZC=0.05
GO TO (2,4),1GO
15 DO 16 I=1,NMAX
16 AMAT(I)=-ERRB(I)
DO 18 I=1,NMAX
ZERO=0
DO 17 LOOP=1,NMAX
17 IF (AMAT(I,LOOP).EQ.0.0) ZERO=ZERO+1
IF (ZERO.LT.NMAX) Go To 18
WRITE (6,30) I
LOOPER=ITRYS+100
GO TO 24
18 CONTINUE
DO 20 LOOP=1,NMAX
ZERO=0
DO 19 I=1,NMAX
19 IF (EMAT(I,LOOP).EQ.0.) IZERO=IZER0+1
   IF (IZER0.LT.NMAX) GO TO 20
   WRITE (6,31) LOOP
   LOOP=ITRYS+100
   GO TO 24
20 CONTINUE
21 CALL MATRIX (EMAT,VMAT,AMAT,NMAX)
   LBIG=0
   VARBIG=0.0
   DO 22 L=1,NMAX
      ABSVAR=ABS(VMAT(L))
      IF (ABSVAR.LE.VLIM*VAR(L)) GO TO 22
      IF (ABSVAR.LE.VARBIG) GO TO 22
      LBIG=L
      VARBIG=ABSVAR
   CONTINUE
22 CONTINUE
   VRATIO=1.0
   IF (LBIG.GT.0) VRATIO=VLIM*VAR(LBIG)/VARBIG
   ERRAVE=0.0
   VMTAVE=0.0
   DELAVE=0.0
   DO 23 L=1,NMAX
      DELVAR(L)=VRATIO*VMAT(L)
      ERRAVE=ERRAVE+ABS(AMAT(L))
      VMTAVE=VMTAVE+ABS(VMAT(L))
      DELAVE=DELAVE+ABS(DELVAR(L))
      VAR(L)=VAR(L)+DELVAR(L)
   CONTINUE
23 ERRAVE=ERRAVE/FLOAT(NMAX)
   VMTAVE=VMTAVE/FLOAT(NMAX)
   DELAVE=DELAVE/FLOAT(NMAX)
   IF (MISMAT.GT.0) GO TO 29
   IF (NOMISS.EQ.0) MISMAT=1
   IF (MISMAT.EQ.0) IG0=1
   WRITE (8,32) LOOP
   DO 25 I=1,NMAX
      WRITE (8,33) AMAT(I),(EMAT(I,L),L=1,6),VMAT(I),DELVAR(I),VAR(I)
   WRITE (8,34) ERRAVE,VMTAVE,DELAVE
   IF (L12PER.LT.ITRYS) GO TO 14
24 CALL ERROR
   RETURN
25 VMTAVX=VMTAVE
   DO 28 I=1,NMAX
      AMAT(I)=-ERR(I)
   CONTINUE
26 IF (L13PER.LT.ITRYS) GO TO 14
27 WRITE (8,35) AMAT,ERRAVE,DELVAR,DELAVE,VMAT,VMTAVE,VAR
   MISMAT=MISMAT+1
   IF (VMTAVE.LT.VCHNGE*VMTAVX) GO TO 26
28 WRITE (8,36)
29 IF (MISMAT.LT.NOMISS) NOMISS=1
   MISMAT=0
   LOOP=0
   IG0=2
   GO TO 5
30 FORMAT (4HOROW,12,16H IS ZERO IN EMAT)
31 FORMAT (7HOCOLUMN,12,16H IS ZERO IN EMAT)
32 FORMAT (8H8 ERRB,28X23HERROR MATRIX AFTER LOOP,14,29X4HVMTAV,6X5H
    1DELVAR,7X14HVARIABLES$$$$$$)
33 FORMAT (1H0,F8.4,8X6F10.4,10XF10.4,F11.4,4XF11.4,6H$$$$$$
34 FORMAT (1H0,F8.4,32X14HAVERROR VALUES,31X,2F11.4,6H$$$$$$)
$IBFTC GUESS DECK
FUNCTION GUESS(M, T, TD, P, PD, W, WD, D, DD, VD) 
IF (M.EQ.0) GUESS=VD*(((T/TD)**1.60)*((DD/D)**0.50) 
IF (M.EQ.1) GUESS=VD*(((P/PD)**1.80)*((DD/D)**0.33) 
IF (M.EQ.2) GUESS=VD*(((W/WD)**0.33)*((DD/D)**1.00) 
IF (M.EQ.3) GUESS=VD*(((W/WD)**0.00)*((P/PD)**0.50) 
IF (M.EQ.4) GUESS=VD*(((W/WD)**0.00)*((P/PD)**0.50) 
IF (M.EQ.5) GUESS=VD*(((T/TD)**1.10)*((DD/D)**0.60) 
IF (M.EQ.6) GUESS=VD*(((P/PD)**1.00)*((DD/D)**0.25) 
IF (M.EQ.7) GUESS=VD*(((P/PD)**0.62)*((DD/D)**0.31) 
RETURN 
END

$IBFTC MATRIX DECK
SUBROUTINE MATRIX (E, V, A, J) 
DIMENSION E(6,6), V(6), A(6), PIV(7), T(6,7) 
AN=N+1 
NM=N-1 
DO 1 I=1,N 
T(I,NN)=A(I) 
DO 1 J=1,N 
1 T(I,J)=E(I,J) 
DO 7 I=1,N 
TEMP=3. 
DO 2 J=1,N 
IF (TEMP.GT.ABS(T(J,I))) GO TO 2 
TEMP=ABS(T(J,I)) 
IFIV=J 
CONTINUE 
2 CONTINUE 
IF1=I+1 
DO 3 J=IPI1,NN 
3' PIV(J)=T(IPIV,J)/T(IPIV,1) 
IFROM=1 
I10=N 
4 IF (IFROM.EQ.IPIV) GO TO 6 
RM=-T(IFROM,1) 
DO 5 J=IPI1,NN 
5 T(I,10,J)=T(IFROM,J)+RM*PIV(J) 
I10=IT0-1 
6 IFROM=IFROM-1 
IF (IFROM.GE.I) GO TO 4 
DO 7 J=IPI1,NN 
7 T(I,J)=PIV(J) 
DO 8 I=1,NN 
J=NN-I 
K=N-I 
DO 8 L=J,N 

8 \[ T(K,NN) = T(K,NN) - T(K,L) * T(L,NN) \]
DO 9 I=1,N
9 \[ V(I) = T(I,NN) \]
RETURN
END

$/BFTC PUTL\DECK$
SUBROJTINE PUTIN
COMMON / POINT / IDATPT
COMMON / ALL /
1WORD, IDES, JDES, KDES, MODE, INIT, IDUMP, IAMTP, 21GASMX, IDBURN, IAFTBN, IDC, IMGD, IDSHOC, IMSHOC, NOZFLT, 3ITRYs, LOOPER, NOMAP, NUMMAP, MAPPED, TOLALL, ERR(6)
COMMON / DESIGN /
1PCNFOD, PCNCGO, T4GU, DUMD1, DUMD2, DELFG, DELFN, DELSFC, 22FDS, PCNFDS, PRFDS, ETAFDS, WAFLDS, PRFCF, ETAFCF, WAFCF, 32CDS, PCNCDS, PRCDOS, ETAADS, WACDS, PRCCF, ETAACF, WAACF, 4T4DS, WFLDOS, DTCDOS, ETAADS, WACDS, JPCDOS, DTCDCF, ETABCF, 5TFHPDS, CHzHPS, ETHPDS, FHHPDF, CTHHPDF, ETHHPDF, DTHHPDF, T2DS, 6TFLPDS, CNLPS, TFLPDS, CNLPCF, TFLPCF, DTHPCF, T2IDS, 7724DS, WFDOS, DTUOS, ETAADS, WACDS, JPCDOS, DTHDCF, ETACDF, 877DS, WFLDOS, DTAFDS, ETAADS, WACDS, JPCDOS, DTHACF, ETACACF, 9955, A25, A6, A7, A8, A9, A28, A29, *PS55, AM55, CVDNOZ, CVMNJZ, A8SAV, A9SAV, A28SAV, A29SAV
COMMON / FRONT /
1T11, P1, H1, S1, T2, P2, H2, S2, 2T21, P21, H21, S21, T3, P3, H3, S3, 3T4, P4, H4, S4, T5, P5, H5, S5, 4T55, P55, H55, S55, BLF, BLC, BLDU, BLOB, 5CNF, PRF, ETAF, WAFC, WAF, WA3, W4, FAR4, 66NC, PRC, ETAC, WACC, WAC, ETAAB, DPACOM, DUMF, 7CNHP, ETATHP, DHTCHP, DHTC, BLHP, LG5, FAR5, CS, 8CNLP, ETALP, DHTCPL, DHTF, BLLP, W635, FAR55, HPACX, 99AM, ALTP, ETAR, ZF, PCNF, ZC, PCNC, WFB, *TFFHP, TFFLP, PCBLF, PCBLC, PCBLDO, PCBLDS, PCBLHP, PCBLLP
COMMON / SID /
COMMON / BACK /
1XT55, XP55, XH55, X555, XT25, XP25, XH25, X525, 2XWFB, XWGD5, XFAR55, XFD, XWDG4, XFR24, XP1, DUMB, 3T6, P6, H6, S6, T7, P7, H7, S7, 4T8, P8, H8, S8, T9, P9, H9, S9, 5NG6, NFA, W7, FAR7, ETAA, DPAFT, V55, V25, 6PS6, V6, AM6, TS7, PS7, V7, AM7, AM25, 77ST8, PS8, V8, AM8, TS9, PS9, V9, AM9, 89AV, FRD, VJD, FGMD, VJM, FGGM, FGPD, FGFPM, 9FGM, GFG, WFT, WFT, FART, FG, FN, SFC
COMMON/DUMMYS/DUMMY(100)
COMMON/LOOPPR/KKGQ, PRFNEW, PRCSNEW
DIMENSION XSAYE(308), XFLX1(1)
EQUIVALENCE (XFLX, HODR)
LOGICAL ERROR
COMMON/ERROR/ERROR

50
**C *** ITITLE=1 WILL READ IN TITLE**

**C *** MODE =0 FOR CONSTANT T4**

**C *** MODF =1 FOR CONSTANT PCNC**

**C *** MODE =2 FOR CONSTANT WFB**

**C *** MODE=3 FOR CONSTANT PCNF**

**C *** INIT =1 WILL NOT INITIALIZE POINT**

**C *** IDUMP =1 WILL DUMP LOOPING WRITE-OUTS IF ERROR OCCURS**

**C *** IDUMP =2 WILL DUMP LOOPING WRITE-OUTS AFTER EVERY POINT**

**C *** IAMTP =0 WILL USE INPUT AM AND MIL SPEC ETAR**

**C *** IAMTP =1 WILL USE INPUT AM AND INPUT ETAR**

**C *** IAMTP =2 WILL USE T2 AS T1=T1+T2 AND STANDARD P1**

**C *** IAMTP =3 WILL USE P2 AND STANDARD T1**

**C *** IAMTP =4 WILL USE T2 AND P2**

**C *** IAMTP =5 WILL USE RAM2 FOR SPECIAL RECOVERY**

**C *** IGASMX=-1 SEPERATE FLOW, INPUT AM6**

**C *** IGASMX=0 SEPERATE FLOW, A6=A25+A55**

**C *** IGASMX=1 WILL MIX DUCT AND MAIN STREAMS, A6=A25+A55**

**C *** IGASMX=2 WILL MIX DUCT AND MAIN STREAMS, INPUT AM6**

**C *** IDBURN=1 FOR DUCT BURNING, INPUT T24**

**C *** IDBURN=2 FOR DUCT BURNING, INPUT WFD**

**C *** IAFIBN=1 FOR AFTERBURNING, INPUT T7**

**C *** IAFIBN=2 FOR AFTERBURNING, INPUT WFA**

**C *** IMCD =1 DUCT NOZZLE WILL BE C-D**

**C *** IMCD =1 MAIN NOZZLE WILL BE C-D**

**C *** NOZFLT=1 FOR FLOATING MAIN NOZZLE**

**C *** NOZFLT=2 FOR FLOATING DUCT NOZZLE**

**C *** NOZFLT=3 FOR FLOATING MAIN AND DUCT NOZZLES**

**C *** ITRYS =N NUMBER OF PASSES THRU ENGINE BEFORE QUtTING**

**DIMENSION ITABLE(400)**

**DATA (ITABLE(I),I=1,3)/0,400,0/**

**CALL ZERO**

**IF (K<GO.EQ.1) GO TO 5**

**IDES=0**

**CALL INPUT (5,6,1,WORD,ITABLE)**

**IF (ERROR.AND.,IAFTBN.GT.0) GO TO 1**

**ERROR=.FALSE.**

**C TABLE IS REFERENCED TO COMMON/ALL/FIRST ENTRY**

**IF (IDES.EQ.0) GO TO 7**

**IF (KGO.NE.2) GO TO 3**

**DO 2 I=1,308**

**2 XFILL(I)=XSAVE(I)**

**CALL INPUT (5,6,1,WORD,ITABLE)**

**3 CONTINUE**

**C SAVE INPUT IN CASE IF LOOP ON PRESSURE RATIOS**

**DO 4 I=1,308**

**4 XSAVE(I)=XFILL(I)**

**GO TO 7**

**5 DO 6 I=1,308**

**6 XFILL(I)=XSAVE(I)**

**WRITE (6,8) PRFDS,PRFNEW,PRCDS,PRCNEW**

**PRCDS=PRCNEW**

**PRFDS=PRFNEW**

**7 CONTINUE**

**KGDO=2**

**IF (IAFTBN.GT.0.OR.IDBURN.GT.0) INIT=1**

**IF (NOZFLY.GT.0) INIT=1**

**IF (M3DE.EQ.0) WRITE (8,9) IDES,AM,ALTP,T4,T24,T7**

**IF (M3DE.EQ.1) WRITE (8,10) IDES,AM,ALTP,PCNC,T24,T7**

**IF (M3DE.EQ.2) WRITE (8,11) IDES,AM,ALTP,WFB,T24,T7**

**CALL ZONLT**

**RETURN**
SUBROUTINE ZERO

COMMON / ALL/
1WORD ,IDES ,JDES ,KDES ,MODE ,INIT ,IDUMP ,IAMTP, 1
2IGASMX, IDBURN, IAFBVN, IDCDO, IMCD ,I OSHOC, IMSHOC, VDZFZT, 2
3ITRYS ,LOOPER, N0MAP, NUMMAP, MAPPEDG, TOLALL, ERR(4), 3
COMMON/DESIGN/QXQ(80) 4
COMMON / FRONT/
1T1 ,P1 ,H1 ,S1 ,T2 ,P2 ,H2 ,S2 , 5
2T21 ,P21 ,H21 ,S21 ,T3 ,P3 ,H3 ,S3 , 6
3T4 ,P4 ,H4 ,S4 ,T5 ,P5 ,H5 ,S5 , 7
4T5 ,P55 ,H55 ,S55 ,T6 ,P6 ,H6 ,S6 , 8
5CNF ,PRF ,ETAF ,WAFC ,WA3 ,W4 ,FAR4 , 9
6CNF, PRF, ETAC ,WACC ,WAC ,ETAB ,DPCOM ,DUMF, 10
7CNHP ,ETATHP ,DHTCHP ,DHTC ,BLHP ,BLU ,BLDB , 11
8CNLP ,ETATLP ,DHTCLP ,DHTF ,BLLP ,WB55 ,FAR55 ,HPEXT, 12
9AM ,ALTP ,ETAR ,ZF ,PCNF ,ZC ,PCNC ,WFB, 13
$TFFHP ,TFLLP ,PCBLF ,PCBLG ,PCBLDO ,PCBLHO ,PCBLLP 14
COMMON / SIDE/
1XP1 ,XWAF ,XWAC ,XBLF ,XBLDU ,XH3 ,DUMS1 ,DUMS2 , 15
2XT21 ,XP21 ,XH21 ,XS21 ,T23 ,P23 ,H23 ,S23 , 16
3T24 ,P24 ,H24 ,S24 ,T25 ,P25 ,H25 ,S25 , 17
4T28 ,P28 ,H28 ,S28 ,T29 ,P29 ,H29 ,S29 , 18
5WAD ,WF D, WG24 ,FAR24 ,ETAD ,DPU2, BYPASS ,DUMS3 , 19
6TS28 ,PS28 ,V28 ,AM28 ,TS29 ,PS29 ,V29 ,AM29 20
COMMON / BACK/
1XT55 ,XP55 ,XH55 ,XS55 ,XT25 ,XP25 ,XH25 ,XS25 , 21
2XWFB ,XWGS5 ,XFAR55 ,XWFD ,XWG24 ,XFAR24 ,XXP1 ,DUMB, 22
3T6 ,P6 ,H6 ,S6 ,T7 ,P7 ,H7 ,S7 , 23
4T8 ,P8 ,H8 ,S8 ,T9 ,P9 ,H9 ,S9 , 24
5W66 ,MFA ,WG7 ,FAR7 ,ETA, DPAFT, V55 ,V25 , 25
6PS6 ,V6 ,AM6 ,TS6 ,PS7 ,V7 ,AM7 ,AM25 , 26
7TS8 ,PS8 ,V8 ,AM8 ,TS9 ,PS9 ,V9 ,AM9 , 27
8VA ,FRD ,VJD ,FGMD ,VJN ,FGMM ,FGPD ,FGPM , 28
9FGM ,FGP ,WFT ,WGT ,FART ,FG ,FN ,SFC 29
COMMON/DUMMS/DUMMY(100) 30
DIMENSION Z1(63), Z2(48), Z3(72) 31
EQUIVALENCE (Z1,T1), (Z2,XP1), (Z3,XT55) 32
IDES=0 33
JDES=0 34
INIT=0 35
IDBURN=0 36
IAF BVN=0 37
IDCDO=0 38
IMSHOC=0 39
T2Q=T2 40

52
P2Q=P2
T4Q=T6
DO 1 I=1,63
1
Z1(I)=0.
DO 2 I=1,48
2
Z2(I)=0.
DO 3 I=1,72
3
Z3(I)=0.
T2=T2Q
P2=P2Q
T4=T4Q
CALL SYE (1)
RETURN
END

$SFTC COINLT DECK
SUBROUTINE COINLT
COMMOV / ALL/
1WORD , IDES , JDES , KDES , MODE , INIT , IDUMP , IAMTP ,
2IAGSMX , IDBURN , IAFTHB , IDCD , IMCD , IDSHOC , IMSHOC , NOZFLT ,
3ITRYs , LOOPER , NOMAP , NUMMAP , MAPEDG , TDLALL , ERR (6)
COMMOV / DESIGN/
1PCNFGJ , PCNCGU , T4GU , DUMD1 , DUMD2 , DELFG , DELFN , DELSFC ,
22FDs , PCNFDS , PRFDs , ETAFDs , WAIFDS , PRF CF , ETAFCF , WAFCF ,
32CDS , PCNCDS , PRCDs , ETACDS , WACOS , PRCCF , ETAACF , WACCF ,
4T4DS , WFBDs , DTCDS , ETAADS , WA3CDS , DPCODS , DTCGCF , ETA BCF ,
5TFHPDS , CNHPS , ETHPDS , TFPVCF , CHPFPC , ETHPCE , DHPFCF , T2DS ,
6TFLPDS , CNLPDS , TFLPDS , TFPCF , CNLPCE , ET LPCF , DHLPCF , T2DS ,
7T24DS , WFDDs , DTDUDS , ETADDS , WA23DS , DPDUODS , DTUCDF , ETADCF ,
8T77DS , WFAOD , DTAODs , ETAADS , WGGCDS , DPFADS , DTAFCF , ETAACF ,
9A55 , A25 , A6 , A7 , A8 , A9 , A28 , A29 ,
$PS55 , AM55 , CVNDOZ , CVMNDOZ , A8SAV , A9SAV , A28SAV , A29SAV
COMMOV / FRONT/
1T1 , P1 , H1 , S1 , T2 , P2 , H2 , S2 ,
2T21 , P21 , H21 , S21 , T3 , P3 , H3 , S3 ,
3T4 , P4 , H4 , S4 , T5 , P5 , H5 , S5 ,
4T55 , P55 , H55 , S55 , BLF , BLC , BLDU , BL0B ,
5CNF , PRF , ETAF , WAFC , WAF , WA3 , WG4 , FAR4 ,
6CNC , PRC , ETAC , WACC , WAC , ETAB , DPCOM , DUMF ,
7CNHP , ETAHPS , DHTCPS , DHTC , BLHP , WGS , FARS , CS ,
8CNLP , ETATLP , DHTCLP , DHTF , BLLP , WGG5 , FARS5 , HPEXT ,
9AM , ALTP , ETAR , ZF , PCNF , ZC , PCNC , WBF ,
$TFFHP , TFFLP , PBCFL , PBCLF , PBCBLD , PBCBLD , PBCBLHP , PCBLLP
COMMOV / SIDE / ZYX (148) / BACK / YZX (T2)
COMMOV / DUMMY / (100)
DATA AWORD / 6HGCOINLTF /
WORD=AWORD
AJ=775 , 26
G=32 , 174049
ALT=ALTP*2.0855531E+07/(2.0855531E+07+ALTP)
CALL ATMD (ALT , T1 , XX1 , XX2 , XX3 , P1 , CS , XX4 , I1ER )
IF ( IAMTP . EQ . 2 ) T1=T1+T2
IF ( IAMTP . EQ . 5 ) CALL RAM2 ( AM , ETAR )
IF ( IAMTP . NE . 1 . AND . IAMTP . NE . 5 ) CALL RAM ( AM , ETAR )
FAR=0.
CALL >ROCOS ( FAR , T1 , CS , XX2 , XX3 , R1 , PHI1 , H1 )
S1=PHI1-R1*ALOG ( P1 )
H2=H1*(AM*CS)**2/(2.*AJ*G)
P2T=1,
DO 1 I=1,10
CALL THERMO (P2T,H2,T2T,S2T,AW,Q0,0.0,0.1)
IF (ABS(S2T-S1).LE.0.0001*S1) GO TO 2
1 P2T=P1*EXP((AW/1.986375)*((S2T-S1)+(1.986375/AW)*ALOG(P2T/P1)))
CALL ERROR RETURN
2 IF (IAMTP.EQ.3.OR.IAMTP.EQ.4) ETAR=P2/P2T
P2=ETAR*P2T
IF (IAMTP.NE.4) CALL THERMO (P2,H2,T2,T2S,XX5,0,0.0,0.1)
IF (IAMTP.EQ.4) CALL THERMO (P2,H2,T2,T2S,XX5,0.0,0.0)
IF (INIT.EQ.1) GO TO 5
IF (IDES.EQ.1) GO TO 3
IF (MODE.EQ.3) GO TO 4
PCNF=3*JESS(MODE,T4,T4DS,PCNC,PCNCD5,WFB,WFBDS,T2,T2DS,PCNFDS)
PCNFJ=PCNF
GO TO 4
3 PCNF=PCNFDS
PCNFJ=PCNF
T2DS=T2
4 ZF=ZFD
5 RETURN
END

$IBFTC ATMOS DECK

SUBROJTINE ATMOS (ZFT,TM,SIGMA,RHO,THETA,DELTA,CA,AMU,K) 1
C THIS IS A SUBROUTINE TO COMPUTE CERTAIN ELEMENTS OF THE 1962 2
C U.S. STANDARD ATMOSPHERE UP TO 90 KILOMETERS. 3
C CALLING SEQUENCE... 4
C
C CALL ATMOS (ZFT, TM, SIGMA, RHO, THETA, DELTA, CA, AMU, K) 5
C ZFT = GEOMETRIC ALTITUDE (FEET) 7
C TM = MOLECULAR SCALE TEMPERATURE (DEGREES RANKINE) 8
C SIGMA = RATIO OF DENSITY TO THAT AT SEA LEVEL 9
C RHO = DENSITY (LB-SEC**2-FT**3) 10
C THETA = RATIO OF TEMPERATURE TO THAT AT SEA LEVEL 11
C DELTA = RATIO OF PRESSURE TO THAT AT SEA LEVEL 12
C CA = SPEED OF SOUND (FT/SEC) 13
C AMU = VISCOSITY COEFFICIENT (LB-SEC/FT**2) 14
C
C K = 1 NORMAL 15
C = 2 ALTITUDE LESS THAN -5000 METERS OR GREATERTHAN 90 KM 17
C = 3 FLOATING POINT OVERFLOW 18
C
C ALL DATA AND FUNDAMENTAL CONSTANTS ARE IN THE METRIC SYSTEM AS 20
C THESE QUANTITIES ARE DEFINED AS EXACT IN THIS SYSTEM. 21
C
C THE RADIUS OF THE EARTH (REFT59) IS THE VALUE ASSOCIATED WITH THE 23
C 1959 ARC ATMOSPHERE SO THAT PROGRAMS CURRENTLY USING THE LIBRARY 24
C ROUTINE WILL NOT REQUIRE ALTERATION TO USE THIS ROUTINE. 25
C DIMENSION HB(10),TM(10),DELTAB(10),ALM(10) 26
DATA(481), TM(10), DELTAB(10), ALM(10)
1 -5.0, 320.65, 1.75363E 00, -6.5, 28
2 0.0, 288.15, 1.00000E 00, -6.5, 29
3 11.0, 216.65, 2.23361E-01, 0.0, 30
4 20.0, 216.65, 5.40328E-02, 1.0, 31
5 32.0, 228.65, 8.56663E-03, 2.8, 32
6 47.0, 270.65, 1.09455E-03, 0.0, 33
C CONVERT GEOMETRIC ALTITUDE TO GEOPOTENTIAL ALTITUDE
HFT=(REFT59/(REFT59+ZFT))*ZFT

C CONVERT FTF AND ZFT TO KILOMETERS
Z=FTTJKM*ZFT
H=FTTJKM*HFT
K=1
TMZ=TM8B(2)
IF (H.LT.-5.0.OR.Z.GT.90.0) GO TO 7
DO 1 M=1,10
IF (H-HB(M)) 2,3,1
1 CONTINUE
GO TO 7
2 M=M-1
3 DELH=H-HB(M)
IF (ALM(M).LT.0.0) GO TO 4
TMK=TM8B(M)+ALM(M)*DELH
4 TMK=TM8B(M)

C GRADIENT IS NON ZERO, PAGE 10, EQUATION 1.2.10-(3)
DELTA=DELTAB(M)*((TM8B(M)/TMK)**(GZ*AMZ/(RSTAR*ALM(M))))
GO TO 5

C GRADIENT IS ZERO, PAGE 10, EQUATION 1.2.10-(4)
DELTA=DELTAB(M)*EXP(-GZ*AMZ*DELH/(RSTAR*TM8B(M)))

C CONVERSION TO ENGLISH UNITS
TM=1.8*TMK
RHD=R*4D0*SIGMA/GZENG
CA=CAZ*SQRT(THETA)
AMU=AMUZ*ALPHA/GZENG
CALL OVERFL (J)
GO TO (6,8),J
6 K=K+2
GO TO 8
7 K=2
8 RETURN
END

$IBFTC RAMS DECK
SUBROUTINE RAM (AM,ETAR)
IF (AM.GT.1.) GO TO 2
ETAR=1.
1 RETURN
2 IF (AM.GT.5.) GO TO 3
ETAR=1.0+0.075*(((AM-1.0)**1.35)
GO TO 1
3 ETAR=30.0/((AM**4)+935.0)
GO TO 1
END

55
CNF = P × NF × THETAD / (100 × THETA)
IF (IDES.EQ.1) WAFC = WAFC × P2 / THETA
IF (ZF.LT.0.) ZF = 0.
IF (ZF.GT.1.) ZF = 1.
CNFS = VF
CALL SEARCH (ZF, CNF, PRF, WAFC, ETAFC, C, NCI(1), NCN, PRX(1, 1), WACX(1, 1), ET)
1
IF (CNF-CNFS) × GT.0.0005 × CNF) MAPEDS = 1
IF (133.EQ.1 OR. IG0.EQ.2) WRITE (8, 11) CNFS, WAF, WLH(1G0)
WAF = WAFC × P2 / THETA
IF (IDES.EQ.1) GO TO 1
PRFCF = (PRFDS-1) / (PRF-1)
ETAFCF = ETAFDS / ETAF
WAFCF = WAFC / WAF
PRF = PRFCF + ETAFCF + WAFCF, T2DS
1
IF (PRF.EQ.(PRF-1) + 1.
ETAFCF = ETAFCF + ETAFCF + WAFCF, T2DS
IF (IDES.EQ.1) GO TO 1
IF (IDES.EQ.1) GO TO 3
IF (M3DE.EQ.2) GO TO 2
T4 = GUESS (3, Y1, Y2, PCNF, PCNFDS, WFB, WFBDS, Y7, Y8, T4DS)
PCNC = JESS (4, Y1, Y2, PCNF, PCNFDS, WFB, WFBDS, Y7, Y8, PCNCDS)
GO TO 6
2
IF (M3DE.EQ.1) GO TO 4
IF (M3DE.EQ.1) GO TO 5
3
T4 = GUESS (7, Y1, Y2, PCNF, PCNFDS, Y5, Y6, T2, T2DS, T4DS)
PCNC = JESS (5, Y4, T4DS, Y3, Y4, Y5, Y6, T2, T2DS, T4DS)
GO TO 6
4
T4 = GUESS (6, Y1, Y2, PCNC, PCNCDS, Y5, Y6, T2, T2DS, T4DS)
GO TO 5
5
PCNC = PCNCDS
T4 = T4DS
WFB = WFBDS
T2DS = T21
6
ZC = ZCDS
PCNCGU = PCNC
T4GU = T4
7
8
IF (M3DE.EQ.3) GO TO 9
IF (ABS((CNF-CNFS) × LE.0.001 × CNF)) GO TO 10
WRITE (8, 13) CNFS, CNF
9
PCNF = 100 × THETA × CNF / THETAD
CALL ERROR
10
CALL GAP
RETURN
C
11
IF (IDES.EQ.1) WRITE (8, 11)
If (IDES.EQ.1) WRITE (8, 11)
Write (8, 11) CNFS, CNF
12
CNF = P × NF × THETAD / (100 × THETA)
13
FORMAT (19HO* * * CNF OFF MAP, F10.4, 2X, A6, 11H* * * $$$$$$$)
FORMAT (11HOFAN DESIGN, 13X8H, PRFCF = E15.8, 8H ETAFCF = E15.8, 8H WA
1FCF = E15.8, 8H T2DS = E15.8)
FORMAT (10HOCNF WAS = E15.8, 11H AND NOW = E15.8, 24H CHECK PCNF I
INPUT$$$$$$)
END

57
$IBFTC COCOMPL DECK
SIBROJTINE COCOMPL

COMMOV / ALL
1WORD, IDES, JDES, KDES, MODE, INIT, IDUMP, IAMTP
2IGASMX, 1DBURN, 1APTBN, 1DCD, 1MCDC, 1DSHOC, 1MSHOC, 1NOZFLT,
3ITRYS, 1LOOPER, 1NOMAP, 1NUMMAP, 1MAPEDG, 1OMALL, 1ERR(6)

COMMOV /DESIGN/
1PCNFGJ, 1PCNCGU, 1T4GU, 1DUMO1, 1DUMO2, 1DELFG, 1DELFG, 1DELSFG,
2ZFDs, 1PCNFDs, 1PRFDs, 1ETFDS, 1WAFDS, 1PRCF, 1ETACF, 1WAFCF,
3ZCDS, 1PCNCDS, 1PRCDS, 1ETACDS, 1WACDS, 1PRCCF, 1ETACCF, 1WACCF,
4TFHPDS, 1CNHPDS, 1ETHPDS, 1THPCF, 1CNHPCF, 1ETHPCF, 1DHPCF, 1T2DS,
5TFLPDS, 1CNLPDS, 1ELLPDS, 1TLPCF, 1CNLPCF, 1EL LPCF, 1DLPCF, 1T2DS,
67CDO, 1WFDS, 1DTODS, 1EFTADS, 1W23DS, 1DPPDS, 1TDDUCF, 1ETADCF,
7877DS, 1WFADS, 1DTAFDS, 1EFTAADS, 1WG6CDS, 1DPAFDS, 1TDAFCF, 1ETAACF,
9A55, 1AS5, 1A6, 1A7, 1A8, 1A9, 1A28, 1A29,
8PS55, 1AM55, 1CVNDOZ, 1CVNDOZ, 1ASAV, 1A9SAV, 1A28SAV, 1A29SAV

COMMOV / FRONT/
11T, 1P1, 1H1, 1S1, 1T2, 1P2, 1H2, 1S2, 18
2T21, 1P21, 1H21, 1S21, 1T3, 1P3, 1H3, 1S3, 19
3T4, 1P4, 1H4, 1S4, 1T5, 1P5, 1H5, 1S5, 20
4T55, 1P55, 1H55, 1S55, 1BLF, 1BLF, 1BLD, 1BLOB, 21
5CNF, 1PRF, 1ETF, 1WAF, 1WAF, 1MA3, 1MG4, 1FAR4, 22
66CNPC, 1PCCF, 1ETAC, 1WACC, 1WAC, 1ETAB, 1PCOM, 1DUMF,
77CNHP, 1ETATHP, 1DHTCHP, 1DHTC, 1BLHP, 1WGM, 1FAR5, 1CS, 24
88CNLP, 1ETATLP, 1DHTCLP, 1DHTLP, 1BLLP, 1WGM5, 1FAR55, 1HPEXT, 25
99AM, 1ALTP, 1ETAP, 1ZF, 1PCNF, 1ZC, 1PCNC, 1WFB, 26
5TFHFP, 1TFDF, 1PBLF, 1PBLCD, 1PCBLD, 1PCBLCD, 1PCBLDP, 1PCBLDP

COMMOV /DUMMY/YZX(48)/BACK/YZX(72)
COMMOV/DUMMY/YZX(100)
COMMOV/COMP/CNX(15), 1PRX(15, 15), 1WAXC(15, 15), 1ETAAX(15, 15)
1NCN, 1PNT(15)

DIMENSION WLIH(2)
DATA 4dORD, 1DWHH/6HCOCOMP, 6H (LD) , 6H (HI) /
WORD=4ORD

THETA=SQRT(T21/518.668)
IF (IDES.EQ.1) THETAD=THETA
CNC=PCNC*THETAD/(100.*THETA)

IF (.NOT.FAN) WACC=(WAF-BLF)*THETA/P21
IF (IDES.EQ.1) WACDS=WACK*P21/THETA
IF (ZC.LT.0.) ZC=0.
IF (ZC.GT.1.) ZC=1.
CNCS=NC

IF (ISPOOL.EQ.1) GO TO 1
CALL SEARCH (ZC, CNC, PRCCF, WACC, ETAC, CXN(1), NCN, PRX(1, 1), WACX(1, 1), ET
1AX(1, 1), NPT(1, 15, 15, 15, 15))
GO TO 2

1PRC=1.
ETAC=1.
CNC=1.
PRCCF=1.

2CONTINUE
IF (MODE.EQ.1) GO TO 3
IF ((CNCS-CNCS).GT.0.0005*CNCS) MAPEDG=1
3IF (ZC.EQ.1.OR.1G0.EQ.2) WRITE (8, 7) CNCS, WLIH(10G)
WAC=WACX*P21/THETA
IF (IDES.EQ.1) GO TO 4
IF (ISPOOL.EQ.2) PRCCF=(PRCDS-1.)/PRC-1,
ETACCF=ETACDS/ETAC
IF (ISPOOL.EQ.1) ETACCF=1.0
WACCF=WAC+ADCFS/WACC
WRITE (6,8) PRCCF,ETACCIF, WACCF, T21DS
4
PRC=PRCCF*(PRC-1.)*1.
ETAC=ETACCIF+ETAC
WAC=WACCF+WAC
WACC=WACCF+WACC
IF (1.40T.FAN) ERR(5)=(WAF-WAC-BLF)/WAC
CALL TC <-- (PRC, ETAC, T21, H21, S21, P21, T3, H3, S3, P3)
IF (PCBLC.GT.0.) BLC=PCBLC+WAC
WA3=WAC-BLC
BLDU=PCBLDU*BLC
BLOB=PCBLB*BLC
BLLP=PCBLLP*BLC
IF IM3DEE.NE.1) GO TO 5
IF (ABS(CNC-CNCS).LT.0.001*CNCS) GO TO 6
WRITE (8,9) CNCS,CNC
CALL ERROR
RETURN
5
PCNC=100.*THETA*CNC/THETA
6
CALL T3COMB
RETURN
C
7
FORMAT (19H0* * CNC OFF MAP,F10.4,2X,A6,11H*#$$$$$$)
8
FORMAT (18HCOMPRESSOR DESIGN,6X,8H PRCCF=,E15.8,8H ETACCIF=,E15.8, 18H WACCF=,E15.8,8H T21DS=,E15.8) 87
9
FORMAT (10HOCNC WAS=,E15.8,11H AND NOW=,E15.8,24H CHECK PCNC I INPUT$$$$$)
END
$IBFTC COCOMB DECK
SUBROUTINE COCOMB
COMMOV / ALL/
1
1WORD IDES IDES IDES MODE INIT IDUMP IAMTP 3
21GASMX IDBURN IAFVND IDCD IMCD IDSHOC IMSHOC NOZFLT 4
3ITRYS LOOPER NOMAP NUMMAP MAPEDG TOLALL ERR(1) 5
COMMOV / DESIGN/
1PCNCFSJ PCNCFSJ PCNCFSJ PCNCFSJ PCNCFSJ PCNCFSJ 7
2ZFD5 PCNCFSJ PCNCFSJ PCNCFSJ PCNCFSJ PCNCFSJ PCNCFSJ 8
3ZCDS PCNCFSJ PCNCFSJ PCNCFSJ PCNCFSJ PCNCFSJ PCNCFSJ 9
4TFDS PCNCFSJ PCNCFSJ PCNCFSJ PCNCFSJ PCNCFSJ PCNCFSJ 10
5TFHPSD CNHPDS ETHPSD TFHPFS CNHPFC ETHPFCEHHPF CDHHPF 11
6TFLPDS CNLPDS TFLPDS CNLPDS TFLPDS CNLPDS TFLPDS 12
7TFDS WFD5 WFD5 WFD5 WFD5 WFD5 WFD5 13
8TFDS DTA4 DTA4 DTA4 DTA4 DTA4 DTA4 14
9ADF5 ADF5 ADF5 ADF5 ADF5 ADF5 15
$PS5 SS5 SS5 SS5 SS5 SS5 16
59
$TFFHP , TFLP , PCBLF , PCBLC , PCBLDU, PCBLOB, PCBLHP, PCBLLP
COMMON/SIDE/ZYX(48)/BACK/ZYX(72)
COMMON/DUMMYS/DIMMY(100)
LOGICAL ERR,FAN
EQUIVALENCE (FAN,DIMMY(11)), (ISPOOL,DIMMY(6))
COMMON/COMB/PSI(15), DELT(15,15), ETA(15,15), NPS,NPT(15)
DIMENSION Q(9), DUMB(15,15)
DATA 4WORD/6HCOCOMB/
WORD=4WORD
Q(2)=3
Q(3)=3
P3PSI=14.696*P3
WA3C=WA3*SORT(T3)/P3PSI
IF (IDES.EQ.1) WA3CDS=WA3C
DPCOM=DPCODS*(WA3C/WA3CDS)
IF (DPCOM.GT.1.0) DPCOM=1.0
P4=P3*(1.-DPCOM)
IF (T4.GT.4000.) T4=4000.
IF (T4.GE.1000.) GO TO 2
T4=1000.
IF (M3DE.EQ.1) MAPEDG=1
TDCO=T4+T3
IF (IDES.NE.1) GO TO 3
DTCODS=DTCO
DTCOCF=DTCODS/DTCO
DTCO=DTCOCF*DTCO
P3PSIN=P3PSI
CALL SEARCH (-1.,P3PSIN,DTCO,ETAB, DUMMY, PSI(1), NPS, DELT(1,1), ETA(1,1), DJMBO(1,1), NPT(1), 15, 15, IGO)
IF (ISO.EQ.7) CALL ERROR
IF (IDES.NE.1) GO TO 4
ETABC=ETABDS/ETAB
ETAB=ETABC*ETAB
HV=((((((12.4954317E-19*T4) -.2034116E-15)*T4+.2783643E-11)*T4+.2051
1501E-37)*T4-.2453116E-03)*T4-.9433296E-01)*T4+.1845537E+05
CALL THERMO (P4,H4,T4,XX1,XX2,0,0,0,0)
WFBX=WFBX/WFB
ERRW=(WFB-WFBX)/WFB
DIR=32RT(WFB/WFBX)
CALL AFQUIR (Q(1),T4, ERRW, 0., 20., 0.0001, DIR,T4, IGO)
GO TO (5,8,6), IGO
T4=TAF4
GO TO 1
CALL ERROR
WFB=WFBX
CALL THERMO (P4,H4,T4,S4,XX2,1,FAR4,0)
WG4=WFB+W3
IF (IDES.EQ.1) WRITE (6,10) WA3CDS,ETABC, DTCOCF
IF (ISPOOL.EQ.1) GO TO 9
CALL 5HPTB
RETURN
FAR5=FAR4
T4=T4
WG5=WG4
S5=S4
H5=H4
P5=P4
ETATHP=1.
CNHP=1.
DHPCF=1
ETHPCF=1
TFHPCF=1
THHCAL=0
DHTCHP=0
DHTC=3
CALL :JLPTB
RETURN

C
FORMAT (17HOCOMBUSTER DESIGN,7X8H WA3CDS=,E15.8,8H ETABCF=,E15.8
1H DTCCF=,E15.8)
END

$IBFTC COHPNTB DECK
SUBROJTIME COHPNTB
COMMON /ALL/
1WORD *IDES ,JOES,KDES,MODE,INIT,IDUMP,IAMTP,
2GASAV,1DDBURN,1AFTBYN,1DCD,1MCD,1DSHOC,1MSHOC,NOIFLT,
3JITRYS,LOOPER,NOMAP,NUMMAP,NMAPELG,TOLALL,ERR(6)
COMMON /DESIGN/
1PCNF6GJ,PCNCGU,T4GJ,DUMD1,DUMD2,DELFG,DELFN,DELSFC,
2ZDFS,PCNDFS,PRDFS,ETADFS,WAFDS,PRFCF,ETAFCF,WACF,
3ZDFS,PCNDFS,PRDFS,ETADFS,WAFDS,PRFCF,ETAFCF,WACF,
4T4DFS,WFDOS,TDTCDS,ETADFS,WAD3DS,PDFCDS,DTDCCS,ETABCF,
5TFHPDFS,CNHDFS,ETHPDFS,TFHPCF,CNHPCF,ETHPCF,DTCCF,T2DS,
6TFLPDFS,CLPDFS,ETLPDFS,TFLPCF,CMNLPCF,ETLPFC,DTLPCF,T21DS,
7TF0DFS,WFDOS,TDUDPS,ETAADS,WAD3DS,PDFUDS,TDUDCF,ETADC,
8TF0DFS,WFDOS,TDUDPS,ETAADS,WG6CDS,DPAFDS,DTAFCF,ETAACF,
COMMON /FRONT/
1T1,P1,H1,S1,T2,P2,H2,S2,
2T21,P21,H21,S21,T3,P3,H3,S3,
3T4,P4,H4,S4,T5,P5,H5,S5,
4T55,P55,H55,S55,BLF,BLC,BLDO,BLDB,
5CNF,PRF,ETAF,WAFC,WAF,WA3,WG4,FAR4,
6CNCF,PRC,ETAC,WACC,WAC,ETAB,DCPDA,DOMF,
7CNHPS,ETATHP,DHTCHP,DHCT,QLHP,WG5,FAR5,CS,
8CNLPS,ETATLF,DHTCLP,DHTF,BLLE,4G55,FAR55,HEPF,
9AM,LALP,ETAR,ZF,PCNF,LC,PCNC,WFB,
$PFFHP,TFFLPS,PCBLF,PCBLC,PCBLDO,PCBLHP,PCBLLO,
COMMON /SIDE/QXQ(48)/BACK/QXQ(72)
COMMON /DUMMYS/DUMMY(100)
EQUIVALENCE(TFFACT, DUMMY(12), CNACT,DUMMY(13), DHCAC,DUMMY(14)
COMMON /TERBH/DHSHV,TFHSHV,CHNVH,ETHSV,THPDS,
COMMON /HTURB/TFX(15),CNX(15,15),DTCHX(15,15),ETATX(15,15),
1NTFFS, VPTFFX(15)
DATA AWORD,WLO,WHI/6HOCOMPNTB,6H(L0),6H(HI) /
WORD=AWORD
IF (IDES.EQ.0) GO TO 1
CNPCHP=CNHPPS*SQRT(T4)/PCNC
CNPCHP=2*VHPCP*PCNC/SQRT(T4)
CNPCHP=CNHP
TFFHP=TFFHP
CALL SEARCH (-1., TFFHP,CNHP,DHTCHP,ETATHP,TFXX(15),NTFFS,CNX(1,1),D
1HTCHX(1,1),ETATX(1,1),NPTFFX(1,1),15,15,1G0)
IF (130.EQ.1.OR.1G0.EQ.11.OR.1G0.EQ.21) WRITE (8,8) TFFHP,S,WLO
IF (133.EQ.2.OR.1G0.EQ.12.OR.1G0.EQ.22) WRITE (8,8) TFFHP,S,WHI

61
IF (IGJ.EQ.10.OR.IGJ.EQ.11.OR.IGJ.EQ.12) WRITE (8,9) CNHPS,WLD
IF (IGJ.EQ.20.OR.IGJ.EQ.21.OR.IGJ.EQ.22) WRITE (8,9) CNHPS,WHI
IF (IGJ.NE.7) GO TO 2
CALL ERROR
RETURN
MAPG0=0
IF (ABS(TFFHPS-TFFHP).LE.0.001*TFFHPS) GO TO 3
MAPG0=1
IF (ABS(CNHPS-CNHP).GT.0.001*CNHPS) MAPG0=3
GO TO 4
3 IF (MAPG0.GT.0) CALL MAPBAC (1,MAPG3,TFFHPS,TFFHP,CNHPS,CNHP,PCVC,
1T4,MODE,NOMAP,NUMMAP)
IF (NOMAP.GT.0) RETURN
TFHCAL=MG0*SQRAT(T4)/(14.696*P4)
BTUEXT=0.706705*HPEXT
DHTCC=BTUEXT+WCAT/((H3-H21))/(WG4*T4)
IF (IDES.EQ.0) GO TO 5
TFHPCF=TFHPDS/TFHCAL
DHPCF=DHTCC/DHTCHP
ETHPCF=ETHPCF/ETATHP
WRITE (6,10) CNHPCF,TFHPCF,ETHPCF,DHPCF
5 TFHCAL=TFHPCF*TFHCAL
DHTCHP=DHPCF*DHTCHP
ETATHP=ETHPCF*ETATHP
DHTC=3*H3+H4+T5+H5+H5+P5
WRITE (6,11) TFHPCF,ETATHP,FAR4,H4,S4,P4,T5,H5,S5,P5)
IF (BLHP.LE.0.0) GO TO 6
FAR5=FAR5/(WA3+BLHP)
WG5=W54+BLHP
H5=(BLHP-H3+WG4*H5)/WG5
CALL THERMO (P5,H5,T5,S5,XX2,1,FAR5,1)
GO TO 7
6 FAR5=FAR4
WG5=W4
7 CALL COLPTB
RETURN
C C
8 FORMAT (19H00000,TFFHP OFF MAP,F10.4,2XA6,11H0000000000)
9 FORMAT (19H00000,CNHPS OFF MAP,F10.4,2XA6,11H0000000000)
10 FORMAT (2OH0,H,P, TURBINE DESIGN,5X7HCNHPCEF=E15.8,8H TFHPCF=E15.8,
1,8H ET-PCF=E15.8,8H DHPCF=E15.8)
END

$IBFTC COLPTB DECK
SUBROUTINE COLPTB
COMMON / ALL/
1WORD ,IDES,JDES,KDES,MDDE,INIT,IDUMP,IAMTP,
2IGASMK,IBURN,IAFTBN,IDCC,IMCD,IMSHOC,NDZFLT,
3ITRYS,LOOPER,NOMAP,NUMMAP,MAPEDG,TDALLY,ERR(6)
COMMON /DESIGN/
1PCNFGJ,PCNGGU,T4GU,DUMDI,DUMD2,DELFG,DELFN,DELSFC,
22FDS, PCNFDS, PRFD, ETAFD, WAFDS, PRGCF, ETAFCF, WAFCF, 8
32CDS, PCNCDS, PRCDS, ETAED, WACDS, PRGCF, ETAEDC, WACCF, 9
4240S, WFBDS, DTCQDS, ETAQDS, WACDS, DPCQDS, DTCQCF, ETAQCF, 10
52FDPS, CNPDPS, ETAPPS, TFHPCF, CNPDCF, ETAPCF, DHPDCF, T2DS, 11
62LPDCS, CNLPDCS, ETLPDCS, TLPDCF, CNLPCF, ETLPCF, DHLPCF, T2DS, 12
72TDS, WFDTS, DTDUDTS, ETAQDS, WACDS, DPDUDS, DTDUQF, ETAQCF, 13
82TDS, WAFDS, DTADFDS, ETAADS, WGCDS, DPAFDS, DTAFCF, ETAACF, 14
9255, A25, A6, A7, A8, A9, A28, A29, 15
95S5, AM5S, CVONOZ, CVONOZ, A85AV, A95AV, A255SVA, A295SAV, 16
COMMON / FRONT / 17
11T1, P1, H1, S1, T2, P2, H2, S2, 18
12T21, P21, H21, S21, T3, P3, H3, S3, 19
13T4, P4, H4, S4, T5, P5, H5, S5, 20
14T55, P55, H55, S55, BLF, BLF, BLFD, BLFD, 21
15CNF, PRF, ETAF, NAFCF, WAF, WA3, WA4, FA14, 22
16CNCF, RPC, ETAC, WACC, WAC, ETA8, DPCOM, DUMF, 23
17CNH, ETATHP, DHTCHP, DHTC, BLHP, W5G, FARS, CS, 24
18CNLP, ETATLP, DHTCPL, DHTF, BLLP, W5G5, FARS5, HPFEXT, 25
19AM, ALTP, ETAR, ZF, PCNF, ZC, PCNC, WFB, 26
20$TFHFP, TFLFP, PCBLF, PCBLP, PCBLDP, PCBLHP, PCBLLP, 27
COMMON/SIDE/QOQ(48)/BACK/QOQ(72) 28
COMMON/DUMMYS/DUMMY(100) 29
EQUIVALENCE (TFACT, DUMMY(15)), (CNACT, DUMMY(16)), (DHACT, DUMMY(17)) 30
LOGICAL, ERR, FAN 31
EQUIVALENCE (FAN, DUMMY(11)), (ISPOOL, DUMMY(6)) 32
COMMON/TERBL/OHLOPS, TFLOSV, CNLOSV, ETLOSV, DHLPOS 33
COMMON / LTURB / TFFX(15), CNX(15, 15), DHTCX(15, 15), ETATX(15, 15), 34
1NTFFS, VPTFF(15) 35
DATA AWORD, WLO, WHI/6HCL0PLTB, 6H (LO), 6H (HI) / 36
WORD=AWORD 37
IF (IDES.EQ.0) GO TO 1 38
CNLPF=CNLPDS*SQRT(T5)/PCNF 39
CNLP=CNLPF*PCNF/SQRT(T5) 40
CNLP=CNLP 41
TFFLP=TFLF 42
CALL SEARCH (-1), TFFLP, CNLP, DHTCLP, ETATLP, TFFX(1), NTFFS, CNX(1, 1), D 43
1HTCX(1, 1), ETATX(1, 1), VPTFF(15), TFFX(15, 15), IG 44
IF (I30.EQ.1.OR.IG0,EQ.11.OR.IG0,EQ.2) WRITE (8,10) TFFLP, WLO 45
IF (I30.EQ.2.OR.IG0,EQ.12.OR.IG0,EQ.2) WRITE (8,10) TFFLP, W4I 46
IF (I30.EQ.10.OR.IG0,EQ.11.OR.IG0,EQ.12) WRITE (8,11) CNLP, WLO 47
IF (I30.EQ.20.OR.IG0,EQ.21.OR.IG0,EQ.22) WRITE (8,11) CNLP, WHI 48
IF (I30.NE.7) GO TO 2 49
CALL ERR 50
RETURN 51
2 MAPGO=0 52
IF (ABS(TFFLP-TFFLP),LE.0.001*TFFLP) GO TO 3 53
MAPGO=1 54
IF (ABS(CNLP-CNLP),GT.0.001*CNLP) MAPGO=3 55
GO TO 4 56
3 MAPGO=2 57
IF (MAPGO,GT.0) CALL MAPBAC (2,MAPGO,TFFLP, CNLP, CNLP, PCNF, 58
1T4, MODE, NOMAP, NUMAP) 59
IF (NMAP.GT.0) RETJRN 60
TFLCAL=WG5*SQRT(T5)/(14.696*P5) 61
DHTCF=WAF*(H21-H2)/WG5*T5 62
IF (ISPOOL.EQ.2) GO TO 5 63
BTUEXT=0.706705*HPFEXT 64
DHTCF=(BTUEXT+WAF*(H21-H2))/(WG5-T5) 65
CONTINUE 66
IF (IDES.EQ.0) GO TO 6 67
TFLPCF=TFLPDS/TFLCAL 68
DHLPCF=DHTCF/DHTCPL 69
63
ETLP=ETLPDS/ETATLP
TFLC AL=TFLPCF*TFLCAL
DHTCLP=DHLPCF*DHTCLP
ETATLP=ET&PCF*ETATLP
DHTF=DHTCF*T5
TFFACT=TFLCAL/TFLPCF
CNACT=CNLP/CNLP CF
DHCACL=DHTCLP/DHLPCF
II=3
I2=4
IF [ISPOOL.NE.1] GO TO 7
I1=1
I2=2
IF [BLLP.LE.0.] GO TO 8
FAR55=IBALP/WA3+BLHP+BLPP)
W55=#G5+BLLP
H55=([BLLP*H5+WG5+H55]/WG5)
CALL T-TERMO (P55,H55,T55,S55,XX2,1,FAR55,1)
GO TO 9
FAR55=FAR5
WG55=#G5
9 CALL FRTOSD
RETURN
C
C
10 FORMAT (19H0****TFFLP OFF MAP,F10.4,2X,A6,11H*********)
11 FORMAT (19H0**** CNLP OFF MAP,F10.4,2X,A6,11H*********)
12 FORMAT (20HOL LP, TURBINE DESIGN,5X,HCNLPCF=,E15.8,8H TFLPCF=,E15.8
1,8H ETLP CF=,E15.8,8H DHLPCF=,E15.8)
END

$IBFTC CODUCT DECK
SUBROUTINE CODUCT
COMMON / ALL/
1WORD, IDES, JDES, KDES, MODE, INIT, IDUMP, IAMTP,
2IGASMX, IDB RN, IAFTBN, IDC D, IDMDC, IDMDC, IMDC, IDMDC, NDZFLT,
3ITRYS, ILOPER, NMAP, NUMMAP, MAPEDG, TOLALL, ERR(6)
COMMON /DESIGN/
1PCNFG, PCNGU, T4GU, DUMD1, DUMD2, DELFG, DELFN, DELSFC,
2ZFD S, PCNFS, PRFDS, ETA FDS, WAFDS, PRF CF, E TAF CF, WAFCF,
3ZCDS, PCNFDS, PRFDS, ETA CDS, WACDS, PRCC F, ETA CCF, WACC F,
4TFDS, WFBDS, DTCODS, ETA BDS, WACDS, DPCODS, DTCODC, ETA BCF,
5TFHPDS, CNHPDS, ETPHD S, TFPDS, CNHPCF, ETHPCF, DTHPCF, T2DS,
6TFLPDS, CNLPCF, TFLPCF, CNLPCF, TFLPCF, CNLPCF, DTLPCF, T2DS,
7T24DS, WFDODS, DT DODS, ETA DDS, WA23DS, DT DUDS, ETA TCF,
8TFDS, WFDODS, DTDODS, ETA DDS, WGD CDS, DPAFDS, DTACF, ETA ACF,
9A5S, A25, A6, A7, A8, A9, A28, A29,
$SP55, AM55, CVNDZ2, CMDNZ2, AB5AV, A9AV, A28AV, A29AV
COMMON/F RONT/XX(80)
COMMON / S I D E/
1P1, WAF, WAC, BLF, BLDU, H3, DUMS1, DUMS2,
2T21, P21, H21, S21, T23, P23, H23, S23,
3T24, P24, H24, S24, T25, P25, H25, S25,
44
**C**

**DRY LSS**

H23=(BLDU*H3+WAX*H21)/WAD

CALL THERMO (P23,H23,T23,S23,XX2,1,0,0,1)

WA23=WAD*SQRT(T23)/P23

IF (IDES.EQ.1) WA23DS=WA23

BYPASS=(WAF-WAC)/WAC

DPDUC=DPDUDS*(WA23/CWA23DS)

IF (DPDUC.GT.1.) DPDUC=1.0

P24=P23+1.0-DPDUC)

CALL >ROCOM (0.,T23,XX1,XX2,XX3,XX4,PHI23,XX6)

IF (L3ASMX.GT.0) IDBUR=0

AM24=AM23

TS24=T23*0.875

DO 2 I=1,15

CALL >ROCOM (0.,TS24,CS24,AK24,CP24,REX24,PHIS24,HS24)

V2A=AM24*CS24

HSCAL=H23-V2A**2/(2.*G*AJ)

DELHS=HSCAL-HS24

IF (ABS(DELHS)LE.0.001*HSCAL) GO TO 3

2 TS24=TS24+DELHS/CP24

GO TO 10

3 C1=P24*SQRT(G/(T23*AJ))CAPSF

IF (IDES.NE.1) GO TO 4

IF (G350.GT.0.) GO TO 4

ASTOA=((AK24+1.)/2.)*(AK24+1.)/(2.*(AK24-1.))*AM24*(1.+(AK24-11.)/2.*)*AM24**(2.2)***(AK24+1.)/(2.*(AK24-1.))**1*(AK24+1.)/(2.*(AK24-1.))**1

WA23CC=WAC23C/SQRT(518.69)

A24=1./ASTOA*WA23CC/EQWCR

GOGO=1.0

4 WQA=WAD/A24

WQAT=C1*SQRT(AK24/REX24)*AM24/(1.+(AK24-1.))*AM24**(2.2)**((AK24+1.)**(1.)/(2.*(AK24-1.)))

DIR=WQA/WQAT

E=W-(WQA-WQAT)/WQA

CALL AFQUR (Q11), AM24,E,0.,30.,0.001,DIR,AM24T,IGO)

GO TO 15,6,10,10

5 AM24=AM24T

IF (AM24.GT.1.0) AM24=0.5

GO TO 1

6 PS24=P24/EXP1(3PHI23-PHIS24)/REX24

IF (IDBUR.GT.0) GO TO 7
C*** NON-DJCT BURNING
T24=T23
WFD=0.
FAR24=0
GO TO 16
8 IF (T24.GT.4000.) T24=4000.
IF (T24.LT.T23) T24=T23
C*** DUCT JOURNING
RH042=CAPSF*PS24/(AJ*REX24*TS24)
PS42=PS24
V42=V24
Q(2)=3.
Q(3)=3.
C *** IF DESIRED, ENTER CALCULATIONS FOR ETAD HERE
HV=(((((1.5945317E-19*T24)-2034116E-01)*T24+.2783643E-11)*T24+.1845537E+05
1051501E-07)*T24-.2653116E-03)*T24-.9433296E-01)*T24+.1845537E+05
CALL THERMO (P24,HA,T24,XX1,XX2,0,0,0,0)
FAR24=(HA-H23)/(HV*ETAD)
IF (FAR24.LT.0.) FAR24=0.
WFDX=FAR24*WAD
IF (TDBC.BNE.2) GO TO 11
ERRW=(WFD-WFDX)/WFD
DIR=SQR(T(WFD/WFDX))
CALL AFQUIR (Q(1),T24,ERRW,0.,20.,0.0001,DIR,T24,T,IGO)
GO TO (9,12,10),IGO
T24=T244
GO TO 8
10 CALL ERROR
11 WFD=WFDX
12 CONTINUE
C *** MOMENT XM LOSS
WG24=WFD+WAD
CALL PRCOM (FAR24,T24,XX1,XX2,XX3,REX24,PHI24,H24)
RH024=CAPSF*PS24/(AJ*REX24*TS24)
V24=WG24/(RH024*A24)
Q(2)=3.
Q(3)=3.
PS24=PS42-0.01
13 RH024=WG24/(V24*A24)
HS24=H24-V24**2/(2.*G*AJ)
CALL THERMO (1.0,HS24,TS24,PHIS24,XX2,1,FAR24,1)
IF (TS24.GE.301.) GO TO 14
CALL THERMO (1.0,HS24,400.,PHIS24,XX2,1,FAR24,1)
V24=SQR(T(2.*G*AJ*(H24-HS24))
GO TO 13
14 PS24=RH024*AJ*REX24*TS24/CAPSF
PS24=PS42+(RH024*V42**2-RH024*V24**2)/(G*CAPSF)
DIR=SQR(T(ABS(PS24/PS24A)))
EP=(PS24-PS24A)/PS24
CALL AFQUIR (Q(1),V24,EP,0.,50.,0.001,DIR,V24,T,IGO)
V24=V24T
IF (V24.LT.25.) V24=25.
GO TO (13,15,10),IGO
15 P24=PS24*EXP((PH124-PHIS24)/REX24)
CALL PRCOM (FAR24,TS24,CS24,XX2,XX3,XX4,XX5,XX6)
AM24=V24*CS24
16 CALL THERMO (P24,H24,T24,S24,XX1,1,FAR24,0)
WG24=WFD+WAD
T25=T24
T25=T24
P25=P24
H25=H24
$IBFTC COMIX
SUBROIITINE COMIX
COMMON / ALL/
1WORD , IDES , JDES , KDES , MODE , INIT , IDUMP , IAMTP ,
2IGASMX , IDBURN , IAFTBN , IDCD , IMCD , IDSHOC , IMSHOC , NOZFLT ,
3ITRYS , LOOPDR , NDOMAP , NUMMAP , MAPEDG , TOLALL , ERR(6)
COMMON / DESIGN/
1PCNCFJ , PCNCGU , T4GU , DUMO1 , DUMO2 , DELFG , DELFN , DELSFC ,
2ZFOS , PCNFOS , PRFDOS , ETAFOS , WAFOS , PRFCF , ETAFCF , WAFCF ,
3ZCDS , PCNCDS , PRCDS , ETACDS , WACDS , PRCCF , ETACC , WACCF ,
4T4DOS , WFDOS , DTCODS , ETADOS , WA3CDS , DPDOS , DTQCF , ETAQCF ,
5FHPOS , CHNHPOS , ETHPOS , TFHPCF , CNHPCF , ETHPCF , DHNPCF , T2DS ,
6FLPDOS , CNLPDS , ETLPDS , TFLPCF , CNLPCF , ETLPCF , DHLPCF , T21DS ,
7T24OS , WFDOS , DTDUS , ETADDS , WA23DS , DPDUS , DTDUCF , ETAUCF ,
87T7DS , WFDADS , DTAFDS , ETAADS , WG6CDS , DPAFDS , DTAFCF , ETAACF ,
9A55 , A25 , A6 , A7 , A8 , A9 , A28 , A29 ,
10

S25=S24
AM25=AM24
IF (IGASMX.GT.0) GO TO 20
WORD=WORD2
A28SAV=A28
A29SAV=A29
NOZ=0
IDNOZ=0
IF (NJFLT.EQ.2 OR \ NOZFLT.EQ.3) NOZ=1
IF (ESES.EQ.1 OR \ IDBURN.GT.0 OR \ NOZ=EQ.1) IDNOZ=1
IF (IDBURN.EQ.1 OR \ NOZFLT.EQ.2 OR \ NOZFLT.EQ.3) IDCD=1
IF (IDBURN.EQ.0 AND \ NOZFLT.EQ.0) IDCD=0
IF (1)лин Од 17
CALL CONVRG (T25, H25, P25, S25, FAR24, WG24, P1, IDNOZ, A28, P25R, T28, H28,
1P28, S29, T28, PS28, V28, AM28, ICON)
GO TO (18, 18, 18, 10), ICON
17 CALL JINDIV (T25, H25, P25, S25, FAR24, WG24, P1, IDNOZ, A28, A29, P25R, T28,
1H28, P28, S28, T28, H29, P29, S29, T28, TS29, PS29, V29, VA29, AM28, AM29,
2ICON)
IDSHOC=ICON
GO TO (19, 19, 19, 10), ICON
18 T29=T28
H29=H28
P29=P28
S29=S28
TS29=TS28
PS29=PS28
V29=V29
AM29=AM28
A29=A28
IDSHOC=ICON+3
ERR(5)=(P25R-P25R) /P25R
IF (1)NOZ.EQ.1) WRITE (6, 21) A28, AM28, A29, AM29
20 CALL FASTBK
RETURN
19
C
C
21 FORMAT (19HPRODUCT NOZZLE DESIGN, 5X8H A28=, E15.8, 8H AM28=, E15.8
1, 8H A29=, E15.8, 8H AM29=, E15.8)
END
$PS55, A55, CV0NQZ, CVHMQJ, 28SAV, 89SAV, 2A85AV, 2A9SAV
COMMON/FONT/QO(80)/SIDE/QO(48)
COMMON BACK
1TS5, P55, H55, T55, T25, P25, H25, S25
2WBO, W955, FAR55, WFD, GF24, FAR24, P1, DUMB
3T6, P6, H6, S6, T7, H7, S7
4TB, P8, H8, S8, T9, H9, S9
5NG6, WFA, W97, FAR7, ETA, DPAFT, V55, V25
6PS6, V6, AM6, TS7, PS7, V7, AM7, AM25
7TS8, PS8, V8, AM6, TS9, PS9, V9, AM9
8VA, FRD, VJD, FGN, VJ, FGM, FGP, FGM
9FGM, FGP, WFT, WGT, FART, FG, WN, SFC
EQUIVALENCE (IF, QO(58)), (PCNF, QO(69))
COMMON/8MYS/8MMY(100)
COMMON/8DOPR/KKG0, PRFNEW, PRCNEW
DATA AWORD/6H COMIX/
DIMENSION QQ(9)
WORD=AWORD
AJ=778*26
CAPSF=2116*2170
G=32*174069
CALL PROCOM (FAR55, T55, XX1, XX2, XX3, XX4, PHI55, XX5)
CALL PROCOM (FAR24, T25, XX1, XX2, XX3, XX4, PHI25, XX5)
IF (S155, EQ, 1) GO TO 12
C *** CALCULATE A55 AND A25 WITH PS55=PS55
IF (PS55, EQ, 0.0) GO TO 3
TS55=(PS55/PS55)*0.286
DO 1 I=1, 15
CALL PROCOM (FAR55, TS55, CSS5, AK55, CP55, REX55, PHIS55, HS55)
PHIS=P*HS55-REX55=AL3G(PS55/PS55)
DELPHI=PHIS-PHIS55
IF (ABS(DELPHI), LE, 0.0001, PHIS) GO TO 6
1 TS55=TS55*EXP(4.0, DEPHI)
2 CALL ERROR
RETURN
3 TS55=0.875, T55
DO 4 I=1, 15
CALL PROCOM (FAR55, TS55, CSS5, AK55, CP55, REX55, PHIS55, HS55)
V55=A555*CSS5
HSCAL=H55-V55/2/I2*G, AJ
DELHS=HSCAL-HS55
IF (ABS(DELHS), LE, 0.0005, HSCAL) GO TO 5
4 TS55=TS55+DELHS/CP55
GO TO 2
5 PS55=PS55/EXP((PHI55-PHIS55)/REX55)
IF (PS55, GT, P25, AND, ID, EQ, 1, AND, IDASMX, GT, 0) GO TO 47
6 IF (H55, GT, HS55) GO TO 7
WRITE (8, 48) P55, PS55, T55, TS55, H55, HS55
CALL ERROR
7 V55=SQR(I2*G, AJ*H55-H555)
RHO=CAPSF*PS55/(AJ*REX55*TS55)
A55=W555/(RHO*V55)
AM55=Y555/CP55
IF (I55, ASMX, GT, 0) GO TO 8
WRITE (6, 49) A55, AM55
IF (I55, ASMX, EQ, -1) GO TO 35
IF (I55, ASMX, EQ, 0) GO TO 43
8 PS25=PS55
TS25=T25*(PS25/P25)*0.286
DO 9 I=1, 15
CALL PROCOM (FAR24, TS55, CS25, AK25, CP25, REX25, PHIS25, HS25)
PHIS=P*TS25-REX25=AL3G(PS25/PS25)
CALL PROCOM (FAR24, TS25, CS25, AK25, CP25, REX25, PHI25, HS25)
V25=AM25*CS25
HSCAL=H25-V25**2/(2.*G*A1)
DELHS=HSCAL-HS25
IF (ABS(DELHS).LE.0.0005*HSCAL) GO TO 25
TS25=TS25+DELHS/CP25
GO TO 2
WQAT=1.*SORT(AK25/REX25)+AM25%(1.*{AK25+1.} AM25%2/2.)*{AK25+1.}
1./2.={AK25-1.})
AMX=AM25
ILOGO=1
GO TO 16
WG6=W24*W55
ERR{S}=PS25-PS55/PS25
WF6=WF0+WF6
FAR6=WF6/(WG6-WF6)
H6=({W24*H25+WG55*H55})/WG6
CALL THERMO (1., T6, PHI6, AMX, 1., FAR6, 1.)
C=PS55+AM55%1.+AK55*AM55%2+PS25*A25%(1.+AK25*AM25%2)
TS6=0.933*T6
DO 32 I=1,15
CALL PROCOM (FAR6, TS6, CS6, AK6, CP6, REX6, PHI6, HS6)
C2=WG6-SQRT(AJ*REX6*TS6/(AK6*G))
C3=C2/(CAPSF*C1)
C4=(AK6-1.)/2.-{C3*AK6}**2
C5=1.-2.*AK6*C3**2
C6=C5*2+4.*C4*C3**2
IF (C5) 28, 29, 30
CALL ERROR
RETURN
AM62G=-C5/(2.*C4)
GO TO 31
AM62G={SORT(C6)-C5}%(2.*C4)
IF (AM62G.LE.0.) GO TO 28
AM66G=SQRT(AM62G)
V6=AM55*CS6
HSCAL=H6-V6**2/(2.*G*A1)
DELHS=HSCAL-HS6
IF (ABS(DELHS).LE.0.0005*HSCAL) GO TO 33
TS6=TS5+DELHS/CP6
GO TO 28
IF (IGASMX.GT.0) GO TO 34
A6G=A25*A55
C7=SQRT(1.+{AK6-1.}*AM62G/2.)
PS6=C2/(CAPSF*A6G*46G*C7)
P6=PS5*EXP(PHI6-PHI56/REX6)
CALL THERMO (P6, H6, T6, S6, XX1, 1., FAR6, 0.)
S6AVE=WG4+S55*WG55%S55)/WG6
IF (S5.9*6AVE) GO TO 35
S6=S6AVE
P6=EXP(AMX*{PHI6-S6])%1.986375
IF (IGASMX.EQ.1) GO TO 45
IF (IGASMX.EQ.-1) GO TO 36
IF (IGASMX.EQ.2) GO TO 37
T6=T55
P6=P55
H6=H55
S6=S55  
WG6=W55  
PS6=P55  
FAR6=FAR55  
AK6=AK55

37 IF (I6S.EQ.0) GO TD 40

*** CALCULATES A6 AS A FUNCTION OF INPUT AM6

T6=S5/(1.0+((AK6-1.0)/2.0)*AM6**2))

DO 38 J=1,15

AK6=AK6

CALL PROCOM (FAR6,TS6,CS6,AK6,CP6,REX6,PHIS6,HS6)

V6=AM5*CS6

DELA6=AK6-AK6

IF (ABS(DELA6)**LE.0.0005*AK6) GO TD 39

38

T6=T5/(1.0+((AK6-1.0)/2.0)*AM6**2))

GO TO 28

39 PS6=P5/(1.0+((AK6-1.0)/2.0)*AM6**2))**((AK6/(AK6-1.0))

AM6AB=AM6

RHO=CAPSF*PS6/(AJ*REX6*TS6)

A6=WG5/(RHO*V6)

WRITE (6,54) A6

GO TO 46

C CALCULATES M6=F(A6DESIGN)

GO TO 40

TS6P=TS6/(1.0+((AK6-1.0)/2.0)*AM6ABD**2))

DO 41 I=1,15

CALL PROCOM (FAR6,TS6P,CS6,AK6,CP6,REX6,PHIS6,HS6)

PS6P=PS6*TS6P/TS6)**((AK6/(AK6-1.0))

RHO6=CAPSF*PS6P/(AJ*REX6*TS6P)

V6=SQR(T2.0*AJ*(H6-HS6))

IF (I5-HS6)**LT.0.0) GO TD 44

A6P=WG6/(RHO6*V6)

DELA6=A6P-A6

V6=WG5/(RHO6*A6)

AM6=V5/CS6

AM62=AM6**2

IF (ABS(DELA6)**LE.0.002*A6) GO TD 42

41

TS6P=TS5/(1.0+((AK6-1.0)/2.0)*AM62)

GO TO 28

42

TS6=TS6P

PS6=PS55

GO TO 46

43 T6=T55

P6=P55

H6=H55

S6=S55

WG6=WG5

PS6=PS55

V6=V55

AM6=AM55

IF (1GASMX.EQ.0) A6=A55

GO TO 46

44 WRITE (6,55) H6,HS6

GO TO 28

45 AM62=AM62G

AM6=AM6G

A6=A25*A55

46 CALL 33AFBN

RETURN

47 KKGO=1

OPRDS=PRFDS*PRCDS

PRFNEW=PRFDS*PS55/P25*1.02

PRCNEW=OPRDS/PRFNEW

70
CALL ENGBAL
RETURN 265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281

$IBFTC COAFBY DECK
SUBROJTINE COAFBN
COMMON / ALL/
1WORD , IDES , JDES , KDES , MODE , INIT , IDUMP , IAMTP ,
2IGASMX , IDBURN , IAFTB4 , IDOD , IDSHOC , IMSHOC , IDQFLT ,
3ITRYS , LOOPER , NOMAP , NUMMAP , MAPEDG , TOLALL , ERR/6
COMMON / DESIGN/
1PCNFGJ , PCNCGU , T4GU , DUMDI , DUMD2 , DELFG , DELFN , DELSFC ,
2ZFDS , PCNFD5 , PRFDS , ETAFDS , WAFDS , PRFCE , ETAFCF , WAFCF ,
3ZCDS , PCNCD5 , PRCDS , ETAOD5 , WACDS , PRCCE , ETAOCF , WACCF ,
4T4DS , WBFD5 , DTCD5 , ETAOD5 , WABDS , DPACDS , DTCACF , ETAACF ,
5TFHPP5 , CNHPDS , ETDP5 , TCHPP5 , CNHPCF , ETDCP5 , DTHPCF , TDP2O ,
6TFLPP5 , CNLPP5 , ETLP5 , TFLPCF , CNLPCF , ETLP5 , DHLPCF , TD2S ,
7T24DS , WBFD5 , DDUDS , ETAOD5 , WABDS , DPUDS , DTDUCF , ETAUCF ,
8T7DS , WAFDS , DATFDS , ETAOD5 , W6G5DS , JPAFDS , DTAFCF , ETAACF ,
9A55 , A25 , A6 , A7 , A8 , A9 , A28 , A29 ,
$PS55 , AM55 , CVNODZ , CVNODZ , AB5AV , A95AV , A5B5AV , A595AV ,
COMMON / FRONT/ QXQ(80) / SIDE/ QYQ(48) / COMMON / BACK/
1T55 , P55 , H55 , S55 , T2S , P2S , H2S , S2S ,
2WFB , W655 , FARS5 , WFD , W7G5 , FAR7 , ETA5 , DPAFT , V5S , V5S ,
3T6 , P6 , H6 , S6 , T7 , P7 , H7 , S7 ,
4T8 , P8 , H8 , S8 , T9 , P9 , H9 , S9 ,
5W66 , WFA , W7G , FAR8 , ETA9 , DPAFT , V55 , V55 ,
6P56 , V6 , AM6 , T57 , PS7 , V7 , AM7 , AM5 ,
7T58 , P58 , V8 , AM8 , TS9 , PS9 , V9 , AM9 ,
8VA , FRD , VJD , FGMDS , VMJ , FGMN , FGPD , FGPM ,
9FGM , FGP , WFT , WGT , FART , FG , FN , SFC ,
COMMON / DUMMYS / DUMMY(100)
EQUIVALENCE (P65AV , DUMMY(7)) , (AM6DSV , DUMMY(8)) , (ETAASV , DUMMY(9)) ,
1IFART75V , DUMMY(10)) .
DIMENSION Q(9)
DATA AWORD/6HCOAFBN/
WORD=AWORD
Q(12)=J.
Q(13)=J.
AJ=77B.26
CAPSF=2116.2170
G=32.174049
C*** P6DS AND AMDS ARE SET FOR GENERALIZATION OF AFTERBURNER

71
**EFFICIENCY MAP GENERALIZATION**

IF (IDES.EQ.1) P6DS=P6*14.696
IF (IDES.EQ.1) AM6DS=AM6
WF6=WFB
IF (TASSMX.GT.0) WF6=WF6+WFD
WA6=WF6-WF6

**DRY LSSS**

WG6C=#36*SQRT(T6)/P6
IF (IDES.EQ.1) WG6CDS=WG6C
DPAFT=3*PASFDS*(WG6C/WG6CDS)
IF (DPAFT.GT.1.) DPAFT=1.
P7=P6*(1.-DPAFT)
A7=AM6

**D**

CALL PROCOM (FAR6,T6,XX1,XX2,XX3,XX4,PHI6,XX6)
WQA=W5/A7
C1=P7*SQRT(G/(T6*AJ))*CPSF
AM7=AM6
TS7=0.875*T6

DO 2 I=1,15
CALL PROCOM (FAR6,T5,C7,A7,CP7,REX7,PHI7,HS7)
V7=AM7*CS7
HSCAL=46-V7**2/(2.*G*AJ)
DELHS=HSCAL-HS7
IF (ABS(DELHS).LE.0.0005*HSCAL) GO TO 3

2 TS7=TS7+DELHS/CP7
GO TO 14

3 WQA=C1*SQRT((AK7/REX7)*AM7/(1.+(AK7-1.)*AM7**2/2.)**((AK7+1.)/(2.*
1*(AK7-1.))))
D1R=WQA/WQAT
E1W=(WQA-WQAT)/WQA
CALL AQOUNR (Q(1),AM7,E1W,40,4001,D1R,AM7,100)
GO TO (4,5,14,100)

4 AM7=AM7
IF (AM7.GE.1.0) AM7=0.9
GO TO 1

5 PS7=P7/EXP((PHI6-PHI7)/REX7)
IF (IAFTBN.GT.0) GO TO 7

**NON-AFTERBURNING**

6 T7=T6
WFA=0.0
FAR7=FAR6
WG7=W5
IF (IDES.EQ.1.AND.T7DS.NE.0.) GO TO 7
GO TO 20

**AFTERBURNING**

7 IF (IAFTBN.EQ.2) T7=T6+2000.
IF (IDES.EQ.1) T7=T7DS
IF (T7.LE.T6) GO TO 6
RM06S=CPSF*PS7/(AJ*REX7*TS7)
PS6S=PS7
V6S=V7
Q(2)=3.
Q(3)=3.

8 IF (T7.GT.4000.) T7=4000.
HV=HINV(-12.94317E-19*T7-.2034116E-15)*T7+.2783643E-11*T7+.2051
1501E-37*T4-.2451115E-03*T4-.9433296E-01*T4+.1845537E+05
CALL THERMO (P7,HA,T7,XX1,XX2,1,FAR6,0)

**TO ALTER DESIGN ABETAA MAP FROM GENERAL TO SPECIFIC MAP**

CALL ETAAB (0.,0.,0.,0.,ETAADS,ETAASV,P6DS,P6DSAV,AM6DS,AM6DSV,IDE
1S,FAR7DS,FAR7SV)

72
T7=T6
GO TO 20
9 P6GS=5*14.696
FARTGS=(HA-H6)/(HV*ETAADS)
DO 10 II=1,15
CALL EAB (FARTGS,AM6,P6GS,ETAA,ETAADS,ETAAVS,P6DS,P6DSAV,AM5DS,A
1M6DSV,IDES,FART7DS,FART7SV)
FART7=(4A-H6)/(HV*ETAA)
DELFA7=ABS(FART7-FARTGS)
IF (DELFA7.LE.0.01*FART7) GO TO 11
10 FARTGS=FART7
CONTINUE
IF (FART7.GT.0.0) GO TO 12
CALL ERROR
12 WFA=FART*WG6
IF (IARFBN.EQ.1) GO TO 15
ERRW=(WF-A-WFA)/WFA
DIR=SQRT(WFA/WFA)
CALL AFQUIR (Q(1),T7,ERRW,0.,30.,1.,0005.,DIR,T77,1G0)
GO TO (13,16,14,1G0)
13 T7=T7T
GO TO 9
14 CALL ERROR
15 WFA=WFA
16 FART7=(WF6+WFA)/WA6
WG7=WG6+WFA
C *** MOMENTUM LOSS
CALL PROCOM (FART7,T7,XX1,XX2,XX3,REX7,PHI7,H7)
RH07=APSF*P7/(A7*REX7*T7)
V7=WG7/(RH07*A7)
Q(2)=0.
Q(3)=0.
PS7=PS65-0.01
17 RH07=gG7/(V7*A7)
HS7=H7-V7**2/(2.*G*A7)
CALL THERMO (1.0,HS7,TS7,PHIS7,XX2,1,FAR7,1)
IF (TS7.GE.301.) GO TO 18
CALL THERMO (1.0,HS7,400.0,PHIS7,XX2,1,FAR7,0)
V7=SQRT(2.*G*A7*(H7-HS7))
GO TO 17
18 PS7=RH07*A7*REX7*TS7/CAPSF
PS7A=P65+(RH65*W65**2-RH07*V7**2)/(G*CAPSF)
DIR=SQRT(1ABS(PS7/PS7A))
EP=(PS7-P5TA)/PS7
CALL AFQUIR (Q(1),V7,EP,0.,50.,1.,001.,DIR,V77,1G0)
V7=V7T
IF (V7.LT.0.1) V7=100.
GO TO (17,19,14,1G0)
19 P7=PS7*EXPI(PH17-PHIS7)*REX7)
CALL PROCOM (FAR7,TS7,CS7,XX2,XX3,XX4,XX5,XX6)
AM7=V7/CS7
20 CALL THERMO (P7,H7,T7,7,XX2,1,FAR7,0)
IF (ID2S.EQ.1) WRITE (6,21) WG6CDS
CALL COMNOZ
RETURN
C
C
C
21 FORMAT (19H0AFTERBURNER DESIGN,5X8H WG6CDS=,E15.8)
END
```
$IBFTC FRBOSD DECK
SUBROJTIME FRBOSD
COMMDOV/ALL/XX(28)/DESIGN/YY(80)
COMMDOV/FRTOSD/FRONT/ 3
1T1   +P1   +H1   +S1   +T2   +P2   +H2   +S2
2T2   +P21  +H21  +S21  +T3   +P3   +H3   +S3
3T4   +P4   +H4   +S4   +T5   +P5   +H5   +S5
4T55  +P55  +H55  +S55  +BLF  +BLC  +BLDU  +BLOB
5CNF  +PRF  +ETA  +WAF  +WF  +WA3  +WG4  +FAR4
6CNF  +PRF  +ETA  +WAC  +WAC  +ETAB  +DPCOM  +DUMF
7CNHP  +ETART  +HTCHP  +HTC  +BLHP  +WG5  +FAR5  +CS
8CNLP  +ETATLP  +HTCL  +HTF  +BLLP  +WG55  +FAR55  +PEXT
9AM   +ALTP  +ETAR  +WF  +PCNF  +ZC  +PCNC  +WFB
$TFFHP  +TFFLP  +PCBLF  +PCBLD  +PCBLU  +PCBLD  +PCBLH  +PCBLLP
COMMDOV/SIDE/ 14
1XP1  +XWAF  +XWAC  +XBLF  +XBLD  +XH3  +DUMS1  +DUMS2
2XT2  +XAF  +XH21  +XS21  +T23  +P23  +H23  +S23
4T28  +P28  +H28  +S28  +T29  +P29  +H29  +S29
5WAD  +WFD  +WG4  +FAR24  +ETAD  +DPDU  +BYPASS  +DUMS3
6TS28 +PS28  +V28  +AM28  +TS29  +PS29  +V29  +AM29
COMMDOV/BACK/ZZ(72)
COMMDOV/DUMMYS/DUMMYY(100)
LOGICAL ERRORS, FARM
EQUIVALENCE {FAN, DUMMYY(11)}, {ISPOOL, DUMMYS(6)}
XP1=P1
XWAF=WHF
XWAC=MAC
XBLF=BLF
XBLD=BLD
XH3=H3
XT21=T21
XP21=P21
XH21=H21
XS21=S21
IF (FAN) CALL CNDUCT
IF (FAN) RETURN
CALL FASTBK
RETURN
END

$IBFTC FASTBC DECK
SUBROJTIME FASTBK
COMMDOV/ALL/XX(28)/DESIGN/YY(80)
COMMDOV/FRTOSD/FRONT/ 3
1T1   +P1   +H1   +S1   +T2   +P2   +H2   +S2
2T2   +P21  +H21  +S21  +T3   +P3   +H3   +S3
3T4   +P4   +H4   +S4   +T5   +P5   +H5   +S5
4T55  +P55  +H55  +S55  +BLF  +BLC  +BLDU  +BLOB
5CNF  +PRF  +ETA  +WAF  +WF  +WA3  +WG4  +FAR4
6CNF  +PRF  +ETA  +WAC  +WAC  +ETAB  +DPCOM  +DUMF
7CNHP  +ETART  +HTCHP  +HTC  +BLHP  +WG5  +FAR5  +CS
8CNLP  +ETATLP  +HTCL  +HTF  +BLLP  +WG55  +FAR55  +PEXT
9AM   +ALTP  +ETAR  +WF  +PCNF  +ZC  +PCNC  +WFB
$TFFHP  +TFFLP  +PCBLF  +PCBLD  +PCBLU  +PCBLD  +PCBLH  +PCBLLP
COMMDOV/SIDE/ 14
1XP1  +XWAF  +XWAC  +XBLF  +XBLD  +XH3  +DUMS1  +DUMS2
2XT2  +XAF  +XH21  +XS21  +T23  +P23  +H23  +S23
```
$1BFTC COMNOZ DECK
SUBROJTIME COMNOZ
COMNOZ / ALL/
1WORD , IDES , JOES , KDES , MODE , INIT , IDUMP ,IAMTP ,
2IGASMK , IDBURN , IDCTC , IDCD , IMCD , ODSHOC , IMSHOC , DOZFLT ,
3ITRYS , LOOPER , NDMAP , NUMMAP , MAPEDG , TOLALL , ERR(6)
COMNOZ / DESIGN/
1PCNFGJ , PCNCGU , TG4U , DUMD1 , DUMD2 , DELFG , DELFN , DELSFC ,
27FDS , PCNFDS , PRFDS , ETAFDS , WAFDS , PRPCF , ETAECF , WACF ,
32COS , PCNCDS , PRCD , ETCODS , WACDS , PRPCF , ETAECF , WACCF ,
4T4OS , WFBDOS , DTCDOS , ETAODS , WACDOS , PRCDF , ETAECF , WACCF ,
5T5HPODS , CNPCHD , ETQPDOS , TFPCF , CHPVER , ETQPCF , DHPCF , T2DS ,
6TFLPODS , CNLPODS , TLFPCF , CNLPCF , ETLPCF , DLPF , T21DS ,
7T7FODS , WFDO , DTDUDS , ETAODS , WAZD , PDUS , DTUCF , ETAECF ,
8T87DS , WAFDS , DTAFDS , ETAODS , WACDS , PDAD , DTAFCF , ETAECF ,
9A55 , A25 , A6 , A7 , A8 , A9 , A28 , A29 ,
$PS55, AM55, CVNNOZ, CVMMN, A8SAV, A8SAV, A8SAV, A29SAV

COMMON/FRONT/QXQ(80)/SIDE/QXQ(48)
EQUIVALENCE (AM, QXQ(15))
COMMON / BACK/

15, 17, 15, 19, 23, 21, 22, 23, 34, 25, 26, 27, 28, 29

LOGICAL ERRER=, FAN
EQUIVALENCE (FIN, DUMMY(11)), (ISPOOL, DUMMY(6))
DATA A, ORD, 6HMNOZZL/
WORD=4, ORD
A8SAV=A8
A9SAV=A9
NOZ=0
IMNOZ=0
IF (NOZFLT.EQ.0 AND NOZFLT.EQ.0) NOZ=1
IF (NOZFLT.EQ.0 OR NOZFLT.EQ.0) NOZ=1
IF (1DES.EQ.1 OR IAFTBN.GT.0 OR NOZM.EQ.1) IMNOZ=1
IF (IAFTBN.EQ.0 AND NOZFLT.EQ.0) IMCDO=0
IF (IAFTBN.EQ.1 OR NOZFLT.EQ.1 OR NOZFLT.EQ.1) IMCDO=1
IF (IMCDO.EQ.0) GO TO 1
CALL JNVNG (7, H7, P7, S7, FAR7, WG7, P1, INMVOZ, AB, P7R, T8, H8, P8, S8, TS8,
1PS8, V9, AM8, ICON)
GO TO (3, 3, 3, 2), ICON
1
CALL JNDIV (7, H7, P7, S7, FAR7, WG7, P1, INMVOZ, AB, A9, P7R, T8, H8, P8, S8, T
19, H9, P9, S9, TS9, TS9, PS9, PS9, PS9, V9, AM9, AM9, AM9, ICON)
IMSHOC=ICON
GO TO (4, 4, 4, 2), ICON
2
CALL ERROR
3
T9=T8
H9=H8
P9=P8
S9=S8
TS9=TS8
PS9=PS9
V9=V8
AM9=AM8
A9=A8
IMSHOC=ICON+3
4
ERR(5)=(P7R-P7)/P7R
IF (ISPOOL.EQ.1) ERR(3)=ERR(6)
IF (1MNOZ.EQ.1) WRITE (6, 5) A8, AM8, AM9, AM9
RETURN
C
C
5
FORMAT (14HONOZZLE DESIGN, 10X8H, A8=, E15.8, 8H, AM8=, E15.8, 8H
1 A9=, E15.8, 8H, AM9=, E15.8
END
WRITE (6,5) (TRASH4(I),I=1,72)
WRITE (6,4)
WRITE (6,7) LOOPER
IF (IJMP.EQ.0) GO TO 1
WRITE (6,6)
CALL SYG (2)
RETURN
1 CALL E4GBAL
RETURN
C
C
2 FORMAT (28H0AN ERROR HAS BEEN FOUND IN ,A6)
3 FORMAT (1HO,A6,9X,5E15.6,I4)
4 FORMAT (2HO )
5 FORMAT (1HO8E15.6)
6 FORMAT (1HL)
7 FORMAT (25H0FAILED TO CONVERGE AFTER,I4,6H LOOPS)
END

$IBFTC SYGS
DECK
SUBROJTINE SYG (ICON)
DIMENSION WORD(132)
DATA 3VEDOL/6H$ /GO TO (1,2),ICON
1 END FILE 8
REWIND 8
RETURN
C
TERMINATE THE FILE
2 WRITE (8,10)
END FILE 8
REWIND 8
RETURN
C
READ RECORD
3 READ (8,11) (WORD(I),I=1,132)
C
CHECK FOR 12 LEADING DOLLAR SIGNS
DO 4 I=1,12
IF (WORD(I)-ONEDOL) 5,4
4 CONTINUE
RETURN
C
CHECK FOR 6 TRAILING DOLLAR SIGNS
5 DO 8 I=1,132
I=I
IF (WORD(I)-ONEDOL) 8,6,8
K=I+5
DO 7 J=I,K
IF (WORD(J)-ONEDOL) 8,7,8
7 CONTINUE
GO TO 9
8 CONTINUE
WRITE (6,12)
RETURN
C
PRINT LINE
9 I=I-1
WRITE (6,11) (WORD(M),M=1,I)
GO TO 3
C
C
10 FORMAT (12H$$$$$$$$$$$$$$)
11 FORMAT (132A1)
12 FORMAT (1HO,12HERROR IN SYG)
END
<table>
<thead>
<tr>
<th>Line</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>$IBFTC PERF for DECK</td>
</tr>
<tr>
<td>2</td>
<td>SUBROJTINE PERF</td>
</tr>
<tr>
<td>3</td>
<td>COMMOV / ALL/</td>
</tr>
<tr>
<td>4</td>
<td>1WORD, IDES, JDES, KDES, MODE, INIT, IDUMP, IAMTP,</td>
</tr>
<tr>
<td>5</td>
<td>21GASMK, IBDURN, IIFTBN, IDC0, IMCD, IODMC, IM2OHC, IM2OH3, NOZL,</td>
</tr>
<tr>
<td>6</td>
<td>3ITRYS, LOPPER, NOMAP, NUMMAP, MAPEDG, TOLALL, ERR(6)</td>
</tr>
<tr>
<td>7</td>
<td>COMMOV / DESIGN/</td>
</tr>
<tr>
<td>8</td>
<td>1PCNF0J, PCNC0U, T4G0, DUMO1, DUMO2, DELF0G, DELFIT, DELSF,</td>
</tr>
<tr>
<td>9</td>
<td>2CFDS, PCNFD0S, PRFDS, ETAFD0S, WAFDS, PRFC0F, ETAFC0F, WAF0C0F,</td>
</tr>
<tr>
<td>10</td>
<td>3ICDS, PCNCDS0, PRCD0S, ETA0CD0S, WAC0S, PRFC0F, ETAFC0F, WAC0F,</td>
</tr>
<tr>
<td>11</td>
<td>4T4D0S, WFBD0S, DTC0DS, ETA0BD0S, WAD03, PRCD0S, DTCP0F, ETA0CF, ETABCF,</td>
</tr>
<tr>
<td>12</td>
<td>5TFH0PS, CNH0PS, ETH0PS, TFH0PCF, CNH0PCF, ETH0PCF, DHH0PCF, T20S,</td>
</tr>
<tr>
<td>13</td>
<td>6TFLPS0, CNL0PS0, ETL0PS0, CNL0PCF0, ETL0PCF0, DHL0PCF0, T210S,</td>
</tr>
<tr>
<td>14</td>
<td>7T20DS, WFBD0S, DTD0US, ETA0DS, WAD03, PD0UD0S, DTD0CF, ETA0CF,</td>
</tr>
<tr>
<td>15</td>
<td>8T70DS, WFAD0S, ETA0DS, ETA0DS, W6G0CD0S, PR0FDS0, ETA0CF, ETA0CF,</td>
</tr>
<tr>
<td>16</td>
<td>9A55, A25, A6, A7, A8, A9, A28, A29,</td>
</tr>
<tr>
<td>17</td>
<td>9PS55, AM55, CVN0D0Z, CVN0D0Z, A8SAV, A9SAV, A28SAV, A29SAV</td>
</tr>
<tr>
<td>18</td>
<td>COMMOV / FRONT/</td>
</tr>
<tr>
<td>19</td>
<td>1T1, P1, H1, S1, T2, P2, H2, S2,</td>
</tr>
<tr>
<td>20</td>
<td>2T21, P21, H21, S21, T3, P3, H3, S3,</td>
</tr>
<tr>
<td>21</td>
<td>3T4, P4, H4, S4, T5, P5, H5, S5,</td>
</tr>
<tr>
<td>22</td>
<td>4T55, P55, H55, S55, BLF, BL0C, BLDU, BLOB,</td>
</tr>
<tr>
<td>23</td>
<td>5CNF, PRF, ETA0F, WAF0C0F, WAF, W0A3, W0G4, FAR04,</td>
</tr>
<tr>
<td>24</td>
<td>6CNC, PRC, ETA0C, WAC0S, WAC0F, ETA0B, DPCM0, W0MF,</td>
</tr>
<tr>
<td>25</td>
<td>7CNHP, ETA0H0P0, DHT0CHP0, DHTC, BL0HP0, AG5, FAR5, CS,</td>
</tr>
<tr>
<td>26</td>
<td>8CN0LP, ETA0LP0, DHT0CLP0, DHTF, BBL0P0, W0G55, FAR55, HPEX0T,</td>
</tr>
<tr>
<td>27</td>
<td>9AM, ALTP, ETA0R0Z, ZF, PCNF, ZC, PCNC, W0BF,</td>
</tr>
<tr>
<td>28</td>
<td>$TFHFP, TFFLP, PCBLF, PCBL0C, PCBL0D0U, PCBL0D0B, PCBL0H0P0, PCBLLP,</td>
</tr>
<tr>
<td>29</td>
<td>COMMOV / SIDE/</td>
</tr>
<tr>
<td>30</td>
<td>1XP1, XWAF, XX0C, XBLF, XBL0DU, X0H3, DUM0S1, DUM0S2,</td>
</tr>
<tr>
<td>31</td>
<td>2XT21, XP21, XX21, XS21, T23, P23, H23, S23,</td>
</tr>
<tr>
<td>32</td>
<td>3T24, P24, H24, S24, T25, P25, H25, S25,</td>
</tr>
<tr>
<td>33</td>
<td>4T28, P28, H28, S28, T29, P29, H29, S29,</td>
</tr>
<tr>
<td>34</td>
<td>5WAD, WFD0S, NG24, FAR24, ETA0D, PD0DU0C, BYPASS0, DUM0S3,</td>
</tr>
<tr>
<td>35</td>
<td>6TS28, PS28, V28, AM28, TS29, PS29, V29, AM29,</td>
</tr>
<tr>
<td>36</td>
<td>COMMOV / BACK/</td>
</tr>
<tr>
<td>37</td>
<td>1XT55, XP55, XX55, XS55, XT25, XP25, XXH25, XS25,</td>
</tr>
<tr>
<td>38</td>
<td>2XWFB, XXG55, XFR055, XWFD, XXG24, XFR024, XXP1, DUM0B,</td>
</tr>
<tr>
<td>39</td>
<td>4T36, P6, H6, S6, T7, P7, H7, S7,</td>
</tr>
<tr>
<td>40</td>
<td>4T7, P8, H8, S8, T9, P9, H9, S9,</td>
</tr>
<tr>
<td>41</td>
<td>5WG6, WFA, NG7, FAR7, ETA0A, DPA0FT, V55, V25,</td>
</tr>
<tr>
<td>42</td>
<td>6PS6, V6, AM6, TS7, PS7, V7, AM7, AM25,</td>
</tr>
<tr>
<td>43</td>
<td>7T5S, PS8, V8, AM8, TS9, PS9, V9, AM9,</td>
</tr>
<tr>
<td>44</td>
<td>87FA0, FRD, VJD, FGM0D, VJM, FGM0M, FGP0D, FGP0M,</td>
</tr>
<tr>
<td>45</td>
<td>9FGM, FGP, WFT, WGT, F0RT, FG, FN, SFC,</td>
</tr>
<tr>
<td>46</td>
<td>COMMOV / DUMMY/DUMMY1100</td>
</tr>
<tr>
<td>47</td>
<td>LOGICAL ERROR, FAN</td>
</tr>
<tr>
<td>48</td>
<td>EQUIVALENCE (FAN, DUMMY11), (ISPOOL, DUMMY6)</td>
</tr>
<tr>
<td>49</td>
<td>DATA AWORD/6H PERF/</td>
</tr>
<tr>
<td>50</td>
<td>WORD=AWORD</td>
</tr>
<tr>
<td>51</td>
<td>G=32.174049</td>
</tr>
<tr>
<td>52</td>
<td>CAPSF=2116,2170</td>
</tr>
<tr>
<td>53</td>
<td>WFT=WFB+WFD+WFA</td>
</tr>
<tr>
<td>54</td>
<td>WAT=WAF-BLOB</td>
</tr>
<tr>
<td>55</td>
<td>WGT=WAT+WFT</td>
</tr>
<tr>
<td>56</td>
<td>FART=AFT/WAT</td>
</tr>
<tr>
<td>57</td>
<td>VAM=AM=CS</td>
</tr>
<tr>
<td>58</td>
<td>FRAV=WAF/W</td>
</tr>
<tr>
<td>59</td>
<td>C TO REMOVE CD NOZ COEFF, CHANGE CALL FRASH0D, CVN0D0Z EQ.</td>
</tr>
<tr>
<td>60</td>
<td>C ALSO SUB CDUCT REMOVE IF (IBDURN, EQ=1) IDC0=1</td>
</tr>
<tr>
<td>61</td>
<td>C ALSO SUB CNOZ REMOVE IF (IIFTBN, EQ=1) IMCD=1</td>
</tr>
<tr>
<td>62</td>
<td>ARAT13=A97A8</td>
</tr>
</tbody>
</table>
PTRATJ = PB / P1
PRATI3 = PS9 / P1
CALL PROCOM (FAR7, TS9, CS9, AK9, CP9, REX9, PHIS9, HS9)
CALL FRASHO (PTRATJ, ARATI3, CVMNOZ)
CVNOZ = CVMNOZ
VJM = VJM * W7/G
FGMM = VJM * W7/G
FGMP = APSF * (PS9 - P1) * A9
IF (IMASMX, GT, 0, OR, NOT, FAN) GO TO 1
PRATI3 = PS29 / P1
ARATI3 = A29 / A28
PTRATL = P28 / P1
FGPH = APSF * (PS9 - P1) * 89
IF (IGRSM#.GT.0, OR, .NOT, FAN1) GO TO 1
CALL PROCOM
CALL FXASH0 (PTRATOJ, ARATI3, CMNOZ)
VJD = CMNOZ * V29
FGHD = WJD + WGE4 / G
FGPD = APSF * (PS29 - P1) * 829
FGM = FGM + FGM
FN = FGMF + FN
SFC = 3500 * WFT / FN
FG = DELFG * FG
FN = DELFN * FN
SFC = DELSFC * SFC
CALL JRTPUT
CALL ERRIR
RETURN
END

$IBFIC OUTPUT
SUBROUTINE OUTPUT
COMMON / ALL/
1 WORD, IDES, JDES, KDES, MODE, INIT, IDUMP, IAMP, 
2 IGRSMX, IDBURN, IAFTB, ICDC, IMCD, IDSHOC, IMSHOC, NOZFLT, 
3 IMYN, LOMPER, NMAP, NUMMAP, MAPEDG, TOLALL, ARR(6)
COMMON / DESIGN/
1 PCNFJ, PCNCGU, T4GU, DUMDI, DUMD2, DELFG, DELFN, DELSFC,
2 FD, PCNFD, PRFD, ETAFD, WAFDS, PRFCF, ETAFCF, WACF
3ZCDS, PCNCDS, PRCD, ETACDS, WACDS, PRCCF, ETAFCF, WACF
4T4DS, WFBDS, DTCDS, ETAADS, WACDS, PDCDS, DTCDCF, ETAFCF
5TFHPS, CNHPS, ETHPS, TFHPCF, CNHPF, ETHPCF, DHHPF, T2DS
6TLPDS, CNLPS, ETLP, TFLPCF, CNLPCF, ETLPCF, DHLPCF, T1DS
7TF24DS, WFDPS, DTDUDS, ETAADS, W24DS, DPDUDS, DTDUFC, ETAFCF
8T7DS, WAFDS, DTAFDS, ETAADS, W66DS, DPAFDS, DTAFCF, ETAACF
9A55, A25, A6, A7, A8, A9, A28, A29
$PS55, AM55, CVNOZ, CVMNOZ, A8SAV, A9SAV, A28SAV, A29SAV
COMMON / FRONT/
11T1, PI, H1, S1, T2, P2, H2, S2, 
2IT2, P21, H21, S21, T3, P3, H3, S3, 
3IT4, P4, H4, S4, T5, P5, H5, S5, 
4IT55, P55, H55, S55, BLF, BLF, BLF, 
5CF, PRF, ETAF, WAF, WAF, WA3, WG4, FAR4, 
6CNC, PCR, ETAC, WACC, WAC, ETAB, DPCOM, DUNF, 
7CNHP, ETAHHP, DHTCHP, DHTC, BLHP, WG5, FASG, CS, 
8CNLP, ETAELP, DHTCLP, DHTF, BLLP, WG55, FAS55, HPEXT, 
9AM, ALTP, ETAR, ZF, PCNF, ZC, PCNC, WFB, 

80
$TFFHP ,TFFLP ,PCBLF ,PCBLC ,PCBLDU ,PCBLOB ,PCBLHP ,PCBLLP
COMM / SIDE/
1XP1 ,XWAF ,XWAC ,XBLF ,XBLDU ,XH3 ,DUMS1 ,DUMS2 ,
2XT21 ,XP21 ,XH21 ,XS21 ,T23 ,P23 ,T23 ,S23 ,
3T24 ,P24 ,H24 ,S24 ,T25 ,P25 ,H25 ,S25 ,
4T28 ,P28 ,H28 ,S28 ,T29 ,P29 ,H29 ,S29 ,
5WAD ,WF24 ,FAR24 ,ETAD ,DPDUC ,BYPASS1 ,DUMS3 ,
6TS28 ,PS28 ,V28 ,AM28 ,TS29 ,PS29 ,V29 ,AM29
COMM / BACK/
1XT55 ,XP55 ,XH55 ,XS55 ,XT5 ,XP5 ,XH25 ,XS25 ,
2XW95 ,XMR5 ,XFA3 ,XWF9 ,XG24 ,XFA4 ,XPP1 ,DUMB ,
3T6 ,P6 ,H6 ,S6 ,T7 ,P7 ,H7 ,S7 ,
4T8 ,P8 ,H8 ,S8 ,T9 ,P9 ,H9 ,S9 ,
5W6 ,WFA7 ,WAF7 ,ETAA ,DPAFT ,V55 ,V25 ,
6PS6 ,V6 ,AM6 ,TS7 ,PS7 ,V7 ,AM7 ,AM25 ,
7TS8 ,PS8 ,V8 ,AM8 ,TS9 ,PS9 ,V9 ,AM9 ,
8VA ,FRD ,VJD ,FGMD ,VJM ,FGMM ,FGPD ,FGPM ,
9FGM ,FGP ,WFT ,WGT ,FAT ,FG ,FN ,SFC
COMM / DUMMYS / DUMMY(100)
EQUIVALENCE (T4PBL,DUMMY(2)),(T41,DUMMY(3))
DIMENSION W(54),ANS1(80),ANS2(80),ANS3(48),ANS4(72)
EQUIVALENCE (ANS1,PCNFGU),(ANS2,T1),(ANS3,XP1),(ANS4,XT55)
DATA AWORD1,AWORD2/6HOUTPUT,6HCOMM0W/
DATA (W11,1),I=1,4/6HSUBSON,6HIC C-D,6H NDZLL,6HE /
DATA (W12,1),I=1,4/6HSHOCK,6HINSOE,6HC-D N,6HDOZZL /
DATA (W13,1),I=1,4/6HSHOCK,6HOUTSD,6HE C-D ,6HNDZLL /
DATA (W14,1),I=1,4/6HSUBSON,6HIC CON,6HVERS,6HNDZLL /
DATA (W15,1),I=1,4/6HSONIC,6HCONVER,6HGET N,6HDOZZL /
WORD=AWORD1
IF (IBURN.GT.0) GO TO 2
IF (IAFTBN.GT.0) GO TO 1
WRITE (6,7) WORD,AM,ALTP,T4,ETAR
GO TO 3
1 WRITE (6,8) WORD,AM,ALTP,T4,T7,ETAR
GO TO 3
2 WRITE (6,9) WORD,AM,ALTP,T4,T24,ETAR
3 CALL ZNOUT (2)
WRITE (6,10) (W1IMSHOC,I),I=1,4),FG,FN,SFC
IF (I3ASMX.GT.0) GO TO 4
WRITE (6,11) (WIDSHOC,I),I=1,4)
4 WRITE (6,12) LOOPER
IF (IDES.NE.1) GO TO 5
WORD=AWORD2
WRITE (6,13) WORD,ZF,PCNF,ZI,PCNI,ZC,PCNC,T4,MODE
WRITE (6,14)
WRITE (6,15) (ANS1(I),I=1,80)
WRITE (6,14)
WRITE (6,15) (ANS2(I),I=1,80)
WRITE (6,14)
WRITE (6,15) (ANS3(I),I=1,48)
WRITE (6,14)
WRITE (6,15) (ANS4(I),I=1,72)
WRITE (6,16)
IF (IDES.EQ.1) GO TO 6
CONTINUE
A8=A8AV
A9=A9AV
A28=A28AV
A29=A29AV
IF (IDUMP.NE.2) GO TO 6
WRITE (6,16)
CALL SYG (2)
2XWF5, XWG55, XFAR55, XWFD, XWG24, XFAR24, XP1, DUMB, 37
3T6, P6, H6, S6, T7, P7, H7, S7, 38
4T8, P8, H8, S8, T9, P9, H9, S9, 39
5NG6, MFA, WC7, FAR7, ETA, DPAFT, V55, V25, 40
6PS6, V6, AM6, TS7, PS7, V7, AM7, AM25, 41
7TS8, PS8, V8, AM8, TS9, PS9, V9, AM9, 42
8VA, FRD, VJY, FGMD, VJM, FGMM, FGP, FGP, 43
9CGM, FCP, WFT, WGT, FART, FG, FN, SFC

COMMD/DUMMY/DUMMY/100

EQUIVALENCE (TFHP, DUMMY(12)), (CNHP, DUMMY(13)), (DHTCHP, DUMMY(14))

EQUIVALENCE (TFFLP, DUMMY(15)), (CNLP, DUMMY(16)), (DHTCLP, DUMMY(17))

DIMENSION PARAM(130), WORDY(380), IOUT(103), ADUT(6), WOUT(6)

EQUIVALENCE (PARAM, PCNFGU)

DATA (IORDY(I), I=1,198) / 50

16HPCNC3U, 6HPCNGU, 6HT4GU, 6HDUMD1, 6HDUMD2, 6HDELFG, 6HDELGN, 51
26HDELSFC, 6HZFDS, 6HPCNFD, 6HPRFDS, 6HEATFD, 6HOWAFD, 6HPRFCF, 52
36HEACFC, 6HWAFCF, 6HZCDS, 6HPCNCD, 6HPRCDS, 6HETACDS, 6HWACDS, 53
46HPPCFC, 6HETACCC, 6HWACCF, 6HTCDS, 6HWFDOS, 6HDTCDOS, 6HETBOS, 54
56HWAC3DS, 6HPCDOD, 6HTCDOF, 6HETABC, 6HTFHPDS, 6HCNHPDS, 6HETHPDS, 55
66HTFHPCF, 6HNPCHPF, 6HEHTPCF, 6HMHPHPCF, 6HTD2OS, 6HTFLPDOS, 6HCNLPS, 56
76HTLDPFS, 6HTLPLCF, 6HETADOS, 6HMT2DOS, 6HT24DS, 57
86HWDOS, 6HTDUDOS, 6HETAODS, 6HWWA23OS, 6HDPDUDOS, 6HDTUDCF, 6HETADCF, 58
96HTDOS, 6HWFADS, 6HTAFADS, 6HETAADS, 6HWG6CD, 6HDPAFDOS, 6HTACFCF, 59
$h6HETACCF, 6HASS, 6HA25, 6HA6, 6HA7, 6HA8, 6HA9, 60
96HAZ2, 6HAP2, 6HPS55, 6HAM55, 6HVDN02, 6HCVMNO2, 6HABSAV, 61
96HAASAV, 6HA28SAV, 6HA29SAV, 6HT1, 6HP1, 6HH1, 6HS1, 62
96HT2, 6HP2, 6HH2, 6HS2, 6HT21, 6HP21, 6HH21, 63
96HS21, 6HT3, 6HP3, 6HH3, 6HS3, 6HT4, 6HP4 / 64

DATA (IORDY(I), I=99,198) / 65

16H4H, 6H5, 6HT5, 6HP5, 6H5H, 6HS5, 6HT5, 65
26H5P5, 6HH55, 6HS55, 6HBLF, 6HBLC, 6HBLDU, 6HBLB, 67
36HCNF, 6HPRF, 6HETAF, 6HWAFC, 6HWA, 6HWA3, 6HWG4, 68
46HFAF, 6HCNC, 6HPRC, 6HEATAC, 6HWAC, 6HAC, 6HETAB, 69
56HDPCC34, 6HDUMT, 6HCNHPM, 6HETATHP, 6HDHTC, 6HDLT, 6BLHP, 70
66HGG5, 6HFAR5, 6HCS, 6HCNLP, 6HETATL, 6HDHTCLM, 6HDHTF, 71
76HBLLP, 6H6W55, 6HFAR55, 6HMPET, 6HAM, 6HALTS, 6ETAR, 72
86HZF, 6HPCNF, 6HZC, 6HPCNC, 6HWF, 6HTFFHMP, 6HTFLPM, 73
96HPCBCL, 6HPCBLC, 6HPCBLD, 6HPCBLB, 6HPCBLH, 6HPCBLP, 6HXP1, 74
46HXM6, 6HXM6, 6HXBFL, 6HXLFDU, 6HXLH, 6HDSUMS1, 6HDUMS2, 75
66HTX21, 6HXP21, 6HXX21, 6HXX21, 6HT23, 6HP23, 6HH23, 75
66HS23, 6HT24, 6HP24, 6HH24, 6HS24, 6HT25, 6HP25, 77
66H26, 6H25, 6HT28, 6HP28, 6HS28, 6HT29 / 78

DATA (IORDY(I), I=190,280) / 79

16H29P, 6HH29, 6HS29, 6HWA0, 6HWF, 6HWW24, 6HFAR24, 80
25HEPAD, 6HPDUDC, 6HBPASS, 6HDMDS3, 6HTS28, 6HPS28, 6HV28, 81
36HAM23, 6HTS29, 6HPS29, 6HV29, 6HAM29, 6HTX55, 6HXP55, 82
46HMX55, 6HX555, 6HXT25, 6HX25, 6HX25, 6HXS25, 6XWFB, 83
56HWW55, 6HXFAR55, 6HWWF, 6HXW24, 6HXFAR24, 6HXPP1, 6HUDBM, 84
66HT6, 6HP6, 6HH6, 6HS6, 6HT7, 6HP7, 6HH7, 85
76HS7, 6HT8, 6H8, 6HS8, 6HT9, 6HP9, 85
86H9H, 6HS9, 6HWW6, 6HWF, 6HWWG7, 6HFAR7, 6HEAA, 87
96HDFPT, 6HV55, 6HV25, 6HPS6, 6HV6, 6HAM6, 6HT57, 88
$6HP57, 6HVT7, 6HAM7, 6HAM25, 6HT8, 6HP58, 6HV8, 89
$6HAM8, 6HTS9, 6HPS9, 6HV9, 6HMA9, 6HVA, 6HFRD, 90
$6H9JD, 6HFGM, 6HFGMM, 6HFPGD, 6HFPGM, 6HFGN, 91
$6HFC, 6HFF, 6HGT, 6HART, 6HF, 6HF, 92

DATA (IORDY(I), I=281,297) / 93

1110, 6HTFFHP, 6HCNHP, 6HDBHTCHP, 6HTFFLP, 6HCNLP, 6HDHTCLP/

DATA THEEND, BLANK, LIMIT / 6HTHEEND, 6H / 297 /

GO TO (1,6), ICON

83
C *** INPUT SECTION
1   DO 4 J=1,103
    NUM=N
    READ (5,11) AIN,CHANGE
    IF (AIN.EQ.THEEND) GO TO 5
    DO 2 J=1,LIMIT
      JJ=J
      IF (AIN.EQ.WORDY(J)) GO TO 3
2 CONTINUE
    WRITE (6,12) AIN
    GO TO 4
3 IOUT(NUM)=JJ
    IF (CHANGE.NE.BLANK) WORDY(JJ)=CHANGE
4 CONTINUE
    WRITE (6,13)
5   NUM=NJM-1
   RETURN
C *** OUTPUT SECTION
6   IF (NJM.EQ.1) GO TO 10
    N=NUM
    J=6
    DO 9 I=1,NUM,6
      IF (N.GT.6) GO TO 7
      J=N
7   N=N-6
    DO 8 I=1,J
      L=I+K-1
      M=IOUT(L)
      WOUT(K)=WORDY(M)
8   AOUT(K)=PARAM(M)
    IF (N.LE.0) GO TO 10
9 CONTINUE
10  RETURN
C
11 FORMAT (A6,6X,A6)
12 FORMAT (10H0THE WORD ,A6,26H NOT FOUND IN COMMON ARRAY)
13 FORMAT (22H0ERROR IN CONOUT INPUT)
14 FORMAT (1H ,25XA6,5(9XA6))
15 FORMAT (1H ,20X6E15.6)
END

$IBFTC THCOMP DECK
SUBROUTINE THCOMP (PR,ETA,T,H,S,P,TJ,HO,SO,PO)
  1   PD=PR
  2   TP=T*PR**0.28572
  3   DO 1 I=1,25
  4   CALL THERMO (PO,HP,TP,SP,X1,0,X2,0)
  5   DELS=SP-S
  6   IF (ABS(DELS).LE.0.00005*S) GO TO 2
  7   TP=TP/EXP(4.*DELS)
  8   CALL ERROR
  9   RETURN
 10  HO=H-((HP-H)/ETA)
 11   CALL THERMO (PO,HO,TJ,SO,X1,0,X2,1)
 12  RETURN
END
$IBFTC PROCOM  DECK
SUBROUTINE PROCOM (FARX,TEX,CSEX,AKEX,CPEX,REX,PHI,HEX)
  IF (FARX.LE.0.067623) GO TO 1
  FARX=3.067623
  1 IF (TEX.GE.300.) GO TO 2
  TEX=300.
  2 IF (TEX.LE.4000.) GO TO 3
  TEX=4000.
  3 IF (FARX.GE.0.0) GO TO 4
  FARX=0.0
  C AIR PATH
  4 CPA=(((1.0115540E-25*TEX-1.4526770E-21)*TEX+7.6217567E-18)*TEX-
  1.519259E-14)*TEX-6.716376E-12)*TEX+6.5519486E-08)*TEX-5.1535879E-
  2E-05)*TEX+2.5020051E-01
  HEA=(((1.2544425E-26*TEX-2.075222E-22)*TEX+1.2702637E-18)*TEX-
  1.30256918E-15)*TEX-1.6794594E-12)*TEX+2.1839826E-08)*TEX-2.576844E-
  2E-05)*TEX+2.5020051E-01)*TEX-1.7558866E+00
  SEA=+2.5020051E-01*ALOG(TEX)+(((1.4450767E-25*TEX-2.4221288E-22
  1.5243153E-18)*TEX-3.7820648E-15)*TEX+2.3392790E-12)*TEX+3.2
  2759763E-08)*TEX-5.1576879E-05)*TEX+4.5432300E-02
  IF (FARX.LE.0.0) GO TO 5
  5 C FUEL/AIR PATH
  CPA=(((7.2678710E-25*TEX-1.3356668E-20)*TEX+1.0212913E-15)*TEX-
  1.2051104E-13)*TEX+9.668793E-10)*TEX-1.3771901E-06)*TEX+1.2258630E-
  2E-03)*TEX+7.3816638E-02
  HEF=(((9.084388E-26*TEX-1.905949E-21)*TEX+1.7021525E-17)*TEX-
  1.8412208E-14)*TEX+2.4921699E-10)*TEX-4.5906332E-07)*TEX+6.129315E-
  2E-04)*TEX+7.3816638E-02)*TEX+3.0581530E+01
  SEF=+7.3816638E-02*ALOG(TEX)+(((1.0382670E-25*TEX-2.226118E-21
  1.0512776E-13)*TEX-1.0512776E-13)*TEX+3.3228928E-10)*TEX-6.8
  2859505E-07)*TEX+1.2258630E-03)*TEX+6.483399E-01
  CPA=(CPA+FARX*CPA)/(1.+FARX)
  HEF=(HEF+FARX*HEF)/(1.+FARX)
  PHIE=(SEF+FARX*SEF)/(1.+FARX)
  AMW=25.97-946186*FARX
  REX=+1.986375/AMW
  AKEX=CPEX/(CPEX-REX)
  CSEX=SQRT(AKEX*REX*TEX*25031.37)
  RETURN
END

$IBFTC SERCH  DECK
SUBROUTINE SERCH (P,A,B,C,D,AX,NA,BX,CX,DX,NO,NAM,NOM,NCODE)
  DIMENSION AX(NAM),BX(NAM,NOM),CX(NAM,NOM),DX(NAM,NOM),NO(NAM),Q(3)
  C *** NEEDS SUBROUTINE AFUIR
  C *** AX AND BX MUST BE STORED LO TO HI
  C *** P=INPJT PROPORTION BETWEEN 0.0 AND 1.0
  C IF NOT INPUT, P MUST EQUAL -1.
  C *** NCODE=00 OK
  C NCODE=01 A LO
  C NCODE=02 A HI
  C NCODE=07 ERRDR
  C NCODE=10 B LO
  C NCODE=20 B HI
  C NCODE=0
  C=C=0.
  D=0.

85
C *** FIND A
   DO 1 I=1,NA
   IH=I
   IF (A.lt.AX(I)) GO TO 2
   1 CONTINUE
   IF (A.eq.AX(IH)) NCODE=2
   A=AX(IH)
   GO TO 3
   IF (I.ge.IH+1) GO TO 3
   NCODE=1
   IH=2
   A=AX(1)
   3 IL=IH-1
   LIMH=VD(IH)
   LIML=VD(IL)

C *** FIND B
   PRM=(A-AX(IL))/(AX(IH)-AX(IL))
   PP=P
   IF (P.ge.0.0) GO TO 6
   BL=BX(IL,1)+PR*(BX(IH,1)-BX(IL,1))
   BH=BX(IL,LIML)+PR*(BX(IH,LIMH)-BX(IL,LIML))
   IF (B.ge.BL) GO TO 4
   NCODE=NCODE+10
   B=BL
   GO TO 5
   4 IF (B.LE.BH) GO TO 5
   NCODE=NCODE+20
   B=BH
   5 PP=0.5
   Q(2)=3.
   Q(3)=3.
   6 BH=PP*(BX(IH,LIMH)-BX(IH,1))+BX(IH,1)
   BL=PP*(BX(IL,LIML)-BX(IL,1))+BX(IL,1)
   DO 7 J=2,LIMH
   JH=J
   IF (B11Lt.BX(IH,J)) GO TO 8
   CONTINUE
   7 CONTINUE
   JL=JH-1
   DO 9 K=1,LIML
   KH=K
   IF (BL.LT.BX(IL,K)) GO TO 10
   8 CONTINUE
   IF (B.ge.BX(IL,K)) GO TO 10
   9 CONTINUE
   10 KL=KH+1
   PR=(BX(IH,JL)-BH)/(BX(IH,JH)-BX(IH,JL))
   CH=BX(IH,JL)-PR*(BX(IH,JH)-BX(IH,JL))
   DH=DX(IH,JL)-PR*(DX(IH,JH)-DX(IH,JL))
   PR=(BX(IL,KL)-BL)/(BX(IL,KH)-BX(IL,KL))
   CL=BX(IL,KL)-PR*(BX(IL,KH)-BX(IL,KL))
   DL=DX(IL,KL)-PR*(DX(IL,KH)-DX(IL,KL))
   BT=BL*PR*(BH-BL)
   CT=CL*PR*(CH-CL)
   DT=DL*PR*(DH-DL)
   IF (P.ge.0.0) GO TO 13
   DIR=SQR(T(B/T))
   ERR=(B-BT)/B
   CALL AFQUIR(Q(1),PP,ERR,0.0,25.0,0.001,DIR,PT,ICON)
   GO TO (11,13,12),ICON
   11 PP=PT
   IF (P.LT.0.0) PP=0.
   IF (P.GT.1.0) PP=1.
   GO TO 5
   12 NCODE=7
$IBFTC MAPBAK DECK
SUBROUTINE MAPBAC (MAP, MAPGO, TFFS, TFF, CNS, CN, PCN, T, MDDE, IGO, NUM)
DATA H+%WL,WT,WS/6H H*+P*56H L*P*56H TFF56HSPEED/
IF (NJMH.GT.0) GO TO 1
NUMH=0
NUML=0
1 IGO=MAPGO+3*(MAP-1)
GO TO (2,3,5,6,7,9), IGO
C *** HIGH PRESSURE TURBINE
2 TFF=TFF+0.1*(TFF-TFFS)
WRITE (8,10) WH, WT, TFFS, TFF
RETURN
3 CN=CN+0.05*(CN-CNS)
IF (MDDE.NE.1) PCN=PCN*(CN/CNS)
IF (MDDE.EQ.1) T=T*(CNS/CN)**2
WRITE (8,10) WH, WS, CNS, CN
IF (NJMH.GT.2) GO TO 4
NUM=1
NUMH=NJMH+1
RETURN
4 DELCN=CN-CNS
IF (DELCN.GE.0) RETURN
TFF=TFF*(1.+DELCN/CN)
WRITE (8,11) WH, WT, TFFS, TFF
RETURN
5 TFF=TFF+0.1*(TFF-TFFS)
WRITE (8,10) WH, WT, TFFS, TFF
GO TO 3
C *** LOW PRESSURE TURBINE
6 TFF=TFF+0.1*(TFF-TFFS)
WRITE (8,10) WL, WT, TFFS, TFF
RETURN
7 CN=CN+0.05*(CN-CNS)
IF (MDDE.NE.3) PCN=PCN*(CN/CNS)
IF (MDDE.EQ.3) T=T*(CNS/CN)
WRITE (8,10) WL, WS, CNS, CN
IF (NJML.GT.2) GO TO 8
NUM=1
NUML=NJML+1
RETURN
8 DELCN=CN-CNS
IF (DELCN.GE.0) RETURN
TFF=TFF*(1.+DELCN/CN)
WRITE (8,11) WL, WT, TFFS, TFF
RETURN
9 TFF=TFF+0.1*(TFF-TFFS)
WRITE (8,10) WL, WT, TFFS, TFF
GO TO 7
C
C
10 FORMAT (1H0,A6,12HTURBINE MAP ,A6,4HNAS=,E13.6,10H AND NOW=,E13.6
1,6H$$$$$$)$
$IBFTC CONVRG DECK

SUBROUTINE CONVRG (TI, HI, PI, SI, FAR, WG, PA, IDES, AO, PR, TO, HO, PO, SO, TS
10, PSO, VO, AMO, ICON)
C ICON=1  SUBSONIC, COMPARE PI WITH PR
C ICON=2  SONIC, COMPARE PI WITH PR
C ICON=4  ERROR
AJ=773.26
CAPSF=2165.21
G=321.174049
CALL PROCOM (FAR, TI, XX1, XX2, XX3, XX4, PHI, XX6)
C *** SONIC CALCULATIONS
J=0
TSS=0.833*TI
1 J=J+1
CALL PROCOM (FAR, TSS, CSS, AKS, CP, REXS, PHI, SS, HSS)
HSCAL=4I-CSS2/(2*G*AJ)
DELHS=HSCAL-HSS
IF (ABS(DELHS)-0.0005*HSCAL) 4,4,2
2 TSS=TSS+DELHS/CP
IF (J<15) 1,1,3
3 ICON=4
RETURN
4 IF (IRES) 12,12,5
C *** ISENTROPIC EXPANSION CALCULATIONS
5 J=0
TSI=TI*(PA/PI)**0.286
6 J=J+1
CALL THERMO (PA, HSI, TSI, SS, SI, XX1, 1, FAR, 0)
IF (ABS(SSI-SI)-0.0001*SI) 8,8,7
7 TSI=TSI/EXP(4.*(SSI-SI))
IF (J<30) 6,6,3
8 VIS=SQR(T2G+AJ*(H-I-HSI))
IF (VIS-SS) 9,11,11
C *** SUBSONIC DESIGN, CALCULATE AO
9 VO=VIS
TSO=TSI
PSO=PA
CALL PROCOM (FAR, TSO, CSS, XX2, XX3, REX, PHI, HS0, SO)
RHO=CAPSF/PSO/(AJ*REX*TSO)
AO=WG/(RHO*VO)
AMO=VO/CSO
PR=PI
ICON=1
10 TO=TI
HO=HI
PO=PI
SO=SI
RETURN
C *** SONIC DESIGN, CALCULATE AO
11 VO=CSS
TSO=TSS
PSO=PI*(TSO/TO)**(AKS/(AKS-1.))
RHO=CAPSF/PSO/(AJ*REX*TSO)
AO=WG/(RHO*VO)
C *** NON-DESIGN, CALCULATE CRITICAL CONDITIONS
12 VO=CSS
TSO=TSS
PSO=PA
RHO=CAPSF*PSO/(AJ*REXS*TSO)
AOCRIT=WG/(RHO*VO)
AMO=1.0
PR=PS3*(TI/TSO)**(AKS/(AKS-1.))
IF (A)-AOCRIT 13, 13, 14
C *** NON-DESIGN, CRITICAL AND SUPERCRITICAL CONDITIONS
13 PSO=PS3*AOCRIT/AO
PR=PR*AOCRIT/AO
TCON=2
GO TO 10
C *** NON-DESIGN, SUBSONIC CALCULATIONS
14 PSO=PA
J=0
TSO=0.833*TSO
15 J=J+1
CALL PROCOM (FAR, TSS, CSS, AK, CP, REX, PHISO, HSO)
RHO=CAPSF*PSO/(AJ*REX*TSO)
VO=WG/(RHO*AO)
HSCAL=HI-VO**2/(2*G*AJ)
DELHS=HSCAL-HSO
IF (ABSDELHS)-0.0005*HSCAL) 17, 17, 16
16 TSO=TSS+DELHS/CP
IF (J-15) 15, 15, 3
17 AMO=V)/CSO
PR=PS3*(TI/TSO)**(AKS/(AKS-1.))
ICON=1
GO TO 10
END

$IBFTC CONDIV DECK
SUBROJNTINE CONDIV (TI, HI, PI, SI, FAR, WG, PA, IDES, AT, AO, PIR, TT, HT, PI, S
1 T, T0, J, 3, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, ICON)
C ICON=1 SUBSONIC, COMPARE PIR WITH PI
C ICON=2 SONIC, SHOCK INSIDE NOZZLE, COMPARE PIR WITH PI
C ICON=3 SONIC, SHOCK OUTSIDE NOZZLE, COMPARE PIR WITH PI
C ICON=4 ERROR
DIMENSION Q(I)
Q(2)=7.0
Q(3)=5.0
AJ=773.26
CAPSF=2116.2170
G=32.174049
CALL PROCOM (FAR, TI, XX1, XX2, XX3, XX4, PHII, XX6)
C *** SONIC CALCULATIONS
14 J=0
15 TSS=0.833*TI
16 J=J+1
CALL PROCOM (FAR, TSS, CSS, AK, CP, REX, PHISS, HSS)
HSCAL=HI-CSS**2/(2*G*AJ)
DELHS = HSCAL - HSS
IF (ABS(DELHS) - 0.0005 * HSCAL) \leq 0.0002  
TSS = TSS + DELHS/CP
IF (J - 15) \leq 1, 3
ICON = 4
RETURN

IF (IDES) \geq 11, 11, 5
C *** SONIC DESIGN, CALCULATE AT
VT = CSS
TST = TSS
PST = PI * (TST/TI)**(AK/(AK - 1.))
RHO = CAPSF * PST/(AJ * RESX * TST)
AT = WG/(RHO * VT)
AMT = 1.0
C *** IDEAL EXPANSION DESIGN, CALCULATE AJ
PSO = PA
J = 0
TSO = TI * (PSO/PI)**0.286
J = J + 1
CALL PROCOM(FAR, TSO, CSO, AK, CP, RESX, PHISO, HSO)
PHICAL = PHI - RESX * ALOG(PI/PSO)
DELPHI = PHICAL - PHISO
IF (A3S(DELPHI) - 0.0001 * PHICAL) \leq 0.0001
TST = TSO * EXP(4.0 * DELPHI)
IF (J - 15) \leq 6, 6, 3
V0 = SQRT(2.0 * G * AJ * (HI - HSO))
AMO = V0/CSO
AO = (AT/AMO)**(2.0*(1.0+(AK - 1.0)*AMO**2/2.0)/(AK + 1.0))**((AK + 1.0)/(2.0*(AK - 1.0)))
PIR = PI
ICON = 3
TO = TI
HO = HI
PO = PI
SO = SI
RETURN
C *** ASSUME SONIC THROAT AND ISENTROPIC EXPANSION TO AO
VT = CSS
AMT = 1.0
TST = TSS
RHO = WG/(AT * VT)
PST = RHO * AJ * RESX * TST/CAPSF
PIR = PST * (TI/TST)**(AK/(AK - 1.))
IF (PST - PA) \geq 12, 27, 27
TSO = 0.95 * TI
MAM = 0
CALL PROCOM (FAR, TSO, CSO, AK, CP, RESX, PHISO, HSO)
AMO = SQRT((TI/TSO) - 1.0)/(AK - 1.0)
AOCAL = (AT/AMO)**(2.0*(1.0+(AK - 1.0)*AMO**2/2.0)/(AK + 1.0))**((AK + 1.0)/(2.0*(AK - 1.0)))
EA = (AO - AOCAL)/AO
DIR = SQRT(AO/AOCAL)
CALL AFFQUIR (Q1, TSO, EA, 0.00100, 0.0001, DIR, TSOT, JCON)
GO TO (14, 18, 3), JCON
TSO = TSOT
IF (TS - TI) \geq 15, 13, 16
TSC = 2.0 * TI/(AK + 1.0)
IF (TS - GT, TSC) GO TO 17
16  TSO=0.98*TI
   GO TO 13
17  IF (Q(2)<LT.30.0.OR.AMO<LT.0.95.OR.MAM<.EQ.1.) GO TO 13
    TSO=2.*TI/(2.+.098*(AK-1.))
    MAM=1
    GO TO 13
18  PSO=PIR*(TSO/TI)**(AK/(AK-1.))
    IF (PSO-PA)<20.19.27
   C *** CRITICAL FLOW, ISENTROPIC EXPANSION TO PA
19   VO=AMO*CSO
   ICON=1
   GO TO 9
   C *** SUBSONIC FLOW
   20  PSO=PA
       Q(2)=J.*
       Q(3)=J.*
       J=0
       TSO=0.833*TI
       J=J+1
       CALL PROCOM (FAR,TSO,CSO,AK,CP,REX,PHISO,HSD)
       RHO=CAPSF/PSO/(AJ*REX*TSO)
       VO=WG/(RHO*AO)
       HSCAL=HI-VO**2/(2.*G*AJ)
       DELHS=HSCAL-HSO
       IF (ABS(DELHS)<0.0005*HSCAL) 23,23,22
22  TSO=TSJ+DELHS/CP
    IF (J-15) 21,21,3
23   PSO=P4
    J=0
24  CALL PROCOM (FAR,TST,CST,AK,CP,REX,PHISO,HST)
    PST=PIR*(TST/TI)**(AK/(AK-1.))
    RHO=PST*CAPSF/(AJ*REX*TST)
    VT=WG/(RHO*AT)
    HSCAL=HI-VT**2/(2.*G*AJ)
    EH=(HSCAL-HST)/HSCAL
    DIR=1.+(HSCAL-HST)/(CP*TST)
    CALL APQUR (Q(1),TST,EH,0.0,20.0,0.0005,DIR,TSTT,ICON)
    GO TO (25,26,3),JCON
25  TST=TSTT
    GO TO 24
26  AMT=VT/CST
   ICON=1
   GO TO 9
   C *** SUPERCRITICAL FLOW, ISENTROPIC EXPANSION TO PA
27   PSO=PA
28  J=0
   TSO=TI*(PSO/PIR)**.286
29  J=J+1
   CALL PROCOM (FAR,TSO,CSO,AK,CP,REX,PHISO,HSD)
   PHICAL=PHI-REX*ALOG(PIR/PSO)
   DELPHI=PHICAL-PHISO
   IF (ABS(DELPHI)<0.0001*PHICAL) 30,30,29
30  TSO=TSJ*EXP(4.*DELPHI)
    IF (J-15) 28,28,3
    VO=SRT(2.*G*AJ*(HI-HSO))
    AMO=V3/CSO
    AOID=(AT/AMO)*(2.*((AK-1.)*AMO**2/2.)/(AK+1.))**((AK+1.)/(2.*(AK
31   ICON=3
32   N=0
    IF (AJ-AOID)<31,9,32
C *** SUPERCRITICAL FLOW, ISENTROPIC EXPANSION TO AO

31 N=1
32 TSO=0.833*TI
33 J=0
34 J=J+1
35 CALL PROCOM (TAR, TSO, CSO, AK, CP, REX, PHISO, HSO)
36 AMO=SRT(T(2.*(T(TI/TSO)-1.)/(AK-1.)))
37 AOCAL=(AT/AMO)*(2.*(1.*(AK-1.)*AMO**2)/(AK+1.))**((AK+1.)/(2.*((AK-1.)))
38 DELA=AO-AOCAL
39 IF (ABS(DELA) 0.0001*AO) 35, 35, 34
40 TSO=TSO+SQRT(AOCAL/TO)
41 IF (J>=50) 33, 33, 3
42 IF (N) 37, 37, 36
C *** UNDEREXPANDED, SHOCK OUTSIDE NOZZLE
43 PSO=PIR*(TSO/TO)**(AK/(AK-1.))
44 VO=AMJ*CSO
45 GO TO 9
C *** OVEREXPANDED, FIND SHOCK POSITION
46 PSX=PIR*(TSO/TO)**(AK/(AK-1.))
47 PSY=PSX*(2.*(AK*AMO**2)/(AK+1.))-(AK-1.)/(AK+1.))
48 IF (PS-PSY) 38, 39, 39
C *** OVEREXPANDED, SHOCK OUTSIDE NOZZLE
49 PSO=PSX
50 VO=AMJ*CSO
51 GO TO 9
C *** OVEREXPANDED, SHOCK INSIDE NOZZLE
52 PSO=PA
53 J=0
54 TSO=0.833*TI
55 J=J+1
56 CALL PROCOM (TAR, TSO, CSO, AK, CP, REX, PHISO, HSO)
57 RHO=CAPSF*PSO/(AJ*REX*TSO)
58 VO=WG/(RHO*AO)
59 HSCAL=HI-VO**2/(2.*S*AJ)
60 DELHS=HSCAL-PSO
61 IF (ARS(DELHS)-0.0005*HSCAL) 42, 42, 41
62 TSO=TSO+DELHS/CP
63 IF (J>=15) 40, 40, 3
64 AMO=V3/CSO
65 TO=TI
66 HO=HI
67 PO=PS3*(T(TO/TSO))**(AK/(AK-1.))
68 SO=PHII-REX*ALOG(P)
69 ICON=2
70 GO TO 10
71 END

$IBFTC THETB DECK
72 SUBROJTH IN THE TURB (DH, ETA, FAR, H, S, P, TO, HO, S, PO)
73 HO=H-9.1
74 HOP=H-DH/ETA
75 PT=P/2.
76 DO 1 I=1, 25
77 CALL THERMO (PT, HOP, TT, ST, AMWT, 1, FAR, 1)
78 DELS=ST-5
79 IF (ABS(DELHS), LE.0.0005*5) GO TO 2
1. PT=P*EXP(DEL*$AMT/1.986375+ALOG(PT/P))
2. CALL ERROR
3. PO=PT
4. CALL THERMO (PO, HO, TO, SO, X1, FAR, 1)
5. RETURN
6. END

$IBFTC THERMO DECK
SUBROUTINE THERMO (PX, HX, TX, SX, AMX, L, FAR, K)
FX=0.
IF (L.EQ.1) FX=FAR
IF (K.EQ.1) GO TO 1
CALL PROCOM (FX, TX, CS, AK, CP, R, PHI, HX)
GO TO 3
1. TX=4.*HX
DO 2 I=1, 15
CALL PROCOM (FX, TX, CS, AK, CP, R, PHI, HX)
DELH=H-X
IF (ABS(DELH) .LE. 0.00001*HX) GO TO 3
2. TX=TX+4.*DELH
WRITE (8, 4)
3. SX=PHI-R*ALOG(PX)
AMX=1.986375/R
RETURN
C
C
4. FORMAT (31H00 CONVERGENCE IN THERMO)$$$$$$
END

$IBFTC AFQUER DECK
SUBROUTINE AFQUER (X, AIND, DEPEND, ANS, AJ, TOL, DIR, ANEW, ICON)
DIMENSION X(9)
C X(1) NAME OF ARRAY TO USE
C AIND= DEPENDANT VARIABLE
C DEPEND= DEPENDANT VARIABLE
C ANS= ANSWER UPON WHICH TO CONVERGE
C AJ= MAX NUMBER OF TRYS
C TOL= PERCENT TOLERANCE FOR CONVERGENCE
C DIR= DIRECTION AND PERCENTAGE FOR FIRST GUESS
C ANEW= CALCULATED VALUE OF NEXT TRY AT DEPENDANT VARIABLE
C ICON= CONTROL =1 GO THRU LOOP AGAIN
C =2 YOU HAVE REACHED THE ANSWER
C =3 COUNTER HAS HIT LIMITS
C X(2)= COUNTER STORAGE
C X(3)= CHOSES METHOD OF CONVERGENCE
C X(4)= THIRD DEPEND VAR
C X(5)= THIRD IND VAR
C X(6)= SECOND DEPEND VAR
C X(7)= SECOND IND VAR
C X(8)= FIRST DEPEND VAR
C X(9)= FIRST IND VAR
C X(10) MUST BE ZERO UPON FIRST ENTRY TO ROUTINE
Y=0.
IF (ANS) 1, 2, 1
1 DEP=DEPEND-ANS
   TOLANS=TOL*ANS
   GO TO 3
2 DEP=DEPEND
   TOLANS=TOL
3 IF (ABS(DEP)-TOLANS) 5,5,4
4 IF (X(2)-AJ) 8,8,7
5 ANEW=AIND
   X(2)=C
   ICON=2
   RETURN
6 ANEW=Y
   X(2)=X(2)+1
   ICON=1
   RETURN
7 ANEW=Y
   X(2)=C
   ICON=3
   RETURN
8 IF (X(3)) 9,9,12
C *** FIRST GUESS USING DIR
9 X(3)=1
   X(8)=DEP
   X(9)=AIND
   IF (AIND) 10,11,10
10 Y=DIR*AIND
   GO TO 5
11 Y=DIR
   GO TO 6
12 IF (X(3)-1.) 13,13,16
C *** LINEAR GUESS
13 X(3)=2
   X(6)=DEP
   X(7)=AIND
   IF (X(8)-X(6)) 14,9,14
14 IF (X(9)-X(7)) 15,9,15
15 A=(X(9)-X(7))/(X(8)-X(6))
   Y=X(9)-A*X(8)
   IF (ABS(10*Y-X(9))-ABS(Y)) 9,9,16
C *** QUADRATIC GUESS
16 X(4)=DEP
   X(5)=AIND
   IF (X(7)-X(5)) 18,17,18
17 IF (X(6)-X(4)) 13,9,13
18 IF (X(6)-X(4)) 19,13,19
19 IF (X(9)-X(5)) 23,20,23
20 IF (X(8)-X(4)) 21,22,21
21 X(9)=X(7)
   X(8)=X(6)
   GO TO 13
22 X(9)=X(7)
   X(8)=X(6)
   X(3)=1
   IF (X(9)) 10,11,10
23 IF (X(8)-X(4)) 24,21,24
24 F=(1/(X(7)-X(5))
   A=(X(3)-X(4)-F*(X(9)-X(5)))/(X(9)-X(7))*(X(9)-X(5))
   B=F-A*X(5)+X(7)
   C=X(4)+X(5)*B-
   IF (A) 27,25,27
25 IF (B) 26,7,26
26     Y=-C/B
        GO TO 47
27     IF (B) 32,28,32
28     IF (C) 30,29,30
29     Y=0.
        GO TO 47
30     G=-C/A
        IF (G) 7,7,31
31     Y=SQRNT(G)
        YY=-SQRNT(G)
        GO TO 37
32     IF (C) 34,33,34
33     Y=-B/A
        YY=0.
        GO TO 37
34     D=4.*A*C/B**2
        IF (1.-D) 13,35,36
35     Y=-B/(2.*A)
        GO TO 47
36     E=SQRNT(1.-D)
        Y=(-B/(2.*A))+(E)*Y
        YY=(-3/(2.*A))+(E)*YY
37     J=4
        DEPMN=ABS(X(I))
        DO 39 I=6,8,2
            IF (DEPMN-ABS(X(I))) 39,39,38
38     J=I
        DEPMN=ABS(X(I))
39     CONTINUE
        K=J+1
        IF (((X(K)-Y)*((K)-YY)) 42,42,40
40     IF (ABS(X(K)))-ABS(Y(K)-YY)) 47,47,41
41     Y=YY
        GO TO 47
42     IF (I=6) 43,44,44
43     JJ=J+2
        KK=K+2
        GO TO 45
44     JJ=J-2
        KK=K-2
45     SLOPE={(X(KK)-X(K))/(X(JJ)-X(J))
            IF (SLOPE*(X(J))-(X(K))-YY)) 46,46,47
46     Y=YY
47     X(9)=X(7)
        X(8)=X(6)
        X(7)=X(5)
        X(6)=X(4)
        GO TO 6
        END

$IBFTC PARABO DECK
SUBROUTINE PARABO (X,Y,XD,YANS)
DIMENSION X(3),Y(3)
A=((X(1)-X(2))*(Y(1)-Y(3))-(X(1)-X(3))*(Y(1)-Y(2)))/(X(1)-X(2))*{(1
1X(1)-X(2))*(X(3)-X(2))}
B=(X(1)**2-X(2)**2)*(Y(1)-Y(3))-(X(1)**2-X(3)**2)*(Y(1)-Y(2))/{
1X(1)-X(2))*(X(1)-X(3))*(X(2)-X(3))}
\[ D = (Y(1)*X(2))^2 - Y(2)*X(1)^2 - B*X(2)*X(1) - (X(2) - X(1))/2 - 1 \]
DATA (PR, 9, J), WAC(9, J), ETA(9, J), J = 1, 8)/
1 1.03000, 139.800, 0.47644, 4.35300, 139.800, 0.60114,
2 7.52200, 139.800, 0.72498, 10.21900, 139.800, 0.77744,
3 11.05300, 139.800, 0.78260, 11.89900, 139.500, 0.77744,
4 13.15000, 139.300, 0.72498, 13.65600, 139.000, 0.69918/
DATA (PR(10, J), WAC(10, J), ETA(10, J), J = 1, 8)/
1 1.03000, 146.200, 0.46612, 3.76500, 146.200, 0.57018,
2 6.43100, 146.200, 0.64224, 9.17600, 146.200, 0.72498,
3 10.21900, 146.200, 0.75078, 11.47900, 146.200, 0.75078,
4 12.71100, 146.200, 0.72498, 14.41200, 146.200, 0.64242/
END

$IBFTC COMB DAT DECK
BLOCK DATA
COMMON / COMB/ PSI(15), DELT(15, 15), ETA(15, 15), N, NP(15) /
DATA N, NP / 15, 15=15 /
DATA PSI /4.9116, 9.8232, 14.735, 19.646, 24.558, 29.470, 34.381, 139.293, 44.207, 73.674, 100.200, 300.400, 500.000/
DATA ETA /
1.600, 7.726, 7.777, 8.064, 8.266, 8.434, 8.855, 8.865, 7.870, 10
78*1.100, 7*.999, 120*1.000/
END

$IBFTC HPT DAT DECK
BLOCK DATA
COMMON / HTURB/TFF(15), CN(15, 15), DH(15, 15), ETA(15, 15), N, NP(15) /
DATA N, NP/10, 9*15, 12, 5*0/
DATA TFF / 39.670, 42.990, 47.460, 48.610, 49.175, 4
1 49.500, 50.000, 50.425, 50.920, 51.575, 5*0./
DATA (CN 1, J), DH(1, J), ETA(1, J), J = 1, 15)/
1 0.1972, 0.0032, 0.6219, 0.3372, 0.0057, 0.7078, 7
2 0.5156, 0.0084, 0.7868, 0.7128, 0.0108, 0.8090, 8
3 0.9382, 0.0133, 0.8090, 1.1442, 0.0152, 0.7963, 9
4 1.3138, 0.0164, 0.7779, 1.5382, 0.0174, 0.7422, 10
5 1.7264, 0.0179, 0.7078, 1.9324, 0.0176, 0.7635, 11
6 2.1500, 0.0167, 0.6068, 2.4058, 0.0144, 0.5309, 12
7 2.5892, 0.0120, 0.4773, 2.7862, 0.0082, 0.4045, 13
8 2.9460, 0.0034, 0.3034/ 14
DATA (CN 2, J), DH(2, J), ETA(2, J), J = 1, 15)/
1 0.1872, 0.0038, 0.6068, 0.3942, 0.0080, 0.7078, 15
2 0.5814, 0.0113, 0.8090, 0.7128, 0.0136, 0.8292, 17
3 0.8442, 0.0156, 0.8363, 0.9804, 0.0176, 0.8393, 18
4 1.1368, 0.0192, 0.8368, 1.2754, 0.0212, 0.8302, 19
5 1.4450, 0.0228, 0.8254, 1.7068, 0.0248, 0.8090, 20
DATA (CN 3, J), D(5, J), ETA(5, J), J=1,15)
1 0.1872, 0.0056, 0.5562, 0.3000, 0.0088, 0.5068,
2 0.5186, 0.0192, 0.5078, 0.3750, 0.0192, 0.8090,
3 0.9754, 0.0236, 0.8494, 1.2574, 0.0238, 0.8697,
4 1.2924, 0.0321, 0.8696, 1.7638, 0.0360, 0.8682,
5 2.3450, 0.0400, 0.8615, 2.3362, 0.0444, 0.8555,
6 2.5450, 0.0496, 0.8520, 2.8076, 0.0540, 0.8494,
7 3.2374, 0.0441, 0.8090, 3.1422, 0.0472, 0.7797,
8 3.3674, 0.0496, 0.7584,
9 3.1518, 0.0661, 0.8570,
10 0.1872, 0.0068, 0.5309, 0.3588, 0.0120, 0.6068,
11 0.5186, 0.0192, 0.5078, 0.3750, 0.0192, 0.8090,
12 0.9754, 0.0236, 0.8494, 1.2574, 0.0238, 0.8697,
13 1.2924, 0.0321, 0.8696, 1.7638, 0.0360, 0.8682,
14 2.3450, 0.0400, 0.8615, 2.3362, 0.0444, 0.8555,
15 2.5450, 0.0496, 0.8520, 2.8076, 0.0540, 0.8494,
16 3.2374, 0.0441, 0.8090, 3.1422, 0.0472, 0.7797,
17 3.3674, 0.0496, 0.7584,
18 3.1518, 0.0661, 0.8570,
19 0.1872, 0.0080, 0.5062, 0.4314, 0.0164, 0.6068,
20 0.5186, 0.0236, 0.7078, 0.9568, 0.0308, 0.8090,
21 1.2010, 0.0372, 0.8696, 1.3834, 0.0416, 0.8697,
22 1.9108, 0.0449, 0.8979, 1.5186, 0.0476, 0.8899,
23 1.7450, 0.0510, 0.8950, 1.8618, 0.0564, 0.9000,
24 1.3558, 0.0576, 0.9010, 2.0000, 0.0600, 0.9000,
25 2.3450, 0.0624, 0.8980, 2.0824, 0.0660, 0.8925,
26 2.1310, 0.0700, 0.8793,
27 0.1872, 0.0088, 0.5051, 0.4834, 0.0196, 0.6068,
28 0.5186, 0.0236, 0.7078, 0.9814, 0.0316, 0.8665,
29 1.2010, 0.0372, 0.8696, 1.3834, 0.0416, 0.8697,
30 1.9108, 0.0449, 0.8979, 1.5186, 0.0476, 0.8899,
31 1.7450, 0.0510, 0.8950, 1.8618, 0.0564, 0.9000,
32 1.3558, 0.0576, 0.9010, 2.0000, 0.0600, 0.9000,
33 2.3450, 0.0624, 0.8980, 2.0824, 0.0660, 0.8925,
34 2.1310, 0.0700, 0.8793,
35 0.1872, 0.0093, 0.4909, 0.3372, 0.0159, 0.5380,
36 0.5364, 0.0232, 0.6068, 0.6754, 0.0284, 0.5573,
37 0.8068, 0.0330, 0.7078, 0.9196, 0.0368, 0.7463,
38 1.0128, 0.0400, 0.7776, 1.1254, 0.0442, 0.6090,
39 1.2196, 0.0480, 0.8191, 1.3138, 0.0524, 0.8302,
$$\text{I}$$

**C**

**D**

**E**

**F**

**G**

**H**

**I**

**J**

**K**

**L**

**M**

**N**

**O**

**P**

**Q**

**R**

**S**

**T**

**U**

**V**

**W**

**X**

**Y**

**Z**

1.3650, 0.0556, 0.8347, 1.4008, 0.0580, 0.8363, 84
2.4450, 0.0612, 0.8322, 1.4638, 0.0640, 0.8241, 85
1.4576, 0.0668, 0.8090/86

DATA (CN(10,J),DH(10,J),ETA(10,J),J=1,12)/
1.0.1977, 0.0132, 0.4257, 0.2841, 0.0180, 0.874, 87
2.0.3504, 0.0228, 0.5056, 0.4686, 0.0268, 0.5359, 88
3.0.3528, 0.0314, 0.5683, 0.6382, 0.0352, 0.5941, 89
4.0.5992, 0.0360, 0.6068, 0.7352, 0.0412, 0.6178, 90
5.0.7696, 0.0440, 0.6260, 0.8068, 0.0476, 0.5310, 91
6.0.8254, 0.0504, 0.6265, 0.8304, 0.0530, 0.5118/92

END

1.0.3582, 0.0018, 0.7120, 0.5336, 0.0026, 0.7300, 93
2.0.7365, 0.0035, 0.7472, 0.9754, 0.0044, 0.7300, 94
3.1.5216, 0.0051, 0.7940, 1.4173, 0.0056, 0.7000, 95
4.1.5201, 0.0059, 0.8850, 1.7673, 0.0061, 0.6730, 96
5.2.3247, 0.0062, 0.6452, 2.2827, 0.0061, 0.6200, 97
6.2.4565, 0.0057, 0.6000, 2.6137, 0.0053, 0.5750, 98
7.2.3166, 0.0044, 0.5310, 2.9456, 0.0035, 0.5000, 99
8.3.3138, 0.0001, 0.3850/100

DATA (CN(2,J),DH(2,J),ETA(2,J),J=1,15)/
1.0.3582, 0.0026, 0.8000, 0.5518, 0.0039, 0.8100, 101
2.0.7919, 0.0054, 0.8200, 1.0672, 0.0069, 0.8300, 102
3.1.2882, 0.0080, 0.9300, 1.4466, 0.0087, 0.8290, 103
4.1.5937, 0.0096, 0.8100, 1.8954, 0.0101, 0.8000, 104
5.2.3519, 0.0104, 0.7850, 2.2273, 0.0107, 0.7600, 105
6.2.3747, 0.0108, 0.7450, 2.6229, 0.0106, 0.7000, 106
7.2.8720, 0.0101, 0.6800, 3.0555, 0.0094, 0.5450, 107
8.3.3138, 0.0077, 0.5900/108

DATA (CN(3,J),DH(3,J),ETA(3,J),J=1,15)/
1.0.3568, 0.0031, 0.8000, 0.5911, 0.0051, 0.8300, 109
2.0.8555, 0.0071, 0.8000, 1.0764, 0.0087, 0.8630, 110
3.1.2519, 0.0099, 0.8670, 1.4354, 0.0111, 0.8700, 111
4.1.5201, 0.0122, 0.8720, 1.8409, 0.0134, 0.8720, 112
5.2.3247, 0.0143, 0.8700, 2.2455, 0.0152, 0.8670, 113
6.2.4302, 0.0157, 0.8600, 2.5956, 0.0162, 0.8500, 114
7.2.7791, 0.0166, 0.8300, 3.0555, 0.0167, 0.8000, 115
8.3.3138, 0.0164, 0.7600/116

DATA (CN(4,J),DH(4,J),ETA(4,J),J=1,15)/
1.0.3582, 0.0039, 0.7950, 0.6237, 0.0038, 0.8000, 117
2.0.5810, 0.0061, 0.8400, 0.8837, 0.0078, 0.8600, 118
3.1.1047, 0.0096, 0.8860, 1.2882, 0.0110, 0.8730, 119
4.1.5090, 0.0126, 0.8800, 1.7482, 0.0141, 0.8830, 120
5.2.2429, 0.0159, 0.8850, 2.2091, 0.0166, 0.8830, 121
6.2.3747, 0.0174, 0.8800, 2.6047, 0.0183, 0.8740, 122
7.2.3720, 0.0191, 0.8600, 3.1291, 0.0195, 0.8350, 123
8.3.3138, 0.0197, 0.8200/124

DATA (CN(5,J),DH(5,J),ETA(5,J),J=1,15)/
1.0.3582, 0.0036, 0.7750, 0.5065, 0.0049, 0.8000, 125
2.0.7365, 0.0071, 0.8480, 0.9754, 0.0092, 0.8600, 126
3.1.2882, 0.0119, 0.8750, 1.5647, 0.0141, 0.8900, 127

100
| 4  | 1.7301 | 0.0155 | 0.8912 | 1.9690 | 0.0172 | 0.8940 | 46  |
| 5  | 2.3893 | 0.0181 | 0.8955 | 2.2637 | 0.0192 | 3.8970 | 47  |
| 6  | 2.6332 | 0.0202 | 0.8961 | 2.6691 | 0.0214 | 0.8900 | 48  |
| 7  | 3.1365 | 0.0226 | 0.8790 | 3.1846 | 0.0235 | 0.8671 | 49  |
| 8  | 3.3138 | 0.0239 | 0.8600 |  |  |  | 50  |

**DATA (CN(6,J),DH(5,J),ETA(6,J),J=1,15)/**

| 1  | 0.3582 | 0.0038 | 0.7600 | 0.6164 | 0.0064 | 0.8000 | 51  |
| 2  | 0.9372 | 0.0087 | 0.8450 | 1.1047 | 0.0113 | 0.8600 | 52  |
| 3  | 1.2882 | 0.0130 | 0.8730 | 1.5283 | 0.0152 | 0.8900 | 53  |
| 4  | 1.7482 | 0.0171 | 0.8950 | 1.9509 | 0.0187 | 0.9000 | 54  |
| 5  | 2.2139 | 0.0209 | 0.9005 | 2.4302 | 0.0226 | 0.9100 | 55  |
| 6  | 2.5510 | 0.0244 | 0.9004 | 2.8619 | 0.0259 | 0.9000 | 56  |
| 7  | 3.1384 | 0.0286 | 0.8900 | 3.2584 | 0.0303 | 0.8800 | 57  |
| 8  | 3.3138 | 0.0319 | 0.8735 |  |  |  | 58  |

**DATA (CN(7,J),DH(7,J),ETA(7,J),J=1,15)/**

| 1  | 0.3582 | 0.0048 | 0.7310 | 0.7728 | 0.0089 | 0.8000 | 59  |
| 2  | 1.3129 | 0.0115 | 0.8300 | 1.2659 | 0.0141 | 0.8600 | 60  |
| 3  | 1.4729 | 0.0162 | 0.8750 | 1.6785 | 0.0181 | 0.8900 | 61  |
| 4  | 1.8409 | 0.0197 | 0.8930 | 2.0247 | 0.0216 | 0.8975 | 62  |
| 5  | 2.1301 | 0.0235 | 0.8999 | 2.3000 | 0.0250 | 0.9000 | 63  |
| 6  | 2.3929 | 0.0265 | 0.8980 | 2.5038 | 0.0284 | 0.8937 | 64  |
| 7  | 2.5583 | 0.0296 | 0.8900 | 2.6137 | 0.0314 | 0.8799 | 65  |
| 8  | 2.5319 | 0.0329 | 0.8710 |  |  |  | 66  |

**DATA (CN(8,J),DH(8,J),ETA(8,J),J=1,15)/**

| 1  | 0.3582 | 0.0048 | 0.7100 | 0.6072 | 0.0078 | 0.7450 | 67  |
| 2  | 0.7919 | 0.0102 | 0.7680 | 0.9754 | 0.0124 | 0.8000 | 68  |
| 3  | 1.2337 | 0.0153 | 0.8380 | 1.4548 | 0.0177 | 0.8600 | 69  |
| 4  | 1.5383 | 0.0201 | 0.8712 | 1.8409 | 0.0226 | 0.8780 | 70  |
| 5  | 1.9509 | 0.0242 | 0.8800 | 2.0801 | 0.0261 | 0.8775 | 71  |
| 6  | 2.1537 | 0.0274 | 0.8760 | 2.2091 | 0.0285 | 0.8722 | 72  |
| 7  | 2.2537 | 0.0299 | 0.8660 | 2.3009 | 0.0314 | 0.8600 | 73  |
| 8  | 2.3051 | 0.0321 | 0.8480 |  |  |  | 74  |

**DATA (CN(9,J),DH(9,J),ETA(9,J),J=1,15)/**

| 1  | 0.3582 | 0.0054 | 0.6780 | 0.5518 | 0.0080 | 0.7000 | 75  |
| 2  | 0.5529 | 0.0096 | 0.7125 | 0.8282 | 0.0119 | 0.7350 | 76  |
| 3  | 1.1329 | 0.0141 | 0.7690 | 1.1691 | 0.0150 | 0.8000 | 77  |
| 4  | 1.2337 | 0.0169 | 0.8060 | 1.3809 | 0.0188 | 0.8225 | 78  |
| 5  | 1.5283 | 0.0209 | 0.8395 | 1.6201 | 0.0233 | 0.8450 | 79  |
| 6  | 1.7482 | 0.0244 | 0.8470 | 1.8409 | 0.0263 | 0.8458 | 80  |
| 7  | 1.3954 | 0.0279 | 0.8330 | 1.9147 | 0.0289 | 0.8235 | 81  |
| 8  | 1.2337 | 0.0303 | 0.8080 |  |  |  | 82  |

**DATA (CN(10,J),DH(10,J),ETA(10,J),J=1,12)/**

| 1  | 0.3582 | 0.0061 | 0.6380 | 0.4782 | 0.0078 | 0.6550 | 83  |
| 2  | 0.5447 | 0.0104 | 0.6700 | 0.7546 | 0.0122 | 0.6850 | 84  |
| 3  | 0.5555 | 0.0139 | 0.7000 | 0.9754 | 0.0157 | 0.7110 | 85  |
| 4  | 1.1347 | 0.0181 | 0.7180 | 1.2015 | 0.0201 | 0.7180 | 86  |
| 5  | 1.2701 | 0.0217 | 0.7170 | 1.3073 | 0.0230 | 0.7140 | 87  |
| 6  | 1.3665 | 0.0244 | 0.7000 | 1.3407 | 0.0251 | 0.6890 | 88  |

**DATA (CN(11,J),DH(11,J),ETA(11,J),J=1,9)/**

| 1  | 0.3582 | 0.0069 | 0.6000 | 0.4918 | 0.0086 | 0.6000 | 89  |
| 2  | 0.5519 | 0.0105 | 0.6120 | 0.6447 | 0.0123 | 0.6170 | 90  |
| 3  | 0.7365 | 0.0141 | 0.6210 | 0.8282 | 0.0159 | 0.6258 | 91  |
| 4  | 0.8837 | 0.0172 | 0.6250 | 0.9391 | 0.0186 | 0.6230 | 92  |
| 5  | 0.9715 | 0.0201 | 0.6009 |  |  |  | 93  |

**END**
$IBFTC ETAAB DECK
SUBROJTINE ETAAB (FAR,EM6,P6,ETA,ETAADS,ETAASV,P6DS,P6DSAV,AM6DS,A
1M6DSV,IDES,FAR7DS,FAR7SV)
DIMENSION FART(25),ETABRT(25),EM6T(7),DEL6M(7),P6T(14),DELP6(14)
DIMENSION X(3),Y(3)
DATA FART/.0390,.0585,.0732,.0878,.0976,.1171,.1268,.1463,.1619,
1.1834,.1951,.2195,.2439,.2927,.3415,.4146,.4634,.5366,.5341,.7317,
2.8293,.9268,1.000,1.0634,1.7/
DATA ETABRT/.9400,.9887,1.0193,1.0306,1.0227,.9572,.9377,.9207,
1.9354,.9626,.9773,1.0193,1.0532,1.0771,1.0781,1.0771,1.0747,1.0668,
21.0573,1.0510,1.0374,1.0192,1.000,.9626,.9151/
DATA EM6T/1.00,1.071,1.190,1.309,1.428,1.547,1.666/
DATA DELM6/0.03013,0.041,0.073,.110,.147,.187/
DATA P6T/.220,.2267,.250,.300,.3333,.3767,.4167,.500,.5833,.6667,
1.75,.8333,.9167,1.0/
DATA JELP6/-1.142,-1.25,-1.10,-.075,-.062,-.05,-.041,-.027,-.019,
1.-.013,-.008,-.004,-.0021,0./
IF (IDES.NE.1) GO T3 5
DO 1 K=1,25
1 ETABRT(K)=ETABRT(K)*ETAADS/ETAASV
DO 2 K=1,25
2 FART(K)=FART(K)*FAR7DS/FAR7SV
DO 3 K=1,7
3 EM6T(K)=EM6T(K)*AM6DS/AM6DSV
DO 4 M=1,14
4 P6T(M)=P6T(M)*P6DS/P6DSAV
ETAASV=ETAADS
P6DSAV=P6DS
FAR7SV=FAR7DS
AM6DSV=AM6DS
RETURN
CONTIVJE
N=0
IF (FAR.GT.0.067) GO TO 8
DO 6 J=1,25
6 IF (FAR.GE.FART(J)) N=J-1
IF (N.NEQ.0) N=1
IF (N.NE.24) N=23
DO 7 I=1,3
7 N=N+1
X(I)=FART(NN)
Y(I)=ETABRT(NN)
call PARABO (X,Y,FAR,ETA1)
go to 9
8 ETA1=-2.*FAR+.1948
9 M=0
DO 10 J=1,7
10 IF (EM5.GE.EM6T(J)) M=J-1
IF (M.NEQ.0) M=1
IF (M.NE.6) M=5
DO 11 I=1,3
11 MM=M-1+1
X(I)=EM6T(MM)
Y(I)=JEL6M(MM)
call PARABO (X,Y,EM5,Cor1)
L=0
DO 12 J=1,14
12 IF (P5.GE.P6T(J)) L=J-1
IF (L.NEQ.0) L=1
IF (L.NE.13) L=12
DO 13 I=1,3
13 LI=L-1+1
XI(I)=P6T(LL)
102
```
NN=N+1-1
XX(I)=PR(NN)
YY(I)=CFR(NN,LL)
CALL PARABO (XX,YY,PRATIO,ANS)
ZZ(L)=ANS
CF=YY(AR(M),AR(M+1),ARATIO,ZZ(1),ZZ(2))
RETURN
WRITE (6,7) ARATIO,AR(10)
DO 6 I=1,3
NN=N+1-1
XX(I)=PR(NN)
YY(I)=CFR(MN,10)
CALL PARABO (XX,YY,PRATIO,CF)
RETURN
C
CALL D
RETURN
C
C
C
FORMAT (8H0,ARATIO=F7.3,4X,26HOUT OF RANGE, USE DATA FOR ,F7.3)
END
```

$IBFTC INPUT DECK

SUBROUTINE INPUT(LJUNIT,LDUNIT,ID,D,IT)
DIMENSION D(I), IT(I)

' DEFINES ARGUMENTS AS ARRAYS

BIT STANDARD RETURN RETURN RETURN RETURN RETURN RETURN 1
NUMBER RETURN 1 2 3 4 5 6
0-2 $ A-Z 0-9( ) = OTH 7
3-4 RETPC RI OTH NLSF 8
C 5-6 DFR TINL OTH PS DF 9
C 7-8 O-9(+)OTH *,A-Z = -(*) 10
C 9 ,,,A-Z OTH 11
C 9-10 *, A-Z 0-9+* = 12
C 10 OTH $A-Z*)=/ 13
C 10-11 O-9,-OTH +'( $A-Z* )=/ 14
C 12 0-9 A-Z OTH 15
C 12-13 A-Z O-9 OTH 16
C 13-14 +* OTH 0-9 17
C 15-16 +* DE- OTH 18
C 17-18 F OTH 19
C 19-20 D E OTH 20
C 21 OTH , 21
C 21-22 A-Z OTH 22
C 23 * OTH 23
C 24 $ OTH 24
C 25 = OTH 25
C 26-27 ( A-Z OTH 26
C 28-29 ) , OTH 27
C 30 ) OTH 28
C 31 ) OTH 29
C ERROR CODE DESIGNATIONS
C ROUTINE TYPE
C 10 IXJTI 100 ILLEGAL CHARACTER
C 20 ITABL 200 NAME TOO LONG
C 30 INMBRI 300 TABLE FULL OR BAD
C 40 INAMEI 400 SCALING ERROR
C 50 BXJTI 500 NAME NOT IN TABLE
C 60 INAMEM 600 $DATA() INCOMPLETE
```

104
C 70 INPUT 700 FORMULA ILL FORMED 39
C 800 FUNCTION UNDEFINED 40
C MCNVRT = TYPE OF LEFT HAND VARIABLE 41
C KCNVRT = TYPE OF CURRENT VARIABLE 42
C ITYPE MEANING 43
C 1 REAL 44
C 2 INTEGER 45
C 3 DOUBLE PRECISION 46
C 4 TYPELESS OR NO CONVERSION 47
C 5 SUBROUTINE 48
C 6 FUNCTION 49
C FORMAT OF TABLE 50
C 1ST W3RD 012 3456 7-31 51
C TYPE NUMBER OF WORDS ADDRESS 52
C NEXT 1 TO 15 WORDS THE NAME, 4 CHARACTERS TO THE WORD 53
C DIMENSION IFT(31), IPTAB(21), ITAB(65) 54
C DIMENSION ANAME(15), IMAGE(80), IMAGE1(81), IPARAM(9) 55
C KSTACK(27), NAME(15), RVALUE(2), STACK(27) 56
C COMMON
C /ICOMVI/ VALUE *ICOMP *IFNTYP ,IMAGE1 ,IRADIX ,ISUB 59
C ,*XCH ,KCNVRT ,KCOUNT ,KDIFF ,KFLD1 ,KFLD2 60
C ,*LMCOMP ,LCNVRT ,LEVEL ,LFRT ,LOOK 61
C ,*MCNVRT ,MDIF ,MODALL ,MSOR 62
C ,*NAME ,*ERROR ,*NONEW ,*NOTARG 63
C ,*SMCHR ,*TEST ,*ERMARK 64
C /ICNSTI/ BLANK *BLANKS *DOLLAR ,*EDS 65
C ,*ICOMMA ,*IDOLAR ,*IFT ,*IPTAB ,*ITAB 66
C ,*KAMLO ,*KBPC ,*KBPW ,*KCPCD ,*KERTYP 67
C ,*KZERO ,*NOPRT ,*TAB1 68
C /IPARAM/ ABORT ,*IUNIT ,*KUNIT ,*LIMALF ,*LOCK ,*LOCX 69
C ,*NOLIST ,*STDIR ,*TRACE 70
C /ISTAKI/ STACK ,*ISTDIM ,*KSTACK ,*LEVLM 71
C INTEGER BLANK *BLANKS ,*EOS ,*IDOLAR ,*TAB1 ,*TRACE 73
C ,*CONTYP ,*STORED 74
C DOUBLE PRECISION STACK, VALUE 75
C LOGICAL ABURT 76
C ' STOP FOR ERROR 77
C LOGICAL DOLLAR ' TRUE IF A $T,$P, OR $C IS BEING PROCESSED 79
C LOGICAL ERMARK ' TRUE IF ANY ERROR HAS OCCURED 81
C LOGICAL LIMALF ' TERMINATE ALF CONSTANTS AT 15 WORDS. 83
C LOGICAL LOCK ' DO NOT STORE ANY INPUT DATA 85
C LOGICAL NOLIST ' DO NOT LIST INPUT CARDS 86
C LOGICAL NONEW ' DO NOT READ A NEW CARD. SET IN INPJ 89
C LOGICAL MODALL ' TRUE, RETURN ALL CHARACTERS 91
C FALSE, DELETE BLANKS AND SKIP FOR E 92
C LOGICAL SHORT ' DO NOT STORE ALPHABETIC AND RADIX CO 94
C DIRECTLY INTO D ARRAY 95
C LOGICAL SMCHR ' SAME CHARACTER IS TO BE USED OVER BY 97
C LOGICAL TEST ' STORE COMMA AT END OF CARD IF NEEDED 99
C END 100

105
EQUVALENCER{STACK,ISTACK),(VALUE,KVALUE,RVALUE),(NAME,ANAME)
EQUVALENCE{(ICOMNI,ISUB),(IMAGE,IMAGE1),(IPARAM,ABERT)
LOGICAL END2

C  
KUNIT=LIUNIT
KUNIT=LUNIT
C  
CALL DEBUGX
LIMALF = *TRUE.
DOLLAR = *FALSE.
NOLIST = *FALSE.
END2 = *FALSE.
MODALL = *FALSE.
ERMARK = *FALSE.
LOCK = *FALSE.
KERTYP = 0
FNTEP = -1
LOCK = 1
L=LEVLIM+3
DO 16 I=1,L
KSTACK(I)=O
LEVEL=O
LFRT=O
NOTARG = ISTDIM
LCNVRT = 3
C  
MCNVRT = 1
KDIF 1
MDIF = 1
1  
NENW = *FALSE.
SMCHR = *TRUE.
CALL ICHAR2{9470,24)
C  
LDOLLAR = KCOUNT
C  
NONE#=*TRUE.
C  
$DATA MUST BE ON A SINGE CARD
C  
D E O T H
CALL ICHARI($3,$9470,19)
C  
A-Z OTH
4 CALL ICHARI{4,$9470,26)
CALL ISUBI
CALL ICHAR2{9470,31)
C  
ERROR IF NO )
NONE#*=FALSE.
IT(I)=ISUB
C  
CALL DEBUG 2 (5HSETVD,IT(I))
C  
CALL DEBUG 2(6HKERSIN,KERSIN)
IF (ID.NE. IT(I)) GO TO 99
C  
RETURN BECAUSE WRONG DATA SET
IF (IT(2).LT. 0) GO TO 9370
C  
TABLE FULL OR BAD
GO TO 19
3 IF (END2) GO TO 99
END2 = *TRUE.
NENW = *FALSE.
5 CALL ICHAR2{1,9}
C  
PASS LETTERS AND **)
GO TO 5
9470 KERTYP = 470
GO TO 999
99 CONTINUE
C CALL DEBUG, 2(6HSTATMT, 99) 162
KCH = IDOLLAR 163
KCOUNT = LDOLLAR 164
C 165 'NEXT CALL BEGINS PROCESSING AT THIS CARD COLUMN
IF(ER4ARK) IT(1) = -IABS(IT(1)) 166
C 167 'WARN PROGRAMER OF POSSIBLE ERROR
RETURN 168
789 WRITE (L, 787) 169
787 FORMAT (44H ERROR HAS OCCURED AND ABORT IS .TRUE., STOP) 170
STOP 171
C BEGIN TO INPUT THE DATA 172
C STATEMENTS 19 AND 20 ARE THE SWITCHHOUSE 173
C CONTROL COMES HERE FOR NEW DIRECTION 174
C LFR7 = 0 INDICATES THAT THE PREVIOUS TASK WAS COMPLETED. 175
19 CONTINUE 176
C CALL DEBUG2(6HSTATMT, 19) 177
TEST = .FALSE. 178
GO TO 21 179
120 DOLLAR = .FALSE. 180
20 CONTINUE 181
C CALL DEBUG2(6HSTATMT, 20) 182
TEST = .TRUE. 183
C ENTER ICHAR4 LOADS LCOMP WITH TAB NO 10 AND RETURNS ON TAB NO 7 184
C OTH , 'A-Z = -(*/ 185
21 CALL ICHAR4 (31, 32, 7, 10) 186
C NOW WE TEST LCOMP ON EACH OF THE 3 POSSIBLE RETURNS. THIS IS A 12 WAY 187
C 0-9, + $ ) 188
GO TO (203, 510, 64, 430), LCOMP 189
C , ' A-Z = 190
31 GO TO (460, 202, 530, 450), LCOMP 191
C - (*/ 192
32 GO TO (520, 420, 440, 441), LCOMP 193
64 IF (LFR7 .NE. 0) GO TO 9770 194
LDOLLAR = KCOUNT 195
C 196 ' CARD COLUMN OF LAST DOLLAR SIGN
C CALL DEBUG 2 (6HSTATMT, 64) 197
TEST = .FALSE. 198
CALL ICHAR4 (9170, 9170, 3, 5) 199
C 200 DE T P C 201
GO TO (99, 100, 150, 630), LCOMP 202
100 CALL ITABLIT(1IT) 203
IF (KERTYP) 999, 120, 999 204
205 C 206 C 207 C 208 C 209 C 210 C 211 C 212 C 213 C 214 C 215 C 216 C 217 C 218 C 219 C 220 C 221 C 222 C 223
C * CONVERT FROM DP TO TYPE OF VARIABLE
   IPT=IFLD(7,25, IPTAB(LOOK))
   IPARAM(IPT) = KVALUE
   ) , OTH
   CALL ICHAR1($151,$9170,28)
   IF(TRACE.GT.0) CALL DEBUGX
C GO TO 120
     * IF TRACE TURNED ON PUT ON HEAD
   GO TO 120
     * GO TO SWITCH HOUSE TEST FOR INSERTIO
C C CONSTANTS = LOGICAL,NUMERIC,ALPHAMERIC,RADIX.
   201 ASSIGV 220 TO CONTYP
C F Radix
   IFNTP = -1
   GO TO 210
   202 ASSIGV 230 TO CONTYP
C C DO TO 210
   203 ASSIGV 250 TO CONTYP
C C NUMERIC,LOGICAL
C C ALL CONSTANTS 210
   210 LOP = 0.
     SHORT = .TRUE.
     MSTOR = 0.
   CALL DEBUG2(6HST,CONT,210)
   IF(LEVEL.EQ.0) LEVEL=3
   IF(KSTACK(LEVEL).EQ.1) GO TO 9770
C GO TO CONTYP,(220,230,250)
C C RADIX CONSTANTS 220
   ( A-Z ) OTH
   220 CALL ICHAR2($9170,30)
   CALL ISUBI
     NAME (2) = 0
   221 CALL ICHAR2($2215,29)
C GO TO 9170
   2215 CALL ICHAR2($9170,12)
C C ERROR IF NO COMMA AFTER BASE NUMBER
   222 IRADIX = ISUB
     SMCHR = .TRUE.
     CALL ISUBI
     MSTOR = MSTOR +1
     NAME(MSTOR) = ISUB
     IF (MSTOR.GE.15) GO TO 225
   OTH
C C ERROR IF NO NUMBER
   224 CALL ICHAR1($223,$9170,28)
     IRADIX = 10
     GO TO 240
   225 ASSIGV 224 TO NEXT
     GO TO 241
C C HOLLERITH CONSTANTS 230
   230 MODALL = .TRUE.
   TEST = .FALSE.
     NAME(2) = BLANKS
231  CALL IYAMER
     IF(.NOT.MODALL) GO TO 240
C      *END OF CONSTANT
C 234  ASSIGN 231 TO NEXT
     GO TO 241
C 240  ASSIGN 260 TO NEXT
     IF (MSTOR-2) 242,2405,241
C 2405  IF (LFRT.EQ.1) GO TO 242
     CALL ICHAR2($2406,29)
     SMCHR = .TRUE.
     GO TO 242
C 2406  SMCHR = .TRUE.
C 241  SHORT = .FALSE.
     IF (LFRT.NE.0) GO TO 265
C 242  CALL DEBUG2(6HST 241,0)
C 243  IF (SHORT) GO TO 255
     DO 245 I= 1,MSTOR
     IF (LJCK) GO TO 245
     D(IOCK) = ANAME(I)
C 245  LOCK = LOCK + 1
C 246  GO TO NEXT, (224,231,260)
C 250  ' ALF OR ) AND RADIX
C 251  ' MORE THAN 15 ELEMENTS IN RADIX FIELD
C 252  ' MORE THAN 15 ELEMENTS IN ALF FIELD
C 253  CONTROL NUMBER + LOGICAL
C 254  TEST = .TRUE.
     CALL INMBRI
C 255  CALL IARITI
     LFRT = 1
     IF (KERTYP) 999,20,999
C 260  RESET STACK BECAUSE IT WAS NOT USED
C 262  LEVEL = 0
C 265  KERTYP = 270
     GO TO 999
C 270  KERTYP = 171
     GO TO 999
C 275  ' SKIP COMMA
C 299  KERTYP = 171
     GO TO 999
C 300  IF (KERTYP) 999,20,999
C 304  TEST EMPTY PARENTHESES
C 305  IF ((KSTACK( LEVEL-1)+KSTACK( LEVEL-2)).NE.0) GO TO 997
C 306  CALL DEBUG 2(6HSTA (),400)
C 307  CALL STACKP
C 308  PRINT STACK
C 309  LEVEL=LEVEL-3
C 310  NEXT
C EMPTY FUNCTION ARGUMENT IS NOT A CURRENT LEFT SIDE
405 VALUE=0.
GO TO 403
404 VALUE(1) = D (LOCK)
VALUE(2) = D(LOCK+1)
CALL ICNVTI(MCNVRT,3)
403 CALL IARITI
GO TO 20
C(**************************
420 IF (LEVEL.EQ.0) LEVEL=3
C CALL DEBUG 2(6HSTAT (,420)
422 LOP=0
LFRT = 1
LEVEL=LEVEL+3
C CALL STACKP
PRINT STACK
C(**************************
430 IF (LEVEL.LT.6) GO TO 997
C CALL DEBUG 2(6HSTAT (,430)
C CALL STACKP
PRINT STACK
431 IF (KSTACK( LEVEL-1) 400, 432, 997
432 DO 433 I=1,3
VALUE=STACK( LEVEL)
KSTACK( LEVEL)=0
LEVEL=LEVEL-1
CALL IARITI
IF (KERTYP) 999, 20, 999
C****************************************(KOP=2)
440 KOP=2
C CALL DEBUG 2(6HSTAT *,440)
GO TO 445
C///////////(/KOP=3)
441 KOP = 3
C CALL DEBUG 2(6HSTAT /,441)
445 IF (LJP NE.0) GO TO 997
C CALL DEBUG 2(6HSTATMT,445)
C CALL STACKP
PRINT STACK
LOP = 1
LFRT = 1
IF (LEVEL.EQ.0) LEVEL=3
IF (KSTACK( LEVEL).NE.1) GO TO 997
444 VALUE=STACK( LEVEL)
KSTACK( LEVEL)=0
LEVEL=LEVEL-1
CALL IARITI
IF (KERTYP*NE.0) GO TO 999
446 KSTACK( LEVEL)=KOP
LEVEL=LEVEL+1
C CALL STACKP
PRINT STACK
GO TO 19
C=-----------------------
450 IF (LEVEL NE.0) GO TO 997
C CALL DEBUG 2(6HSTAT =,450)
451 LOP=0
LOCX = LOOK
110
HCNRVR = KCNVRT
HCOR = LDH
LCMVRV = 3
LEVEL=3
LFRT=0
KSTACK(3)=0
C CALL DEBUG2(WHLOCX,LOCX)
C CALL DEBUG2(WHMVCNR,MVCNR)
GO TO 19

C+++++++++++++++400 CONTINUE
C CALL DEBUG 2(WHSTAT,460)
C CALL STACKP
C INCREMENT LOCX WITHOUT STORING
C LFRT WILL BE ZERO AFTER $ EXPRESSION OR FOR CONSECUTIVE COMMAS
C GO TO 19
C
C++++++++++++++++++440  IF KSTACK(3) .NE.1) GO TO 997
C DO 471 I = 1,2
VALUE=STACK( LEVEL)
KSTACK( LEVEL)=0
LEVEL=LEVEL-1
C CALL IARITI
IF KERRYP .NE.0) GO TO 999
C IF KSTACK(1) .EQ.0) GO TO 999
KSTACK(1)=0
C CALL IARITI (LCNRVT,MCNRVT)
C CALL DEBUG (WHSTAT,510)
GO TO 521
C++++++++++++++++++520  KOP=4
C CALL DEBUG 2(WHSTAT,520)
C CALL DEBUG 2(WHSTAT,520)
C CALL DEBUG 2(WHSTAT,520)
C CALL DEBUG 2(WHSTAT,520)
C CALL DEBUG 2(WHSTAT,520)
C CALL DEBUG 2(WHSTAT,520)
C CALL DEBUG 2(WHSTAT,520)
528 CALL IARITI
   IF (KERTYP .NE. 0) GO TO 999
527 KSTACK( LEVEL) = KOP
   LEVEL = LEVEL + 2
   GO TO 19

C ABCDEFGHJKLMNOPQRSTUVWXYZABCDEFHIJKLMNOPQRSTUVWXYZABCDEFHJL

530 LOP = 0
   C CALL DEBUG 2(16HSTAT A, 530)
      IF (.NOT. (LEVEL .NE. 0).AND. (KSTACK( LEVEL) .EQ. 1)) GO TO 997
535 TEST = .TRUE.
      CALL INAMEI($999, 0, IT)
      IF (IFNTYP) 531, 621, 640
531 LFRT = 1
      IF (LEVEL .NE. 0) GO TO 540
532 STACK( 3) = VALUE
533 CALL DEBUG 2(16HSTATMT, 532)
      KSTACK( 3) = 1
      CALL STACKP ' PRINT STACK
534 GO TO 20
540 CONTINUE
   C CALL DEBUG 2(16HSTATMT, 540)
   CALL IARITI
      IF (KERTYP) 999, 20, 999
   C PROCESS SUBROUTINES AND FUNCTIONS
   C COMMA SEPARATING FUNCTION ARGUMENTS
   600 IF (KSTACK( LEVEL - 3) .LT. 6) GO TO 997
   C CALL DEBUG 2(16HSTAT F, 600)
   601 IF (KSTACK( LEVEL) - 1) 602, 603, 997
   602 STACK( LEVEL) = 0.
   603 DO 610 I = 1, 2
      VALUE = STACK( LEVEL)
      KSTACK( LEVEL) = 0
      LEVEL = LEVEL - 1
      CALL IARITI
      IF (KERTYP .NE. 0) GO TO 999
   610 CALL STACKP ' PRINT STACK
   611 KSTACK( LEVEL) = KSTACK( LEVEL - 1) + 1
      LEVEL = LEVEL + 3
   C CALL STACKP ' GO TO 19
549 CALL ICHAR2($9170, 12)
550 CONTINUE
   C SKIP *ALL* IN $CALL
   C JTH OPERATORS
      CALL ICHAR2($9170, 12)
   631 CONTINUE
      DOLLAR = .FALSE.
      TEST = .TRUE.
      CALL INAMEI($999, 0, IT)
      IF (IFNTYP .NE. 1) GO TO 640
      KVALUE = -KVALUE
   640 CONTINUE
      ' INDICATE THAT NO RESULT IS TO BE STORED FOR SJB
      FUNCTION NAME
   621 CONTINUE
      ' FUNCTIONS
   620 CALL DEBUG2(16HSTAT FNC, 621)
   621 CALL DEBUG2(16HIFNTYP, IFNTYP)
C CALL DEBUG2(6HKVALUE,KVALUE)
IFNTYP = -1
IF (LEVEL.EQ.0) LEVEL = 3
IF(NDTARG . GT. LEVEL) NOTARG = LEVEL
LFRT = 1
STACK(LEVEL) = KVALUE
KSTACK( LEVEL) = 6
LEVEL = LEVEL + 3
C CALL STACKP
C 
PRINT STACK
C CALL ICHAR2($622,30)
GO TO 19
C 
* IF THERE ARE ARGUMENTS
622 SMCHR = .TRUE.
IF (KVALUE .LT. 0) LEVEL = LEVEL - 3
GO TO 405
640 IF (IFNTYP - 2) .GT. 7770 .LT. 700
C LOCK THE SUBROUTINE THAT STORES LOCX THE SUBSCRIPT OF D ARR
700 IF (IFNTYP .NE. 3) GO TO 9770
IFNTYP = -1
IF (LEVEL .NE. 0) GO TO 9770
LFRT = 0
CALL ICHAR2($9170,30)
C 
* ERROR IF NO ( AFTER LOCX
704 ASSIGN 704 TO NEXT
C 
A-Z OTH
702 CALL ICHAR2($9170,9)
C CALL INAMEI($999,D,IT)
IF (IFNTYP .LT. -1) GO TO 9770
GO TO NEXT,(704,708)
704 LOOKX = LOOK
C 
OTH 
CALL ICHAR2($706,29)
C 
* ERROR IF NO COMMA
706 GO TO 9770
706 ASSIGNED 708 TO NEXT
GO TO 702
708 KVALUE = LOCX - LOOKX + 1
CALL ICNVTI(2,KCNVRT)
C 
* CONVERT FROM INT TO TYPE OF 2ND ARG
710 IF (LOCX) GO TO 710
D(LOOK) = RVALUE(1)
710 CALL ICHAR2($9170,31)
C 
* ERROR IF NO
C TEST = .TRUE.
C OTH 
CALL ICHAR2($19,29)
C 
* SKIP THE COMMA THAT MUST FOLLOW
9170 KERTYP = 170
GO TO 999
9370 KERTYP = 370
GO TO 999
997 CONTINUE
9770 KERTYP = 770
GO TO 999
999 CALL IERROR1
LFRT = 0
LOP = 0
NAME(1) = 0
GO TO 19
C 
* NOW GO TO SWITCH HOUSE . . GOOD
$IBFTC BLOCK DECK
BLOCK DATA
C COMMON
  */ICOMM1/ VALUE *ICOMP,*IFNTYP,*IMAGE1,*IRADIX,*ISUB
  *KCH,*KCHNVRT,*KCOUNT,*KDFJF,*KFID1,*KFID2
  *LCOMP,*LCMVRT,*LEVEL,*LFRT,*LOOK
  *MCNVRT,*MDIF,*MODALL,*MSTOR
  *NAME,*NERRDR,*NONEW,*NOTARG
  *SMCHR,*TEST,*ERMARK
  */ICNST/ BLANK,*BLANKS,*DOLLAR,*EOS
  *ICOMMA,*IDOLAR,*IFT,*IPTAB,*ITAB
  *KAM10,*KBPC,*KBPW,*KCPD,*KERTYP
  *KZERO,*NOPRT,*TAB1
  */IPARAM/ ABORT,*KUNIT,*KUNIT,*LIMALF,*LOCK,*LOCH
  *NLIST,*NSTDIR,*TRACE
  */ISTAKI/ STACK,*ISTDIM,*KSTACK,*LEVELM
C INTEGER BLANK,*BLANKS,*EOS,*IDOLAR,*TAB1,*TRACE
DOUBLE PRECISION STACK,VALUE
LOGICAL ABORT,DOLLAR,ERMARK,LIMALF,LOCK,NLIST,NONEW,MODALL,
*NSTDIR,*SMCHR,*TEST
DATA VERROR,TRACE/0,0/
DATA ITAB/65*/20572347824/
DATA ITAB(49)/-11890146992/ 27
  *, ITAB(44)/11923699376/, ITAB(61)/29355228304/
  *, ITAB(29)/29053238816/, ITAB(12)/-3484760752/
  *, ITAB(28)/-11821997744/, ITAB(17)/-11903778480/
  *, ITAB(33)/-12041137840/, ITAB(45)/-12192136880/
  *, ITAB(50)/-12208914096/, ITAB(13)/-12041137840/
  *, ITAB(60)/-11957255792/, ITAB(11)/-11890146992/
  *, ITAB(1)/-20572347824/, ITAB(2)/-29063724720/
  *, ITAB(3)/29063724720/, ITAB(4)/-29063724720/
  *, ITAB(3)/29063724720/, ITAB(6)/-29063724720/
  *, ITAB(7)/29063724720/, ITAB(10)/-29063724720/
  *, ITAB(8)/20572347824/, ITAB(20)/18961735088/
  *, ITAB(21)/17335053258/, ITAB(22)/173350565296/
  *, ITAB(23)/23256571312/, ITAB(26)/20035476912/
  *, ITAB(36)/22182960560/, ITAB(38)/22182960560/
  *, ITAB(40)/18424854176/, ITAB(42)/19498606000/
  *, ITAB(51)/22719831472/, ITAB(52)/1788773120/
DATA BLANK,BLANKS,EOS,ICOMMA,IDOLAR,KAM10,KBPC,KBPW,KCH,KCPD,
  *KZERO,NOPTRT,TAB1,IMAGE1(2),IMAGE1(3)/
  *48,-17997958192,10,59,43,68
  *,7,6,36,43,89,49
  *,0,-1155507248,64,-1155507248,-1155507248/
DATA ABORT,IMAGE1(1),IMAGE1(81),IRADIX,ISTDIM,
  *KCOUNT,KDFJF,KSTACK,LEVELM,NSTDIR,STACK/
  *FALSE,*1HE,*1H,10,27,0,1,27+0,24,*TRUE,*27*0,00/
DATA IFT/54
  0,31,-17716740112,21051935792,-17716740128
  23481748528,-17716740144,-3864988720,-17716740160
  -17716740176,-20009401392,-17176740192,1913857200
  -17716740208,-821767088,-17716740224,-8428915760
  -17716740240,-8433110064,-9126805536
  -9954237936,-9126805552,-3863968816,-9126805696
  -3863883248,0
DATA IPTAB/
  0, 21, -536870928, 18565733616, -536870976, 62
  -3650164950, -536870992, -3863882800, -536871024, -60155240199, 63
  -535971040, -6221022825, 177167401280, -2580698739, 64
  -2793802547, 17716740192, -3863968816, 65
  0/
END

$IBFTC IARITI DECK
C ARITHMETIC OPERATIONS FOR INPUT R. J. A. S. 2
C SUBROUTINE IARITI
C CALLED FROM INPUT
DIMENSION IFT( 31), IPTAB( 21), ITAB( 65)
DIMENSION ANAME(15), IMAGE(80), IMAGE(81), IPARAM(9)

  * KSTACK(27), NAME(15), VALUE(2), STACK(27)
C COMMON
  */ICOMNI/, VALUE, ICOMP, IFNTYP, IMAGE1, IRADIX, ISUB
  */KCH, KCNVRT, KCOUNT, KDIFF, KFLD1, KFLD2
  */LCOMP, LCNVRT, LEVEL, LFRT, LOOK
  */MCNVRT, MDIFF, MODALL, MSTOR
  */IMAGE, NERROR, NOFIOX, NOTARG
  */SMCHR, TEST, ERMARK
  */ICNSTI/, BLANK, DOLLAR, EOS
  */ICOMMA, IDOLAR, IFT, IPTAB, ITAB
  */KAM10, KBPC, KBPW, KPCD, KERTYP
  */KZERO, NORT, TAB1
  */IPARAM/, ABORT, KUNIT, KUNIT, LIMADF, LOCK, LOCK
  */NOLIST, NSTDIR, TRACE
  */ISTACK/, STACK, ISTDIM, KSTACK, LEVLI
C INTEGER BLANK, BLANKS, DOLLAR, IDOLAR, TAB1, TRACE
DOUBLE PRECISION STACK, VALUE
LOGICAL ABORT, DOLLAR, ERMARK, LIMADF, LOCK, NOLIST, NOFIOX, MODALL,

  NSTDIR, SMCHR, TEST
  EQUIVALENCE (STACK, ISTACK), (VALUE, KVALUE, RVALUE), (NAME, ANAME)
  EQUIVALENCE (ICOMNI, ISUB), (IMAGE, IMAGE1), (IPARAM, ABORT)
C CALL DEBUG(6HIARITI)
C CALL STACK
  PRINT STACK
C
  IF (KERTYP NE 0) GO TO 100
  IF (LEVEL LE 0) OR (LEVEL GT LEVLI) ) GO TO 120
  BRANCH ON KSTACK( LEVEL)= -K 0, 1, 2, 3, 4, 5, 6, 7, 8 AND UP
  IF (K LE 8) GO TO 60
  K=MAX(1, KSTACK( LEVEL)+2)
  IF (K NE 8) GO TO 60
3  GO TO (120, 90, 120, 20, 30, 40, 50), K
  VALUE=STACK( LEVEL)*VALUE
  GO TO 90
  VALUE=STACK( LEVEL)/VALUE
  GO TO 90
  VALUE=STACK( LEVEL)+VALUE
  GO TO 90
  VALUE=STACK( LEVEL)-VALUE
  GO TO 90
  LEVEL= LEVEL-K+9
  DO 61 I=LEVEL1, LEVEL

115
61 KSTACK( I ) = 0
   LEVEL = LEVEL1 - 1
C CALL DEBUG2( 15HLEVEL , LEVEL )
C CALL DEBUG2( 15HVALUE , VALUE )
   IF ( LOCK ) GO TO 62
   CALL IXOTI( VALUE , STACK( LEVEL ) )
62 CONTINUE
C CALL DEBUG2( 15HVALUE , VALUE )
   IF ( LEVEL .LE. NOTARG ) NOTARG = ISTDIM
   IF ( STACK( LEVEL ) .LT. 0.0D0 ) GO TO 110
   GO TO 90
90 STACK( LEVEL ) = VALUE
   KSTACK( LEVEL ) = 1
100 CONTINUE
C CALL STACKP
   PRINT STACK
C CALL DEBUGR
   RETURN
110 LFRT = 0
C   * SPECIAL TREATMENT FOR SUBROUTINES
C CALL DEBUG2( 6HSTATWT , 110 )
   KSTACK( LEVEL ) = 0
   LEVEL = 0
C   OTH
   CALL ICHAR2( $100 , 29 )
   KERTYP = 150
   GO TO 100
120 KERTYP = 750
   GO TO 100
   END

$IBFTC ICHAR4 DECK
SUBROUTINE ICHAR4( * , * , LIST1 , LIST2 )
DIMENSION IFIT( 31 ) , IPTAB( 21 ) , ITAB( 65 )
DIMENSION ANAME( 15 ) , IMAGE( 80 ) , IMAGE1( 81 ) , IPARAM( 9 )
   , KSTACK( 27 ) , NAME( 15 ) , RVALUE( 2 ) , STACK( 27 )
C COMMON
   /ICOMNI/ VALUE , ICOMP , IFNTYP , IMAGE1 , IRADIX , ISUB
   , KCH , KCVRT , KCOUNT , KDIF , KFLD1 , KFLD2
   , LCOMP , LCVRT , LEVEL , LFRT , LOOK
   , MCNRVT , MDIF , MODALL , MSTOR
   , NAME , NERR , NEW , NOTARG
   , SMCHR , TEST , EMARK
   /ICNSTI/ BLANK , BLANKS , DOLLAR , EOS
   , ICOMMA , IDOLAR , IFT , IPTAB , ITAB
   , KAMIO , KBPC , KBPW , KPCD , KERTYP
   , KZERO , NOPRNT , TAB1
   /IPARAM/ ABORT , KUNIT , KUNIT , LIMALF , LOCK , LOCKX
   , NOLIST , NSTDIM , TRACE
   /ISTAKI/ STACK , ISTDIM , KSTACK , LEVIM
C INTEGER BLANK , BLANKS , EOS , IDOLAR , TAB1 , TRACE
   , COMPRA
DOUBLE PRECISION STACK , VALUE , IEBUA , DEBNA( 2 )
DATA DEBNA/ 6HICHR , SHICHR/
LOGICAL ABORT , DOLLAR , EMARK , LIMALF , LOCK , NOLIST , FNEW , MODALL ,
   NSTDIM , SMCHR , TEST
EQUIVALENCE (STACK,ISTACK),(VALUE,KVALUE,RVALUE),(NAME,ANAME)  
EQUIVALENCE (ICOMMI,ISUB),(IMAGE,IMAGE1),(IPARAM,ABORT)  
LOGICAL GTC  
C  
TRUE IF NEW CARD WAS READ  
C  
RE-PROCESS THE SAME CHARACTER AS LAST  
KFLD2=LIST2  
IBITS=2  
ASSIGN 36 TO COMPR  
IDEFUN=4  
GO TO 10  
ENTRY ICHAR1(*,*,LIST1)  
IBITS=2  
IDEFUN=3  
GO TO 4  
ENTRY ICHAR2(*,LIST1)  
IBITS=1  
IDEFUN=2  
4  
KFLD2=-1  
ASSIGN 37 TO COMPR  
IF(MODALL)GO TO 40  
10  
ASSIGN 20 TO IGETR  
12  
KFLD1=LIST1  
IDEFUN=DEBGN(1)  
ASSIGN 110 TO NEXT  
GOTC=.FALSE.  
IF(ISCHR)GO TO 35  
C  
GO TO 200  
GET CHARACTER RETURN TO 20  
20  
IF(KCH.EQ.BLANK)GO TO 200  
BYPASS BLANKS  
C  
IF(KCH.NE.EOS)GO TO 30  
EOS= END OF STATEMENT CHARACTER  
C  
ASSIGN 200 TO INERW  
C  
PROVIDE TO GET FIRST CHARACTER FROM NEXT C  
30  
IF(ITEMP.AND.GOTC)GO TO 100  
ITEMP=ITAB(KCH+1)  
ICOMP=IFLD(LIST1,IBITS,ITEMP)  
GO TO COMPR,(36,37)  
36  
LCOMP=IFLD(LIST2,2,ITEMP)+1  
CONTINUE  
IF(TRACE. LT.4)GO TO 38  
C  
CALL DEBUG2(IDEFUN,IDEFUN)  
38  
SMCHR=.FALSE.  
IF(ICOMP-1)381,382,383  
381  
RETURN  
382  
RETURN1  
383  
RETURN2  
40  
ASSIGN 35 TO IGETR  
GO TO 12  
C  
SPECIAL ROUTINE TO INSERT COMMA AT END OF CARD IF  
C  
THE NEXT CARD BEGINS WITH $ OR A LEFT MEMBER  
C  
TO THIS ROUTINE, A LEFT MEMBER BEGINS WITH A-Z  
C  
FOLLOWED BY ANY OF 0-9(A-Z FOLLOWED BY =  
100  
IDEFUN=DEBGN(2)  
ITEMP=ITAB(KCH+1)  
ICOMP=IFLD(0,3,ITEMP)  
GO TO NEXT,(110,120)
* 110 FOR 1ST CHR ON NEW CARD, 120 FOR FOLL

110  NO\\NEd=.TRUE.,
     ASS\NISN 120 TO NEXT
120  $ A-Z 0-9() = OTH
     GO TO (130,200,140,140,140), ICMP
130  K\NCOUNT=0
     KCH=\NCOMMA
     NO\\NEd=.FALSE.,
     \NGOTCO = .FALSE.,
     GO TO 35
140  K\NCOUNT=0
     NO\\NEd=.FALSE.,
     \NGOTCO = .FALSE.,
     GO TO 200
     ' COMMA IS CHARACTER RETURNED
     \NCOUNT=K\NCOUNT+1
     ' FIRST \NON-BLANK CHARACTER ON CARD IS RETURNED
     ' ROUTINE TO GET NEXT CHARACTER
     IF(K\NCOUNT.LT.KCPCD)GO TO 210
     ASS\NISN 210 TO \NINWR
     GO TO 300
     ' CARD COLUMN OF NEW CHARACTER
     K\NCOUNT=K\NCOUNT+1
     K\NCOUNT=IMAGE(K\NCOUNT)
     GO TO IG\NTR,(20,35)
     35 IS USED ONLY FOR MODALL .TRU
     ' ROUTINE TO PRINT OLD CARD AND READ NEXT
     IF(NOS\NEd=WIS, TRUE, , STORE TAB\IS IN KCH AND RETURN
     ' GET CARRIAGE CONTROL (IMAGE(81) IS A BLANK)
     IF((\NCH,EQ,NOPRNT).OR.NOL\NIS), GO TO 305
     WRITE(KUNIT,398)KCH,IMAGE
     398 \NFORMAT(1A1,5x,BO\NIS1)
     305 \NREAD(K\NISUNIT,399)K\NCH,IMAGE
     399 \NFORMAT(BO\NIS1)
     \NGOTC=.TRUE.,
     K\NCOUNT=0
     GO TO \NINWR,(200,210)
     210 K\NCOUNT=K\NCOUNT+1
     GO TO 30
     ' TAB\IS IS RETURNED, INDICATES END OF C

END

$\NBF\NIS IC\N\NIS T\N DECK

COMMON IC\N\NISVT(I\N\NISFROM,\NITO)
     DIM\NIS\NION I\N\NISFT(31), I\N\NISPA\NIS21), I\N\NISBT(65)
     DIM\NIS\NION AN\NISNAME(15), IMAGE(80), \NISIMAGE1(81), \NISPARAM(9)
     *   , \NISKST\NISSTACK(27), \NISNAME(15), \NISVALUE(2), \NISSTACK(27)

IC\N\NISVT / VALUE , IC\N\NISMP , IFNT\NISYP , IMAGE1 , IR\NISDIX , ISUB
     , K\NCH , K\NISCN\NISVT , K\NISCOUNT , K\NISDIF , KFLD1 , KFLD2
* LCOMP  LCNVRT  LEVEL  LFRT  LOOK
* MCNVRT  MDIF  MODALL  MSTJR
* NAME  VERROR  NONE  NDARG
* SMCHR  TEST  EMARK
*/ICNSTI/ BLANK  BLANKS  DOLLAR  EOS
* ICOMMA  IDOLAR  IFT  IPTAB  ITAB
* KAMIO  KBPC  KBPW  KCPCD  KERTYP
* KZERO  NOPRT  TAB1
*/IPARAM/ ABORT  KUNIT  KOUNIT  LIMALF  LOCK  LOCK
* NLIST  YSDIR  TRACE
*/ISTAKI/ STACK  ISTDIM  KSTACK  LEVLIM
C
INTEGER BLANK  BLANKS  EOS  IDOLAR  TAB1  TRACE
DOUBLE PRECISION STACK, VALUE
LOGICAL ABORT, DOLLAR, EMARK, LIMALF, LOCK, NLIST, NONE, MODALL,
* NSTDIR, SMCHR, TEST
EQUIVALENCE (STACK, ISTACK), (VALUE, KVALUE, RVALUE), (NAME, ANAME)
EQUIVALENCE (ICOMNI, ISUB), (IMAGE, IMAGE1), (IPARAM, ABORT)
DIMENSION NTYPE(4)
DOUBLE PRECISION NTYPE
DATA NTYPE / 6HREAL, 6HINT, 6HDSP, 6HNOCONV/
IF((IFROM .LE. 0).OR.(IT0 .LE. 0)) GO TO 100
IF (IFROM - 4) 1,99,100
1 IF (IT0 - 4) 2,99,100
2 ITOM2 = IT0 - 2
3 IF (IFROM-2) 3,4,5
4 IF (ITOM2) 99,10,20
5 IF (ITOM2) 30,99,40
6 IF (ITOM2) 50,60,99
10 KVALUE = RVALUE(1)
GO TO 99
20 VALUE = RVALUE(1)
GO TO 99
30 RVALUE(1) = KVALUE
GO TO 99
40 VALUE = KVALUE
GO TO 99
50 RVALUE(1) = VALUE
GO TO 99
60 KVALUE = VALUE
99 CONTINUE
C CALL DEBUG3 (6HICNVTI,0,DO,3)
C CALL DEBUG3 (INTYPE(IFROM), NTYPE(IT0), 3)
RETURN
100 WRITE(10UNIT,101) IFROM, IT0
101 FORMAT(35H ARGUMENTS OF ICNVTI BAD, IFROM = ,I13,8H, IT0 = ,I13,
* 41H (1 TO 4 ALLOWABLE). CHECK IPTAB IN COMNJ)
GO TO 99
END

$IBFTC IERORI DECK
SUBROUTINE IERORI
DIMENSION IFT( 31), IPTAB( 21), ITAB( 65)
DIMENSION ANAME(15), IMAGE(80), IMAGE1(81), IPARAM(9)
   ,KSTACK(27), NAME(15), RVALUE(2), STACK(27)
C
COMMON

*ICOMNI/ VALUE ,ICOMP ,IFNTYP ,IMAGE1 ,IRADIX ,ISUB 5
* ,KCH ,<CNVRT ,KCOUNT ,KCRN ,KFLD1 ,KFLD2 8
* ,LCOMP ,LNCNVRT ,LEVEL ,LFRT ,LOOK 9
* ,MCNVRT ,MDIF ,MODALL ,MSTOR 10
* ,NAME ,NERROR ,NONEW ,NOTARG 11
* ,SMCHR ,TEST ,ERMARK 12
*/ICNSTI/ BLANK ,BLANKS ,DOLLAR ,EOS 13
* ,ICOMMA ,IDOLAR ,IFT ,IPRTAB ,ITAB 14
* ,KAM10 ,KBPC ,KPW ,KCPDC ,KERTYP 15
* ,KZERO ,NOPRTN ,TAB1 16
*/IPARAM/ ABORT ,KIUNIT ,KOUNIT ,LIMALF ,LOCK ,LOCX 17
* ,NLIST ,NSTDIR ,TRACE 18
*/ISTAK/ STACK ,ISTDIM ,KSTACK ,LEVLIM 19

C
INTEGER BLANK ,BLANKS ,EOS ,IDOLAR ,TAB1 ,TRACE 21
DOUBLE PRECISION STACK ,VALUE 22
LOGICAL ABORT ,DOLLAR ,ERMARK ,LIMALF ,LOCK ,NLIST ,NONEW ,MODALL 23
*NSTDIR ,SMCHR ,TEST 24
EQUIVALENCE (STACK ,ISTACK) , (VALUE ,KVALUE ,RVALUE) , (NAME ,ANAME) 25
EQUIVALENCE (ICOMNI ,ISUB) , (IMAGE ,IMAGE1) , (IPARAM ,ABORT) 26
INTEGER H(14,7) 27
EQUIVALENCE (H(1,1),H1) , (H(1,2),H2) , (H(1,3),H3) , (H(1,4),H4) 28
,(H(1,5),H5) , (H(1,6),H6) ,(H(1,7),H7) 29
DATA KA/1H+/* 30
DATA B/1H/ 31
DATA C/1H+/ 32
DATA D/1H+/ 33
INTEGER MSGTYP(2,2) , H1(4) , H2(4) , H3(4) , H4(4) , H5(4) , H6(4) , H7(4) 34
DATA MSGTYP(1,1)/24H*DIAGNOSTIC**ERROR** / 35
DATA 41(1) /24HINAPPROPRIATE CHARACTER / 36
DATA 42(1) /24HNAME TOO LONG / 37
DATA 43(1) /24HTABLE FULL OR DESTROYED / 38
DATA 44(1) /24HSSECRET INCOMPLETE OR MISSING / 39
DATA 45(1) /24HNAME NOT IN TABLE / 40
DATA H5(1) /24HFUNCTN OR SUB NOT ABAND/41
DATA 47(1) /24HFORMLA ILL-FORMED / 42
C
CALL DEBUGC(6HIER0I) 43
NONEW = .FALSE. 44
NERROR = NERROR + 1 45
C
ERMARK = .TRUE. 46
C
IOP = 0 47
C
! COUNT DIAGNOSTICS 48
! WARN PROGRAMER OF POSSIBLE ERROR 49
! COUNT OPERATORS +,-,=(* 50
C
MODE = 1 51
IF(NOTARG.LT.LEVEL) MODE = 2 52
KCI=K3 53
IF(KCJNT.EQ.0) KCI=KC 54
L = KCOUNT + 1 55
MODALL = .FALSE. 56
TEST = .TRUE. 57
M =IABS(KERTYP) 58
J = M/100 59
K = 2 60
IF(DOLLAR) MODE = 2 61
IF(KERTYP.GT.0) LOCK = .TRUE. 62
IF(KERTYP.LT.0) K=1 63
WRITE (COUNT,90) (MSGTYP(I,K),I=1,2),M,(H(I,J),J=1,4),KCI, 64
IMAGE ,KB ,I=1,L),KA ,KB ,I=L ,81,NERROR ,LOCK 65
90 FORMAT (1H ,*2A6,*2H ,*(I3 ,2H) ,*(4A6 ,2X ,81'A1 /45X ,83A1) 66
,14X,11HERROR COUNT ,I4 ,13H LOCK = .L1 ,I1H ) 67
IF(NERROR.GT.64) STOP 68

120
IF(KETYP.LT.0) GO TO 99
SMCHR = .TRUE.

30 CALL ICHAR($31,$65,7,10)
C 0-9,+,$
GO TO (30,65,39,80),LCOMP
C A-Z =
31 GO TO (60,65,30,60),LCOMP
39 SMCHR = .TRUE.

40 L = KCOUNT + 1
KC1 = KB
IF(KCOUNT.EQ.0) KC1 = KC
WRITE(KOUNIT,91) KC1,IMAGE,(KB,I=1,L),KD,(KB,I=1,81)
91 FORMAT(14X,32HBEGIN PROCESSING AT SYMBOL ,81A1/45X,83A1
.14X$48HLook FOR DIAGNOSTICS BUT DON'T STORE ANYTHING.//)
IFNTVP = -1
DO 1 I=ISTDIM
1 KSTACK(I) = 0
LEVEL = 0
NOTARG = ISTDIM
CONTINUE
KERTVP = 0
CALL DEBUG
RETURN
C 
C MODE ACTION
C 1 SKIP TO NEXT $ OR = OR , (UNCONDITIONAL)
C 2 SKIP TO NEXT $ (UNCONDITIONAL) OR TO = OR , AFTER 
END

$IF TC ILOOKI DECK
SUBRJUTINE ILOOKI(*,IT)
C CALLED FROM
C ITABL1
C INAMEI
DIMENSION IT(1)
DIMENSION IF'T(31), IPTAB(21), ITAB(65)
DIMENSION ANAME(15), IMAGE(80), IMAGE1(81),IPARM(9)
KSTACK(27),NAME(15),RVALUE(2),STACK(27)
COMMON
*/ICONVI VALUE,ICOMP,IFNTYP,IMAGE1,IRADIX,ISUB
KCH,KCNVRT,KCOUNT,KDIF,KFLD1,KFLD2
LCOMP,LCNVRT,LEVEL,LFRT,LOOK
MCNVRT,MDIF,MODALL,MSTOR
NAME,VERROR,NONEW,NOTARG
INTEGER BLANK, $BLANKS$, DOLLAR, ERMARK, TRACEx
DOUBLE PRECISION STACK, VALUE
LOGICAL ABORT, DOLLAR, ERMARK, LIMALF, LOCK, VOLIST, NOWNEW, MODALL,
nstdir, SMCHR, TEST
EQUIVALENCE (STACK, ISTACK), (VALUE, KVALUE, RVALUE), (NAME, ANAME)
EQUIVALENCE (ICOMNI, ISUB), (IMAGE, IMAGE1), (IPARAM, ABORT)
LOOK=3

IF (IT(L0OK) .EQ. 0) GO TO 7
  ' FIRST USABLE POSITION

J = IFLD(3, 4, IT(LOOK))
  ' RETURN WITH NEXT AVAILABLE ENTRY

IF (J .NE. MSTOR) GO TO 8
  ' MAX POSSIBLE J IS 15

DO 4 <= 1, J
  ' CHECK NAME FOR MATCH

LOOKK = LOOK + K
IF (NAME(K) .NE. IT(LOOK)) GO TO 8

4 CONTINUE

CALL DEBUG2(6H*L0OKF, LOOK)
  ENTRY WAS FOUND

5 RETURN 1

7 CONTINUE

CALL DEBUG2(6H*L0OKN, LOOK)
  NO ENTRY WAS FOUND

8 LOOK = LOOK + J + 1

9 CONTINUE

GO TO 1
C INTEGER BLANK, BLANKS, EOS, IDOLAR, TAB1, TRACE
D DOUBLE PRECISION STACK, VALUE
L LOGICAL ABORT, DOLLAR, ERMARK, LIMALF, LOCK, NOLIST, NONEW, MODALL,
E EQUIVALENCE (STACK, ISTACK), (VALUE, KVALUE, RVALUE), (NAME, ANAME)
E EQUIVALENCE (ICOMNI, ISUB), (IMAGE, IMAGE1), (IPARAM, ABORT)
D DIMENSION IT(1)
C DIMENSION D(1)
C ISUB = 1
C KDIFF = 1
C CALL INAMEN
C CALL ILOOK1($1, IT)
C CALL ILOOK1($64, IFT)
9540 KERTYP=540
C GO TO 99
1 CONTINUE
C CALL DEBUG2(6HSTATMT, 1)
C ITL = IT(LOOK)
C KCNVRT=IFLD(0,3, ITL)
C CALL DEBUG 2 (6HKCNVRT, KCNVRT)
C R I O T S F
C GO TO(12, 12, 10, 12, 62, 61), KCNVRT
10 KDIFF = 2
C { OTH
12 CALL ICHAR2($90, 30)
C } GO TO 90 FOR NO SUBSCRIPT
C A-Z 0-9 OTH
C CALL ICHAR1($82, $9140, 12)
C CALL INAMEN
C CALL ILOOK1($83, IT)
C GO TO 9540
82 SMCHR = .TRUE.
C CALL ISUBI
C GO TO 84
83 I TYPE=IFLD(0, 3, IT(LOOK))
C LOC=IFLD(7, 25, IT(LOOK))
C RVALUE(1) = D(LOC)
C RVALUE(2) = D(LOC+1)
C CALL ICNVTI (ITYPE, 2)
C ISUB = KVALUE
C ) OTH
84 CALL ICHAR2($9140, 31)
C } ERROR IF NO
C GO TO 91
90 SMCHR = .TRUE.
91 LOC=IFLD(7, 25, ITL)
C CALL DEBUG 2 (5HLLOC L, LOC)
C LOC = LOC + (ISUB-1)*KDIFF
50 RVALUE(1) = D(LOC)
C RVALUE(2) = D(LOC+1)
C CALL DEBUG (5HIT(L),ITL,2)
C CALL ICNVTI (KCNVRT,3)
52 LOOK = LOC
98 CONTINUE
C CALL DEBUG (5HVALUE,VALUE,4)
C CALL DEBUG 2 (6HLOOK,LOOK)
C CALL DEBUG 2 (5HLOC,LOC)
C CALL DEBUG (6HKCNVRT,KCNVRT)
C CALL DEBUGR RETURN
61 IFNTYP = 0
   FUNCTION
90
62 IFNTYP = 1
   SUBROUTINE
93
63 KVALJE=IFLD(7,25,ITL)
GO TO 98
64 IFNTYP = 0
   LIBRARY FUNCTION OR SUBROUTINE
97
C KVALJE=IFLD(7,25,IFT(LOOK))
94
C MSTOR = 0
KCNVRT=IFLD(0,3,IFT(LOOK))
98
C IF(KCNVRT.EQ.5)IFNTYP=KVALUE
99
C PROGRAM NUMBER (USED BY IXQTI)
C 5 FOR SUBROUTINES, 6 FOR FUNCTIONS
102
C PROVIDE FOR EXECUTION OF INPUT SUBRO
104
C CALL DEBUG(6HLIBF,KVALUE)
105
C MEANING OF IFNTYP
106
C VALUE
NAME IS
107
C -1(NORMAL) AN ORDINARY VARIABLE
108
C 0 FUNCTION (USER OR FORTRAN MATH)
109
C 1 USER SUBROUTINE
110
C 2 RADIX (INPUT FUNCTION)
111
C 3 LOCX (INPUT SUBROUTINE)
112
C $CALL LOCX(Y,I) CAUSES I TO BE SET SO THAT Y(I)OFFERS THE CURRENT LEFT SIDE.
113
C LEFT SIDE.
114
GO TO 98
115
9140 KERTYP =140
116
99 CONTINUE
117
C CALL DEBUG(6HRETURN,1)
118
C CALL DEBUGR
119
C RETURN 1
   REGISTER RETURN
120
END
121
122

$IBFTC INAMEN DECK
SUBROUTINE INAMEN
C CALLED FROM
C INAMEI
C ITALBI
C INPUT
C CALL DEBUGC(6HINAMEV)
C DIMENSION IFT(31), IPTAB(21), ITAB(65)
C DIMENSION ANAME(15), IMAGE(80), IMAGE(81), IPARAM(9)
C ,KSTACK(27), NAME(15), RVALUE(2), STACK(27)
C 124
COMMON /ICOMNI/ VALUE ,ICOMP ,IFNTYP ,IMAGE1 ,IRADIX ,ISUB
         ,KCH ,KCNVRT ,KCOUNT ,KDIF ,KFLD1 ,KFLD2 
         ,LCOMP ,LCNVRT ,LEVEL ,LFRT ,LOOK 
         ,MCNVRT ,MDIF ,MODALL ,MSTOR 
         ,NAME ,VERROR ,NEW ,NOTARG 
         ,SMCHR ,TEST ,ERMARK 
/ICNST/ BLANK ,BLANKS ,DOLAR ,EOS 
         ,ICOMMA ,IDOLAR ,IFT ,iptab ,ITAB 
         ,KAM10 ,KBPC ,KBPW ,KPCD ,KERTYP 
         ,KZERO ,NPRTN ,TAB1 
/IPARAM/ ABORT ,KIUNIT ,KUNIT ,LIMALF ,LOCK ,LOCK 
         ,NLIST ,NSTDIX ,TRACE 
/ISTACK/ STACK ,ISTDIM ,KSTACK ,LEVELM 

C INTEGER BLANK ,BLANKS ,DOLAR ,TAB ,TRACE 
DOUBLE PRECISION STACK ,VALUE ,DNAME 
LOGICAL ABORT ,DOLLAR ,ERMARK ,LIMALF ,LOCK ,NLIST ,NOW ,MODALL 
NSTDIX ,SMCHR ,TEST 
EQUIVALENCE (STACK ,ISTACK) , (VALUE ,<VALUE ,RVALUE) , (NAME ,ANAME) 
EQUIVALENCE (ICOMNI ,ISUB) , (IMAGE ,IMAGE1) , (IPARAM ,ABORT) 
EQUIVALENCE (NAME ,DNAME) 

C COLLECTS NAME (UP TO 15 WORDS) TERMINATED BY ANY SPECIAL CHAR 
ASSIGN 6 TO NEXT 
  IF(MODALL) ASSIGN 2 TO NEXT 
  MSTOR = 0 
  J = <BPW 
  NAME(2) = BLANKS 
  1 GO TO NEXT , (2,6,7) 
C         * 0TH 
  2 CALL ICHAR2($8 ,23) 
  TEST = *TRUE , 
  MODALL = *FALSE . 
  CALL ICHAR2($99,23) 
C         * PAIR OF APOSTROPHYS DOES NOT EN 
  TEST = *FALSE , 
  MODALL = *TRUE . 
  GO TO 8 
  6 ASSIGN 7 TO NEXT 
  GO TO 8 
C         A-Z=0-9 0TH 
  7 CALL ICHAR2($99,12) 
  8 IF(J<LT<KBPW) GO TO 9 
  IF(MSTOR.<EQ.15) GO TO 10 
  MSTOR = MSTOR + 1 
  NAME(MSTOR) = BLANKS 
  J = 0 
  9 NAME(MSTOR)=IFLD4(KCH,J,KBPC,NAME(MSTOR)) 
      J=J<KBPC 
    GO TO 1 
  10 IF(MODALL) GO TO 99 
    KERTYP = -260 
    CALL IEORI 
  95 CALL ICHAR2($99,12) 
C         * SKIP REST OF NAME 
  GO TO 95 
  99 CONTINUE 
  SMCHR = *TRUE , 
C CALL DEBUG3(6HNAME ,DNAME ,3) 
C CALL DEBUG2(5HMSTOR ,MSTOR) 
C CALL DEBUGR 
RETURN 
END 

125
$IBMTC INMBRI DECK

SUBROUTINE TO TRANSLATE A NUMERIC FIELD

SUBROUTINE INMBRI

CALLED FROM INPUT

INMBRI IS CALLED WITH FIRST DIGIT IN KCH

CALL DEBUG(6HINMBRI)

DIMENSION IFT( 31), IPTAB( 21), ITAB( 65)

DIMENSION ANAME(15), IMAGE(80), IMAGE1(81), IPARAM(9)

* , KSTACK(27), NAME(15), RVALUE(2), STACK(27)

COMMON

/ICOMNI/ VALUE, ICOMP, IFNTPY, IMAGE1, IRADIX, ISUB

, KCH, KCNVRT, KCOUNT, KDIF, KFLD1, KFLD2

, LCOMP, LCNVRT, LEVEL, LFRT, LOOK

, MCNVRT, MDIF, MODALL, MSTOR

, NAME, NERROR, NONEW, NOTARG

, SMCHR, TEST, ERMARK

/ICNSTI/ BLANK, BLANKS, DOLLAR, EOS

, IDOLAR, IFIT, IPTAB, ITAB

, KAM10, KBPC, KBPW, KCPCD, KERTYP

, KZERO, NOPRNT, TAB1

/IPARAM/ ABORT, KIUNIT, KOUNT, LIMALF, LOCK, LOCX

, NOLIST, NSTDIR, TRACE

/ISTAKI/ STACK, ISTDIM, KSTACK, LEVX

INTEGER BLANK, BLANKS, EOS, IDOLAR, TAB1, TRACE

DOUBLE PRECISION STACK, VALUE, DNBR, H, FD(4)

LOGICAL ABORT, DOLLAR, ERMARK, LIMALF, LOCK, NOLIST, NONEW, MODALL,

NSTDIR, SMCHR, TEST

EQUIVALENCE (STACK, ISTACK), (VALUE, KVALUE, RVALUE), (NAME, ANAME)

EQUIVALENCE (ICOMNI, ISUB), (IMAGE, IMAGE1), (IPARAM, ABORT)

LOGICAL LVALUE

EQUIVALENCE (KVALUE, LVALUE)

LOGICAL SWITCH

DIMENSION LD(4)

DATA LD(1), LD(2), LD(3), LD(4)/8, 4, 2, 1/

DATA FD(1), FD(2), FD(3), FD(4)/1.0D8, 1.0D4, 1.0D2, 1.0D0/

DNBR=0

ICSC = 0

THE NUMBER COLLECTED S3 FAR

IPF = 1

THE CHARACTERISTIC SCALE FACTOR

IESE = 0

SIGN OF EXPONENT

ASSIGN 1 TO NEXT

SMCHR = .TRUE.

CALL ICHAR2($70, 21)

SMCHR = .TRUE.

GO TO 70 FOR LOGICAL CONSTANTS

CALL ICHAR4($2, $3, 13, 15)

GO TO 50

DNBR = DNBR*10.0D0+FLOAT(KCH-KZERO)

GO TO NEXT, (1, 15)

ICSC = ICSC - 1

GO TO 1

SMCHR = .TRUE.

* ENTER HERE FOR INITIAL DECIMAL POINT

10 IF (SWITCH) GO TO 9130

126
C CALL DEBUG2(6HSTAT *,20)
ASSIGN 15 TO NEXT
SWITCH = *TRUE*
GO TO 1
37 IF=1
GO TO J 36
30 CONTINUE
C CALL DEBUG 2(6HSTAT E,30)
C ← OTH 0-9
CALL ICHAR4($50, $35, 13, 15)
C ←
GO TO J(35, 37), LCOMP
35 SMCHR= *TRUE*
36 CALL ISUBI
C * TEST WILL BE TRUE
44 ICSC = ICSC+ISIGN(ISUB, IPF)
C * RESOLVE SCALE FACTORS
50 CONTINUE
C CALL DEBUG 2(6HSTAT ,50)
C CALL DEBUG3 (5HONBR1, DNBR ,4)
M = 1. DO 63 I=1, 4
IESC=IABS(ICSC)
DO 63 I=1, 4
61 IF (IESC*LT. LD(I)) GO TO 63
62 IESC=IESC-LD(I)
H=H*FD(I)
GO TO J 61
63 CONTINUE
IF (IESC*LT.0) GO TO 65
64 DNBR = DNBR*H
GO TO 98
C T F OTH
70 CALL ICHAR1($73, $19, 17)
LVALUE= *TRUE*
71 IF(NOTARG.GT.LEVEL) LCNVRT=4
C * LOGICAL CONSTANTS NOT CONVERTED
RVALUE(2)=0.
A-Z OTH
72 CALL ICHAR1($9130, $99, 21)
C *DISCARD REST OF WORD, MUST FIND
GO TO J 72
LVALUE= *FALSE*
GO TO J 71
65 DNBR = DNBR/H
98 VALUE = DNBR
SMCHR= *TRUE*
99 CONTINUE
C CALL DEBUG3(6HDNBR 2, DNBR, 4)
C CALL DEBUG3(5HVALUE, VALUE, 4 )
C CALL DEBUG2(6HICSC , ICSC)
C CALL DEBUG2 (6HIPF, IPF)
C CALL DEBUG 2 (6HIESC 1, IESC)
C CALL DEBUGR
RETURN
9130 KERTYP = 130
GO TO 99
END
SUBROUTINE TO CONSTRUCT TABLE ENTRIES

DIMENSION IFT(31), IPTAB(21), ITAB(65)
DIMENSION NAME(15), IMAGE(80), IMAGEI(81), IPARAM(9)
  *    KSTACK(27), NAME(15), RVALUE(2), STACK(27)

COMMON/ ICOMNI/ VALUE, ICOMP, IFNTYP, IMAGE1, IRADIX, ISUB
  *  KCH, KCVNRT, KCOUNT, KDIFF, KFLD1, KFLD2
  *  LCOMP, LCNVRT, LEVEL, LFRT, LOOK
  *  MCNVRT, MDIF, MODALL, MSTOR
  *  NAME, NERROR, NONEW, NOTARG
  *  SMCHR, TEST, ERMARK
  */ICNSTI/ BLANK, DOLLAR, EOS
  *  ICOMMA, IDOLAR, IFT, IPTAB, ITAB
  *  KAM10, KBPC, KBPW, KPCD, KERTYP
  *  KZERO, KOPRT, TAB1
  */IPARAM/ ABORT, KUNIT, KUNIT, LIMALF, LOCK, LOCX
  *  IOLIST, NSTDIR, TRACE
  */ISTACI/ STACK, ISTDIM, KSTACK, LEVLIM

INTEGER BLANK, BLANKS, EOS, IDOLAR, TAB1, TRACE
DOUBLE PRECISION STACK, VALUE
LOGICAL ABORT, DOLLAR, ERMARK, LIMALF, LOCK, NOLIST, NONEW, MODALL,
  NSTDIR, SMCHR, TEST
EQUIVALENCE (STACK, ISTACK), (VALUE, KVALUE, RVALUE), (NAME, ANAME)
EQUIVALENCE (ICOMNI, ISUB), (IMAGE, IMAGEI), (IPARAM, ABORT)

DIMENSION IT (1)

KDIF = 1
ISUBX = 1
ITYPE = 1

* ALWAYS INITIALIZE TO .REAL.

CALL ICHARI ($1, $9120, 26)
  * SKIP *ABLE* IN TABLE

CONTINUE
CALL DEBUG2(6HSTATMT, 3)
CALL ICHARI4($4, $9120, 7, 9)
  1  0-9* OTH
    GO TO (30, 98, 10, $9120), LCOMP
  2  A-Z OTH DTH
    GO TO (3, 20, $9120, $9120), LCOMP
  $9120  KERTYP = 120
    *   ILLEGAL CHARACTER
    GO TO 98

CONTINUE
CALL DEBUG2(6HST 0-9, 10)
SMCHR = .TRUE.
CALL ISUBI
ISUBX = ISUB
CALL ICHARI2($9120, 25)
  * ERROR IF NO =

GO TO 3
CONTINUE
CALL DEBUG2(6HSTATMT, 30)
KDIF = 1
C SPLIT TYPES (INT, REAL, DP, NO CONVERSION, FUNCTION, SUBROUTINE) 61
CALL ICHAR4($32, $33, 3, 5) 62
IF (LCOMP .GT. 1) GO TO 9120 63
KDIF = 2 64
GO TO 33 65
32 IF (LCOMP .EQ. 3) GO TO 9120 66
IYPE = LCOMP 67
GO TO 34 68
33 IYPE = LCOMP + 2 69
C A-Z OTH . 70
34 CALL ICHARI($9120, $3, 21) 71
GO TO 34 72
20 CONTINUE 73
C CALL DEBUG2(6HSTATMT, 20) 74
CALL INAMEN 75
CALL INAME 76
50 CONTINUE 77
C CALL DEBUG2(6HSTATMT, 50) 78
ITBUFF = IFLD4(IYPE, 0, 3, ITBUFF) 79
ITBUFF = IFLD4(MSTOR, 3, 4, ITBUFF) 80
ITBUFF = IFLD4(ISUBX, 7, 25, ITBUFF) 81
C CALL DEBUG2 (5HITYPE, IYPE) 82
C CALL DEBUG2 (5HSTOR, MSTOR) 83
C CALL DEBUG2 (5HSUBX, ISUBX) 84
C CALL DEBUG4 (6HITBUFF, ITBUFF, 2) 85
ISUBX = ISUBX + KDIF 86
CALL ILOOKI($56, IT) 87
IF((IT(2) .NE. 0) .AND. (IT(1) .GT. IT(2)) ) GO TO 9320 88
IT(1) = ITBUFF 89
DO 55 K = 1, MSTOR 90
LOOK = LOOK + 1 91
55 IT(LOOK) = NAME(K) 92
C CALL DEBUG2 (6HSTATMT, 55) 93
LOOK = LOOK + 1 94
IT(LOOK) = 0 95
GO TO 3 96
56 IT(LOOK) = ITBUFF 97
C CALL DEBUG2 (6HSTATMT, 56) 98
GO TO 3 99
98 CONTINUE 100
C CALL DEBUGR 101
RETURN 102
9320 KERTYP = -320 103
CALL IERORI 104
GO TO 3 105
END 106

$IBFTC ISUBI DECK
SUBROUTINE ISUBI
ISUBI FINDS SUBSCRIPTS AND INTEGER CONSTANTS
ISUBI CALLED FROM
ISJB BEGINS PROCESSING WITH THE NEXT CHARACTER.
CALL DEBUG (SHISUBI)
ISUBI COLLECTS INTEGER OF BASE IRADIX TERMINATED BY A SPECIAL CHARACTER
DIMENSION IFT(31), IPTAB(21), ITAB(65)
DIMENSION ANAME(15) , IMAGE(80), IMAGE1(81), IPARAM(9)
   , KSTACK(27), NAME(15), RVALUE(2), STACK(27)

C
COMMON
   /ICOMNI/ VALUE , ICOMP , IFNTYP , IMAGE1 , IRADIX , ISUB
   , KCH , KCVNRT , KCOUNT , KDF1 , KFLD1 , KFLD2
   , LCOMP , LCVNRT , LEVEL , LFLR , LOOK
   , MCVNRT , MDIF , MODALL , MSTOR
   , NAME , NERROR , NNEW , NOTARG
   , SMCHR , TEST , ERESULT , NMODE
   /ICNSTI/ BLANK , BLANKS , DOLLAR , EOS
   , ICOMMA , MDOLLAR , IFT , IPTAB , ITAB
   , KAM10 , KBPC , KBPW , KCPD , KERTYPE
   , KZERO , NOPERNT , TAB1
   /IPARAM/ ABORT , KUNIT , KUNIT , LIMALF , LOCK , LOCX
   , NOLIST , NSTDIR , TRACE
   /ISTAX/ STACK , ISTDIM , KSTACK , LEVILI
C
INTEGER BLANK , BLANKS , DOLLAR , EOS , IDOLLAR , TAB1 , TRACE
DOUBLE PRECISION STACK , VALUE
LOGICAL ABORT , DOLLAR , ERESULT , LIMALF , LOCK , NSTDIR , NNEW , MODALL,
   NSTDIR , SMCHR , TEST
EQUIVALENCE (STACK, ISTACK), (VALUE, KVALUE, RVALUE), (NAME, ANAME)
EQUIVALENCE (ICOMNI, ISUB), (IMAGE, IMAGE1), (IPARAM, ABORT)
ISUB = 0

C
   A-Z 0-9 OTH
80  CALL ICHARI($10,$99,12)
   IDIGIT = KCH - KAM10
   * VALUE OF LETTER USED AS DIGIT
30  IF (IDIGIT .GE. IRADIX) GO TO 99
   ISUB = ISUB * IRADIX + IDIGIT
   * ACCUM TOTAL. IRADIX
40  GO TO 80
99  SMCHR = .TRUE.
   * ALLOW SAME CHARACTER TO BE READ
C
   CALL DEBUG2(6H+XSUB, ISUB)
   CALL DEBUGR
   RETURN
10  IDIGIT = KCH - KZERO
C
   GO TO 30
   END

$IBFTC IXQTI  DECK
SUBROJTIME IXQTI (ARG1, ARG2)
C
   USER MAY PUT HIS OWN COMMON STATEMENTS IN THIS ROUTINE AND
   USE THEM TO SUPPLY ARGUMENTS TO HIS CALLS IF HE DESIRES
   COMMON
   */IPARAM/ ABORT , KUNIT , KUNIT , LIMALF , LOCK , LOCX , NOLIST , NSTDIR
   , TRACE
   DOUBLE PRECISION ARG1(27), ARG2, ARG22
   DIMENSION ARG2(2)
   EQUIVALENCE (ARG2, ARG22)
C
   M = DABS(ARG1(1))
   IF (M .LT. 1 .OR. M .GT. 16) GO TO 99
   GO TO {1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16}, M
1 ARGL = COS(ARGL)
   GO TO 100
2 ARGL = EXP(ARGL)
   GO TO 100
3 ARGL = ALOG(ARGL)
   GO TO 100
4 ARGL = SIN(ARGL)
   GO TO 100
5 ARGL = SQRT(ARGL)
   GO TO 100
6 ARGL = ATAN2(ARG2, ARGL)
   GO TO 100
7 ARG22 = ARGS(2)
C PRINT FUNCTION
   WRITE (KOUNIT,101) ARG2, ARGL
101 FORMAT(1H 2A6,3H = ,D26.17)
   GO TO 100
8 ARGL = SNGL(ARG2) ** IFIX(SNGL(ARGL+DSIGN(*5D0,ARGL)))
   GO TO 100
9 ARGL = ABS(SNGL(ARG2)) ** SNGL(ARGL)
   GO TO 100
10 ARGL = ABS(ARGL)
   GO TO 100
C DISC(KMINENT) FUNCTION
11 IF (ARGL) 102,103,104
102 ARGL = ARGS(2)
   GO TO 100
103 ARGL = ARGS(3)
   GO TO 100
104 ARGL = ARGS(4)
   GO TO 100
12 CALL LOCKX(ARGL)
   GO TO 100
13 CONTINUE
   GO TO 100
14 CONTINUE
   GO TO 100
15 CONTINUE
   GO TO 100
16 CONTINUE
   GO TO 100
100 CONTINUE
C CALL DEBUGR
RETURN
99 KERTYP =-610
   CALL IERORI
   GO TO 100
END

$IBFTC LOCKX  DECK
SUBROBINE LOCKX(J)
   COMMV
   */PARAM/ ABORT ,KUNIT ,KUNIT ,LIMALF ,LOCK ,LOCK
   ,NLIST ,NSTDIR ,TRACE
   LOGICAL J,LOCK
   LOCK = J
   RETURN
END

*IBFCT DEBUG* DECK

SUBROUTINE DEBUG

*INITIALIZATION*

C DIMENSION ISUBN(20)
C TRACE = 0 NO PRINTING
C TRACE = 1 PRINT DEBUG 2+3 CALLS ONLY
C TRACE = 2 PRINT DEBUG 2+3 CALLS ONLY
C TRACE = 3 PRINT DEBUG 2+3 AND STACK PRINT
C TRACE = 4 PRINT DEBUG 2+3 AND STACK PRINT AND CALLS FROM CALL
D DIMENSION IF (27), IPTAB (21), ITAB (65)
D DIMENSION ANAME (15), IMAGE (80), IMAGE (81), IPARAM (9)
* KSTACK (27), NAME (15), RVALUE (2), STACK (27)

C COMMON
C /ICOMNI/, VALUE, ICOMP, IFNTYP, IMAGE, IRADIX, ISUB
C , KCH, KCNVRT, KCOUNT, KDIFF, KFLD1, KFLD2
C , LCOMP, LCNVRT, LEVEL, LFR, LLOOK
C , MCNVRT, MDIF, MODALL, MSTDOR
C , NAME, NERROR, NOW, NOTARG
C , SMCHR, TEST, ERMARK
C /ICNST/, BLANK, BLANKS, DOLLAR, EDS
C , ICOMMA, IDOLLAR, IFT, IPTAB, ITAB
C , KAM10, KBPC, KBWP, KCPD, KERTYP
C , KZERO, NOPRNT, TAB1
C /IPARAM/, ABORT, KUNIT, KUNIT, LIMAF, LOCK, LOCX
C , NOLIST, NSTDOR, TRACE
C /ISTAKI/, STACK, ISTDIM, KSTACK, LEVLIM

C INTEGER BLANK, BLANKS, EDS, IDOLLAR, TAB1, TRACE
D DATA BADCAL, 6BADCAL/
D DOUBLE PRECISION ALFARG, ISUBN, STACK, VALUE, DBLANK, DARG
D LOGICAL ABORT, DOLLAR, ERMARK, LIMAF, LOCK, NOLIST, NOW, MODALL,
D NSTDOR, SMCHR, TEST
D EQUIVALENCE (STACK, ISTACK), (VALUE, KVALUE, RVALUE), (NAME, ANAME)
D EQUIVALENCE (ICOMNI, ISUB), (IMAGE, IMAGE), (IPARAM, ABORT)
* (DBLANK, BLANK)
I ISUBN (1) = DBLANK
I IF (TRACE, EQ, 0) GO TO 99
I WRITE (KUNIT, 410)
I GO TO 99
I ENTRY DEBUGC (ISUBNA)
D DOUBLE PRECISION ISUBNA

C * NEW SUBROUTINE CALLED
C IF (ISUBC, GT, 10) GO TO 98
ALFARG = DBLANK
C NUMARG = -1
ISUBC = ISUBC + 1
ISUBN (ISUBC) = ISUBNA
C IF (TRACE, GE, 4) GO TO 400
C GO TO 99

C 98 ALFARG = BADCAL
C * CALLS MIGHT GET OUT OF RANGE
C NUMARG = ISUBC
C GO TO 400

C ENTRY DEBUGC
C
IF (ISUBC.LT.1) GO TO 98
ALFAR = DBLANK
NUMAR = -1
ISUBN(ISUBC) = DBLANK
ISUBC = ISUBC - 1
IF (TRACE .GE. 4) GO TO 400
GO TO 99
C
ENTRY DEBUG2(ALFAR, NUMAR)
DOUBLE PRECISION ALFAR
NUMAR = NUMAR
ALFAR = ALFAR
IF (TRACE .LT. 1) GO TO 99
400 WRITE (KOUNIT,405) (ISUBN(I),I=1,4), ALFARG, NUMARG,
,KCOUNT, SMCHR, KCH, KFLD1, ICMP, KFDD2, LCMP
99 RETURN
C
ENTRY DEBUG3(ALFAR, DNUMAR, ITYPE)
DOUBLE PRECISION DNUMAR
ENTRY DEBUG4(ALFAR, DNUMAR, ITYPE)
DARG = DNUMAR
NUMARG = NUMAR
ALFARG = ALFAR
IF (TRACE .LT. 1) GO TO 99
GO TO (30, 40, 50, 60), ITYPE
30 WRITE (KOUNIT, 406) (ISUBN(I), I=1,4), ALFARG, NUMARG,
,KCOUNT, SMCHR, KCH, KFDD1, ICMP, KFDD2, LCMP
GO TO 99
40 WRITE (KOUNIT, 407) (ISUBN(I), I=1,4), ALFARG, NUMARG,
,KCOUNT, SMCHR, KCH, KFDD1, ICMP, KFDD2, LCMP
GO TO 99
50 WRITE (KOUNIT, 408) (ISUBN(I), I=1,4), ALFARG, NUMARG,
,KCOUNT, SMCHR, KCH, KFDD1, ICMP, KFDD2, LCMP
GO TO 99
60 WRITE (KOUNIT, 409) (ISUBN(I), I=1,4), ALFARG, NUMARG,
,KCOUNT, SMCHR, KCH, KFDD1, ICMP, KFDD2, LCMP
GO TO 99
405 FORMAT(1H, 1H, 29X, 5(A6,1X), I24,
1H, I3, L2, 1H), A6, 2H (I3, I2, 1H), 2H (I3, I2, 1H))
406 FORMAT(1H, 29X, 5(A6, 1X), E24.8,
1H, I3, L2, 1H), A6, 2H (I3, I2, 1H), 2H (I3, I2, 1H))
407 FORMAT(1H, 29X, 5(A6, 1X), 11X, O12, 1X,
1H, I3, L2, 1H), A6, 2H (I3, I2, 1H), 2H (I3, I2, 1H))
408 FORMAT(1H, 29X, 5(A6, 1X), 16X, A8,
1H, I3, L2, 1H), A6, 2H (I3, I2, 1H), 2H (I3, I2, 1H))
409 FORMAT(1H, 29X, 5(A6, 1X), 024.8,
1H, I3, L2, 1H), A6, 2H (I3, I2, 1H), 2H (I3, I2, 1H))
410 FORMAT(29X, 11HDEBUG TRACEx52X, 9HI--4COUNT , 5X, 8HI--LIST1/
68X, 9ID ARG OR, 15X, 9HI I-SMCHR, 5X, 10HI I--ICMP /
29X, 13H SUBROUTINES CALLED, 11X, 7HALF ARG, 7X, 11H NUMERIC ARG, 9X,
3HI, 1X, 5H KCH-I, 5X, 16HI I I--LIST2 /
92X, 3HI I, 5X, 1HI, 5X, 18HI I I--LCMP)
END
$IBFTC STACKP DECK
SUBROJXINE STACKP
DIMENSION IFT(27), IPTAB(21), ITAB(65)
DIMENSION ANAME(15), IMAGE(80), IMAGE1(81), IPARAM(9)
   *KISTACK(27), NAME(15), RVALUE(2), STACK(27)

C
COMMON
   */ICOMNI/ VALUE, ICOMP, IFNTYP, IMAGE1, IRADIX, ISUB
   */CH, KCH, KCOUNT, KDIF, KFLD1, KFLD2
   */LCOMP, LCVNRT, LEVEL, LFRT, LOOK
   */MCNVRT, MDIF, MODALL, MSTOR
   */NAME, NERROR, NONEW, NOTARG
   */SMCHR, TEST, ERMARK
   */ICNSTI/ BLANK, DOLLAR, EDS
   */ICOMMA, IDOLAR, IFT, IPTAB, ITAB
   */KAMIO, KBPC, KBPW, KCPD, KERTYP
   */ZERO, NOPRNT, TAB1
   */IPARAM/ ABORT, KUNIT, KUNIT, LIMALL, LOCK, LOCX
   */NLIST, NSTDIR, TRACE
   */ISTACK/ STACK, ISTDIM, KSTACK, LEVLIM

C
INTEGER BLANK, DOLLAR, EOS, IDOLAR, TAB1, TRACE
DOUBLE PRECISION STACK, VALUE
LOGICAL ABORT, DOLLAR, EMARK, LIMALL, LOCK, NLIST, NONEW, MODALL,
   NSTDIR, SMCHR, TEST
EQUIVALENCE (STACK, ISTACK), (VALUE, KVALUE, RVALUE), (NAME, ANAME)
EQUIVALENCE (ICOMNI, ISUB), (IMAGE, IMAGE1), (IPARAM, ABORT)
IF(TRACE .LT. 3) GO TO 99
LEVEL = LEVEL + 3
WRITE(KUNIT, 85) LEVEL, VALUE, KVALUE, RVALUE, LOCX
WRITE(KUNIT, 89)(STACK(I), I=1, LEVEL)
WRITE(KUNIT, 86)(KSTACK(I), I=1, LEVEL)
FORMAT (11D12.4)
FORMAT (11 I12)
FORMAT (7H LEVEL=I3,7H VALUE=D25.17,8H KVALUE=I3,3H ,0,211X,012),
   6H LOCX=I6)
99 RETURN
END

$IBMAP IFLD DECK
TTL IFLD AND IFLD4 PROGRAM
ENTRY IFLD
ENTRY IFLD4
* FUNCTION (I1,N,DATA)
*     START EXTRACTION WITH BIT I+1
*     EXTRACT N BITS FROM DATA AND RETURN IN REGESTER 1
IFLD SAVE
   CAL 3,4 I
   STA LGL1
   CAL 4,4 N
   STA LGL2
   LDQ 5,4 DATA
   LGL LGL ** **=I
   ZAC

134
COMMON
  /ICON/ VALUE ,ICOMP ,IFNTYP ,IMAGE ,IRADIX ,ISUB
  ,KCH ,KCNVRT ,KCOUNT ,KDIFF ,KFDL1 ,KFDL2
  ,LCOMP ,LCNVRT ,LEVEL ,LFT ,LOOK
  ,MCNVRT ,MDIF ,MODALL ,MSTOR
  ,NAME ,NERR ,NONEW ,NOTARG
  ,SMCHR ,TEST ,ERMARK
/ICON/ BLANK ,BLANKS ,DOLLAR ,EOS
  ,ICOMMA ,IDOLAR ,IFT ,IPTAB ,ITAB
  ,KAM10 ,KBPC ,KBPL ,KCPCD ,KERTYP
  ,KZERO ,NOPRNT ,TAB1
/ICON/ ABORT ,KUNIT ,KUNIT ,LIMALL ,LOCK ,LOCK
  ,Nolist ,NDTR ,TRACE
/ISTACK/ STACK ,ISTDIM ,STACK ,LEVIM

INTEGER BLANK ,BLANKS ,EOS ,IDOLAR ,TAB1 ,TRACE
DOUBLE PRECISION STACK ,VALUE
LOGICAL ABORT ,DOLLAR ,ERMARK ,LIMALL ,LOCK ,NOLIST ,NONEW ,MODALL
ISTDIR ,SMCHR ,TEST
EQUIVALENCE (STACK ,ISTACK) , (VALUE ,KVALUE ,RVALUE) , (NAME ,ANAME)
EQUIVALENCE (ICON/ ,ISUB) , (IMAGE ,IMAGE) , (IPARAM ,ABORT)
IF(TRACE .LT. 3) GO TO 99
LEVEL = LEVEL + 3
WRITE(KUNIT ,85) ,LEVEL ,VALUE ,KVALUE ,RVALUE ,LOCK
WRITE(KUNIT ,89) ,STACK(I) ,I=1 ,LEVELX)
WRITE(KUNIT ,86) ,KSTACK(I) ,I=1 ,LEVELX)
99 RETURN

END

$IBMAP IFLD DECK
SUBROUTINE IFLD
DIMENSION IFLD(27) , IFLD(21) , IFLD(65)
DIMENSION ANAME(15) , IMAGE(80) , IMAGE(81) , IPARAM(9)

* COMMON
  /ICON/ VALUE ,ICOMP ,IFNTYP ,IMAGE ,IRADIX ,ISUB
  ,KCH ,KCNVRT ,KCOUNT ,KDIFF ,KFDL1 ,KFDL2
  ,LCOMP ,LCNVRT ,LEVEL ,LFT ,LOOK
  ,MCNVRT ,MDIF ,MODALL ,MSTOR
  ,NAME ,NERR ,NONEW ,NOTARG
  ,SMCHR ,TEST ,ERMARK
/ICON/ BLANK ,BLANKS ,DOLLAR ,EOS
  ,ICOMMA ,IDOLAR ,IFT ,IPTAB ,ITAB
  ,KAM10 ,KBPC ,KBPL ,KCPCD ,KERTYP
  ,KZERO ,NOPRNT ,TAB1
/ICON/ ABORT ,KUNIT ,KUNIT ,LIMALL ,LOCK ,LOCK
  ,Nolist ,NDTR ,TRACE
/ISTACK/ STACK ,ISTDIM ,STACK ,LEVIM

INTEGER BLANK ,BLANKS ,EOS ,IDOLAR ,TAB1 ,TRACE
DOUBLE PRECISION STACK ,VALUE
LOGICAL ABORT ,DOLLAR ,ERMARK ,LIMALL ,LOCK ,NOLIST ,NONEW ,MODALL
ISTDIR ,SMCHR ,TEST
EQUIVALENCE (STACK ,ISTACK) , (VALUE ,KVALUE ,RVALUE) , (NAME ,ANAME)
EQUIVALENCE (ICON/ ,ISUB) , (IMAGE ,IMAGE) , (IPARAM ,ABORT)
IF(TRACE .LT. 3) GO TO 99
LEVEL = LEVEL + 3
WRITE(KUNIT ,85) ,LEVEL ,VALUE ,KVALUE ,RVALUE ,LOCK
WRITE(KUNIT ,89) ,STACK(I) ,I=1 ,LEVELX)
WRITE(KUNIT ,86) ,KSTACK(I) ,I=1 ,LEVELX)
99 RETURN

END

$IBMAP IFLD DECK
TTL IFLD AND IFLD4 PROGRAM
ENTRY IFLD
ENTRY IFLD4

* FUNCTION (I,N,DATA)
* START EXTRACTION WITH BIT I+1
* EXTRACT N BITS FROM DATA AND RETURN IN REGISTER 1
IFLD SAVE
CAL* 3 , 4 , I
STA LGL1
CAL* 4 , 4 , N
STA LGL2
LDQ* 5 , 4 , DATA
LGL1 LGL ** =I
ZAC
LGL2 LGL ** ==N
XCL
XCA
RETURN IFLD
SPACE 2
* FUNCTION OR SUBROUTINE IFLD4(NUDATA,I,N,DATA)
* PUT LOW ORDER N BITS OF NUDATA IN DATA
* STARTING WITH THE ITH+1 BIT.
* THE MODIFIED DATA IS LEFT IN REG 0 FOR FUNCTION TYPE USAGE
* I.E. IX=IFLD4(NUDATA),I,N,DATA)
* DATA IS UNCHANGED IX CONTAINS THE CHANGED DATA
IFLD4 SAVE
CAL* 4,4 I
STA I1
STA I2
CAL* 5,4 N
STA N1
STA N2
ZAC
LDQ* 6,4 DATA
I1 LGL ** ==I
SLW TEMP
N1 LGL ** ==N
CAL* 3,4 Z
N2 LGR ** ==N
CAL TEMP
I2 LGR ** ==I
XCA
RETURN IFLD4
TEMP PZE
END
APPENDIX B

HUFF INPUT ROUTINE

The first version of the Huff Input Routine was reported in reference 6. The Huff Input Routine provides more versatility in reading input data into the computer than the NAMELIST feature in FORTRAN. The Huff Input Routine has the ability to make simple arithmetic manipulations (such as conversion of units) during loading and to load alphanumeric data. While not an indispensable feature, it has been found to be quite convenient. The Huff Input Routine also allows for the automatic printout of data cards at execution time.

The following sections contain a general description of the Huff Input Routine and its usage.

Usage

The programmer transfers control to the INPUT routine with a standard FORTRAN IV call

CALL INPUT (5, 6, 1, X, ITABLE)

Argument 1 is the system input tape number (5 on the Lewis system). Argument 2 is the system output tape number (6 on the Lewis system). Argument 3 is the identifying number of a data group. This value is compared with an identification number occurring on the input card ($DATA card). If the values agree, the data are processed until another $DATA or end-of-data ($END) card is encountered. If the values do not agree, no data are processed and control is returned to the calling program. Argument 4 is the Array X, which serves as a reference point for the storing of input data. Since all data are stored relative to X, the programmer must provide fixed relations between the location of X and other locations to be loaded (e.g., through the use of common blocks and/or equivalence statements). In this case, X is "WORD," the first name in the labeled common block "ALL." Common blocks ALL, DESIGN, FRONT, SIDE, BACK, and DUMMY are in all routines and hence are loaded sequentially so that the location of all variables is known. Argument 5 is the array ITABLE, which contains the names of the variables used on the cards and their subscript location relative to X. Sufficient space must be provided in the calling program for storing the table of names. This is done by a DIMENSION statement. The dimension of ITABLE(3) must initially be zero.
Types of Input Statements

$DATA statement. - The $DATA statement identifies a group of data with an identification number. It must be the first statement on a card. For example, $DATA(1) or $D(1) on the first card of a data group causes the value 1 to be compared with argument 3 in the calling sequence. If unequal, control is returned to the calling program. If equal, data are loaded until the next $DATA or $END statement is reached.

$TABLE statement. - The $TABLE statement makes a list of names needed for loading data. Consider for example that the real variable names VELOCITY, MASS, and RADIUS are to be assigned to memory locations X(1), X(2), and X(3), respectively. The card would be punched $TABLE (.REAL., 1 = VELOCITY, 2 = MASS, 3 = RADIUS). These variables will be treated as real in any subsequent loading of data. A limit of 15 computer words is placed on the length of a name. Since .REAL. is what designates the mode of a name, a name may begin with any alphabetic letter. For example, the statement $TABLE (.INTEGER., 20 = INDEX, SUBSCRIPT, I) will place these names in the table and any values subsequently loaded will be stored in X(20), X(21), and X(22), respectively, as integers. In a similar manner $TABLE (.DOUBLE PRECISION., 10 = RADIUS DOUBLE, .LOGICAL., 12 = SWITCH1) causes the name RADIUS DOUBLE to be stored in the table as a double-precision variable equivalent to X(10) and X(11), and the logical variables SWITCH 1 and SWITCH 2 will be equivalent to X(12) and X(13).

Note that $TABLE statements are loaded as Number = Name to avoid confusion with loading statements.

Loading Statement

The loading statement loads data by taking the name of a variable previously appearing in a $TABLE statement and setting it equal to a value which may be of several forms.

Numeric values. - Standard FORTRAN language is used; for example, VELOCITY = 3.4, MASS = 32 (no decimal point is needed and MASS will have the REAL value 32), RADIUS = 4E21, and INDEX = 3. Data can be continued from one card to another; for example, SUBSCRIPT may appear at the end of one card and = 47 on the next card.

Subscripts may be used. Since 3 = RADIUS, RADIUS(2) = 6, 10, 12,. 14 will put real numbers in X(4), X(5), X(6), and X(8) and leave X(7) unchanged because of the double comma. If new values are assigned to a variable before the next $D(1) card, the new value will override the previous one. For example, RADIUS(2) = 8 will override the RADIUS(2) = 6 card.

Internally addressed values. - An internally addressed value is one that refers to the contents of memory by name. RADIUS(7) = RADIUS(3); RADIUS(INDEX) causes RADIUS(7) to be replaced by the value of RADIUS(3) and RADIUS(8) to also be replaced
by \textsc{RADIUS}(3) since \textsc{INDEX} = 3.

The statement \textsc{RADIUS}(7) = \textsc{RADIUS}(\textsc{INDEX} + 1), however, is ILLEGAL.

\textbf{Arithmetic expressions.} - Provisions have been made to allow arithmetic operations to be performed on data at execution time. The operations + (addition), - (subtraction), *(multiplication), and /(division) and the functions, included among which are \textsc{SQRT}, \textsc{EXP}, \textsc{SIN}, \textsc{COS}, and \textsc{PWR}(x, y) (= x^y), may be used with name or numbers (or any expression that has a value) to compute the value of an arithmetic expression. Parentheses may be used to indicate the order of performing the operations. The computations are analyzed from left to right and any intermediate results are stored in up to 24 locations in the core (the stack) which is sufficient for fairly complex expressions. All numeric operations are carried out in double-precision floating-point FORTRAN arithmetic. As an example, \textsc{RADIUS}(2) = \textsc{RADIUS}(2) \times \textsc{SQRT}(\textsc{RADIUS}(2)) or \textsc{RADIUS}(2) = \textsc{PWR}(\textsc{RADIUS}(2), 1.5) will set \textsc{RADIUS}(2) = 8^{3/2}.

\textbf{Alphanumeric expressions.} - Alphanumeric data may be entered by placing the variable name in the "\textsc{REAL}" list and then setting the variable equal to the data by first enclosing in parentheses the length of the word to be read in. As an example,

\begin{equation}
Q = (A39)\text{THIS IS AN EXAMPLE OF ALPHANUMERIC DATA}
\end{equation}

The (A39) gives the length of the data including imbedded blanks. Of course, since on the IBM 7094 there are six characters per word, \( Q \) must internally be dimensioned to at least 7.

\textbf{Printing Input Cards}

Each input card processed will normally be written on the tape specified by the second argument of the calling sequence. An end-of-statement symbol read on the card will cause interpretation of the card to stop at that point and permit comments to be placed on the remainder of the card to be printed with the output. In order to avoid printing the card at all, the nonprint character is placed in the next column following the end-of-statement character. The developers of the routine selected the sign + for both characters. This is punched as a colon on an IBM Model 29 Keypunch and corresponds to a 2-8 punch.

If the character following the end-of-statement symbol is other than a nonprint character, it is inserted as the printer control character in the first position of the output format before the card is written on the output tape. If no end-of-statement character occurs on the card, a blank printer control character is used. Comment cards having the end-of-statement character as the first nonblank character will be printed and may be placed anywhere except in a continued alphabetic field.
In summary, the end-of-statement character has the effect of moving the end of the card forward to the column ahead of the end-of-statement character. The column following it is printer control.

If the control parameter NOLIST is true, printing is suppressed for all cards.
APPENDIX C

SYMBOLS

General Symbols Internal to Program

Variables in program are formed by combining these symbols.

Station Numbers

See figures 1 to 4 for each type of engine.

Thermodynamic Properties

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>AM</td>
<td>Mach number</td>
</tr>
<tr>
<td>FAR</td>
<td>fuel-air ratio, f/a</td>
</tr>
<tr>
<td>H</td>
<td>enthalpy, Btu/lbm</td>
</tr>
<tr>
<td>P</td>
<td>total pressure, atm</td>
</tr>
<tr>
<td>PS</td>
<td>static pressure, atm</td>
</tr>
<tr>
<td>S</td>
<td>entropy, Btu/°R/lbm</td>
</tr>
<tr>
<td>T</td>
<td>total temperature, °R</td>
</tr>
<tr>
<td>TS</td>
<td>static temperature, °R</td>
</tr>
<tr>
<td>V</td>
<td>velocity, ft/sec</td>
</tr>
</tbody>
</table>

Component Symbols

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>A,AFT</td>
<td>afterburner</td>
</tr>
<tr>
<td>B</td>
<td>burner</td>
</tr>
<tr>
<td>C</td>
<td>inner compressor</td>
</tr>
<tr>
<td>COM</td>
<td>combustor</td>
</tr>
<tr>
<td>D</td>
<td>fan duct</td>
</tr>
<tr>
<td>F</td>
<td>first or fan compressor</td>
</tr>
</tbody>
</table>
M core nozzle
NOZ nozzle
OB overboard
T total
THP inner (high pressure) turbine
TLP outer (low pressure) turbine

Engine Symbols

BL bleed, lbm/sec
CN ratio of corrected speed to design corrected speed
DHT turbine delta enthalpy, Btu/lbm
DHTC turbine delta enthalpy (temperature corrected), \((H_{in} - H_{out})/T_{in}\), Btu/°R/lbm
DP pressure drop, \(\Delta P/P\)
DT temperature change, °R
ETA efficiency
ETAR ram recovery, \(P_2/P_1\)
HPEXT horsepower extracted
PCBL fractional bleed
PCN percent of design shaft speed
PR pressure ratio
TFF turbine flow function, \(\text{lbm} \sqrt{\text{°R}/(\text{psia})(\text{sec})}\)
WA airflow, lbm/sec
WF fuel flow, lbm/sec
WG gas flow, lbm/sec
Z ratio of pressure ratios

Miscellaneous Symbols

A area, ft
ALTP altitude, ft
142
<table>
<thead>
<tr>
<th>Symbol</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AM</td>
<td>Mach number of aircraft</td>
</tr>
<tr>
<td>BYPASS</td>
<td>bypass ratio (fan duct air/air entering intermediate compressor)</td>
</tr>
<tr>
<td>C</td>
<td>when following component symbol, signifies &quot;corrected&quot;</td>
</tr>
<tr>
<td>CF</td>
<td>correction factor, when used following component symbol</td>
</tr>
<tr>
<td>CS</td>
<td>ambient speed of sound, ft/sec</td>
</tr>
<tr>
<td>CV</td>
<td>nozzle velocity coefficient</td>
</tr>
<tr>
<td>DEL</td>
<td>delta degradation coefficient</td>
</tr>
<tr>
<td>DS</td>
<td>design value</td>
</tr>
<tr>
<td>DUM</td>
<td>dummy value</td>
</tr>
<tr>
<td>FG</td>
<td>gross thrust, lbf</td>
</tr>
<tr>
<td>FGM</td>
<td>momentum thrust, lbf</td>
</tr>
<tr>
<td>FGP</td>
<td>pressure thrust, lbf</td>
</tr>
<tr>
<td>FN</td>
<td>net thrust, lbf</td>
</tr>
<tr>
<td>FRD</td>
<td>ram drag, lbf</td>
</tr>
<tr>
<td>GU</td>
<td>initial or guessed values</td>
</tr>
<tr>
<td>ITRYS</td>
<td>number of loops through engine before quitting</td>
</tr>
<tr>
<td>LOOP</td>
<td>variable counter</td>
</tr>
<tr>
<td>LOOPER</td>
<td>number of loops through engine counter</td>
</tr>
<tr>
<td>SFC</td>
<td>specific fuel consumption, lbm/lbf/hr</td>
</tr>
<tr>
<td>TOLALL</td>
<td>tolerance on convergence</td>
</tr>
<tr>
<td>VA</td>
<td>velocity of aircraft, ft/sec</td>
</tr>
<tr>
<td>VJ</td>
<td>jet velocity, ft/sec</td>
</tr>
</tbody>
</table>

**Input Symbols**

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ALTP</td>
<td>altitude, ft</td>
</tr>
<tr>
<td>AM</td>
<td>Mach number of aircraft</td>
</tr>
<tr>
<td>AM6</td>
<td>design afterburner entrance Mach number</td>
</tr>
<tr>
<td>AM23</td>
<td>design duct-burner entrance Mach number</td>
</tr>
</tbody>
</table>
design low-pressure-turbine exit Mach number
area at afterburner entrance (calculated from AM6), ft\(^2\)
main nozzle throat area (can be changed at off-design), ft\(^2\)
fan duct nozzle throat area (see A8), ft\(^2\)
design corrected speed - inner turbine
design corrected speed - outer turbine
nozzle thrust coefficient (tabular lookup)
nozzle thrust coefficient (tabular lookup)
gross-thrust delta degradation multiplier
net-thrust delta degradation multiplier
specific-fuel-consumption delta degradation multiplier
afterburner design pressure drop, \(\Delta P/P\)
combustor design pressure drop, \(\Delta P/P\)
duct design pressure drop, \(\Delta P/P\)
combustor design temperature increase (automatically set to \(T_4 - T_3\)), \(^\circ\)R
afterburner efficiency (not required)
afterburner efficiency at design
combustor efficiency at design
inner compressor adiabatic efficiency at design
duct-burner combustion efficiency
front (outer) compressor adiabatic efficiency at design
inlet pressure recovery (ram recovery)
high-pressure- (inner) turbine design adiabatic efficiency
low-pressure- (outer) turbine design adiabatic efficiency
logical control for fan and turbojet engines
horsepower extraction
index on afterburning desired
index on ram or inlet operation desired
index on duct burning desired
duct nozzle convergent-divergent when IDCD equals 1 (design or off-design)
<table>
<thead>
<tr>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>IDES</td>
<td>index for design point; must be set equal to 1 to design engine; zeroed automatically</td>
</tr>
<tr>
<td>IDUMP</td>
<td>index for dumping of error matrix</td>
</tr>
<tr>
<td>IGASMX</td>
<td>index for mixed-flow or non-mixed-flow turbofan</td>
</tr>
<tr>
<td>IMCD</td>
<td>main nozzle convergent-divergent when IMCD equals 1 (design or off-design)</td>
</tr>
<tr>
<td>ISPOOL</td>
<td>index for number of compressors for turbojet engines</td>
</tr>
<tr>
<td>ITRYS</td>
<td>index for maximum number of iterations</td>
</tr>
<tr>
<td>MODE</td>
<td>independent variable designator for engine operation</td>
</tr>
<tr>
<td>NOZFLT</td>
<td>index for floating main or duct nozzle</td>
</tr>
<tr>
<td>PCBLC</td>
<td>ratio of compressor bleed to turbines to compressor airflow</td>
</tr>
<tr>
<td>PCBLDU</td>
<td>ratio of compressor bleed leaked into fan duct to total compressor bleed flow</td>
</tr>
<tr>
<td>PCBLF</td>
<td>ratio of bleed from outer compressor to fan airflow dumped overboard (i.e., leakage)</td>
</tr>
<tr>
<td>PCBLHP</td>
<td>fraction of PCBLC used for high-pressure (inner) turbine (cooling)</td>
</tr>
<tr>
<td>PCBLLP</td>
<td>fraction of PCBLC used for low-pressure (outer) turbine (cooling)</td>
</tr>
<tr>
<td>PCBLOB</td>
<td>ratio of inner compressor bleed to compressor airflow (overboard for customer use)</td>
</tr>
<tr>
<td>PCNC</td>
<td>inner-compressor shaft speed as a percent of design shaft speed</td>
</tr>
<tr>
<td>PCNCDS</td>
<td>design inner-compressor shaft speed</td>
</tr>
<tr>
<td>PCNF</td>
<td>outer-compressor shaft speed as percent of design</td>
</tr>
<tr>
<td>PCNFDS</td>
<td>design outer-compressor shaft speed as percent of design</td>
</tr>
<tr>
<td>PRCDS</td>
<td>design inner-compressor pressure ratio</td>
</tr>
<tr>
<td>PRFDS</td>
<td>design outer-compressor pressure ratio</td>
</tr>
<tr>
<td>PS55</td>
<td>static pressure at low-pressure-turbine exit</td>
</tr>
<tr>
<td>P2</td>
<td>compressor-face total pressure (for nonstandard days only), atm</td>
</tr>
<tr>
<td>TFHPDS</td>
<td>design inner-turbine flow function</td>
</tr>
<tr>
<td>TFLPDS</td>
<td>design outer-turbine flow function</td>
</tr>
<tr>
<td>TOLALL</td>
<td>tolerance on error matrix</td>
</tr>
<tr>
<td>T2</td>
<td>compressor-face total temperature (for nonstandard days only), T1 + T2</td>
</tr>
<tr>
<td>T24</td>
<td>duct-burner exit temperature, °R</td>
</tr>
</tbody>
</table>
T4 combustion exit temperature, °R
T4DS design combustion exit temperature, °R
T7 afterburner exit temperature, °R
T7DS design afterburner exit temperature, °R
WACCDS design inner-compressor corrected airflow (turbofans only), lbm/sec
WAFCDS design outer-compressor corrected airflow, lbm/sec
WFA fuel flow rate to afterburner (IAFTBN = 2 only), lbm/sec
WFB fuel flow rate to main burner (MODE = 2 only), lbm/sec
WFBDS design fuel flow rate to main burner (MODE = 2 only), lbm/sec
ZCDS, ZFDS design ratio of inner compressor and fan compressor pressure ratios, respectively; equals pressure ratio at design point on design speed line minus value of pressure ratio of lowest point on speed line divided by high (surge) value minus low value of pressure ratio on the design speed line

Output Symbols

A area, ft²
ALTP altitude, ft
AM Mach number
BLC bleed flow out of compressor, lbm/sec
BLF bleed flow out of fan (dumped overboard), lbm/sec
BLHP bleed flow into high-pressure turbine, lbm/sec
BYPASS ratio of airflow into fan duct to airflow into inner compressor
CNC corrected shaft speed - inner compressor
CNF corrected shaft speed - fan
CNHP corrected shaft speed - high-pressure turbine, PCNC/√T_{in}
CNHPCF corrected speed - high-pressure-turbine correction factor
CNLP corrected speed - low-pressure turbine, PCNF/√T_{in}

1Some symbols, such as T4, are followed by station numbers, see appropriate figure for each engine to determine station numbers.
CNLPCF  corrected speed - low-pressure-turbine correction factor
CVDNOZ  velocity coefficient of fan nozzle
CVMNOZ  velocity coefficient of core nozzle
DHHPCF  high-pressure-turbine delta enthalpy correction factor
DHLPCF  low-pressure-turbine delta enthalpy correction factor
DHTC    work done by high-pressure turbine, Btu/lbm
DHTCHP  enthalpy change temperature corrected - high-pressure turbine,
         Btu/°R/atm/lbm
DHTCLP  enthalpy change temperature corrected - low-pressure turbine,
         Btu/°R/atm/lbm
DHTF    work done by low-pressure turbine, Btu/lbm
DPCOM   $(\Delta P/P)_{\text{combustor}}$
DPDUC   $(\Delta P/P)_{\text{fan duct}}$
DTCOCF  temperature-rise-across-combustor correction factor
ETAB    combustor efficiency
ETABCF  combustor efficiency correction factor
ETAC    inner-compressor adiabatic efficiency
ETACCF  inner-compressor efficiency correction factor
ETAD    duct-burner efficiency
ETAF    fan adiabatic efficiency
ETAFCF  fan efficiency correction factor
ETATHP  high-pressure-turbine adiabatic efficiency
ETATLP  low-pressure-turbine adiabatic efficiency
ETHPCF  high-pressure-turbine efficiency correction factor
ETLPCF  low-pressure-turbine efficiency correction factor
FAR     fuel-air ratio, f/a
FG      gross thrust, lbf
FGM     momentum thrust, lbf
FGP     pressure thrust, lbf
FN      net thrust, lbf
FRD  ram drag, lbf
HPEXT horsepower extracted, hp
P  pressure, atm
PCBLC fraction of compressor exit air bled for cooling or lost to cycle
PCBLDU fraction of bled air out of compressor which leaks into fan duct
PCBLF fraction of fan exit airflow lost overboard
PCBLHP fraction of compressor bleed air put into high-pressure turbine
PCBLLP fraction of compressor bleed air put into low-pressure turbine
PCNC inner compressor shaft speed as fraction of design
PCNF fan compressor shaft speed as fraction of design
PRC pressure ratio of inner compressor
PRCCF pressure-ratio-of-inner-compressor correction factor
PRF pressure ratio of fan or outer compressor
PRFCF pressure-ratio-of-fan-or-outer-compressor correction factor
PS static pressure, atm
SFC specific fuel consumption, lbm/(lbm/hr)
T  temperature, °R
TFFHP high-pressure-turbine flow function, \( \frac{(\text{lbm})\sqrt{\text{O}R}}{(\text{in.})^2}/\text{(sec)(lbf)} \)
TFFLP low-pressure-turbine flow function, \( \frac{(\text{lbm})\sqrt{\text{O}R}}{(\text{in.})^2}/\text{(sec)(lbf)} \)
TFHPCF high-pressure-turbine flow function correction factor
TFLPCF low-pressure-turbine flow function correction factor
T2DS design exit temperature of fan, °R
T21DS design exit temperature of inner compressor, °R
V  velocity, ft/sec
VA velocity of aircraft, ft/sec
VJD fan duct exhaust velocity, ft/sec
VJM core exhaust velocity, ft/sec
WA airflow, lbm/sec
WA3CDS corrected airflow in combustor at design, lbm/sec
WAC inner-compressor airflow, lbm/sec
<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>WACC</td>
<td>inner-compressor corrected airflow, lbm/sec</td>
</tr>
<tr>
<td>WACCF</td>
<td>inner-compressor corrected airflow correction factor</td>
</tr>
<tr>
<td>WAD</td>
<td>fan duct airflow, lbm/sec</td>
</tr>
<tr>
<td>WAF</td>
<td>fan airflow, lbm/sec</td>
</tr>
<tr>
<td>WAFC</td>
<td>fan corrected airflow, lbm/sec</td>
</tr>
<tr>
<td>WAFCF</td>
<td>fan corrected airflow correction factor</td>
</tr>
<tr>
<td>WFA</td>
<td>fuel flow rate to afterburner, lbm/sec</td>
</tr>
<tr>
<td>WFB</td>
<td>fuel flow rate to combustor, lbm/sec</td>
</tr>
<tr>
<td>WFD</td>
<td>fuel flow rate to duct burner, lbm/sec</td>
</tr>
<tr>
<td>WFT</td>
<td>total fuel flow rate, lbm/sec</td>
</tr>
<tr>
<td>WG</td>
<td>gas flow rate, lbm/sec</td>
</tr>
<tr>
<td>WGT</td>
<td>total gas flow rate, lbm/sec</td>
</tr>
<tr>
<td>ZC</td>
<td>ratio of inner compressor pressure ratios</td>
</tr>
<tr>
<td>ZF</td>
<td>ratio of fan pressure ratios</td>
</tr>
</tbody>
</table>
REFERENCES


<table>
<thead>
<tr>
<th>Variable</th>
<th>Two-spool turbofan</th>
<th>Mixed-flow turbofan</th>
<th>Two-spool turbojet</th>
<th>One-spool turbojet</th>
</tr>
</thead>
<tbody>
<tr>
<td>Variable 1</td>
<td>ZF</td>
<td>ZF</td>
<td>ZF</td>
<td>ZF</td>
</tr>
<tr>
<td>Variable 2</td>
<td>PCNF</td>
<td>PCNF</td>
<td>PCNF</td>
<td>PCNF</td>
</tr>
<tr>
<td>Variable 3</td>
<td>ZC</td>
<td>ZC</td>
<td>ZC</td>
<td>TFFLP</td>
</tr>
<tr>
<td>Variable 4</td>
<td>PCNC</td>
<td>PCNC</td>
<td>PCNC</td>
<td>TFFLP</td>
</tr>
<tr>
<td>Variable 5</td>
<td>TFFHP</td>
<td>TFFHP</td>
<td>TFFHP</td>
<td>TFFLP</td>
</tr>
<tr>
<td>Variable 6</td>
<td>TFFLP</td>
<td>TFFLP</td>
<td>TFFLP</td>
<td>TFFLP</td>
</tr>
<tr>
<td>Error 1</td>
<td>TFHCAL - TFFHP</td>
<td>TFHCAL</td>
<td>TFHCAL</td>
<td>TFHCAL</td>
</tr>
<tr>
<td>Error 2</td>
<td>DHTCC - DHTCHP</td>
<td>DHTCC</td>
<td>DHTCC</td>
<td>DHTCC</td>
</tr>
<tr>
<td>Error 3</td>
<td>TFLCAL - TFFLP</td>
<td>TFLCAL</td>
<td>TFLCAL</td>
<td>TFLCAL</td>
</tr>
<tr>
<td>Error 4</td>
<td>DHTCF - DHTCLP</td>
<td>DHTCF</td>
<td>DHTCF</td>
<td>DHTCF</td>
</tr>
<tr>
<td>Error 5</td>
<td>P25R - P25</td>
<td>PS25 - PS55</td>
<td>WAF - WAC - BLF</td>
<td>WAC</td>
</tr>
<tr>
<td>Error 6</td>
<td>P7R - P7</td>
<td>P7R</td>
<td>P7R</td>
<td>P7R</td>
</tr>
<tr>
<td>Matrix size</td>
<td>6 x 6</td>
<td>6 x 6</td>
<td>6 x 6</td>
<td>3 x 3</td>
</tr>
<tr>
<td>Variable</td>
<td>Units</td>
<td>Definition</td>
<td>Two-spool turbofan</td>
<td>Mixed-flow turbofan</td>
</tr>
<tr>
<td>----------</td>
<td>-------------</td>
<td>-------------------------------------------</td>
<td>--------------------</td>
<td>---------------------</td>
</tr>
<tr>
<td>PRFDS</td>
<td></td>
<td>Fan pressure ratio</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>WAFCDS</td>
<td>lb/sec</td>
<td>Fan corrected airflow</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>ETAFDS</td>
<td></td>
<td>Fan efficiency</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>ZFDS</td>
<td></td>
<td>Design Z of fan</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>PCNFDS</td>
<td></td>
<td>Corrected speed of fan</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>PRCDs</td>
<td></td>
<td>Compressor pressure ratio</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>WACCDS</td>
<td>lb/sec</td>
<td>Compressor corrected airflow</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>ETACDS</td>
<td></td>
<td>Compressor efficiency</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>ZCDS</td>
<td></td>
<td>Design Z of compressor</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>PCNCDs</td>
<td></td>
<td>Corrected speed of compressor</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>ETABDS</td>
<td></td>
<td>Combustor efficiency</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>DPCODS</td>
<td></td>
<td>Combustor pressure drop, $\Delta P/P$</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>T4DS</td>
<td>$O_R$</td>
<td>Turbine inlet temperature</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>TFHPDS</td>
<td>$\frac{lb\sqrt{R}}{R}$ (sec)(psia)</td>
<td>High-pressure-turbine flow function</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>CNHPDS</td>
<td></td>
<td>Corrected speed - high-pressure turbine</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>ETHPDS</td>
<td></td>
<td>Efficiency - high-pressure turbine</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>TFLPDS</td>
<td>$\frac{lb\sqrt{R}}{R}$ (sec)(psia)</td>
<td>Low-pressure-turbine flow function</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>CNLPDS</td>
<td></td>
<td>Corrected speed - low-pressure turbine</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>ETLPDS</td>
<td></td>
<td>Efficiency - low-pressure turbine</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>DPDUDS</td>
<td></td>
<td>$\Delta P/P$ of fan duct</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>DPAFDS</td>
<td></td>
<td>$\Delta P/P$ of afterburner</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>FAN</td>
<td></td>
<td>Logical variable</td>
<td>TRUE.</td>
<td>TRUE.</td>
</tr>
<tr>
<td>ISPOOL</td>
<td></td>
<td>Number of spools</td>
<td>2</td>
<td>2</td>
</tr>
</tbody>
</table>
Figure 1. - Schematic of non-mixed-flow duct-burning and/or afterburning turbofan.

Figure 2. - Schematic of mixed-flow afterburning turbofan.
Figure 3. - Schematic of two-spool turbojet.

Figure 4. - Schematic of one-spool turbojet.
Figure 5. - Flow chart for GENENG computer program.
High pressure ratio, $PR_{\text{high}}$

Design pressure ratio, $PR_{\text{des}}$

Low pressure ratio, $PR_{\text{low}}$

Corrected airflow, $WAC$

Surge line

Constant corrected speed, $CN$

Design point

Constant efficiency, $ETA$

Figure 6. - Example of a specific fan-compressor map. $Z = (PR_{x} - PR_{\text{low}})/(PR_{\text{high}} - PR_{\text{low}})$

Constant input pressure, $P_3$

Efficiency, $ETA$

Temperature rise, $DT = T_4 - T_3$

Figure 7. - Example of combustor map.
Figure 8. Example of specific turbine map.

2. Efficiency correction factor as a function of afterburner inlet Mach number.

3. Efficiency correction factor as a function of afterburner inlet total pressure ratio.

Figure 9. Example of a generalized afterburner combustion efficiency performance map.

Figure 10. Performance map for nozzle, giving velocity coefficient.
"The aeronautical and space activities of the United States shall be conducted so as to contribute... to the expansion of human knowledge of phenomena in the atmosphere and space. The Administration shall provide for the widest practicable and appropriate dissemination of information concerning its activities and the results thereof."

—National Aeronautics and Space Act of 1958

NASA SCIENTIFIC AND TECHNICAL PUBLICATIONS

TECHNICAL REPORTS: Scientific and technical information considered important, complete, and a lasting contribution to existing knowledge.

TECHNICAL NOTES: Information less broad in scope but nevertheless of importance as a contribution to existing knowledge.

TECHNICAL MEMORANDUMS: Information receiving limited distribution because of preliminary data, security classification, or other reasons.

CONTRACTOR REPORTS: Scientific and technical information generated under a NASA contract or grant and considered an important contribution to existing knowledge.

TECHNICAL TRANSLATIONS: Information published in a foreign language considered to merit NASA distribution in English.

SPECIAL PUBLICATIONS: Information derived from or of value to NASA activities. Publications include conference proceedings, monographs, data compilations, handbooks, sourcebooks, and special bibliographies.

TECHNOLOGY UTILIZATION PUBLICATIONS: Information on technology used by NASA that may be of particular interest in commercial and other non-aerospace applications. Publications include Tech Briefs, Technology Utilization Reports and Technology Surveys.

Details on the availability of these publications may be obtained from:

SCIENTIFIC AND TECHNICAL INFORMATION OFFICE

NATIONAL AERONAUTICS AND SPACE ADMINISTRATION

Washington, D.C. 20546