AFAPL-TR-76-43 VOLUME II

# AIRCRAFT HYDRAULIC SYSTEMS DYNAMIC ANALYSIS

## VOLUME II
## TRANSIENT ANALYSIS
## (HYTRAN)
### COMPUTER PROGRAM
### TECHNICAL DESCRIPTION

MCDONNELL AIRCRAFT COMPANY
MCDONNELL DOUGLAS CORPORATION
ST. LOUIS, MISSOURI



*February 1977*

TECHNICAL REPORT AFAPL-TR-76-43, VOLUME II



AIR FORCE AERO PROPULSION LABORATORY
AIR FORCE SYSTEMS COMMAND
WRIGHT-PATTERSON AIR FORCE BASE, OHIO 45433

NOTICE

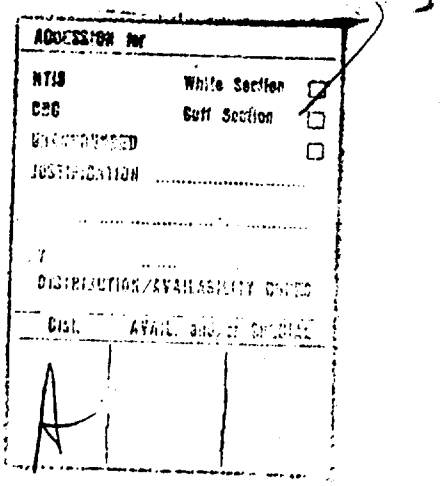
When Government drawings, specifications, or other data are used for any purpose other than in connection with a definitely related Government procurement operation, the United States Government thereby incurs no responsibility nor any obligation whatsoever; and the fact that the government may have formulated, furnished, or in any way supplied the said drawings, specifications, or other data, is not to be regarded by implication or otherwise as in any manner licensing the holder or any other person or corporation, or conveying any rights or permission to manufacture, use, or sell any patented invention that may in any way be related thereto.

This report was submitted by McDonnell Douglas Corporation, under contract F3615-74-C-2016.

The effort was sponsored by the Air Force Aero Propulsion Laboratory, Air Force Systems Command, Wright-Patterson A.F.B., Ohio, under Project No. 3145-30-18 . h AFAPL/POP/, and was under the direction of Paul Lindquist and William Kinzig.

Neil Pierce and Gerry Amies of McDonnell Douglas Corporation were technically responsible for the work.

This report has been reviewed by the Information Office, (ASD/OIP) and is releasable to the National Technical Information Service (NTIS). At NTIS, it will be available to the general public, including foreign nations.




Copies of this report should not be returned unless return is required by security considerations, contractual obligations, or notice on a specific document.

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE *(When Data Entered)*

## REPORT DOCUMENTATION PAGE

READ INSTRUCTIONS BEFORE COMPLETING FORM

1. REPORT NUMBER: AFAPL-TR-76-43 VOL II

2. GOVT ACCESSION NO.

3. RECIPIENT'S CATALOG NUMBER

4. TITLE *(and Subtitle)*: AIRCRAFT HYDRAULIC SYSTEM DYNAMIC ANALYSIS / VOLUME II, TRANSIENT ANALYSIS (HYTRAN) COMPUTER PROGRAM TECHNICAL DESCRIPTION.

5. TYPE OF REPORT & PERIOD COVERED: Interim Technical Report,

6. PERFORMING ORG. REPORT NUMBER

7. AUTHOR(s): Gerry Amies, Ray Levek, Dave Struessel

8. CONTRACT OR GRANT NUMBER(s): F3615-74-C-2016

9. PERFORMING ORGANIZATION NAME AND ADDRESS: McDonnell Douglas Corporation, P O Box 516, St. Louis, Missouri 63166

10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS: 3145-30-18

11. CONTROLLING OFFICE NAME AND ADDRESS: Air Force Aero Propulsion Laboratory, Air Force Systems Command, Wright-Patterson Air Force Base, Ohio 45433

12. REPORT DATE: February 1977

13. NUMBER OF PAGES

14. MONITORING AGENCY NAME & ADDRESS *(if different from Controlling Office)*: F33615-74-C-2016

15. SECURITY CLASS. *(of this report)*: UNCLASSIFIED

15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT *(of this Report)*

Approved for public release, distribution unlimited

3145 30 AFAPL

17. DISTRIBUTION STATEMENT *(of the abstract entered in Block 20, if different from Report)*

TR-76-43-Vol-2

18. SUPPLEMENTARY NOTES

19. KEY WORDS *(Continue on reverse side if necessary and identify by block number)*

| | | |
|---|---|---|
| Computer Program | Waterhammer | Reservoir |
| Hydraulic System | Hydraulic Pump | Restrictor |
| Transient Response | Actuator | Valve |
| Technical Manual | Hydraulic Filter | |

20. ABSTRACT *(Continue on reverse side if necessary and identify by block number)*

The hydraulic transient analysis (HYTRAN) computer program has been developed to simulate the response of a hydraulic system to sudden changes in flow demand by the system loads.

For selected component temperatures, pump RPM, and initial steady state conditions, the program will calculate the pressures and flow amplitudes resulting from changes in flow demand or some other controller input.

It will predict transient pressures due to waterhammer and the onset of cavitation due to the opening and closing of valves.

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

403117

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE *(When Data Entered)*

The engineering input data to the program is normally available to a design engineer. When specialized components are required that are not covered by existing subroutines, these may be simulated by adding to the program.

TABLE OF CONTENTS

TABLE OF CONTENTS (Continued)

TABLE OF CONTENTS (Continued)

## LIST OF FIGURES

LIST OF FIGURES (Continued)

LIST OF FIGURES (Continued)

# 1.0 INTRODUCTION

The hydraulic transient analysis (HYTRAN) computer program is intended for use by designers with an interest in the detailed performance of an aircraft hydraulic system or the response of a load, where the supply system is an integral part of that response.

An aircraft hydraulic system is basically a power source connected to several loads. Under steady state conditions, where only the pump and fluid are moving, the flows and pressures at various points in the system can be calculated using non-time dependent formulae. However the unsteady flow conditions which are more normal, cannot be analyzed using simple formulae. The pump is basically a closed loop servo which has a time varying output and responds continuously to system pressure changes. These changes propagate through the system at the speed of sound, which is about 4000 ft/sec. The system components respond to these pressure and flow changes, and to external load and control disturbances.

The program simulates the complete system and calculates the value of all the flows, pressures and state variables throughout the system.

This allows the designer to study the dynamic response of any variable, such as a check valve poppet position, an actuator piston velocity, the pump swash plate acceleration, etc., since all these variables are calculated as part of the system simulation.

The program is composed of five basic parts; input, steady state calculation, line simulation, component simulation, and output.

The designer inputs data describing the lines, components, and system configuration. Since the simulation is only as good as the data, some of the information required for components such as a pump, is very detailed.

Fortunately there are only a few components like this and often these are common to many systems, e.g. DC-10 pumps are used on the 747, L1011 and A300 aircraft.

The steady state section of the program balances the pressures and flows in the system and calculates the initial values for all the system state variables. Once the initial values are established at zero time, the program starts by calculating for a small change in time (delta T), new flows and pressures at the junction between the line segments.

The lines are divided into segments, the length of each segment being greater than or equal to the velocity of sound in the line divided by the time interval delta T. There is a whole number of segments for each line.

Once the new pressures and flows have been established for the line junctions, the program calculates new values for the state variables of all the components, and the flows and pressures at the junctions between the components and the lines.

The program continues to march forward in delta T time intervals, first calculating the line and then the component variables.

The output part of the program selects the variables that are required as output or output plots, at specified time steps, since it is not always necessary to plot every value that was calculated. When the program calculations are completed, the output is then printed and plotted.

The controlling input to the system will usually be a sudden load demand from a surface actuator or some similar load function. This is input as a time dependent valve motion or input demand.

The output is essentially a time history of selected system variables which have been disturbed by the controlling input.

Since the program actually advances in discrete time steps, it can be integrated into other simulations, if the cost of running can be tolerated.

This report is a technical description of the HYTRAN Program, including a detailed listing of the main program and subprograms, and the theoretical basis and assumptions made in the calculations.

Volume 1 of this report is a users manual which describes how the program can be used, the method of inputting data and the interpretation of the output.

## 2.0 TECHNICAL SUMMARY

The HYTRAN program is intended for use by engineers with different interests. Some will be concerned with the performance of the hydraulic system as a whole, while others will be interested in the detailed performance of individual components.

HYTRAN uses a building block approach which allows the programmer to meet these needs by adding special component subroutines as required to the existing component subroutine library.

The program is supported by a number of specialized utility routines, which have been included to avoid program incompatibility with other computer systems. In the development of HYTRAN, the emphasis was placed on the performance of the hydraulic system as a whole, and its components are considered only to the extent to which they affect the total system response.

The transient analysis is a digital simulation process, which treats the fluid lines with distributed parameters, applying the concepts of wave mechanics, and including the effects of nonlinear friction. The fluid line equations are solved with the help of the method of characteristics. The dynamic equations of the components are either algebraic or ordinary differential equations. These form the boundary conditions of the lines and are solved simultaneously with the associated line characteristic equations. A numerical scheme is used to make the grid of characteristics compatible with the integration techniques used by the components.

The input to the system is normally a valve motion, which causes a disturbance to propagate through the mathematical model. The output of the program is the time histories of pressure and flows at any point in the system and other variables of interest such as actuator positions.

In the simulation of the components, the precision of the model used will depend upon its use. If the user is studying the dynamic stability of a pump system, then an accurate model is required. If, however, the user is studying an actuator out at the end of the line system, the pump response could be simulated using a simpler model; hence saving some running costs. In a similar manner, actuator friction has a significant effect on its small amplitude response, but such friction is of little interest if the actuator is being used as large demand load in the study of pump stability.

The dynamics of components such as pumps are very dependent on the dynamic properties of the connecting lines and components; hence it is important in simulations involving these components that an accurate system simulation be used.

The results which are obtained from HYTRAN are solutions of the differential and algebraic equations used to describe the system dynamics. The solutions are obtained by methods of numerical analysis, such as Runge Kutta numerical integration procedures, method of characteristics, and Lagrange interpolations. They are, therefore, subject to the errors which are inherent in numerical methods, but which can be kept small enough to be of no practical influence. Of more importance than the numerical inaccuracies are the underlying assumptions and restrictions imposed upon the basic equations.

A digital simulation has been chosen because of some important advantages over the simulation on an analog computer. These are, in particular, the high accuracy in conjunction with an almost unlimited memory capacity, the reliability, the absence of scaling problems, and the difficulty of modeling wave phenomena on analog computers.

The numerical aspects of digital simulation are described in a variety of textbooks. The concepts of the method of characteristic are explained in Appendix A and in more general terms in the description of the line subroutine.

## 3.0 MAIN PROGRAM

The main or executive program section of HYTRAN is named HYTR. HYTR controls the flow of the program, and keeps track of the counters for time variables. The BLOCK DATA and FLUID subroutines are also included in this section.

HYTRAN is a program that has evolved over a number of years, with changes being made from time to time to simplify the programming procedures and additions being made to expand its capabilites.

Since the program is still in the development stage, changes and additions will continue to be made whenever significant improvements can be achieved.

Some cost savings can be made by the use of OVERLAYS or SEGMENTS, but it was decided that the use of these devices should be left to the individual user.

## 3.1 HYTR Program

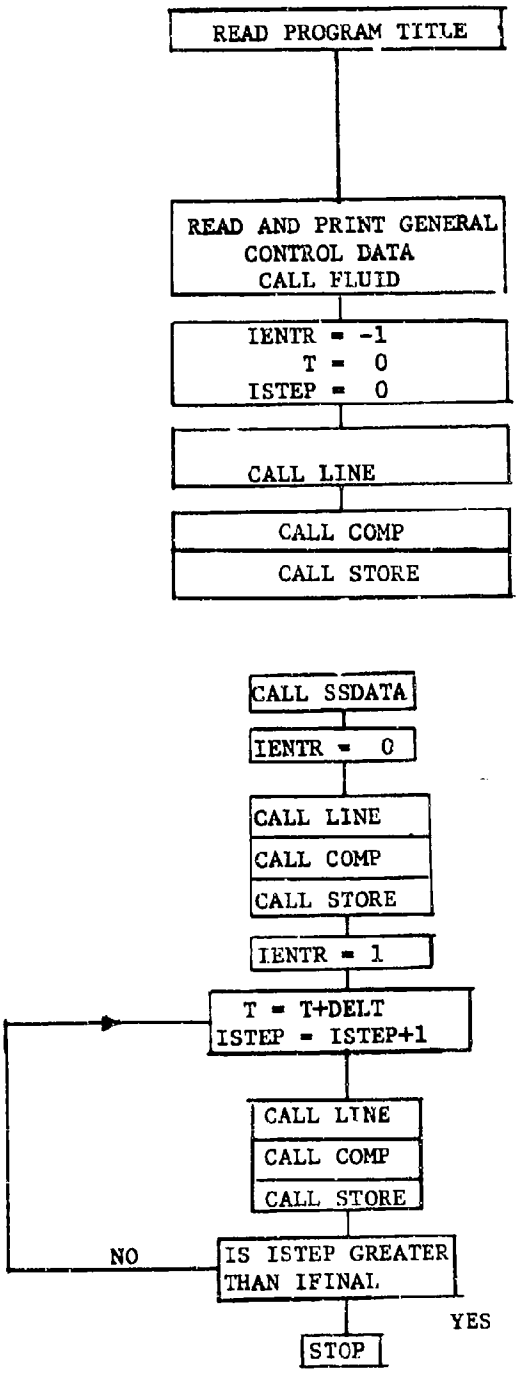
HYTR is the main or executive program of HYTRAN. The program flow is directed from HYTR, but there are no significant calculations made there. Only general control data is read in HYTR. The main program card is set up to read from a file called DATA. This should be changed to suit the user's own data inputting scheme. Extensive use is made of common and equivalences, so that care is required in modifying variables that are contained therein.

The first section of HYTR reads and prints data from general control cards and calls FLUID to calculate values of bulk modulus, viscosity and density. FLUID subroutine then prints all the calculated data. In event the fluid data is input, FLUID will then just print the input data.

The second section calls LINE which reads line data then calculates constants, initializes variables and writes data to output. Next, COMP is called to read and print component data and to set constants and initialize variables for all components. STORE is called to read all the output data requirements and then SSDATA is called which reads in the system arrangement data and performs steady state calculations.

The third section again calls LINE and COMP to initialize all the system state variables. STORE is then called to store pressures, flows and/or component variables.

The fourth section advances the time step by DELT. LINE and COMP are called to do the transient calculations. This step is repeated until the sum of the time intervals is equal to the input total run time. During this iterative procedure, variables to be plotted are stored by STORE.




HYTR FLOW DIAGRAM
FIGURE 3.1-1

When time T exceeds the final time specified in the input, the program stops.

3.1.1 Math Model. Not applicable.

3.1.2 Assumptions.

The basic assumptions in HYTR are as follows:

- o Fluid temperature is assumed constant during the entire run.
- o Flow is one-dimensional, that is, the fluid properties are constant across any transverse cross section of the pipe.
- o Pipes have circular cross sections.
- o Stresses in pipes are always below the elastic limit.
- o Pipe geometry is such that the "thin wall" case is valid.
- o Pipe and liquid are perfectly elastic (all energy dissipation is due to shearing stresses at the walls).

3.1.3 Computation Methods. Not applicable

3.1.4 Approximations.

HYTR approximations are those inherent in numerical analysis. They are kept small enough by error control to be of no practical influence.

3.1.5 Limitations.

HYTR currently has the following constraints:

- o Temperature range ... -65°F to 300°F
- o Pressure range ... 0 psia to 5000 psia
- o Maximum number of components ... 60
- o Maximum number of lines ... 79
- o Maximum number of legs ... 70
- o Maximum number of nodes ... 50
- o Maximum number of plots ... 60

### 3.1.6 Variable Names

| Variable | Description | Units |
|---|---|---|
| I | Counter | -- |
| IF | Fluid type indicator | -- |
| IFINAL | Number of transient iterations | -- |
| M | Dummy Variable | -- |
| MM | Address = LSTART ( ) + NLPT-1 | -- |
| N | Counter | -- |
| PRESS | Working Pressure | PSIA |
| X | Updated characteristics | PSI |

### 3.1.7 SUBROUTINE LISTING

```
      PROGRAM HYTR(INPUT,OUTPUT,DATA,TAPE5=DATA,TAPE6=OUTPUT,
     1 SDEN,TAPE1=SDEN,A140CD,TAPE3=A140CD)
C *** REVISED AUGUST 5, 1975 ***
      COMMON NTELPL,NTOLPL,IPT,IPOINT,NPTS,INEL,KNEL,NTOPL,NLPLT(61,3),
     1 VSTORE(6150)
      COMMON/SUB/PAR1(150,9),PM(1500),QM(1500),P(300),Q(300),C(300)
     1,Z(300),RHO(20),S2ORHO(20),VISC(20),BULK(20),TEMP(20),PVAP(20)
     2,ATPRES,T,DELT,TFINAL,PLTDEL,PI,TITLE(20),LEGN,ICON
     3,NTEMP(99),LSTART(150),NLPT(150),LTYPE(99),NC(99),INX,INZ
     4,INV,ISTEP,NLINE,NEL,IND,IENTR,MNLINE,MNEL,MNLEG,MNNODE,MNPLOT
     5,MNLPTS,MDS
C***
      READ(5,470)(TITLE(I),I=1,20)
      WRITE(6,480) TITLE
      ISTEP=0
      PI=3.1416
      T=0.0
C***
C     THIS READ STATEMENT INPUTS THE FOLLOWING DATA
C     DELT  =DELTA TIME BETWEEN CALCULATIONS   SEC
C     TFINAL=FINAL TIME   SEC
C     PLTDEL=DELTA TIME BWTWEEN PLOT POINTS   SEC
C     TEMP  =TEMPERATURE OF OIL   DEG F
      READ(5,450) DELT,TFINAL,PLTDEL,TEMP(1),DTEMP,PRESS,PRESR,ATPRES
      WRITE(6,485) TFINAL,DELT,PLTDEL
      IF(DELT.EQ.0) GO TO 251
      NPTS=1.01 + TFINAL/PLTDEL
      IPOINT=0.5+PLTDEL/DELT
      IFINAL=0.5+TFINAL/DELT
C     THIS READ STATEMENT INPUTS THE FOLLOWING DATA
C     NLINE =NUMBER OF LINES
C     NEL   =NUMBER OF COMPONENTS
C     IF  = FLUID TYPE
C     VISC  =FLUID VISCOSITY
C     RHO   =FLUID DENSITY
C     BULK  =FLUID BULK MODULUS
      READ(5,460) NLINE,NEL,IF,INV,VISC(1),RHO(1),BULK(1),PVAP(1)
C     PRESS IS USED TO ESTABLISH FLUID PROPERTIES   PSIG
C     SET SYSTEM PRESSURE TO ITS DEFAULT VALUE
      IF(PRESS.EQ.0.0) PRESS=3000.
C     SET RETURN PRESSURE TO ITS DEFAULT VALUE
      IF(PRESR.EQ.0.0) PRESR=50.0
C     SET THE VAPOUR PRESSURE TO ITS DEFAULT VALUE
      IF(PVAP(1).EQ.0.0) PVAP(1)=2.0
C     SET ATMOSPHERIC PRESSURE TO ITS DEFAULT VALUE
      IF(ATPRES.EQ.0.0) ATPRES=14.5
C     SET TEMP AND DTEMP TO THIER DEFAULT VALUES
      IF(TEMP(1).EQ.0.0) TEMP(1)=100.
      IF(DTEMP.EQ.0.0)DTEMP=10.0
      CALL FLUID(TEMP,DTEMP,PRESS,PRESR,IF,VISC,BULK,RHO,PVAP)
```

3.1.7 (Continued)

```
      N=20
      IF(IF.EQ.0) N=1
      DO 100 I=1,N
  100 S2ORHO(I)=SQRT(2./RHO(I))
C
      IENTR=-1
C
      CALL LINE
C
      INEL=0
      CALL COMP
      CALL SBRAN
C
C
      CALL SECOND(CPU)
      WRITE(6,9999)CPU
 9999 FORMAT(10X,*CPU TIME IN SECONDS =*,F10.3)
      CALL SSDATA
      CALL SECOND(CPU)
      WRITE(6,9999)CPU
C
C *** THIS SECTION CALLS LINE AND COMP TO INITIALIZE ALL THE
C     SYSTEM VARIABLES TO THEIR STEADY STATE VALUES
C
C     IF (T.EQ.0.0) STOP
      IENTR=0
      INEL=0
C
      CALL LINE
C
      CALL  COMP
C
      CALL STORE
C
C     TRANSIENT CALCULATION SECTION
C
      IENTR=1
C
  150 CONTINUE
C
      T=T+DELT
      ISTEP=ISTEP+1
C     IF(T.GT.0.0)GO TO 251
C
      CALL LINE
      CALL SBRAN
C
C     DO ELEMENT CALCULATIONS
      X=CHARUP(I,T)
C
```

3.1.7 (Continued)

```
      CALL  COMP
      N=-1
      DO 250 I=1,NLINE
      N=N+2
      M=LSTART(I)
      MM=M+NLPT(I)-1
      PM(M)=P(N)
      PM(MM)=P(N+1)
      QM(M)=-Q(N)
  250 QM(MM)=Q(N+1)
C
      CALL STORE
C
      IF (ISTEP.LT.IFINAL) GO TO 150
C
      STOP
  251 CONTINUE
      STOP 3100
  450 FORMAT (8E10.0)
  460 FORMAT(4I5,6E10.0)
  470 FORMAT(20A4)
  480 FORMAT(25X,20A4,//)
  485 FORMAT(20X,44H THE TRANSIENT RESPONSE IS FROM T=0.0  TO T=    ,
     1 F7.3,35H SECONDS AT TIME INTERVALS OF DELT=    ,
     2 F7.5,//,30X, 48HWITH OUTPUT POINTS PLOTTED AT INTERVALS OF    ,
     3 F7.5,9H  SECONDS   , //)
      END
```

## 3.2 BLOCK DATA

Block data is used to initialize values in COMMON/SUB/and COMMON/COMPD/. The maximum number of various input values in COMMON/SUB/ are established using the following data initialization statement.

```
      DATA MNLINE,MNEL,MNLEG,MNNODE,MNPLOT,MNLPTS,MDS
     + /150,99,70,55,60,1500,4500/
```

Maximum and minimum values for each individual component are initialized in COMMON/COMPD/ as follows:

```
      DATA LT/100*0/
      DATA L11/0,4,0,4,0,0,4,2,0,0/
      DATA L1220/90*0/
      DATA L21/24,2,0,5,0,3,2,2,1,0/
      DATA L22/32,8,0,7,0,4,4,2,0,0/
      DATA L23/24,2,0,5,0,3,2,2,1,0/
      DATA L2430/8,12,0,12,0,0,8,8,0,0,60*0/
      DATA L31/6,6,0,3,0,1,2,2,1,0/
      DATA L32/6,5,0,4,0,0,3,3,1,0/
      DATA L33/10,20,0,3,0,1,2,2,1,0/
      DATA L34/20,20,0,3,0,2,2,2,1,0/
      DATA L3540/60*0/
      DATA L41/4,0,0,2,0,1,2,2,1,0/
      DATA L4250/90*0/
      DATA L51/37,43,0,10,0,5,3,3,1,0/
      DATA L52/10*0/
      DATA L53/37,31,115,34,0,5,3,3,1,0/
      DATA L54/37,43,0,10,0,5,3,3,1,0/
      DATA L5560/60*0/
      DATA L61/2,0,0,12,0,1,10,1,0,0/
      DATA L62/6,10,0,7,0,2,5,2,0,0/
      DATA L63/10,13,0,9,0,2,7,2,0,0/
      DATA L6470/70*0/
      DATA L71/5,11,0,2,0,1,2,1,0,0/
      DATA L72/5,11,0,2,0,1,2,1,0,0/
      DATA L7380/80*0/
      DATA L81/8,8,0,2,0,1,2,2,1,0/
      DATA L82/24,25,0,8,0,3,6,6,0,0/
      DATA L83/8,5,0,4,0,1,4,2,0,0/
      DATA L8490/70*0/
      DATA L91/0,1001,0,3,0,0,1,1,0,0/
      DATA L9293/81,1,0,5,0,3,0,0,0,0,24,3,0,5,0,3,0,0,0,0/
```

```
DATA L9495/10*0,112,15,0,7,0,2,0,0,0,0/
DATA L9697/20*0/
DATA L98/81,1,0,13,0,2,0,0,0,0/
DATA L99100/0,50,0,12,16*0/
DATA L101/32,20,0,12,0,2,2,2,1,0/
DATA L102/12,15,63,32,0,2,2,2,1,0/
DATA L103/40,100,0,12,0,2,6,2,0,0/
DATA L104/5,13,0,5,0,1,2,2,1,0/
DATA L105/40,100,0,12,0,2,6,2,1,0/
DATA L106/0,16,0,15,0,0,6,6,1,0/
DATA L107/2,18,0,15,0,1,6,6,1,0/
DATA LEND/430*0/
END
```

## 3.3 COMMON USAGE

One blank and one labeled common statements are used in HYTR program to share storage and pass arguments between the various subroutines.

1. Blank Common is used to pass plot and steady state variables

```
COMMON NTELPL,NTOLPL,IPT,IPOINT,NPTS,INEL,KNEL,NTOPL,NLPLT(61,3),
1 VSTORE(6150)
```

2. Common SUB is used to pass arguments concerning fluid, time, constants, addresses, characteristics, pressures, flows, compo connections and program input limitations

```
COMMON/SUB/PARM(150,9),PM(1500),QM(1500),P(300),Q(300),C(300)
1,Z(300),RHO(20),S2ORHO(20),VISC(20),BULK(20),TEMP(20),PVAP(20)
2,ATPRES,T,DELT,TFINAL,PLTDEL,PI,TITLE(20),LEGN,ICON
3,KTEMP(99),LSTART(150),NLPT(150),LTYPE(99),NC(99),INX,INZ
4,INV,ISTEP,NLINE,NEL,IND,IENTR,MNLINE,MNEL,MNLEG,MNNODE,MNPLOT
5,MNLPTS,MDS
```

Subroutine COMP uses one labeled common statement.

Common COMPD is used to pass variables used in component calculations

```
COMMON/COMPD/D(4500),L(1500),LE(99,4)
```

The maximum input limits of the program are set forth in BLOCK DATA. In order to increase any of the limits, the initialized data statement in BLOCK DATA must be changed.

Note: The maximum number of lines that can be input is equal to the dimension of the C array divided by 2 minus 2 since array elements MNLINE and MNLINE-1 are used to store "blanked off" pressure and atmospheric pressure, respectively.

It should be noted that arrays initially allocate storage for specific data (e.g., PARM(N,1) is input line length for line N), however these arrays, or portions of these arrays are frequently "updated" or used to store entirely different data as the program progresses. This is quite often done when values of an array are needed only for the early phases of the program. In view of this, the following description of Common arrays (and variables) may only apply to first or primary usage.

Array Initialization For Components

Used In BLKDTA

| Array Location | |
|---|---|
| 1 | Number of real data points, D( ) |
| 2 | Number of temporary variables, DT( ) |
| 3 | Number of double precision variables, DD( ) |
| 4 | Number of integer variables, L( ) |
| 5 | Not used |
| 6 | Minimum number of data cards |
| 7 | Maximum number of connections |
| 8 | Minimum number of connections |
| 9 | Connection default 0=Blocked<br>1> Open to Atmosphere |
| 10 | Not used |

### 3.3.1 Variables Names

| Variable | Description | Unit |
|---|---|---|
| ATPRES | Atmospheric Pressure | PSIA |
| BULK( ) | Adiabatic Bulk Modulus of Fluid | PSI |
| C( ) | Characteristic Array | PSI |
| D( ) | Component Real Data Array | -- |
| DELT | Program Time Step | SEC |
| IENTR | Subroutine Entry Point Indicator | -- |
| IND | Number Assigned to Component by User | -- |
| INEL | Plot Point Address | -- |
| INV | Number of Component Active Connections | -- |
| INX | Counter | -- |
| INZ | Number of Elements in Leg | -- |
| IPOINT | Counter for Number of Points Stored | -- |
| IPT | Counter | -- |
| KNEL | Plot Number, = 0 When Plots Are Not Required | -- |
| L( ) | Component Integer Data Array | -- |
| LE(N) | Address of Real Data for Component N | -- |
| LSTART(N) | Address of the First Point of Line N | -- |
| LTYPE(N) | Component N Type | -- |
| MNEL | Maximum Number of Components | -- |
| MNLEG | Maximum Number of Legs | -- |
| MNLINE | Maximum Number of Lines | -- |
| MNLPTS | Maximum Number of Line Points | -- |
| MNNODE | Maximum Number of Nodes | -- |

| Variable | Description | Unit |
|---|---|---|
| MNPLOT | Maximum Number of Plots | -- |
| MPARME | Maximum Number of Component Real Data Values | -- |
| NC(N,I) | Line Number Attached to Connection I of Component N, and Temporary Storage Area | -- |
| NEL | Number of Components Input | -- |
| NLINE | Number of Lines Input | -- |
| NLPLT(,1) | Address of Pressure or Flow to be Plotted in DM( ) & QM( ) or Component Variable in D( ) | -- |
| NLPLT(,2) | Line Number from which Pressures and/or Flows are Plotted/or Component Number | -- |
| NLPLT(,3) | Point of Plots - Distance from Upstream Line End in DELX's/or Component Variable Number | -- |
| NLPT(N) | Number of Line Points in Line N | -- |
| NPTS | Number of Plot Points | -- |
| NTELPL | Number of Element Plots Input by User | -- |
| NTOLPL | Number of Line Plots | -- |
| NTOPL | Number of Total Plots | -- |
| P(··) | Array of Line End Pressures | -- |
| PARM(N,1) | Length of Line N | IN |
| PARM(N,2) | I.D. of Line N | IN |
| PARM(N,3) | Wall Thickness of Line N | IN |
| PARM(N,4) | Modulus of Elasticity of Line N | PSI |
| PARM(N,5) | Length of Characteristic Line Segment for Line N | IN |
| PARM(N,6) | Characteristic Impedance of Line N | PSI/CIS |
| PARM(N,7) | Velocity of Sound in Line N | IN/SEC |

| Variable | Description | Unit |
|---|---|---|
| PARM(N,8) | Laminar Flow Constant of Line N for given Diameter, Viscosity and Density | -- |
| PARM(N,9) | Turbulent Flow Constant of Line N for Given Diameter, Viscosity and Density | -- |
| PI | Constant 3.1416 | -- |
| PLTDEL | Plot Time Interval | SEC |
| PM ( ) | Array of Pressures for Points of all Lines | PSIA |
| PVAP | Vapor Pressure | PSIA |
| Q ( ) | Array of Line End Flows | CIS |
| QM( ) | Array of Flows for Points of all Lines | CIS |
| RHO ( ) | Density Array | $LB/SEC^2/IN^4$ |
| S2ORHO ( ) | SQRT(2/RHO) Array | -- |
| T | Current Main Program Calculation Time | SEC |
| TEMP ( ) | Temperature Array | °F |
| TFINAL | Final Calculation Time | SEC |
| TITLE( ) | Program Run Title Array | -- |
| VISC( ) | Fluid Viscosity Array | $IN^2/SEC$ |
| VSTORE( ) | Array for Storage of Line and Component Variable Data Required for Plotting | -- |
| Z ( ) | Characteristic Impedance Array | PSI/CIS |

3.3.2 Variables Equivalenced In Common - These variables which share the same storage as the VSTORE( ) array, are used in the steady state section of the program.

| Name | Description | Dimension |
|---|---|---|
| PQLEG( ) | Real Data Storage for LEGCAL Variables and Formulae | |
| LCS( ) | Integer Storage for Steady State and LEGCAL Counters | |
| ILEP( ) | Same as above in CALC | |
| ILEG( ) | Leg Element Numbers, Arranged in Pairs | |
| PN( ) | Calculated Node Pressures | PSI |
| P( ) | Same as above in CALC | |
| QN( ) | Overboard Flow From Nodes | CIS |
| QL( ) | Same as above in CALC | |
| PEX( ) | Equivalent External Pressure Node | CIS |
| PD( ) | Leg Constant Pressure Drop (-PQLEG(INEL,5) | CIS |
| CALC2( ) | Array used by CALC to send data to simult. returns with Node Pressures which are not always in the same sequence as the Input Data | PSI |
| IP( ) | Counter Array | |
| G( ) | Leg Conductances | CIS/PSI |
| QNEW( ) | New Predicted LEG Flow | CIS |
| MLEP( ) | Node Integer Data | |
| QL( ) | Leg Flow Guess used for Interation | |
| CALC1( ) | Only in the SSDATA Common Used in CALC and simult. with varying dimensions | |

## 3.4 FLUID SUBROUTINE

Subroutine FLUID calculates and writes the fluid density, adiabatic bulk modulus, and kinematic viscosity at the fluid steady state temperatures and pressures specified in the main program input data. Data are currently included in FLUID for three hydraulic fluids; MIL-H-5606B, MIL-H-83282, and SKYDROL 500B. FLUID is presently dimensioned to accept data on three additional fluids. Fluid data sources and calculation of properties are discussed in Appendix B, along with tabulations of data presently included in FLUID. Data sources are also contained in the FLUID subroutine itself via comment records. Volume 1 of this report describes the option whereby the user may specify and input his own fluid data to the HSFR program, if the user desires to not use FLUID.

The FLUID subroutine argument requires the input data values for the maximum and temperature steps, maximum and minimum pressures and the fluid type identification number. Also, the argument includes the variable names of the three fluid properties, since they are not in "common". Parameters are then dimensioned for nine input data points and six fluids. Data statements are then used to input the name of each fluid, the nine temperature data points for each fluid, and the bulk modulus and viscosity data corresponding to the nine temperature points for each fluid. Only two points are used for density input data since a straight line interpolation is used over the entire temperature range for density calculations.

Subroutine INTERP is then called to estimate the fluid property value at the actual fluid operating temperature. Viscosity is then converted from metric (centistokes) to English units (newts). Density, bulk modulus, and viscosity values are then corrected to the steady state operating pressure

Finally FLUID writes the computed properties before returning control to the main program.

3.4.1 Math Model - Not applicable.

3.4.3 Computation Method - FLUID calls subroutine INTERP (Ref. paragraph 8.3) which derives the fluid property at operating conditions from the tables of FLUID property data.

3.4.4 Approximations - Not applicable.

3.4.5 Limitations - Not applicable.

3.4.6 Variable Names

| SYMBOL | NAME | UNITS |
|---|---|---|
| A | Temperature ratio | - |
| ABULK(-,-) | Array for ten adiabatic bulk modulus input data points for six fluids | PSI |
| ATEMP(-,-) | Array for ten temperature data points for six fluids | °F |
| AVISC(-,-) | Array for ten viscosity input data points for six fluids | CENTISTOKES |
| B | Viscosity correction exponent | - |
| BRHO(-,-) | Array for two density input data points for six fluids | LB*SEC**2/IN |
| BTEMP(-) | First and next to last temperature input points | °F |
| BULK( ) | Bulk modulus at operating conditions | PSI |
| COEFF | Viscosity correction factor | - |
| IF | Input data fluid type identification number | - |
| IFLUNM(-,-) | Array for fluid names | - |
| IK | Number of input temperature points | - |
| IND | Error indicator | - |
| J | Integer counter | - |
| PRESS | Input data fluid operating pressure | PSIG |

3.4.6 Variable Names (Cont'd)

| SYMBOL | NAME | UNITS |
|---|---|---|
| RHO( ) | Density at operating conditions | LBS*SEC**2/IN |
| TEMP( ) | Input data fluid operating temperature | °F |
| VISC( ) | Viscosity at operating conditions | IN**2/SEC |

### 3.4.7 Subroutine Listing

```
      SUBROUTINE FLUID(TEMP,DTEMP,PRESS,PRESR,IF,VISC,BULK,RHO,PVAP)
C**** REVISED SEPTEMBER 3, 1975 ****
      DIMENSION ATEMP(10,6),AVISC(10,6),ABULK(10,6),BRHO(2,6),
     1BTEMP(2),COEFF(6),IFLUNM(3,6),IX(6)
     2 ,TEMP(20),VISC(20),RHO(20),BULK(20),PVAP(20)
C
C     SECOND SUBSCRIPT REFERS TO FLUID TYPE (IF PARAMETER)
C
      DATA IFLUNM/
     1 4H MIL,4H-H-5,4H606 ,
     2 4H MIL,4H-H-3,4H3282,
     3 4HSKYD,4HROL ,4H5003,
     4 9*4H     /
C
      DATA ATEMP /
     1-65.,-40.,0.,50.,100.,150.,200.,250.,300.,300.,
     2-65.,-40.,0.,50.,100.,150.,200.,250.,300.,300.,
     3-65.,-40.,0.,50.,100.,150.,200.,250.,300.,300.,
     430*10. /
C
      DATA BTEMP /
     1-65.,275. /
C
C       RHO,BULK AND VISC DATA ARE FOR 0.0 PSIG
C
C     RHO DATA SOURCE:
C     1-MDC REPORT A2536 DATED 4/74
C     2-MDC REPORT A2686 DATED 4/74
C     3-MONSANTO DATA SHEET DATED 6/67(DOUGLAS HYD MANUAL)
      DATA BRHO /
     13.57E-5,7.63E-5,
     23.49E-5,7.3E-5,
     310.3E-5,8.9E-5,6*10./
C
C     BULK DATA SOURCE:
C     1-LETTER TO G.AGIES FROM J.W.HOOMAN DATED 11/70
C     2-TECHNICAL REPORT AFAL-TR-73-91 DATED 4/73
C     3-LETTER TO G.AGIES FROM J.W.HOOMAN DATED 11/70
      DATA ABULK /
     13.47E5,3.25E5,2.9E5,2.48E5,2.08E5,1.73E5,1.42E5,1.19E5,.98E5,
     A.98E5,
     23.47E5,3.25E5,2.9E5,2.48E5,2.08E5,1.73E5,1.42E5,1.19E5,.98E5,
     A.98E5,
     34.26E5,4.05E5,3.64E5,3.13E5,2.7E5,2.29E5,1.94E5,1.62E5,1.38E5,
     A1.38E5,30*10. /
C
C     VISC DATA SOURCE:
C     1-MDC REPORT A2686 DATED 4/74
C     2-MDC REPORT A2686 DATED 4/74
C     3-MONSANTO DATA SHEET DATED 6/67(DOUGLAS HYD MANUAL)
```

3.4.7 (Continued)

```
      DATA AVISC /
     11993.5,482.3,134.4,34.85,14.47,7.46,4.58,3.19,2.39,2.39,
     211446.9,2019.3,269.45,48.87,15.95,7.46,4.24,2.83,2.04,2.04,
     33485.5,598.07,104.19,27.9,11.7,6.5,4.18,2.89,2.15,2.15,
     A30*10./
C
      DATA IK/3*9,3*10/
      DATA COEFF/.335,.33,.42,3*10./
C
      IF(IF.EQ.0) GO TO 100
      DO 50 J=1,10
      TEMP(J)=TEMP(1)-(J-1)*DTEMP
      TEMP(J+10)=TEMP(J)
      I=J
      CALL INTERP(TEMP(I),ATEMP(1,IF),ABULK(1,IF),20,IK(IF),
     1TBULK,IND)
      CALL INTERP (TEMP(J),BTEMP,BRHO(1,IF),10,2,
     1TRHO,IND)
      CALL INTERP (TEMP(J),ATEMP(1,IF),AVISC(1,IF),11,
     1 IK(IF),TVISC,IND)
      CALL INTERP(TEMP(J),ATEMP(1,IF),AVISC(1,IF),12,
     1 IK(IF),COEFF(IF),IND)
C
C     VISC IS CONVERTED FROM CENTISTOKES TO NEWTS
      TVISC=TVISC*1.555E-3
C     DENSITY, VISC AND BULK ARE ADJUSTED TO PRESSURE 'PRESS'
      PRES=PRESS
      GO TO 70
   60 I=I+10
      PRES=PRESR
   70 CONTINUE
      RHO(I)=TRHO*(1.+PRES/2.5E5)
      BULK(I)=TBULK+12.*PRES
      A=560./(TEMP(J)+460.)
      B=((COEFF(IF))**A)*PRES*2.3E-4
      VISC(I)=TVISC*EXP(B)
      PVAP(I)=2.0
      IF(I.LT.11) GO TO 60
   50 CONTINUE
  300 WRITE(6,601) (IFLUID(J,IF),J=1,3),PRESS,PRESR,TEMP(1),DTEMP,
     1 VISC(1),VISC(11),RHO(1),RHO(11),BULK(1),BULK(11),
     2 PVAP(11),TEMP(1)
  601  FORMAT(//16X,15HFLUID DATA FOR ,3A4,4H AT ,F7.1,9H PSIG, - ,
     1 F7.1,9H PSIG AND ,F6.1,7H DEG F ,4H IN ,F6.1,12H DEG F STEPS ,//,
     2 35X,14HVISCOSITY     -,E10.3,5X,E10.3, 3LIN**2/SEC ,// ,
     3 35X,14HDENSITY       -,E10.3,5X,E10.3,17H(LB-SEC**2)/IN**4,//,
     4 35X,14HBULK MODULUS -,E10.3,5X,E10.3,3HPSI,//,
     5 35X,14HVAPOUR PRESS.-,E10.3,4H AT ,F6.1,7H DEG F  )
      GO TO 222
  100 CONTINUE
```

3.4.7 (Contined)

```
      IF=6
      WRITE(6,602) (IFLUNM(J,6),J=1,3),TEMP(1),VISC(1),RHO(1),BULK(1)
  602 FORMAT(//25X,16H FLUID DATA FOR ,3A4,20H      AT      PSIG AND,F6.1
     A7H DEG F ,//,35X,
     1         14HVISCOSITY     -,E10.3,2X,9HIN**2/SEC,/,35X,
     2         14HDENSITY       -,E10.3,2X,17H(LB-SEC**2)/IN**4,
     3         /,35X,14HBULK MODULUS -,E10.3,2X,3HPSI )
  227 CONTINUE
      RETURN
      END
```

# 4.0 STEADY STATE SUBROUTINES

The steady state subroutines comprising SSDATA, CALC and LEGCAL provide the transient section with the distribution of pressures and flows in the system.

The steady state programs need to know how each constant flow path is connected, where the flow splits and adds, and where there is a net displacement or overboard flow.

This data is input after the component information. The input data used gives great flexibility and is very easy to modify.

The steady state program can cope with system configurations that are very complex and it is particularly valuable with closed loop systems and intertwined flow paths.

## 4.1 SUBROUTINE SSDATA

The SSDATA subroutine reads the input data which specifies one or more system configurations. Each system is input and worked on separately.

SSDATA is a simple input routine, with very little calculation. The data storage is divided into two sections; the basic leg data is contained in the LCS array, and the elements in the leg are stored in the ILEG array.

When all the data has been read in, it is written to the output so that a check can be made for errors in each data field.

The subroutine CALC is then called which calculates the steady state conditions throughout the system. After the call to the CALC subroutine, the pressures at the end of any zero flow legs in the system are initialized to the proper pressure.

4.1.1 Math Model - Not applicable.

4.1.2 Assumptions - Not applicable.

4.1.3 Computation Method - The first set of input data to be read in is the number of nodes, NNODE, the number of legs, NLEG, the number of constant pressure nodes, NCPN, the number of zero flow legs, NZFLEG, and the number of systems, NSYST.

Subroutine CALC requires a variable size array with an NNODE x NNODE dimension. The LCS array contains the leg number, the upstream and downstream node numbers, the number of elements in the leg, and the address of the leg element data in ILEG( ). Other variables in LCS are used as indicators and counters by LEGCAL.

The LEG element data is stored in ILEG( ) in data pairs. If the first value is equal to zero the second is the line number. If the first value is nonzero, it is the component number and the second is the connection number. There are LCS (I, 5) pairs of data for each LEG #I with the first

value stored at LCS (I, 4) in ILEG.

If there are any zero flow legs in the system, the pressures at the leg ends are initialized to the appropriate steady state pressures. The up and downstream pressures and flows are then written into the PM and QM arrays. The components connected to the zero flow legs are also initialized accordingly.

Should there be more than one system, the SSDATA su' routine will read in all the data for the new system and compute the steady state flows and pressures and initialize all the zero flow legs.

4.1.4 Assumptions - Not applicable.

4.1.5 Limitations - The steady state data is essentially a restatement of previously inputed transient data, in a form that can be followed during the steady state calculations. A sorting routine would eliminate the need for inputing steady state data, by generating it from the data inputed for the system components.

4.1.6 Variable Names

| Name | Description |
|---|---|
| I | Do Loop Counter |
| II | Do Loop Counter |
| JJ | Number of Elements in a LEG |
| J | Do Loop Counter |
| K | Address Counter |
| NCPN | Number of Constant Pressure Nodes |
| NLEG | Number of LEGS |
| NNODE | Number of Nodes |
| NSYST | Number of Systems |
| NZFLEG | Number of Zero Flow Legs |

### 4.1.7 Subroutine Listing

```
      SUBROUTINE SSDATA
C**** REVISED AUGUST 5, 1975 ****
      COMMON NTELPL,NTOLPL,IPT,IPOINT,NPTS,INLL,KNEL,NTOPL,NLPLT(61,3),
     1 PQLEG(90,12),LCS(90,10),ILEG(1400),PN(90),QN(90),PEX(90)
     2 ,G(90)
     3 ,QL(90),CALC1(55,55)
      COMMON/SUB/PARM(150,9),PM(1500),QM(1500),P(300),Q(300),C(300)
     1,Z(300),RHO(20),S2ORHO(20),VISC(20),BULK(20),TEMP(20),PVAP(20)
     2,ATPRES,T,DELT,TFINAL,PLTDEL,PI,TITLE(20),LEGN,ICON
     3,KTEMP(99),LSTART(150),NLPT(150),LTYPE(99),NC(99),INX,INZ
     4,INV,ISTEP,NLINE,NEL,IND,IENTR,MNLINE,MNEL,MNLEG,MNNODE,MNPLOT
     5,MNLPTS,MDS
      COMMON/COMPD/D(4500),L(1500),LE(99,4)
      DATA NZF/1/
C
C **** READ THE NUMBER OF NODES,LEGS,AND CONSTANT PRESSURE
C       NODES.
C
   50 CONTINUE
      READ(5,60)NNODE,NLEG,NCPN,NZFLEG,NSYST
  110 K=0
C
C ***  READ IN DATA FOR EACH LEG
C
      DO 200 II=1,NLEG
      READ(5,61)(LCS(II,J),J=1,3),JJ,QL(II),PQLEG(II,9),
     1 PQLEG(II,10)
      LCS(II,7)=1
      PQLEG(II,2)=0.0
      PQLEG(II,11)=0.0
      PQLEG(II,12)=0.0
      LCS(II,4)=JJ
      JJ=JJ*2
C     READ LEG ELEMENT DATA
      READ(5,60) (ILEG(K+J),J=1,JJ)
   61 FORMAT(4I5,4F10.0)
      LCS(II,5)=K+1
  200 K=K+JJ
C
C ****  WRITE OUT THE INPUTED DATA
C
      IF(NZF.EQ.0) GO TO 55
      WRITE(6,70) NNODE,NLEG,NCPN
   55 WRITE(6,71)
   60 FORMAT(16I5)
   70 FORMAT(1H1,55X,23HSTEADY STATE INPUT DATA,//,
     1  30X,17HNUMBER OF NODES =,I3,5X,16HNUMBER OF LEGS =,I3,
     2  5X,35HNUMBER OF CONSTANT PRESSURE NODES =,I3,//)
   71 FORMAT(45X,25HLEG CONNECTION INPUT DATA,
     4  //,10X,6HLEG NO,9X,12HUPST NODE NO,4X,12HDWST NODE NO,4X,
```

4.1.7 (Continued)

```
     5 14HNO OF ELEMENTS,5X,10HFLOW GUESS,5X,10HUPST PRESS,5X,
     6 10HDWST PRESS)
      WRITE(6,80)(((LCS(I,J),J=1,4),QL(I),PQLEG(I,9),
     1 PQLEG(I,10)),I=1,NLEG)
   80 FORMAT(10X,I5,10X,I5,10X,I5,10X,I5,9X,5X,F10.5,
     1  5X,F10.5,5X,F10.5)
      WRITE(6,90)
   90 FORMAT(1H0,9X,30HLEG NO      ELEMENTS IN LEG----)
      DO 300 II=1,NLEG
      K=LCS(II,5)
      JJ=LCS(II,4)*2-1+K
      WRITE(6,152) II,(ILEG(J),J=K,JJ)
  300 CONTINUE
  152 FORMAT(10X,I3,7X,10(I3,3H --,I3,1H,),//,20X,10(10(I3,3H --,I3,1H,)
     +,//,20X))
  154 FORMAT(1H1,50X,31HSTEADY STATE CALCULATION DATA    ,//,
     1  49X, 33HLEG FORMULAE GENERATED BY LEGCAL     ,//,
     2  6X,46HLEG NUMBER--FLOW GUESS-LOWER LIMIT-UPPER LIMIT  ,
     3  50H-CONST DELTP------Q TERM-Q**1.75TERM---Q**2 TERM
     4 ,//)
      IF(NZF.EQ.0) GO TO 380
        WRITE(6,154)
C
      CALL CALC(NNODE,NLEG,CALC1,NCPN)
  380 IF(NZFLEG.EQ.0) GO TO 500
      IF (NZF.EQ.0) GO TO 400
      NZF=0
      WRITE(6,390) NZFLEG
  390 FORMAT(38X,25HNUMBER OF ZERO FLOW LEGS=,I3)
      NLEG=NZFLEG
      GO TO 110
  400 CONTINUE
      DO 450 II=1,NZFLEG
      INEL=II
      PQLEG(INEL,1)=0.000001
      PQLEG(INEL,2)=1.0
      JJ=LCS(II,2)
      IF(LTYPE(JJ)/10.NE.1.AND.LTYPE(JJ).NE.82)WRITE(6,999)
      IF(LTYPE(JJ)/10.NE.1.AND.LTYPE(JJ).NE.82) STOP 4105
      JJ=LL(JJ,2)
      PRES=D(JJ)
      PQLEG(INEL,11)=PRES
      PQLEG(INEL,12)=PRES
      IF(PRES.EQ.0.0)WRITE(6,999)
  999 FORMAT(10X,33HPROGRAM STOP IN SUBROUTINE SSDATA)
      IF(PRES.EQ.0.0) STOP 4106
      JJ=LCS(INEL,4)
      K=LCS(INEL,5)
      J=K+JJ*2-1
      DO 450 I=K,J,2
```

4.1.7 (Continued)

```
      IND=ILEG(I)
      KNEL=ILEG(I+1)
      IF(IND.EQ.0) GO TO 430
      CALL COMPE
      GO TO 450
  430 IN1=LSTART(KNEL)
      IN2=IN1+NLPT(KNEL)-1
      PM(IN1)=PRES
      PM(IN2)=PRES
      QM(IN1)=0.0
      QM(IN2)=0.0
  450 CONTINUE
  500 IF(NSYST.EQ.0) RETURN
      NZF=1
      GO TO 50
      END
```

## 4.2 SUBROUTINE CALC

The CALC subroutine is responsible for the steady state calculations in the system. CALC is called from the SSDATA subroutine. The subroutine will compute the pressures at all the system nodes and flows in all the legs, using pressure drop data obtained from LEGCAL. Figure 4.2-1 is a generalized flow diagram of CALC.

On entry into CALC the first phase performed by the subroutine will be to identify any constant pressure nodes input by the user in the data deck. This procedure is necessary to initialize the appropriate calculation arrays. After the initialization, the computation phase begins. All the legs will be assigned conductance values from the LEGCAL subroutine. These conductance values, along with constant factors, will then be inserted into two matrices. The SIMULT subroutine will be called to compute the new pressure values. These pressure values at the nodes are then used to calculate the new flow rates for the legs in the system. When all the flows pass the convergence test, the flows and pressures are written to labeled common arrays and program control is passed back to SSDATA. If the number of iterations exceeds 50, the subroutine also terminates with the most recent calculated values of flow and pressure.

4.2.1 Math Model - The development of the CALC subroutine to analyze complex flow systems results from the assumption that all resistance factors in a line can temporarily be assumed linear. The net flow around any node can then be written as the sum of all the flows entering and leaving that node or $Q_{NET} = 0$.

If $R_{12}$ is a resistance factor used to describe a resistance in a leg, then $R_{12} = \Delta P_{12}/Q_{12}$.

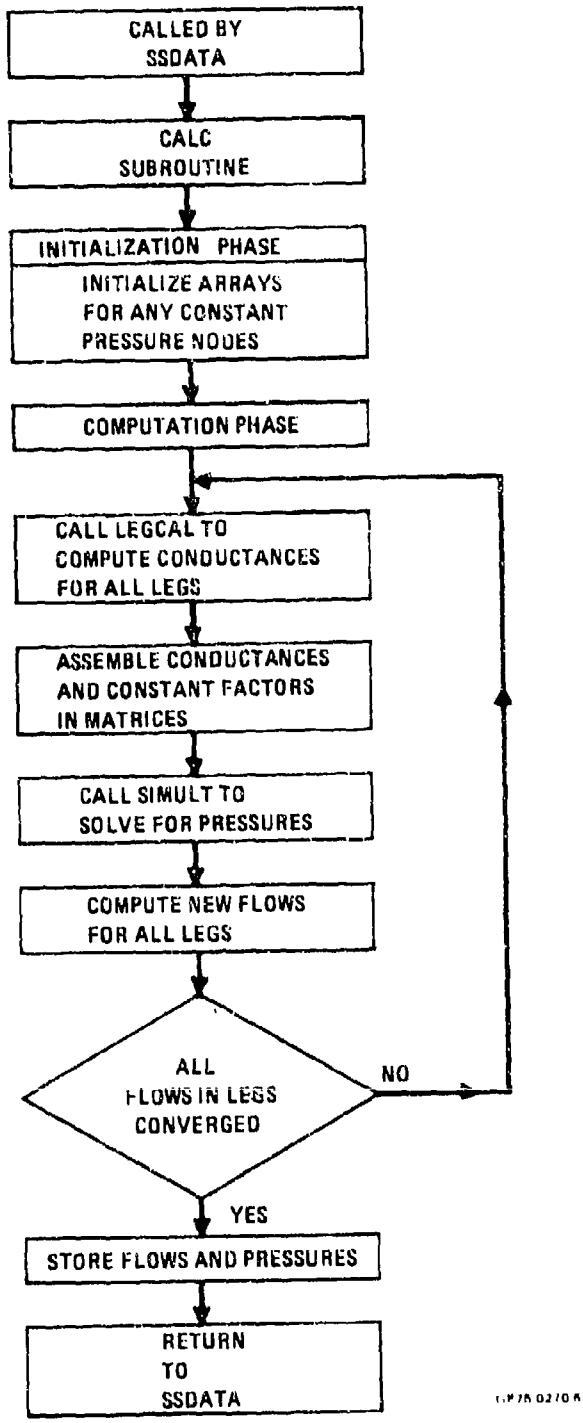



**FIGURE 4.2-1**
**CALC GENERALIZED FLOW DIAGRAM**

where:

$R_{12}$ = Resistance from node 1 to node 2 of the leg

$\Delta P_{12}$ = Pressure drop from node 1 to node 2 of the leg

$Q_{12}$ = Flow in the leg

Conductance is then defined as:

$$G_{12} = \frac{1}{R_{12}}$$

where:

$G_{12}$ = Conductance from node 1 to node 2 of the leg

Then:

$$Q_{12} = G_{12} P_{12}$$

The net flow at any node (where three or more legs come together) must be zero.

Therefore the flow requirement is satisfied if:

$$\Sigma_J G_{IJ} [P_I - P_J \pm \Delta P_{IJ}] - \Sigma_K \pm Q_{IK} = 0$$

Where:

$P_I$ = pressure at node I

$P_J$ = pressure at node J

$\Delta P_{IJ}$ = a pressure rise or loss (from a pump or actuator) in leg IJ

$Q_{IK}$ = fixed flow in leg IK connected to node I

Equations of the above form are input to a matrix for solution of pressures at nodes. These matrix solution pressures are used in conjunction with the calculated conductance (G) to calculate a new flow guess in each leg. When two successive flow guesses for all legs in the system are within a specific tolerance such as .001 CIS, the solution has converged. Refer to Appendix A SSFAN Technical Manual, MDC A3059, Vol. II, for a more detailed mathematical development.

4.2.2 CALC Subroutine Description - The CALC subroutine is divided into two phases. The first phase deals directly with the input data for constant pressure node identification. Five arrays are generated which are used in the calculation of node pressures and leg flows in phase two. Specifically, these arrays are NODEX, an array of external constant pressure nodes, EXPRESS, an array of the constant pressure node values, LEGEX, the legs connected to the constant pressure nodes, ISGN, which is set equal to +1 if the constant pressure node is upstream of a leg and -1 if it is downstream, and PEX which is set to a constant value for every external pressure node. Figure 4.2-2 describes phase one operation. If there are no constant pressure nodes, this phase is omitted, otherwise PQLEG (I, 9) and PQLEG (I, 10) are searched for any constant pressure values. If the constant pressure is in an upstream node, NODEX (J) is set equal to the node number, the pressure value is placed into EXPRESS (J), and ISGN (J) is set equal to 1. Should the constant pressure be in a downstream node the same arrays are filled except ISGN (J) is set to a -1. Also for any constant pressure node found in PQLEG, LEGEX (J) is set to the leg number and a constant value is added to PEX (NODEX (J)). With this sort completed the first phase of CALC is finished.

Phase two operation of the CALC subroutine begins with initializing the conductance matrix - CALC1, and the constant matrix - QN, to zero values. (See Figure 4.2-3 for a flow diagram of the phase two operation.) A call is now made to the subroutine LEGCAL for each leg in the system. The total number of legs is passed through the subroutine argument. LEGCAL will return the value of conductance to the G array in the unlabeled common.

After all the conductance values are calculated for each leg, they must

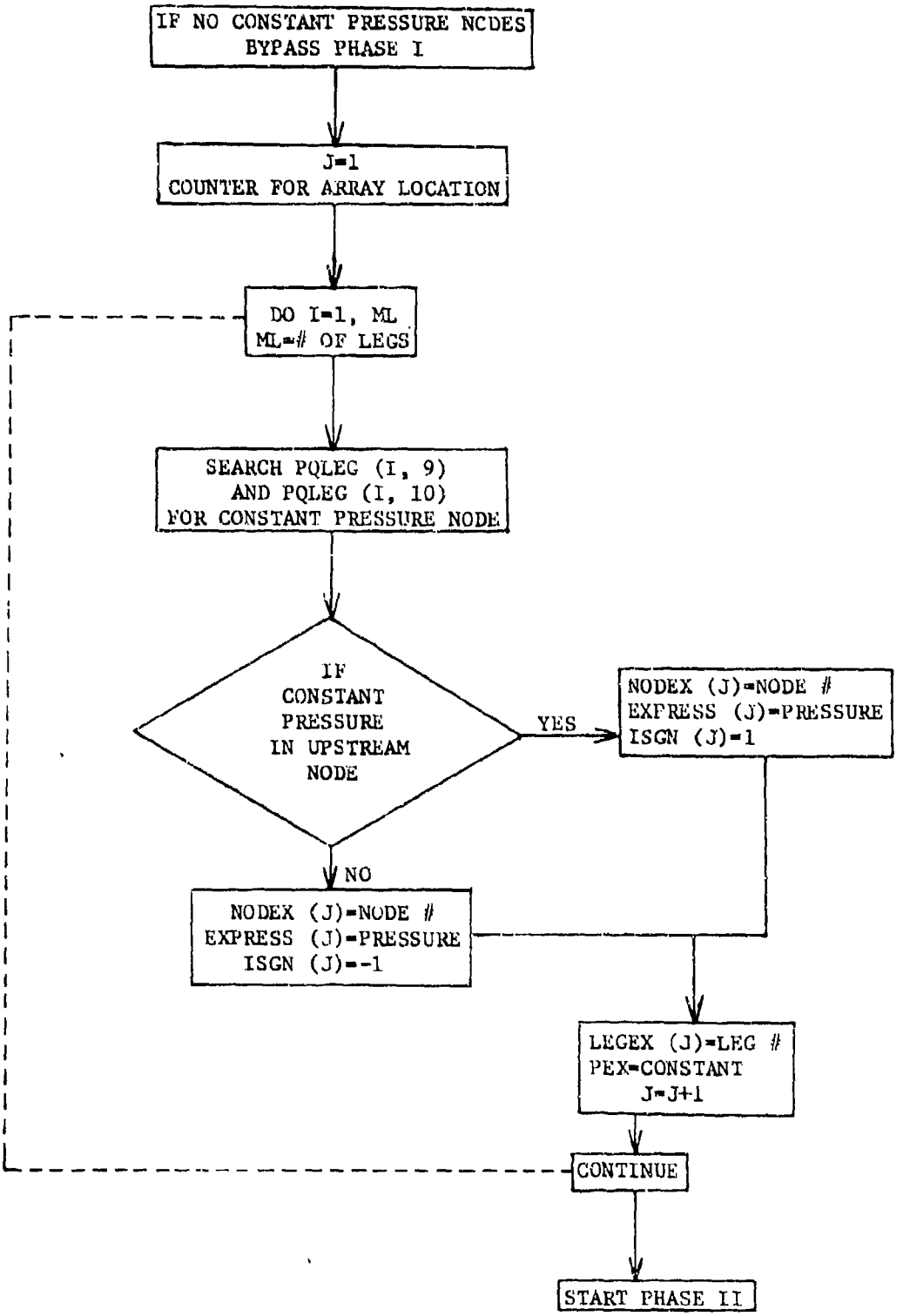



FIGURE 4.2-2
CALC SUBROUTINE PHASE ONE OPERATION

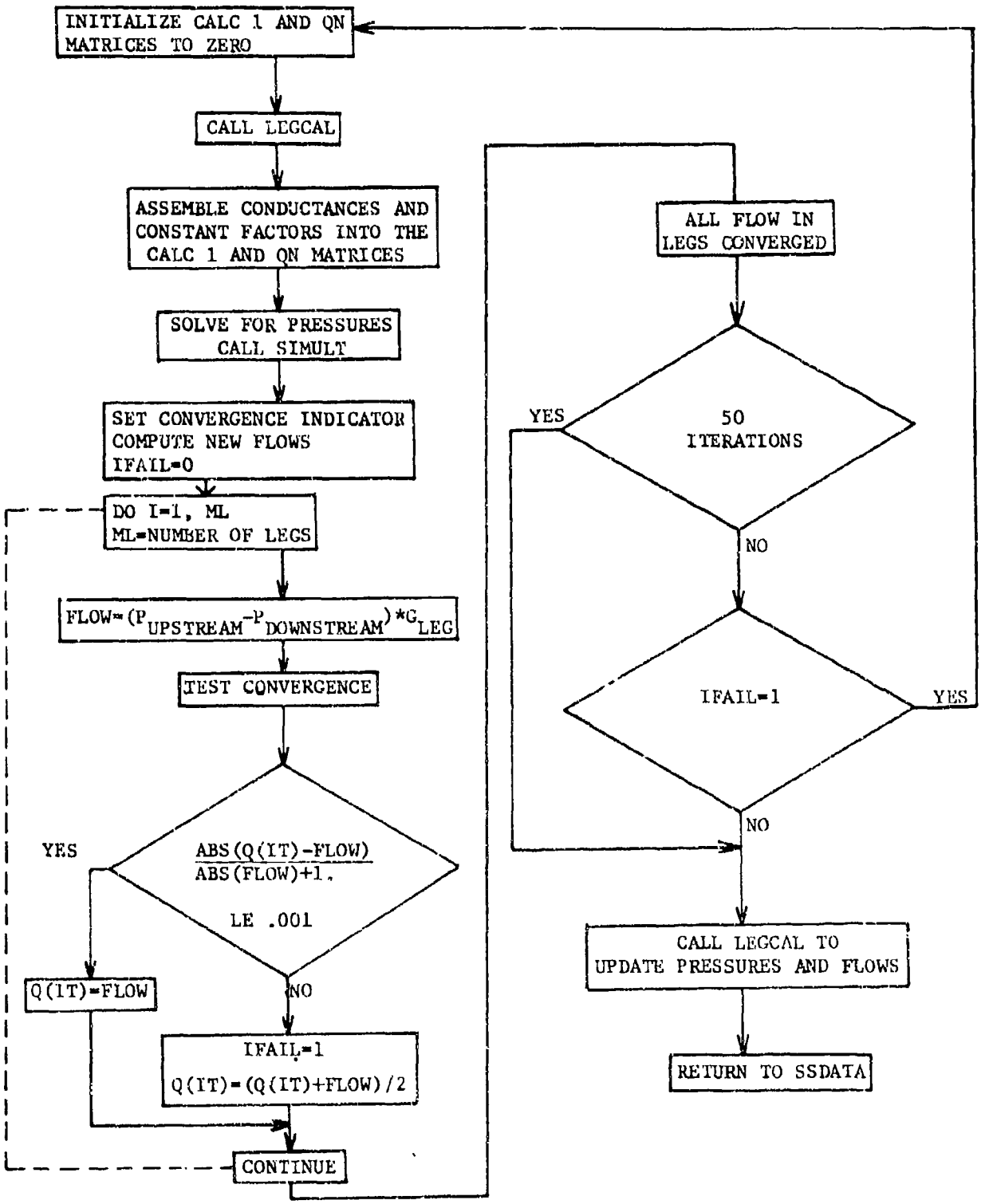



FIGURE 4.2-3
CALC SUBROUTINE PHASE TWO OPERATION

be entered into the CALC1 matrix. This is accomplished by a simple DO loop. The main diagonal element of CALC1 contains the sum of all the leg conductances surrounding a node. Each off-diagonal element equals the sum of all conductances around a node for multiple legs connected to common nodes, and for the remaining legs connected to the same node.

The QN matrix will contain the constant terms of the system of linear equations that describe the model. Constant pressure drops in legs, external flows and constant pressure sources are all inserted into this matrix. Any constant pressure source or pressure drop is multiplied by the conductance of the leg it is associated with. If leg (6) has a pressure drop term - PQLEG(6, 5), then PQLEG(6, 5) will be multiplied by the conductance for leg (6) which is G(6), making the resulting term a flow. Thus, all external flows have no multiplication factor.

With both CALC1 and QN filled, the SIMULT subroutine is called to solve for pressures in the system. The answers are returned through the QN matrix and then put into the P array which contains all the system pressures. Now a new flow is calculated for each leg in the system based on the recent calculation of the pressures. The new flow is equal to the difference of pressures between the nodes of the leg plus any constant pressure drops all multiplied by the conductance of the leg.

The solution for flows in all the legs are final when all the previous flows (Q) and the latest calculated flows (FLOW) are within a specified tolerance. For all flows if

$$\text{ABS}\ \frac{(\text{FLOW}-\text{Q(IT)})}{\text{ABS(FLOW)}+1} \leq .001 \qquad (1)$$

then the flows have converged.

If equation (1) is not satisfied in each leg of the system a new value of flow will be computed in each leg by the following equation:

$$Q(IT) = \frac{Q(IT) + FLOW}{2} \tag{3}$$

These new flows will then be given to LEGCAL for computation of new conductance values for another iteration. If all the legs do not converge after fifty iterations, the cycle will stop and all the current values will be used as the steady-state variables. Before transfer is made back to SSDATA a last call is made to LEGCAL to distribute pressure drops and flows for the steady state conditions.

4.2.3 Computations. The only direct computation made in the solution of the steady state values in CALC is the calculation of FLOW. The purpose of this of course, is to establish an error tolerance in flows that is reduced through iterations to meet the convergence criteria as discussed in the previous section. The majority of the CALC subroutine handles the bookkeeping necessary to manipulate the leg and node numbers to compute system pressures and flows.

4.2.4 Approximations. The coefficients of the CALC1 matrix are linearily approximated to represent the system conductances. Inherent approximations exist in some of the constant data in QN.

4.2.5 Limitations. Most limitations exist in the areas of physical discontinuities. CALC was written to solve a flow balance in a system. Any flow discontinuities that occur, such as in a simple unbalanced actuator, must have mathematical formula to describe what happens to the flow. CALC also requires the leg pressure drops to be continuous over a specified flow range. When this does not occur, as in a check valve, the proper input from the check valve subroutine must be feed to CALC so it may respond to the changed

conditions. Please refer to Appendix D SSFAN Technical Manual for a more thorough discussion on the limitations of CALC.

4.2.6 Variable Names

| Variables | Description | Dimensions |
|---|---|---|
| CALC1 | M*M matrix of conductances | -- |
| EXPRESS | Array of constant pressures | PSI |
| FLOW | Latest value of leg flow | CIS |
| G | Array of conductances | CIS/PSI |
| IFAIL, IFLAG | Indicators | -- |
| ILEP | Array of leg numbers with the corresponding pressure on each end | -- |
| ISGN | Array giving location of constant pressure node in leg<br>+1 - upstream<br>-1 - downstream | -- |
| ITER | Iteration counter | -- |
| LEGEX | Array of leg numbers connected to constant pressure nodes | -- |
| M | Number of nodes | -- |
| ML | Total number of legs | -- |
| NCPN | Number of constant pressure nodes | -- |
| NODEX | Array of external pressure nodes | -- |
| P | Array of node pressures | PSI |
| PQLEG (J, 5) | Location of pressure drops or increases | PSI |
| PEX | Array of external pressure constants | PSI |
| Q | Array of leg flows | CIS |
| QN | Flow gain or loss at a pressure node changed to an M matrix of constants | CIS |

### 4.2.7 Subroutine Listing

```
      SUBROUTINE CALC(M,ML,CALC1,NCPN)
C**** REVISED AUGUST 5, 1974 ****
      COMMON NTELPL,NTOLPL,IPT,IPOINT,NPTS,INEL,KNEL,NTOPL,NLPLT(61,3),
     1 PQLEG(90,12),ILEP(90,10),ILEG(1400),P(90),QN(90),PEX(90)
     2 ,G(90),Q(90)
      DIMENSION CALC1(M,M),NODEX(10),EXPRESS(10),LEGEX(10),ISGN(10)
      DO 5 I=1,M
      P(I)=0.
 5    PEX(I)=0.0
      J=1
      DO 100 I=1,ML
      IFLAG=0
      IF(PQLEG(I,9).GT.0.) GO TO 110
 50   IF(PQLEG(I,10).LE.0.) GO TO 100
      IF(IFLAG.EQ.1) GO TO 100
      IFLAG=1
      NODEX(J)=ILEP(I,3)
      EXPRESS(J)=PQLEG(I,10)
      ISGN(J)=-1
      GO TO 120
 110  NODEX(J)=ILEP(I,2)
      EXPRESS(J)=PQLEG(I,9)
      ISGN(J)=1
 120  LEGEX(J)=I
      PEX(NODEX(J))=100.0+PEX(NODEX(J))
      J=J+1
      GO TO 50
 100  CONTINUE
      ITER=1
C     INITIALIZE CALC1 AND QN ARRAYS TO ZERO
 200  DO 220 L1=1,M
      DO 210 K1=1,M
 210  CALC1(L1,K1)=0.
 220  QN(L1)=0.
C     COMPUTE G*S FOR CALC ARRAYS
      CALL LEGCAL(ML)
      DO 310 K=1,ML
      I=ILEP(K,2)
      J=ILEP(K,3)
      CALC1(I,I)=CALC1(I,I)+G(K)
      CALC1(J,J)=CALC1(J,J)+G(K)
      CALC1(I,J)=CALC1(I,J)-G(K)
      CALC1(J,I)=CALC1(I,J)
 310  CONTINUE
C     BUILD QN ARRAY
      DO 400 JX=1,ML
      IF(PQLEG(JX,5).EQ.0.)GO TO 400
      JY=ILEP(JX,2)
      QN(JY)=QN(JY)-PQLEG(JX,5)*G(JX)
      JY=ILEP(JX,3)
```

4.2.7 (Continued)

```
      QN(JY)=QN(JY)+PQLEG(JX,5)*G(JX)
  400 CONTINUE
      DO 30 I=1,M
      CALC1(I,I)=CALC1(I,I)+PEX(I)
   30 CONTINUE
C     WRITE(6,2005)((CALC1(I,J),J=1,M),I=1,M)
C     WRITE(6,2005)(QN(I),I=1,M)
 2005 FORMAT(1X,10E12.5)
C     WRITE(6,2005)(PEX(I),I=1,M)
C     WRITE(6,2005)(PQLEG(I,5),I=1,ML)
 2010 FORMAT(1X,10I10)
      IF(NCPN.EQ.0) GO TO 610
      DO 600 J=1,NCPN
      NX=NODEX(J)
      LX=LEGEX(J)
  600 QN(NX)=QN(NX)+ISGN(J)*Q(LX)+EXPRESS(J)*100.0
  610 CONTINUE
      CALL CKSOLV(CALC1,QN,M,KS)
      DO 500 I=1,M
  500 P(I)=QN(I)
C     CALCULATE NEW FLOW RATES
      WRITE(6,9000)(QN(I),I=1,M)
      WRITE(6,9001)
 9000 FORMAT(1H0,(5X,14HNODE PRESSURES,2X,8F12.3,/))
 9001 FORMAT(1H0)
      IFAIL=0
      DO 435 IT=1,ML
      IU=ILEP(IT,2)
      IV=ILEP(IT,3)
      FLOW=((QN(IU)+PQLEG(IT,5)-QN(IV))*G(IT))
C     TEST NEW FLOW RATES
      IF(ABS(FLOW-Q(IT))/(ABS(FLOW)+1.).GT.0.001)GO TO 434
C     RECALCULATE FLOW RATES
      Q(IT)=FLOW
      GO TO 435
  434 IFAIL=1
      Q(IT)=(Q(IT)+FLOW)/2.
  435 CONTINUE
      IF(ITER.EQ.50)GO TO 520
      IF(IFAIL.EQ.0)GO TO 520
      ITER=ITER+1
      GO TO 200
  520 CONTINUE
      IF(ITER.EQ.50)WRITE(6,9999)
 9999 FORMAT(10X,38HEXCEEDED 50 ITERATIONS IN STEADY STATE,
     + 19H-PROGRAM CONTINUING)
C     MAKE A LAST CALL TO ALL LEGS TO DISTRIBUTE PRESSURE
C     DROPS AND FLOWS CALCULATED FOR STEADY STATE CONDITIONS
      KNEL=-1
      DO 5000 I=1,ML
```

4.2.7 (Continued)

```
5000 ILEP(I,7)=5
     CALL LEGCAL(mL)
     RETURN
     END
```

## 4.3 SUBROUTINE LEGCAL

LEGCAL is called by the CALC to obtain a leg conductance and fixed pressure drops for a given flow guess for a particular leg.

The call variable ML is the total number of legs in the system. The constant pressure drop such as that across a check valve, relief valve, actuator, or pump, is passed via PQLEG(NLEG, 5) in common. A positive PQLEG(NLEG, 5) is a pressure rise such as at a pump, a negative is a drop such as across a check valve. The leg conductance is passed via G(NLEG) in common.

LEGCAL has two ways of obtaining these values, the first is by calling all the elements in the leg. The second is by using a formulae generated for the leg previously.

Using the formulae is the most efficient way, but some components which have external references such as actuators and reservoirs require special treatment. An indicator system is set up to determine which method should be used.

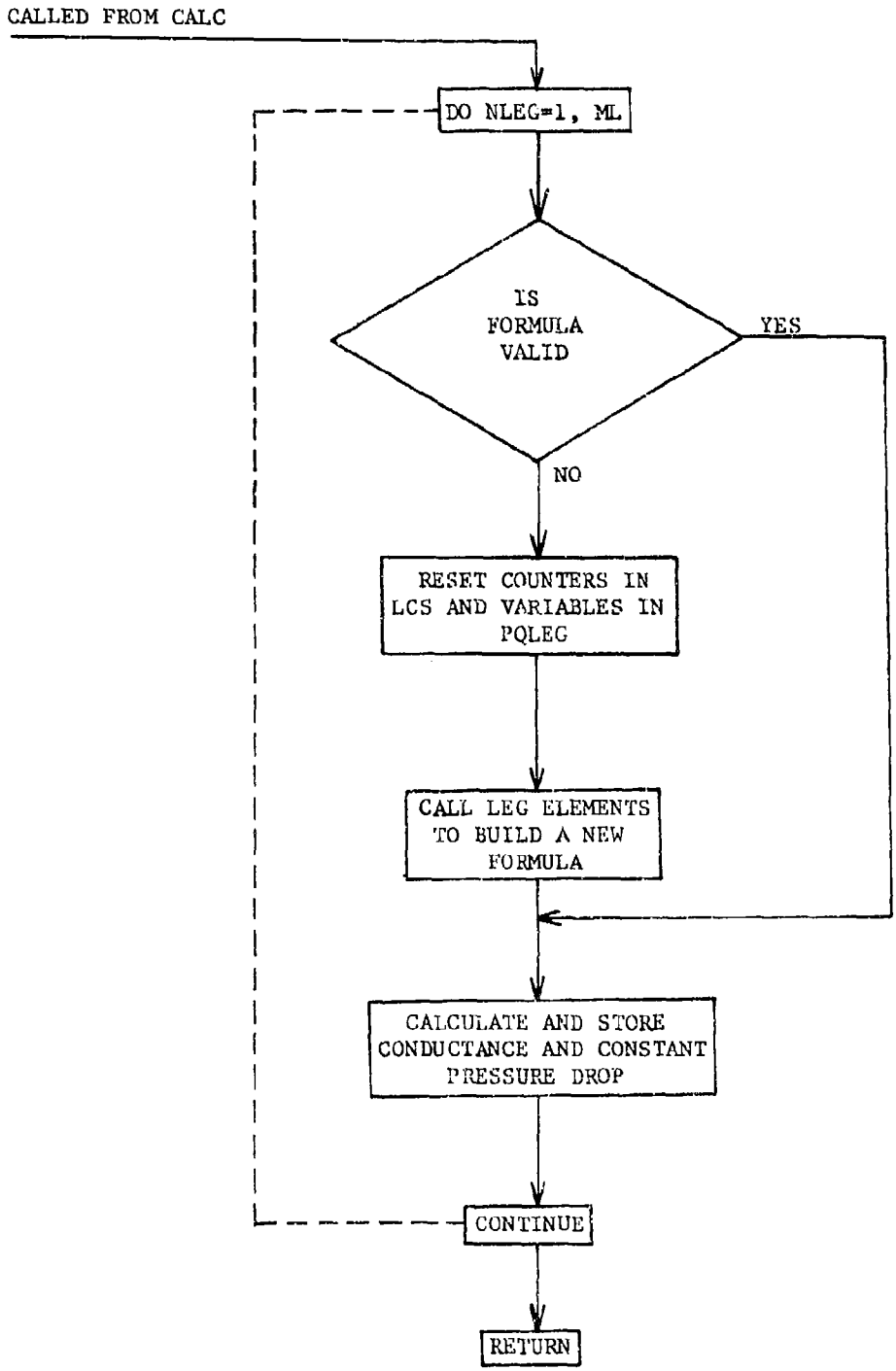



FIGURE 4.3-1
LEGCAL ORGANIZATION

### 4.3.1 Theory

LEGCAL calls the elements in a leg to build up a formulae for pressure drop for a given flow. The formulae takes the form

```
DELTP = -PQLEG(NLEG,5) + (PQLEG(NLEG,6)*Q
        + PQLEG(NLEG,7)*Q**1.75
        + PQLEG(NLEG,8)*Q**2      )  *QS
```

Q is the absolute value of the flow and QS the sign

PQLEG(NLEG,5) is the constant pressure drop (+ ve for gain, - ve for loss)

PQLEG(NLEG,6) is the constant for flow dependent loss

PQLEG(NLEG,7) is the constant for turbulent losses

PQLEG(NLEG,8) is the constant for square law losses such as orifices.

The formulae is valid for

$$PQLEG(NLEG,4) \geq Q \geq PQLEG(NLEG,3)$$

If LCS(NLEG,6) = QS or ZERO and LCS(NLEG,7) is not equal to 5

The leg conductance (inverse of resistance) is calculated from the leg pressure drop formulae, excluding the constant pressure drop value

```
G(NLEG) = QS*Q/P
```

where

```
P = QS*(PQLEG(NLEG,6)*Q
       +PQLEG(NLEG,7)*Q**1.75
       +PQLEG(NLEG,8)*Q**2) .
```

The Q and QS have been left in for clarity, the conductance is always positive.

Using this formulae, the conductance can be flow dependent, hence it has to be updated whenever the flow guess is changed.

### 4.3.2 Assumptions

The assumption that the pressure drop can be described using the formulae is generally valid. If for some reason an element in a leg cannot be described in this manner then a pseudo description can be used without loss of accuracy.

This would involve taking the actual pressure drop and dividing by Q, the result, a pseudo resistance, would then be added in to PQLEG (NLEG,6), as though it were a linear function, with the counters set to prevent re-use of the formulae for a different flow guess.

### 4.3.3 Computation Method

The variable Q1, the new flow guess is first split into its absolute value and its sign, $\pm 1.0$. Tests are then made to see if a flow formulae exists or can be used.

The formulae does not exist if PQLEG(NLEG,2) = 0.0

and cannot be used if LCS(NLEG,7) = 5.

Additional tests are if the new flow is within the formulae flow range such that PQLEG(NLEG,3) $\leq$ Q $\leq$ PQLEG(NLEG,4), and if the flow sign is acceptable such that LCS(NLEG,6)*QS $\geq$ 0.0.

If all these test are passed, then the formulae is OK and can be used to calculate a new conductance.

If not, then the formulae must be regenerated.

The first thing that has to be done is to reset the counters and zero the arrays. The new formulae is then built up by calling each element in turn starting at the element connected to the upstream node. When the element is a line, it is taken care of directly without a call statement, all other components are called via their steady state entry.

The variables IND, and KNEL are the component number and the connection number respectively.

The common variables INZ and INX are set equal to the number of elements in the leg and the actual element that is being calculated respectively. This allows particular component subroutines to determine which end of the leg they are connected to, and hence which node is located at the component.

4.3.4 Approximations

The use of a formulae requires some approximations but these are usually related to approximations in the component model and are an integral part of the componen model. In general we think this method is quite good but it could be easily extended to a higher order approximation if it was found desirable.

4.3.5 Limitations

So far we have not found any limitations to the technique used in LEGCAL itself.

However, some of the component subroutines called by LEGCAL such as the bootstrap reservoir pump and actuators, are complicated by the interaction between the flow guesses, flow direction and node pressures.

Some of these subroutines use calculations which, though conforming to the basic calculation technique, do not fall into any simple category and have to be treated individually.

### 4.3.6 Variable Names

| Variable | Description | Dimensions |
|---|---|---|
| DELTP | Line Pressure Drop | PSI |
| I | Ith Element in a Leg | -- |
| KNEL | Component Connection Number | -- |
| IND | Component or Line Number | -- |
| K | Do Loop Counter | -- |
| LCS4 | Number of Elements in Leg | -- |
| LCS5 | Address of Leg Data in ILEG | -- |
| ML | Total Number of Legs | -- |
| NTYPE | Integer of Comp Type #/10 | -- |
| P | Flow Dependent Leg Pressure Drop | PSI |
| Q | ABS Value of Leg Flow | CIS |
| QT | Leg Transition Flow | CIS |
| Q1 | Leg Flow Guess | -- |
| QS | Flow Sign CIS | -- |

For variables in common refer to Paragraph 3.3.

## 4.3.7 Subroutine Listing

```
      SUBROUTINE LEGCAL (ML)
C**** REVISED AUGUST 5, 1975 ****
      COMMON NTELPL,NTOLPL,IPT,IPOINT,NPTS,INEL,KNEL,NTOPL,NLPLT(61,3),
     1 PQLEG(90,12),LCS(90,10),ILEG(1400),PN(90),QN(90),PEX(90)
     2 ,G(90),QL(90)
      COMMON/SUB/PARM(150,9),PM(1500),QM(1500),P(300),Q(300),C(300)
     1,Z(300),RHO(20),S2ORHO(20),VISC(20),BULK(20),TEMP(20),PVAP(20)
     2,ATPRES,T,DELT,TFINAL,PLTDEL,PI,TITLE(20),LEGN,ICON
     3,KTEMP(99),LSTART(150),NLPT(150),LTYPE(99),NC(99),INX,INZ
     4,INV,ISTEP,NLINE,NEL,IND,IENTR,MNLINE,MNEL,MNLEG,MNNODE,MNPLOT
     5,MNLPTS,NDS
C     FIND THE SIGN OF THE FLOW GUESS AND ITS ABSOLUTE VALUE
      DO 200 NLEG=1,ML
      Q1=QL(NLEG)
      QA=ABS(Q1)
      IF(QA.LE..00001)QA=.00001
      QS=SIGN(1.0,Q1)
C
C      CHECK TO SEE IF THE FORMULAE IS VALID
C
      IF(PQLEG(NLEG,2).EQ.0.0.OR.LCS(NLEG,7).EQ.5)
     1 GO TO 400
      IF(PQLEG(NLEG,3).GT.QA.OR.PQLEG(NLEG,4).LT.QA)GO TO 400
      IF(LCS(NLEG,6)) 5,1000,7
   5  IF(QS.LT.0.0) GO TO 1000
      GO TO 400
   7  IF(QS.GE.0.0)GO TO 1000
C
C     RESET VARIABLES AND COUNTERS
C
 400  CONTINUE
      PQLEG(NLEG,1)=QA
      PQLEG(NLEG,2)=QS
      PQLEG(NLEG,3)=0.0
      PQLEG(NLEG,4)=1.0E6
      DO 401 N=6,9
      PQLEG(NLEG,N-1)=0.0
 401  LCS(NLEG,N)=0
C
C     CALCULATE THE FORMULAE FOR THE LEG PRESSURE DROP
C
      LCS(NLEG,7)=1
      INEL=NLEG
      LCS4=LCS(NLEG,4)
      LCS5=LCS(NLEG,5)
      PQLEG(NLEG,11)=PN(LCS(NLEG,2))
      PQLEG(NLEG,12)=PN(LCS(NLEG,3))
      INZ=LCS4
C     CALL EACH ELEMENT IN THE LEG
C
```

4.3.7 (Continued)

```
      DO 600 K=1,LCS4
      I=LCS5+(K-1)*2
      INX=K
      IND=ILEG(I)
      KNEL=ILEG(I+1)
C     IF THE ELEMENT IS A LINE GO TO 500
      IF(IND.EQ.0) GO TO 500
      CALL COMPE
      GO TO 600
C *** THIS SECTION ADDS THE VALUES INTO THE FORMULAE FOR THE LINES
C
  500 CONTINUE
      QT=PARA(KNEL,4)
      IF(QA.GT.QT)GO TO 505
      IF(QT.LT.PQLEG(NLEG,4)) PQLEG(NLEG,4)=QT
      PQLEG(NLEG,6)=PARA(KNEL,8)+PQLEG(NLEG,6)
      DELTP=QA*QS*PARA(KNEL,8)
      GO TO 598
  505 IF(QT.GT.PQLEG(NLEG,3)) PQLEG(NLEG,3)=QT
      PQLEG(NLEG,7)=PARA(KNEL,9)+PQLEG(NLEG,7)
      DELTP=PARA(KNEL,9)*QS*QA**1.75
  598 CONTINUE
      PA(LSTART(KNEL))=PQLEG(NLEG,11)
      PQLEG(NLEG,11)=PQLEG(NLEG,11)-DELTP
      PA(LSTART(KNEL)+NLPT(KNEL)-1)=PQLEG(NLEG,11)
      QA(LSTART(KNEL))=QA*QS
      QA(LSTART(KNEL)+NLPT(KNEL)-1)=QA*QS
  600 CONTINUE
C
C     THE FORMULAE FOR DELTA P CAN BE USED
C
 1000 CONTINUE
      C(NLEG)=1.0/(PQLEG(NLEG,6)+PQLEG(NLEG,7)*QA**.75+PQLEG(NLEG,8)*QA
      PQLEG(NLEG,1)=QA
      PQLEG(NLEG,2)=QS
      WRITE(6,50) NLEG,QA,(PQLEG(NLEG,I),I=3,8),C(NLEG)
   50 FORMAT(1H0      LEG NO ,I3,7F12.3,E20.8)
  200 CONTINUE
      RETURN
      END
```

## 5.0 LINE AND ASSOCIATED SUBPROGRAMS

Subroutine LINE and associated subprogram calculates the flow and pressure conditions for all interior points within the line.

The values of the flows and pressures at the ends of the line are calculated in the applicable component subroutine.

Line is called by the main program at the beginning of each new time step and calculates new values of pressures and flows. Line uses FRIC and DFRICD functions to calculate the steady state friction and dynamic friction pressure drops.

The theory of the method of characteristics has been extensively documented, but is often described in terms that are not readily understandable.

For convenience, a summary of the method is given in paragraph 5.1.1. A detailed derivation is given in Appendix A.

## 5.1 SUBROUTINE LINE

LINE divides each line into segments, the length of each segment being greater than or equal to the velocity of sound in fluid multiplied by the time step, DELT. LINE then calculates the flows and pressures for all the interior line points as well as the end point characteristics. These characteristics are subsequently used by component subroutines to calculate pressures and flows at the ends of each line where the lines and components interface.

### 5.1.1 Math Model

The equation of motion and the continuity equation, describing the one-dimensional unsteady flow of a compressible fluid in an elastic fluid line, equations (1) and (2), involve the actual distributed parameters and include the nonlinear viscous terms. These partial differential equations are transformed into total differential equations by use of the method of characteristics. A finite difference method is then used to place the total differential equations in a form suitable for numerical solution on a digital computer.

Basic Equations

The equation of motion is:

$$\frac{\partial P}{\partial X} + \frac{RHO}{A} * \frac{\partial Q}{\partial t} + F(Q) = 0 \qquad (1)$$

The condition of continuity yields:

$$\frac{\partial Q}{\partial X} + \frac{A}{RHO * a^{**2}} * \frac{\partial P}{\partial t} = 0 \qquad (2)$$

| | | |
|---|---|---|
| where | P = Pressure | psia |
| | Q = Flow | cis |
| | A = Flow area of line | $in^2$ |
| | t = Time | sec |

RHO = Fluid mass density (lb-sec**2)/in.**4

F(Q) = Pressure loss as a function of flow psi

a = speed of the wave propagation = (BULK/RHO)*1/2 in./sec

BULK = Bulk modulus of the fluid psi

X = Distance along line in.

Method of Characteristics

The basic equations (1) and (2) can be transformed into a pair of total differential equations, the validity of which is restricted to certain lines in the x-t plane called the characteristics. Integration along the characteristics, Fig. 5.1-1, yields the following algebraic equations.

For convenience let

$$Z = \frac{RHO*a}{A}$$

(the number, $Z_c$, is the characteristic impedance of a frictionless line) and let

$$\text{C left:}\quad C_L = P_{VW} - Z*Q_{VW} + \frac{a\ \Delta\ t}{2}\ [F(Q_{WT}) + F(Q_{VW})] \quad (3)$$

$$\text{C right:}\quad C_R = P_{WX} - Z*Q_{WX} + \frac{a\ \Delta\ t}{2}\ [F(Q_{WT}) + F(Q_{WX})] \quad (4)$$

then

$$P + ZQ_P - C_R = 0 \quad (5)$$

$$CR = P_{WT} - Z * Q_{WT}$$

$$P_P - Z_C Q_P - C_L = 0 \quad (6)$$

$$CL = P_{WT} + Z * Q_{WT}$$

$$P_{WT} = + \frac{C_R + C_L}{2} \quad (7)$$

$$Q_{WT} = \frac{C_L - C_R}{2*Z} \quad (8)$$

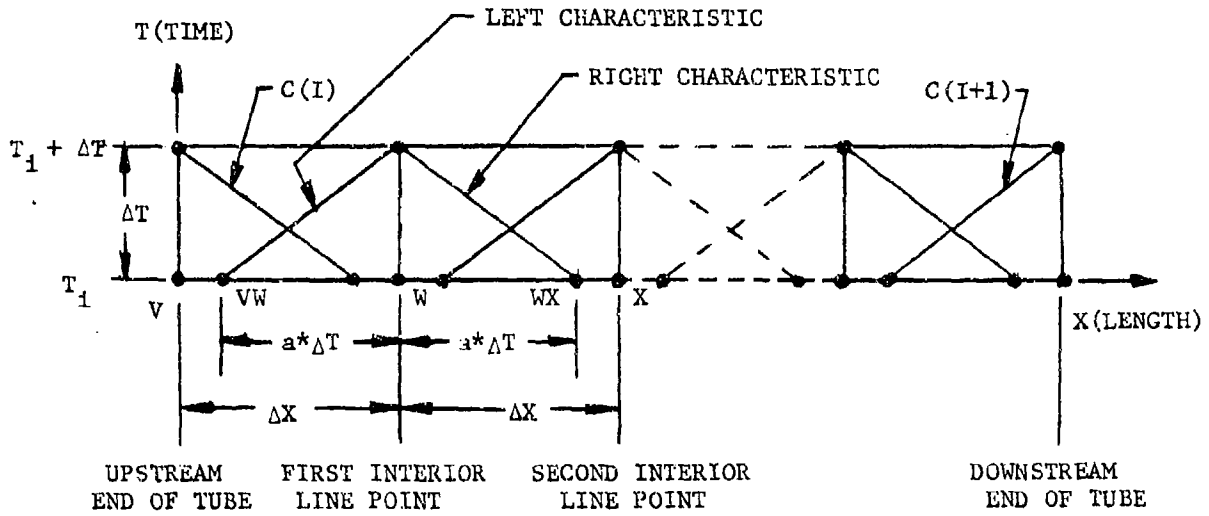



FIGURE 5.1-1 GRID OF CHARACTERISTICS WITH INTERPOLATION

Equations (7) and (8) apply only to the interior sections of the fluid line. Points at the boundary use only one of the characteristic equations to be solved simultaneously with the boundary condition. $C_R$ and $C_L$ are still functions of the unknown, $Q_{WT}$. Therefore, an iteration is used to calculate $Q_{WT}$. $C_R$ and $C_L$ are first determined using only a rectangular rule to approximate the friction.

$$C_L = P_{VW} + Z*Q_{VW} - a*\Delta t*F(Q_{VW}) \qquad (3a)$$

$$C_R = P_{WX} - Z*Q_{WX} + a*\Delta t*F(Q_{WX}) \qquad (4a)$$

This gives a very close approximation of $Q_P$, which then can be improved by using the trapezoidal rule to evaluate $C_R$ and $C_L$.

There are two ways to apply these equations to a line system: either one has to modify the wave speeds slightly to make $n = L/(a*\Delta t)$ an integer in each individual line (L is the length of a line), or one has to use the interpolating grid of Fig. 5.1-1 which has been applied in HYTRAN and is discussed in more detail by Streeter and Wylie (5).

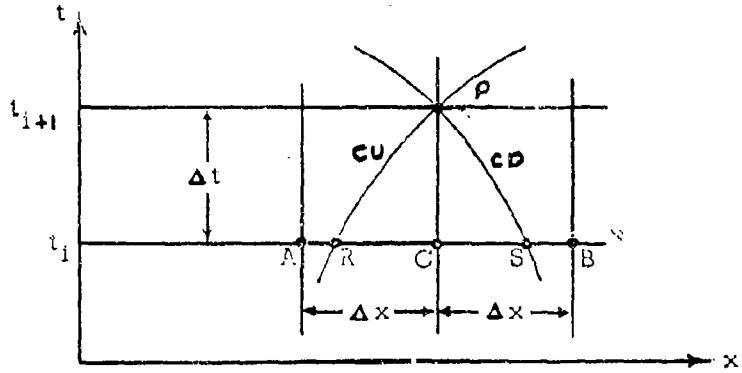



FIGURE 5.1-2 CHARACTERISTICS IN X,t PLANE

5.1.2 Assumptions

1. Transition from laminar to turbulent flow is assumed to occur at a Reynolds number of 1200.

2. Friction factors used are based on circular cross-section, smooth I.D., drawn tubing.

3. See Appendix A for assumptions associated with method of characteristics.

5.1.3 Computation Methods

Section 1000

The type of line N is isolated and the speed of sound and characteristic impedance are calculated

```
IF(LDN(N,1).EQ.1) GO to 70
A = 1.8*(PARM(N,3)+PARM(N,2)/2.0)*BULK(IT)
A = BULK(IT)/RHO(IT)*(1.0+A/(PARM(N,4)*PARM(N,3))))
GO to 75
70 BULKH = BULK(IT)*PARM(N,4)/(BULK(IT)+PARM(N,4))
A = BULKH/RHO(IT)
75 A=SQRT(A)
PARM(N,6) = RHO(IT)*A*4.0/(PI*PARM(N,2)**2)
```

If the speed of sound is not large enough to produce two line points, a fix-up is taken and the percent error in the adjusted speed of sound is printed

```
CONST = PARM(N,1)/DELT
IF(CONST.GE.A) GO to 78
PC = 100.-CONST*100./A
WRITE(6,300)N,PC
A = CONST
78 PARM(N,7) = A
NLPT(N) = CONST/A+1.01
```

DELX is then calculated

```
PARM(N,5) = PARM(N,1)/(NLPT(N)-1)
```

Equivalent line length of bends and fittings is calculated for line N

```
EQUIVL = PARM(N,2)*(LDN(N,3)*12+
         LDN(N,4)*57.0+LDN(N,5)/45.*
         4.65+LDN(N,6)/90.*7.5)
```

The equivalent line length is then added to the actual line length and is used to calculate the laminar and turbulent flow constants

```
PARM(N,8) = 128./PI*VISC(IT)*RHO(IT)/(DIA**4.)*
            (PARM(N,1)+EQUIVL)
PARM(N,9) = .213*RHO(IT)*(VISC**.25)/(DIA**4.75)
            *(PARM(N,1)+EQUIVL)
```

The above steps are repeated until the data for all lines are calculated. This data is then printed. A constant for subsequent interpolation calculations, transition flow and address of line points are calculated and stored for each line. Finally, variables used for plugged and open connections are initialized.

Dynamic friction entry DYFRI is called to calculate and store line constants for the dynamic pressure loss equation. The returned value of DYFRI is not used.

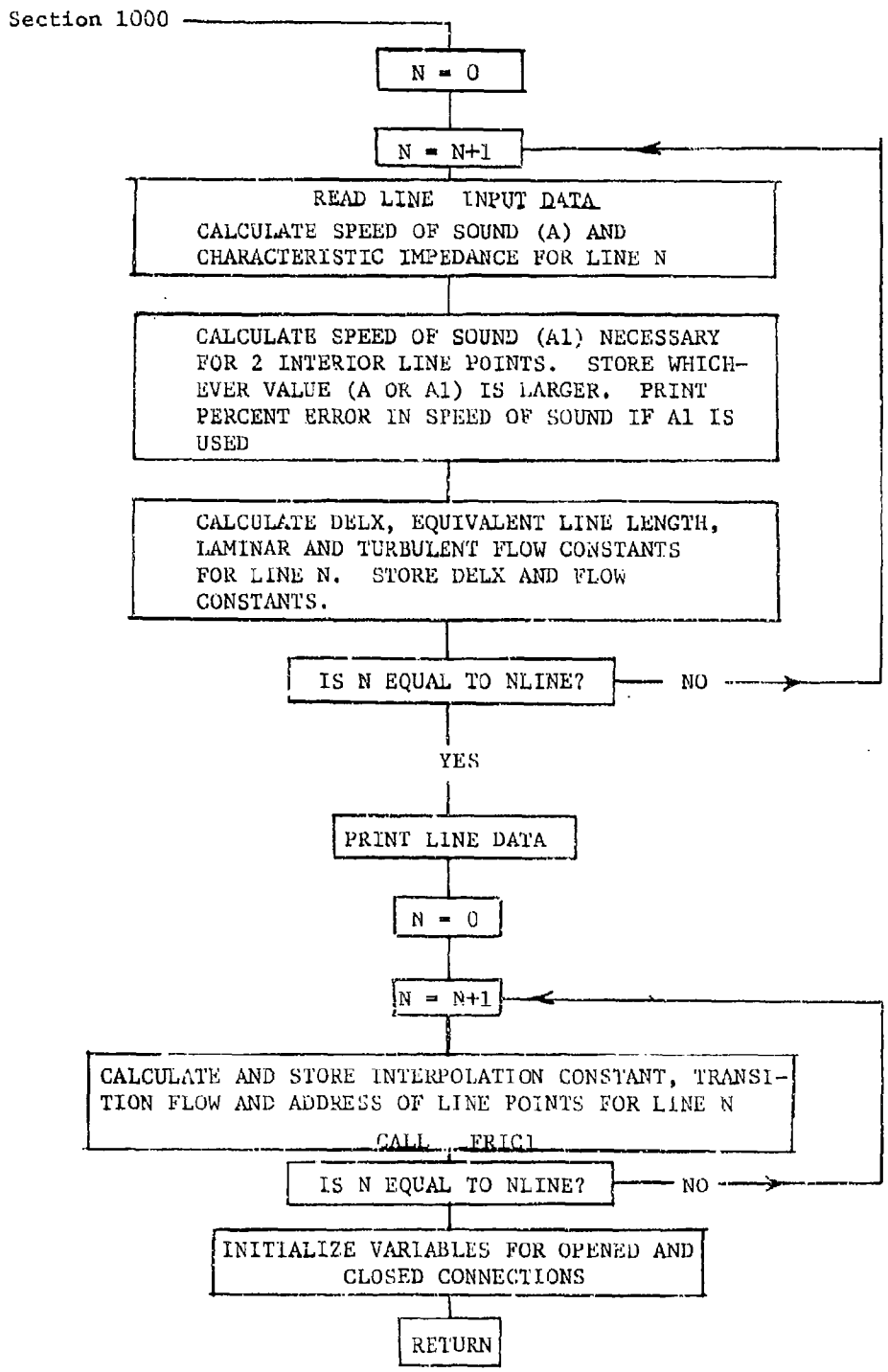



FIGURE 5.1-3 LINE SUBROUTINE FLOW DIAGRAM

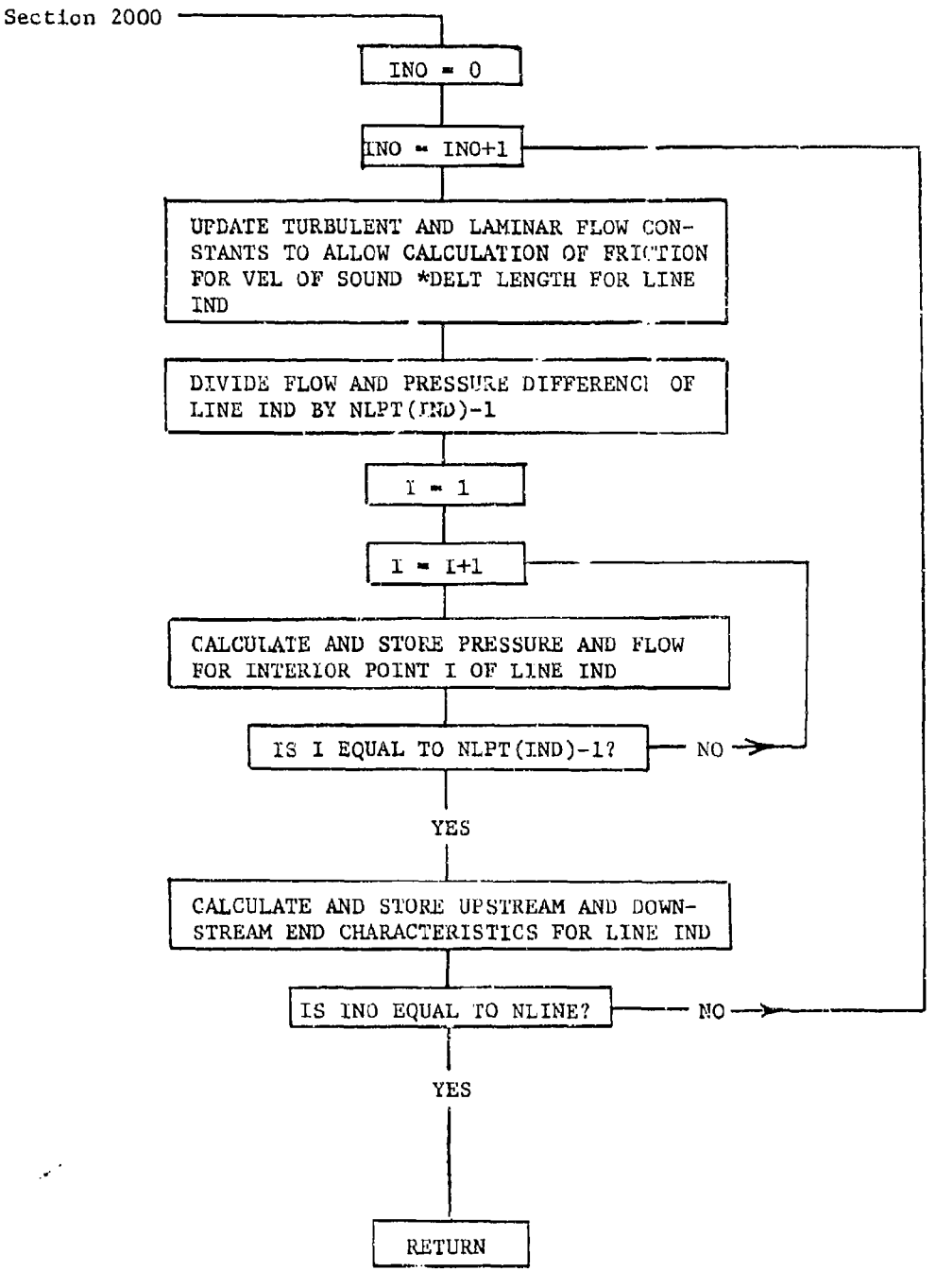



FIGURE 5.1-3 CONT.

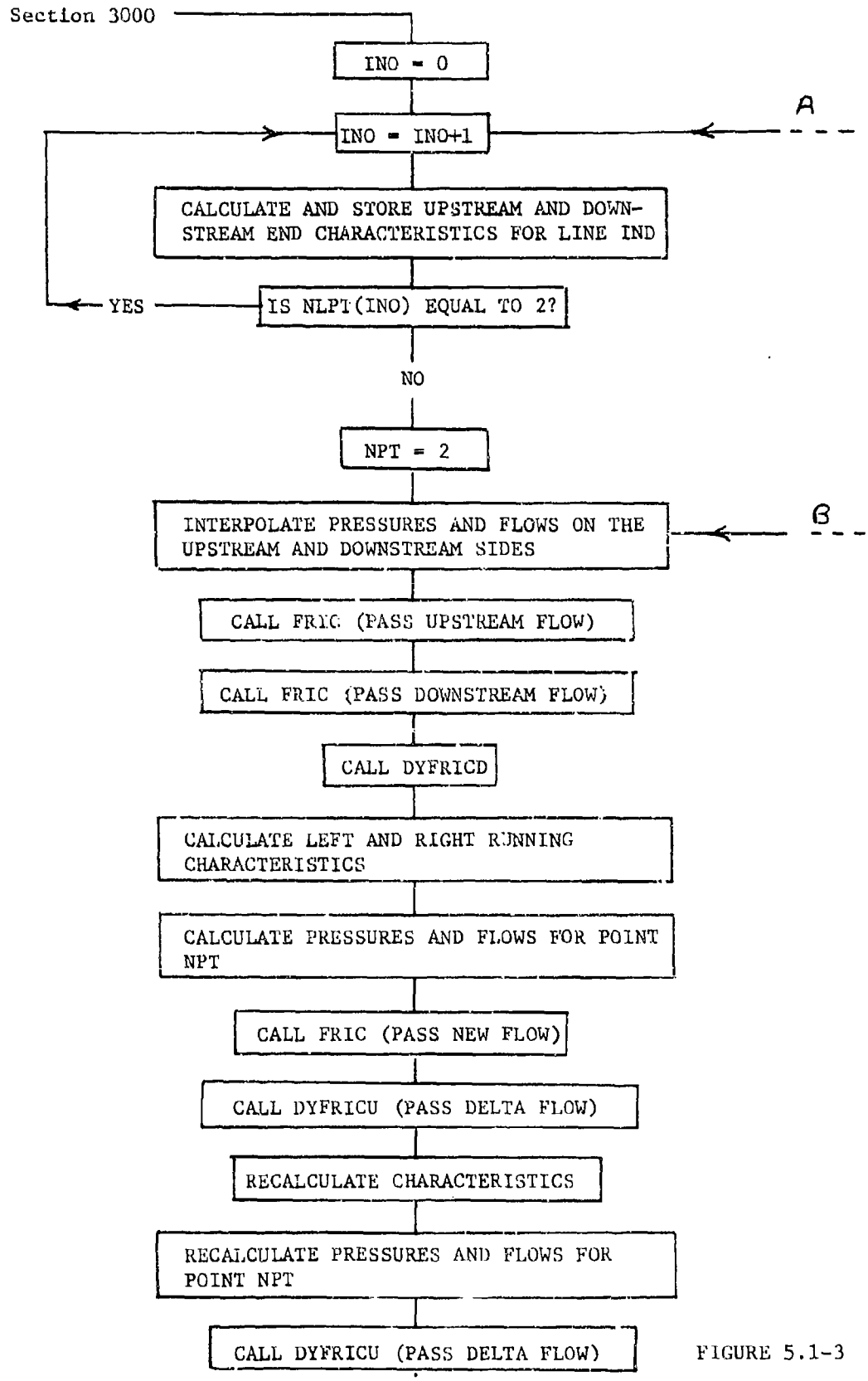



FIGURE 5.1-3 CONT.

Section 3000 (Cont)

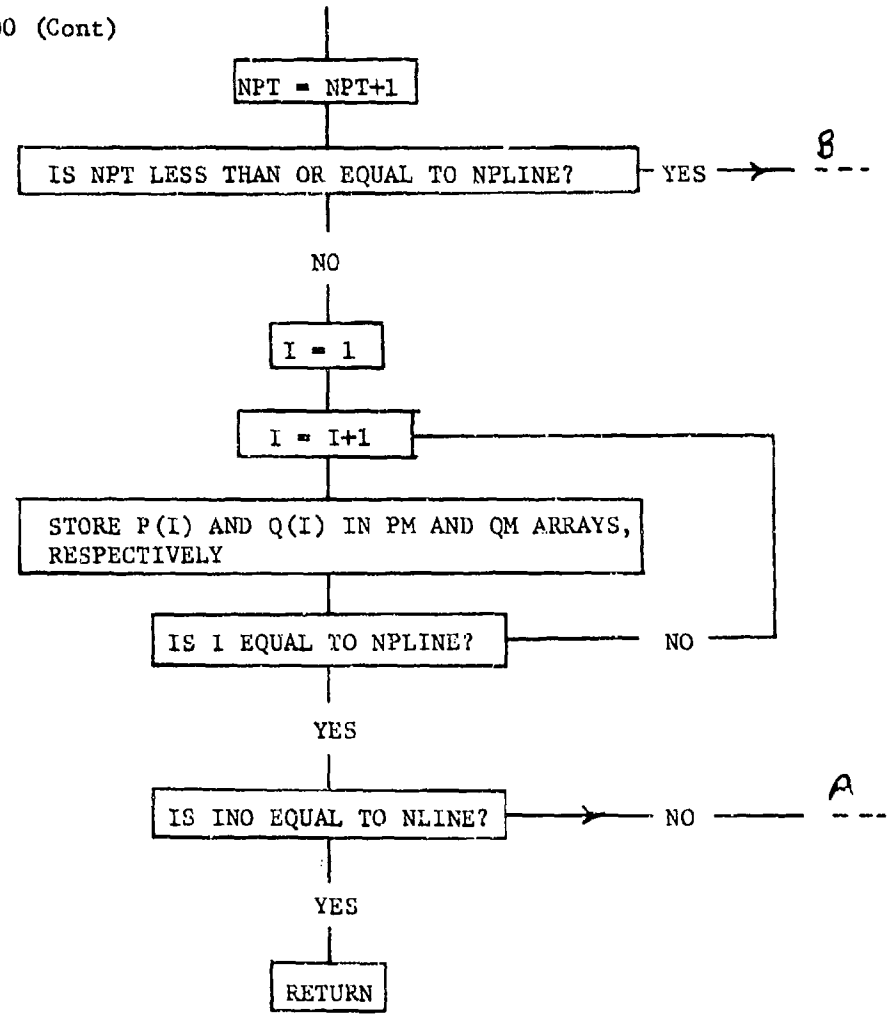



FIGURE 5.1-3 CONT.

Section 2000

The laminar and turbulent flow constants are updated to allow calculation of pressure drop for velocity of sound * DELT length of each line.

```
PARM(IND,8)=PARM(IND.8)*PARM(IND.7)*DELT/PARM(IND,1)
PARM(IND,9)=PARM(IND,9)*PARM(IND,7)*DELT/PARM(IND,1)
```

The pressure and flow differences for each line are divided by the number of line increments (number of line points -1).

```
DELP = (PPM1 - PPM2)/(N-1)
DELQ = (QQM1 - QQM2)/(N-1)
```

The pressures and flows are next calculated and stored for each interior line point. Pressures and flows on the upstream and downstream sides are interpolated then upstream and downstream end point characteristics are calculated and stored for all lines

```
C(LUP) = PP1-ZC*QP1+FRIC(QP1)
C(LDWN) = PM1+ZC*QM1-FRIC(QM1)
```

Section 3000

A DO loop is used to calculate the values for each line.

```
DO 140 IN = 1, NLINE
```

Pressures and flows on the upstream and downstream sides are interpolated.

```
PPI = PM(M)-PARM(IND,3)*(PM(M)-PM(M+1))
QP1 = QM(M)-PARM(IND,3)*(QM(M)-QM(M+1))
PMI = PM(M+N-1)-PARM(IND,3)*(PM(M+N-1)-PM(M+N-2))
QMI = QM(M+N-1)-PARM(IND,3)*(QM(M+N-1)-QM(M+N-2))
```

Upstream and downstream end point characteristics are calculated and stored.

```
C(1ND*2-1) = PPI-ZC*QPI+FRIC(QPI)
C(IND*2) = PMI+ZC*QMI-FRIC(QMI)
```

If there are only 2 points for the given line, control then passes to the end of the DO loop.

```
IF (N.LE.2) GO to 140
```

If there are interior points, the flows and pressures at these points are then calculated. Pressures and flows on the upstream and downstream sides of interior point NPT are interpolated and steady state friction for each of these flows is calculated by calling FRIC.

```
FRICM = FRIC(QMI)
FRICP = FRIC(QPI)
```

The decayed dynamic friction value is calculated by calling DFRICD.

```
DYFRIC = DFRICD(QMI,MPT,JJ,M)
```

The characteristics are now calculated for interior point NPT.

```
CL = +PMI+ZC*QMI-FRICM-DYFRIC
CR = +PPI-ZC*QPI+FRICP+DYFRIC
```

The first approximation of the pressure and flow at point NPT are calculated.

```
P (NPT) =  (CR+CL)/2.
Q (NPT) = (CL-CR)/(2.*ZC)
```

Friction is recalculated using the new flow.

```
FRICR=FRIC (Q(NPT))
```

Dynamic friction is updated using the difference between new and old flows.

```
DYFRIC=DFRICU(DELQ,MPT,JJ,M)
```

The characteristics are updated using the average of the new and old values of friction along with updated dynamic friction.

```
CL = CL+(FRICM-FRICR)/2. - DYFRIC

CR = CR+(FRICR-FRICP)/2. + DYFRIC
```

The pressure, flow and dynamic friction are calculated a final time.

```
P(NPT) = + (CR+CL)/2

Q(NPT) = (CL-CR)/2.*ZC)

DYFRIC = DFRICU(DELQ,MPT,JJ,M)
```

Once the pressures and flows for all points have been calculated, they are stored in the PM and QM arrays.

```
DO 130 I = 2, NPLINE

PM(M+I-1) = P(I)

QM(M+I-1) = Q(I)

130 CONTINUE
```

5.1.4 Approximations

See Appendix A

5.1.5 Limitations

See Appendix A

5.1.6 Variable Names

| | | |
|---|---|---|
| A | Velocity of Sound in Fluid | IN/SEC |
| BULKH | Equivalent Bulk Modulus of Hose | PSI |
| CL | Left Running Characteristic | PSI |
| CONST | Constant-Line Length/DELT | IN/SEC |
| CR | Right Running Characteristic | PSI |
| DELP | Pressure Drop for DELX Length | PSI |

| | | |
|---|---|---|
| DELQ | Flow Difference for DELX Length | CIS |
| DIA | Line I.D. | IN |
| DYFRIC | Dynamic Friction | PSI |
| EQUIVL | Equivalent Length of Line | IN |
| FRICM | Pressure Drop for DELX Length Using Flow QMI | PSI |
| FRICP | Pressure Drop for DELX Length Using Flow QPI | PSI |
| FRICR | Pressure Drop for DELX Length Using Flow Q (NPT) | PSI |
| IT | Fluid Temp/Pressure Number | -- |
| INO | Counter | -- |
| JJ | LSTART(IND)+NLPT(IND)-1 | -- |
| LDWN | Downstream Line End Address | -- |
| LLPT | NLPT(IND)-1 | -- |
| LUP | Upstream Line End Address | -- |
| M | Dummy Variable | -- |
| MPM1 | Address of Current Upstream Line Point | -- |
| MPP1 | Address of Current Downstream Line Point | -- |
| MPT | Address LSTART(IND)+1 | -- |
| N | Counter | -- |
| NPLINE | NLPT(IND)-1 | -- |
| NPT | Current Interior Line Point | PSI |
| PC | Percent Error | -- |
| PLAST | Past Pressure Value | PSI |
| PMI | Interpolated Pressure on the Upstream Side | PSI |
| PPI | Interpoloated Pressure on the Downstream Side | PSI |
| PPM1 | Dummy Variable | -- |
| PPM2 | Dummy Variable | -- |

| | | |
|---|---|---|
| QLAST | Past Flow Value | -- |
| QMI | Interpolated Flow on the Upstream Side | CIS |
| QPI | Interpolated Flow on the Downstream Side | CIS |
| QQM1 | Dummy Variable | -- |
| QQM2 | Dummy Variable | -- |
| REN | Reynolds Number | -- |
| ZC | Dummy Variable | -- |

## 5.1.7 Subroutine Listing

```
      SUBROUTINE LINE
C *** REVISED MAY 1, 1976 ***
      COMMON NTELPL,NTOLPL,IPT,IPOINT,NPTS,INEL,KNEL,NTOPL,NLPLT(61,3),
     1 PQLEG(90,12),LCS(90,10),ILEG(1400),PN(90),QN(90)
      COMMON/SUB/PARM(150,9),PM(1500),QM(1500),P(300),Q(300),C(300)
     1,Z(300),RHO(20),S2ORHO(20),VISC(20),BULK(20),TEMP(20),PVAP(20)
     2,ATPRES,T,DELT,TFINAL,PLTDEL,PI,TITLE(20),LEGN,ICON
     3,KTEMP(99),LSTART(150),NLPT(150),LTYPE(99),NC(99),INX,INZ
     4,INV,ISTEP,NLINE,NEL,IND,IENTR,MNLINE,MNEL,MNLEG,MNNODE,MNPLOT
     5,MNLPTS,MDS
C     THIS SUBROUTINE SIMULATES LINES AS DISTRIBUTED PARAMETER SYSTEMS
C     USING THE METHOD OF CHARACTERISTICS
      DIMENSION LDN(150,7),LC(300)
      EQUIVALENCE(ILEG(1),LDN(1)),(C(1),LC(1))
      IF(IENTR)1000,2000,3000
 1000 CONTINUE
C
C     THIS DO READS LINE DASH NUMBERS(LDN) AND LINE PARAMETERS(PARM)-SEE
C     N      =INDIVIDUAL LINE NUMBER
C     LDN(.,1)=LINE TYPE
C     LDN(.,2)=PERCENTAGE INCRLASE IN LINE LOSS
C     LDN(.,3)=NUMBER OF 45 DEG ELBOWS
C     LDN(.,4)=NUMBER OF 90 DEG ELBOWS
C     LDN(.,5)=TOTAL OF BEND ANGLES .LT. 90 DEG    DEG
C     LDN(.,6)=TOTAL OF BEND ANGLES .GE. 90 DEG    DEG
C     LDN(.,7)= FLUID TEMPERATURE/PRESSURE NUMBER
C***  PARM(.,1)=LENGTH
C     PARM(.,2)=LINE OD    IN.
C     PARM(.,3)=WALL THICKNESS/CONSTANT
C     PARM(.,4)=MODULUS OF ELASTICITY    PSI/TRANSITION FLOW FOR GIVEN
C                 REN,DIA + DENSITY    IN**3/SEC
      DO 1100 I=1,NLINE
      READ(5,1300) N,(LDN(N,J),J=1,7),(PARM(N,J),J=1,4)
 1300 FORMAT ( 8I5,4L10.0)
      IF(LDN(N,7))10,20,30
   10 LDN(N,7)=10-LDN(N,7)
      GO TO 30
   20 LDN(N,7)=1
   30 CONTINUE
      IF(I.NE.N) WRITE(6,430) N
  430 FORMAT ( 43H  THE LINE CARDS ARE OUT OF ORDER AT NUMBER , I5)
 1100 CONTINUE
C     REN    =REYNOLD*S NUMBER
C     EQUIVL=EQUIVALENT LENGTH
C***  PARM(.,5)=DELX
C***  PARM(.,6)=CHARACTERISTIC IMPEDANCE
C***  PARM(.,7)=VELOCITY OF SOUND IN THE LINE
C     PARM(.,8)=LAMINAR FLOW CONSTANT FOR GIVEN DIA,VISC + DENSITY
C     PARM(.,9)=TURBULENT FLOW CONSTANT FOR GIVEN DIA,VISC + DENSITY
C     PARM(.,2) IS INPUTED AS LINE OD AND CONVERTED TO LINE ID
```

5.1.7 (Continued)

```
      REN=1200
      N=1
      DO 80 N=1,NLINE
      IT=LDN(N,7)
      M=N*2
      LC(M)=1
      LC(M-1)=1
      IF(LDN(N,1).LT.10) GO TO 65
      LC(M)=-1
      LDN(N,1)=LDN(N,1)-10
   65 IF(LDN(N,1).EQ.1) GO TO 70
      PARM(N,2)=PARM(N,2)-PARM(N,3)*2.0
      A=1.8*(PARM(N,3)+PARM(N,2)/2.0)*BULK(IT)
      A=BULK(IT)/(RHO(IT)*(1.0+A/(PARM(N,4)*PARM(N,3))))
      GO TO 75
   70 BULKH=BULK(IT)*PARM(N,4)/(BULK(IT)+PARM(N,4))
      A=BULKH/RHO(IT)
   75 A=SQRT(A)
      PARM(N,6)=RHO(IT)*A*4.0/(PI*PARM(N,2)**2)
      CONST=PARM(N,1)/DELT
      IF(CONST.GE.A) GO TO 78
      PC=100.-CONST*100./A
      WRITE(6,300)N,PC
      A=CONST
   78 PARM(N,7)=A
      NLPT(N)=CONST/A+1.01
      PARM(N,5)=PARM(N,1)/(NLPT(N)-1)
      EQUIVL=PARM(N,2)*(LDN(N,3)*12+LDN(N,4)*57.0+LDN(N,5)/45.*4.65+
     1       LDN(N,6)/90.*7.5)+PARM(N,1)*LDN(N,2)/100
      DIA=PARM(N,2)
      PARM(N,8)=128./PI*VISC(IT)*RHO(IT)/(DIA**4.)*(PARM(N,1)+EQUIVL)
      PARM(N,9)=.213*RHO(IT)*(VISC(IT)**.25)/(DIA**4.75)*(PARM(N,1)
     1+EQUIVL)
   80 CONTINUE
      WRITE(6,340)
      WRITE(6,350)((I),(PARM(I,J),J=1,7),I=1,NLINE)
      M=-1
      LSTART(1)=1
      DO 90 N=1,NLINE
      IT=LDN(N,7)
      PARM(N,3)=PARM(N,7)*DELT/PARM(N,5)
      PARM(N,4)=.7854*REN*PARM(N,2)*VISC(IT)
      M=M+2
      IF(LSTART(N)+NLPT(N).GT.MNLPTS-2) GO TO 252
      LSTART(N+1)=NLPT(N)+LSTART(N)
      QPI=DYFRI(DELQ,IT,I,1)
      Z(M)=PARM(N,6)
      Z(M+1)=Z(M)
   90 CONTINUE
C***
```

5.1.7 (Continued)

```
C       INITIALIZE VARIABLES USED FOR PLUGGED AND OPEN
C       CONNECTIONS, (MNLINE)=PLUGGED, (MNLINE-1)=OPEN
        LSTART(MNLINE)=MNLPTS-1
        NLPT(MNLINE)=2
        N=MNLINE*2
        C(N)=0.0
        C(N-1)=ATPRES
        Z(N)=1.0E25
        Z(N-1)=1.0E-10
C***    JUNK WILL BE STORED IN PM(MNLPTS-4) THRO PM(MNLPTS-1)
C***    LIKEWISE FOR QM()
        IF(NLINE.GT.MNLINE-1) GO TO 252
        RETURN
  252   WRITE(6,475) NLINE,LSTART(NLINE),NLPT(NLINE)
        WRITE(6,999)
  999   FORMAT(10X,31HPROGRAM STOP IN SUBROUTINE LINE)
        STOP 5101
  475   FORMAT(5X,9HERROR #5  ,3I10)
 2000   CONTINUE
        DO 2020 INO=1,NLINE
        IND=INO
        PARM(IND,8)=PARM(IND,8)*PARM(IND,7)*DELT/PARM(IND,1)
        PARM(IND,9)=PARM(IND,9)*PARM(IND,7)*DELT/PARM(IND,1)
        ZC=PARM(IND,6)
        N=NLPT(IND)
        M=LSTART(IND)
        PPM1=PM(M)
        PPM2=PM(M-1+N)
        DELP=(PPM1-PPM2)/(N-1)
        QQM1=QM(M)
        QQM2=QM(M-1+N)
        DELQ=(QQM1-QQM2)/(N-1)
        LLPT=N-1
        DO 2010 I=2,LLPT
        PPM1=PPM1-DELP
        QQM1=QQM1-DELQ
        PM(M-1+I)=PPM1
        QM(M-1+I)=QQM1
 2010   CONTINUE
        LUP=IND*2-1
        LDWN=IND*2
        P(LUP)=PM(M)
        Q(LUP)=-QM(M)
        P(LDWN)=PM(M+N-1)
        Q(LDWN)=QM(M+N-1)
C***
C       CALCULATE CR FOR UPSTREAM ENDPOINT
        PPI=PM(M)-PARM(IND,3)*(PM(M)-PM(M+1))
        QPI=QM(M)-PARM(IND,3)*(QM(M)-QM(M+1))
        PMI=PM(M+N-1)-PARM(IND,3)*(PM(M+N-1)-PM(M+N-2))
```

**5.1.7 (Continued)**

```
      QMI=QM(M+N-1)-PARM(IND,3)*(QM(M+N-1)-QM(M+N-2))
      C(LUP)=PPI-ZC*QPI+FRIC(QPI)
C     CALCULATE CL FOR DOWNSTREAM END POINT
      C(LDWN)=PMI+ZC*QMI-FRIC(QMI)
 2020 CONTINUE
      RETURN
 3000 CONTINUE
      DO 140 INO=1,NLINE
      IND=INO
      ZC=PARM(IND,6)
      N=NLPT(IND)
      M=LSTART(IND)
      JJ=M+N-1
      NPLINE=N-1
C     CALCULATE CR FOR UPSTREAM ENDPOINT
      PPI=PM(M)-PARM(IND,3)*(PM(M)-PM(M+1))
      QPI=QM(M)-PARM(IND,3)*(QM(M)-QM(M+1))
      PMI=PM(M+N-1)-PARM(IND,3)*(PM(M+N-1)-PM(M+N-2))
      QMI=QM(M+N-1)-PARM(IND,3)*(QM(M+N-1)-QM(M+N-2))
      C(IND*2-1)=PPI-ZC*QPI+FRIC(QPI)
C     CALCULATE CL FOR DOWNSTREAM END POINT
      C(IND*2)=PMI+ZC*QMI-FRIC(QMI)
      IF(N.LE.2) GO TO 140
C
C     ALGORITHM FOR INTERIOR POINTS OF LINES
      NPT=2
      MPT =M+1
  100 CONTINUE
      MPP1=MPT+1
      MPM1=MPT-1
C     INTERPOLATION ROUTINE
C     PPI - INTERPOLATED PRESSURE ON THE PLUS(DOWNSTREAM) SIDE
C     PMI - INTERPOLATED PRESSURE ON THE MINUS(UPSTREAM) SIDE
C     SIMILAR NOTATION IS USED FOR FLOWS
      QPI=QM(MPT)-PARM(IND,3)*(QM(MPT)-QM(MPP1))
      PPI=PM(MPT)-PARM(IND,3)*(PM(MPT)-PM(MPP1))
      PMI=PM(MPT)-PARM(IND,3)*(PM(MPT)-PM(MPM1))
      QMI=QM(MPT)-PARM(IND,3)*(QM(MPT)-QM(MPM1))
C     FIRST APPROXIMATE CL + CR
      FRICM=FRIC(QMI)
      FRICP=FRIC(QPI)
      DYFRIC=DFRICD(QMI,MPT,JJ,M)
      CL=-PMI-ZC*QMI+FRICM+DYFRIC
      CR=-PPI+ZC*QPI-FRICP-DYFRIC
C     CALCULATE PRESSURE(P) AND FLOW(Q) USING THE ABOVE APPROXIMATIONS
C     P(NPT)= -(CR+CL)/2.
      Q(NPT)= (CR-CL)/(2.*ZC)
      FRICR=FRIC(Q(NPT))
      DELQ=Q(NPT)-QM(MPT)
      DYFRIC=DFRICU(DELQ,MPT,JJ,M)
```

5.1.7 (Continued)

```
C       PLAST=P(NPT)
        QLAST=Q(NPT)
C       RECALCULATE CL + CR USING PREVIOUS ESTIMATE OF Q
        CL=CL-(FRICM-FRICR)/2.+DYFRIC
        CR=CR-(FRICR-FRICP)/2.-DYFRIC
C       RECALCULATE P + Q USING IMPROVED CL + CR
        P(NPT)= -(CR+CL)/2.
        Q(NPT)= (CR-CL)/(2.*ZC)
        DELQ=Q(NPT)-QLAST
        DYFRIC=DFRICU(DELQ,NPT,JJ,M)
        MPT =MPT+1
        NPT=NPT+1
        IF(NPT.LE.NPLINE) GO TO 100
C       AFTER FINDING P + Q FOR THE WHOLE LINE, PUT THESE VALUES INTO PM +
        DO 130 I=2,NPLINE
        PM(M+I-1)=P(I)
        QM(M+I-1)=Q(I)
  130 CONTINUE
  140 CONTINUE
  300 FORMAT(/,5X,44HFIX-UP TAKEN AT LINE 18,VEL OF SOUND IN LINE,I4,2X,
     12HIS,F8.1,17HPER CENT IN ERROR)
  340 FORMAT(//,10H LINE DATA,/,9H LINE NO.,8X,6HLENGTH,9X,8HINTERNAL
     1,7X,4HWALL,11X,10HMODULUS OF,5X,4HDELX,11X,14HCHARACTERISTIC,
     212H VELOCITY OF ,/,32X,
     3          3HDIA,12X,9HTHICKNESS,6X,10HELASTICITY,20X,9HIMPEDANCE
     4,11H       SOUND)
  350 FORMAT(/1X,I5,10X,F8.4,7X,F8.4,7X,F8.4,7X,  E10.3,5X,F8.4,7X,F8.4,
     16X,F11.4)
  360 FORMAT(1X,5X,I5,6E12.5)
        RETURN
        END
```

## 5.2 SUBROUTINE FRIC

FRIC is a function subroutine that is used to calculate steady state pressure drops for laminar and turbulent flows.

### 5.2.1 Math Model

Steady state pressure drops are computed using the Darcy-Weisbach equations

$$FRIC = Q*128*VISC(IT)*RHO(IT)*PARM(IND,1)/(P1*DIA^4) \text{ for laminar flow}$$

$$FRIC = Q*Q^{.75}*.213*RHO(IT)*VISC(IT)^{.25}*PARM(IND,1)/DIA^{4.75} \text{ turbulent flow}$$

Note: The equations assume a Reynolds number of 1200. Since in each equation all variables except flow are actually constants for a given line and fluid, laminar and turbulent flow constants PARM(IND,8) and PARM(IND,9) are calculated in the 1000 section of LINE subroutine for use in this subroutine. The transition flow for each line PARM(IND,4) is also calculated in LINE's 1000 section.

The pressure drop equation can then be written

$$FRIC = Q*PARM(IND,8) \quad \text{for laminar flow}$$

$$FRIC = Q*Q1^{.75}*PARM(IND,9) \text{ for turbulent flow.}$$

### 5.2.2 Assumptions

1. Transition from laminar to turbulent flow is assumed to occur at a Reynolds number of 1200. Flows having a Reynolds number greater than 1200 are considered turbulent while flows having a Reynolds of 1200 or less are assumed laminar.

2. The friction factors used are based on circular cross-section, smooth I.D., drawn tubing.

### 5.2.3 Computation Methods

Not applicable.

5.2.4 Approximations

Pressure drops calculated for flows in the Reynolds number range of 1200 to 3000 are approximate since a transition flow equation was not developed. The turbulent equation is used in this range.

5.2.5 Limitations

FRIC should not be used to calculate pressure drops across non-circular cross section passages or across rough I.D. tubing.

5.2.6 Variable Names

| Variable | Description | Units |
|---|---|---|
| Q | Fluid Flow | $IN^3/SEC$ |
| Q1 | Absolute Value of Fluid Flow | $IN^3/SEC$ |
| FRIC | Pressure Drop | PSI or PSI/Length (depending on the form of the constant) |

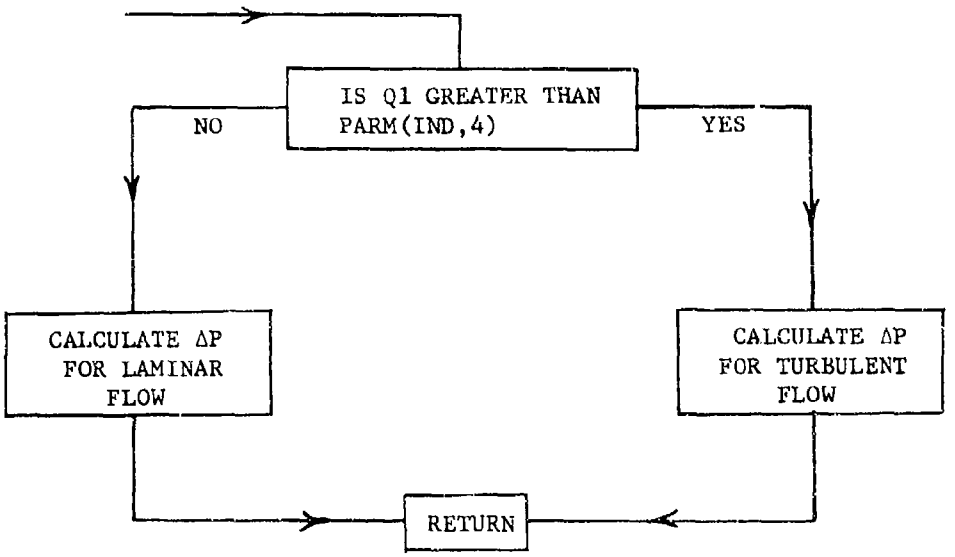



FIGURE 3.2-1 FRIC SUBROUTINE FLOW DIAGRAM

### 5.2.7 Subroutine Listing

```
      FUNCTION FRIC(QA)
C *** REVISED AUGUST 5, 1975 ***
      COMMON/SUB/PARM(150,9),PM(1500),QM(1500),P(300),Q(300),C(300)
     1,Z(300),RHO(20),S2ORHO(20),VISC(20),BULK(20),TEMP(20),PVAP(20)
     2,ATPRES,T,DELT,TFINAL,PLTDEL,PI,TITLE(20),LEGN,ICON
     3,KTEMP(99),LSTART(150),NLPT(150),LTYPE(99),NC(99),INX,INZ
     4,INV,ISTEP,NLINE,NEL,IND,IENTR,MNLINE,MNEL,MNLEG,MNNODE,MNPLOT
     5,MNLPTS,MDS
      Q1=ABS(QA)
      IF(Q1.GT.PARM(IND,4)) GO TO 10
      FRIC=QA*PARM(IND,8)
      RETURN
   10 FRIC=QA*Q1**.75*PARM(IND,9)
      RETURN
      END
```

## 5.3 SUBROUTINE DFRICD

Function DFRICD is used by LINE to calculate pressure loss due to dynamic friction caused by local fluid acceleration. The theory used (see Reference paragraph 9.4) provides an efficient method for determining dynamic friction that gives close correlation with experimental results.

### 5.3.1 Math Model

The theory used for the basis of this subroutine is described in the paper noted in Reference paragraph 9.4. Equation (29) of the referenced paper is the expression used to describe the combined steady state and dynamic pressure losses under laminar flow conditions.

$$f(t+\Delta t) = \frac{8\rho\nu}{a^2} V(t) + \frac{4\rho\nu}{a^2} (y_1 + y_2 + y_3)$$

Substituting equation (35) into equation (33) produces the following expressions for $y_1$, $y_2$ and $y_3$

$$y_1(t + \Delta t) = y_1(t) * e^{-8000*\Delta*\nu/a^2} + 40.*(V(t + \Delta t) - V(t))$$

$$y_2(t + \Delta t) = y_2(t) * e^{-200*\Delta t*\nu/a^2} + 8.1*(V(t + \Delta t) - V(t))$$

$$y_3(t + \Delta t) = y_3(t) * e^{-26.4*\Delta t*\nu/a^2} + (V(t + \Delta t) - V(t)).$$

Each expression for y at time t + Δt contains a decayed pressure at t added to a pressure loss caused by a change in fluid velocity. Converting velocity to volumetric flow rate and I.R. to I.D. yields

$$f(t+\Delta t) = \frac{128 * \rho * \nu * Q(t)}{\pi * D^4} + \frac{16 * \rho * \nu}{D^2}$$

$$\left[ [y_1(t) * e^{-8000 * \Delta t * \nu * 4/D^2} + \frac{4*40}{\pi * D^2} * (Q(t+\Delta t) - Q(t)) \right]$$

$$+ [y_2(t) * e^{-200 * \Delta t * \nu * 4/D^2} + \frac{4*8.1}{\pi * D^2} (Q(t+\Delta t) - Q(t))]$$

$$+ [y_3(t) * e^{-26.4 * \Delta t * \nu * 4/D^2} + \frac{4}{\pi * D^2} (Q(t+\Delta t) - Q(t))] \Bigg] .$$

Since the first term in the equation is steady state pressure drop for laminar flow, it is omitted in this subroutine. The dynamic pressure drop equation can then be written

$$f(t+\Delta t) = \frac{16*\rho*\nu}{D^2} * y_1(t) * e^{-8000*\Delta t*\nu*4/D^2}$$

$$+ \frac{16 * 160}{\pi * D^4} * (Q(t+\Delta t) - Q(t))$$

$$+ \frac{16*\rho*\nu}{D^2} * y_2(t) * e^{-200*\Delta t*\nu*4/D^2}$$

$$+ \frac{16 * 32.4}{\pi * D^4} * (Q(t+\Delta t) - Q(t))$$

$$+ \frac{16*\rho*\nu}{D^2} * y_3(t) * e^{-26.4*\Delta t*\nu*4/D^2}$$

$$+ \frac{16*4}{\pi * D^4} * (Q(t+\Delta t) - Q(t))$$

This equation is used to calculate turbulent as well as laminar flow pressure drops under dynamic conditions since little is known about the effects of dynamic friction under turbulent flow conditions.

In the subroutine, pressure loss at time T to be decayed is expressed by:

$$\frac{16*\rho*\nu}{D^2} * Y_{1,2,3}$$ , and

is actually the latest Y value calculated by entry DFRICU. The first time through the 3000 section of line (i.e., T = DELT), the decayed pressure values are set to zero.

5.3.2 Assumptions

See Reference (4).

5.3.3 Computation Methods

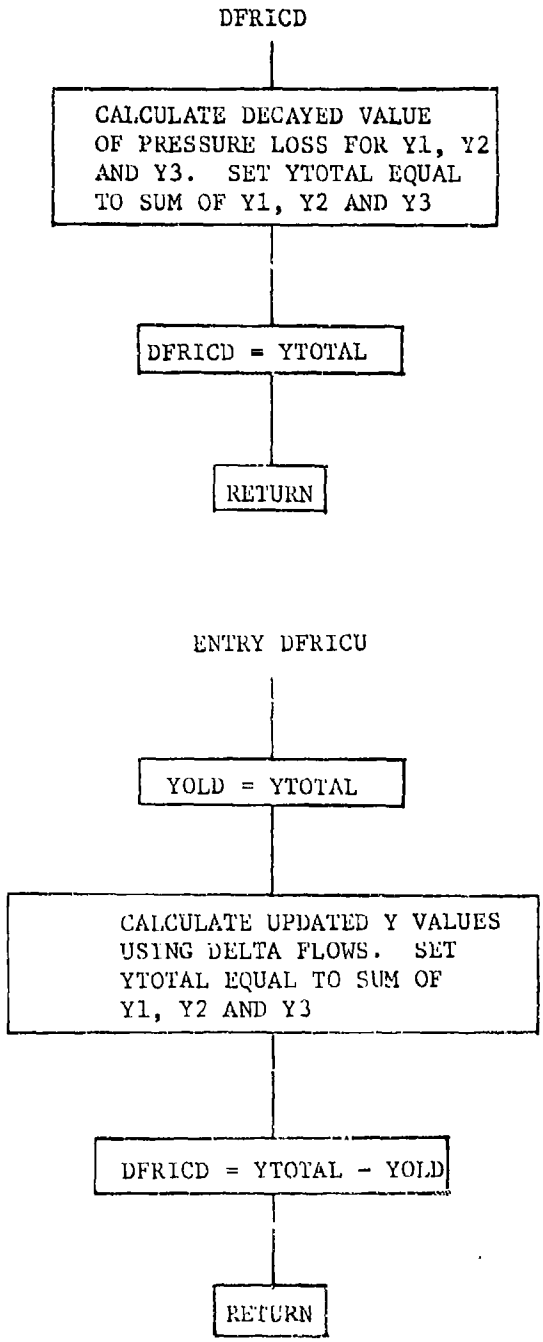
Entry DYFRI is called from the 2000 section of LINE. The purpose of this entry is to calculate and store the dynamic pressure loss equation constants for all LINES. The value returned to LINE ,Y1(1), is not actually used. DFRICD is next called from the 3000 section of LINE. New Y values are calculated using past Y values multiplied by their decay factors. Y1, Y2 and Y3 are summed and set equal to YTOTAL. This decayed value of pressure loss, YTOTAL, at program time T - DELT is returned to LINE which uses it in the first iteration of each time step to calculate flows and pressures at a given interior line point. DFRICU entry is made from the 3000 section of LINE and is used to update the Y values of DFRICD using changes in flow. YTOTAL is set equal to YOLD and all Y values are updated. Y1, Y2 and Y3 are next summed and set to YTOTAL. YOLD is subtracted from YTOTAL and the resulting value is returned to LINE.

5.3.4 Approximations

This method of calculating frequency dependent friction is in itself an approximation of the exact expression. The method was chosen to conserve computer core and computation time.

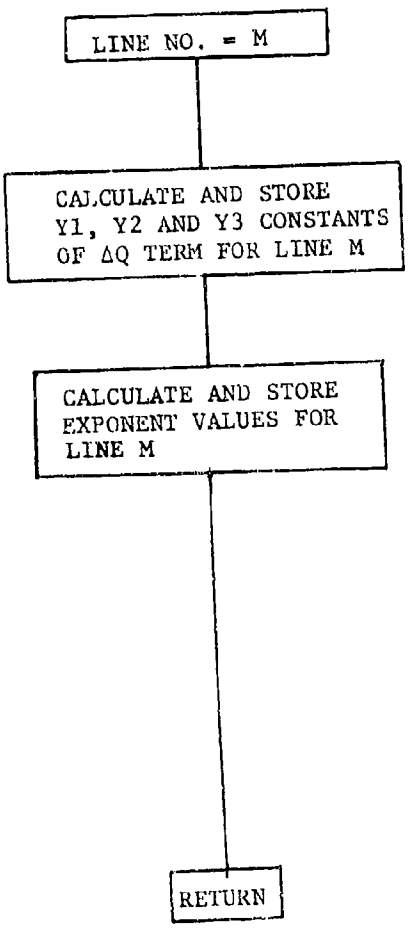
5.3.5 Limitations

See Reference (4).




DFRICD FLOW DIAGRAM
FIGURE 5.3-1

ENTRY DYFRI




ENTRY DYFRI FLOW DIAGRAM
FIGURE 5.3-2

### 5.3.6 Variable Names

| Variable | Description | Units |
|---|---|---|
| CONST | Constant in dynamic pressure loss equation | -- |
| DELQ | New flow - Old flow | CIS |
| DFR1CD | Dummy variable | -- |
| DIA | I.D. of tube | IN |
| EX1 | Constant in dynamic pressure loss equation | -- |
| EX2 | Constant in dynamic pressure loss equation | -- |
| EX3 | Constant in dynamic pressure loss equation | -- |
| JJ | Address LSTART(N) + NLPT(N) -1 | -- |
| M | Address LSTART(N) | -- |
| MM | Address LSTART(N) | -- |
| MPT | Address LSTART(N) +1 | -- |
| M | Counter indicating assigned line number | -- |
| YOLD | Dummy variable | -- |
| YTOTAL | Decayed value of pressure | PSI |
| Y1 ( ) | Array for Y1 valves | -- |
| Y2 ( ) | Array for Y2 values | -- |
| Y3 ( ) | Array for Y3 values | -- |

## 5.3.7 Subroutine Listing

```
      FUNCTION DFRICD(DELQ,MPT,JJ,M)
C *** REVISED AUGUST 5, 1975 ***
      COMMON/SUB/PARM(150,9),PM(1500),QM(1500),P(300),Q(300),C(300)
     1,Z(300),RHO(20),S20RHO(20),VISC(20),BULK(20),TEMP(20),PVAP(20)
     2,ATPRES,T,DELT,TFINAL,PLTDEL,PI,TITLE(20),LEGN,ICON
     3,KTEMP(99),LSTART(150),NLPT(150),LTYPE(99),NC(99),INX,INZ
     4,INV,ISTEP,NLINE,NLL,IND,IENTR,MNLINE,MNEL,MNLEG,MNNODE,MNPLOT
     5,MNLPTS,MDS
      DIMENSION Y1(1500),Y2(1500),Y3(1500)
      DATA Y1,Y2,Y3/4500*0.0/
      Y1(MPT)=Y1(MPT)*Y1(JJ)
      Y2(MPT)=Y2(MPT)*Y2(JJ)
      Y3(MPT)=Y3(MPT)*Y3(JJ)
      YTOTAL=Y1(MPT)+Y2(MPT)+Y3(MPT)
      DFRICD=YTOTAL
      RETURN
      ENTRY DFRICH
      YOLD=YTOTAL
      Y1(MPT)=Y1(MPT)+DELQ*Y1(M  )
      Y2(MPT)=Y2(MPT)+DELQ*Y2(M  )
      Y3(MPT)=Y3(MPT)+DELQ*Y3(M  )
      YTOTAL=Y1(MPT)+Y2(MPT)+Y3(MPT)
      DFRICD=YTOTAL-YOLD
      RETURN
      ENTRY DYFRI
C     IBM 360 - ZERO Y1,Y2,Y3 ARRAYS HERE
      RAD=PARM(M,2)/2.0
      MM=LSTART(M)
      JJ=MM+NLPT(M)-1
      CONST=PARM(M,7)*DELT*RHO(MPT)*VISC(MPT)/(PI*RAD**4.)
      Y1(MM)=160.*CONST
      Y2(MM)=32.4*CONST
      Y3(MM)=4.*CONST
      CONST=DELT*VISC(MPT)/RAD**2.
      EX1=-8000.*CONST
      EX2=-200.*CONST
      EX3=-26.4*CONST
      Y1(JJ)=EXP(EX1)
      Y2(JJ)=EXP(EX2)
      Y3(JJ)=EXP(EX3)
      DFRICD=Y1(1)
      RETURN
      END
```

## 5.4 SUBROUTINE CHAR

The function subroutine CHAR was set up to obtain interpolated values of the line characteristics, CU( ) and CD( ) between TIME = T - DELTA and T.

The interpolations are required by subroutines using integration, such as PUMP51, which require evaluation of the line characteristics at time intervals smaller then DELT..

The subroutine is divided into three parts, using two entry points.

ENTRY CHARIN which is called from the 2000 section of the component subroutine sets up the address system, for the number of active connections, and returns the storage addresses to the calling routine.

ENTRY CHARUP is called from HYTR after the call to LINE and before the call to COMP. This call adds the latest characteristic values and calculates the interpolation formula constants ready for use.

Function CHAR calls obtains interpolated values of the line characteristic for TIME where TIM = 0.0 when TIME = T-DELTA

and TIM = 1.0 when TIME = T,

using the constants derived in the call to ENTRY CHARUP in the interpolation formulas.

### 5.4.1 Theory

The Lagrange Interpolation formula was taken from Reference (3) Chapter 3. A second order interpolation was used which requires the solution of formula.

$$Y(X) = \sum_{k=0}^{i} Lk(X) * Y(Xk)$$

where

$$Li(X) = (X - Xo) \cdots (X-Xi-1)(X - Xi+1) (X-Xn) \Big/ \left[ (Xi - Xo) \cdots (Xi - Xi-1)(Xi-Xi+1) \cdots (Xi - Xn) \right]$$

and Y(Xk) is the value of Y at time X.

for i = -1, 0, and +1 this becomes

$$L(-1)(X) = (X + 0)*(X - 1)/\left[(-1-0)*(-1-1)\right]$$

$$= (X^2 - X)/2$$

$$L(o)(X) = (X + 1)*(X-1)/\left[(0+1)*(0-1)\right]$$

$$= (X^2 - 1)* -1$$

$$L(1)(X) = (X+1)*(X+0)/\left[(1+1)*(1-0)\right]$$

$$= (X^2+X)/2$$

$$Y(X) = X^2*(Y(-1)/2 - Y(0) + Y(1)/2$$

$$+ X*(-Y(-1)/2 + Y(1)/2)$$

$$Y(0)$$

This formula is then used to obtain an interpolated value of Y between X = 0 and X = 1.

The fixed time system was chosen to simplify the calculations,

### 5.4.2 Assumptions

The Lagrange interpolation technique has limitations when applied to rapidly changing functions, in that it tends to produce overshoot. Since there is no pre-determined function that can be used and the interpolation is always using the last piece of data, any interpolation routine be it first, second or third order will suffer from some deficiencies. Hence the choice of technique has to be based on the advantages and disadvantages of each method.

5.4.3 Computation

The function subroutine uses the entries to initialize and update the data, to avoid time consuming tests in the running portion of the program. The initialization section under ENTRY CHARIN is called from the 2000 section of the calling subroutine. The CUCD( ) array is initialized to the steady value of the line characteristic and the line number is stored in the NCUCD( ) array. It will be -ve if its the upstream end of the line. The number of connections for each component is passed to the subroutine via the common variable INX. The address of the first line connection is returned to the subroutine by the value of CHAR, which is then converted to an integer address in the calling program.

5.4.4 Approximations

Not applicable.

5.4.5 Limitations

The Lagrange interpolation technique does not produce a smooth curve from one time interval to the next, however the typical characteristic data form a relatively smooth curve so the interpolation error is not very great.

5.4.6 Variable Names

| Name | Description | Dimension |
|---|---|---|
| CHAR | Valve returned to calling routine or integer address | PSI |
| CUCD( ) | Array of characteristic values and formula for interpolation | |
| I | DO loop counter | |
| N | Line number +ve or -ve | |

| Name | Description | Dimension |
|---|---|---|
| NCUCD | Storage for N values | |
| N1 | Rotating addresses | |
| N2 | Rotating addresses | |
| N3 | Rotating addresses | |
| N4 | Rotating addresses | |
| TIM | No. dimensional time | |

## 5.4.7 Subroutine Listing

```
      FUNCTION CHAR (J,TIME)
C**** REVISED AUGUST 5,1975 ****
      COMMON/SUB/PARM(150,9),PM(1500),QM(1500),P(300),Q(300),C(300)
     1,Z(300),RHO(20),S2ORHO(20),VISC(20),BULK(20),TEMP(20),PVAP(20)
     2,ATPRES,T,DELT,TFINAL,PLTDEL,PI,TITLE(20),LEGN,ICON
     3,KTEMP(99),LSTART(150),NLPT(150),LTYPE(99),NC(99),INX,INZ
     4,INV,ISTEP,NLINE,NEL,IND,IENTR,MNLINE,MNEL,MNLEG,MNNODE,MNPLOT
     5,MNLPTS,MDS
      DIMENSION CUCD(4,99),NCUCD(99)
      COMMON /COMPD/D(4500),L(1500),LE(99,4)
      DATA I,N1,N2,N3,N4/1,1,2,3,4/
      DATA NCUCD(1)/1/
      TIM=1.0-(T-TIME)/DELT
      CHAR=CUCD(N2,J)+CUCD(4,J)*TIM+CUCD(N3,J)*TIM**2
      RETURN
      ENTRY CHARUP
      IF(I.EQ.1) RETURN
      N4=N3
      N3=N2
      N2=N1
      N1=N4
      DO 300 I=1,KCD
      N=NCUCD(I)
      CUCD(N1,I)=C(N)
      CUCD(4,I)=CUCD(N1,I)/2-CUCD(N3,I)/2
      CUCD(N3,I)=CUCD(N1,I)/2-CUCD(N2,I)+CUCD(N3,I)/2
  300 CONTINUE
      CHAR=KCD+.01
      RETURN
      ENTRY CHARIN
      CHAR=I+.01
      DO 20 K=1,INX
      N=L(LL(J,4)+K-1)
      CUCD(1,I)=C(N)
      CUCD(2,I)=C(N)
      CUCD(3,I)=C(N)
      NCUCD(I)=N
      KCD=I
      I=I+1
   20 CONTINUE
      RETURN
      END
```

## 6.0 COMPONENT SUBROUTINES

The components modeled vary from the simple restrictor to a very fast response pump. Each model is broken down into its most basic equations of motion and flow, the sum total of all these equations can be very complex, but individually they are usually simple.

New subroutines can be added without difficulty and, if the computer system can tolerate unsatisfied external references that are not called during execution then component subroutines not in use, can be omitted from the input deck or file when not required.

In working with the program it is necessary to get a good grasp of the fundamentals involved in the simulation, even the most carefully checked routine can have traps built in which are not always found until the output data is carefully examined by someone who khows what it should look like.

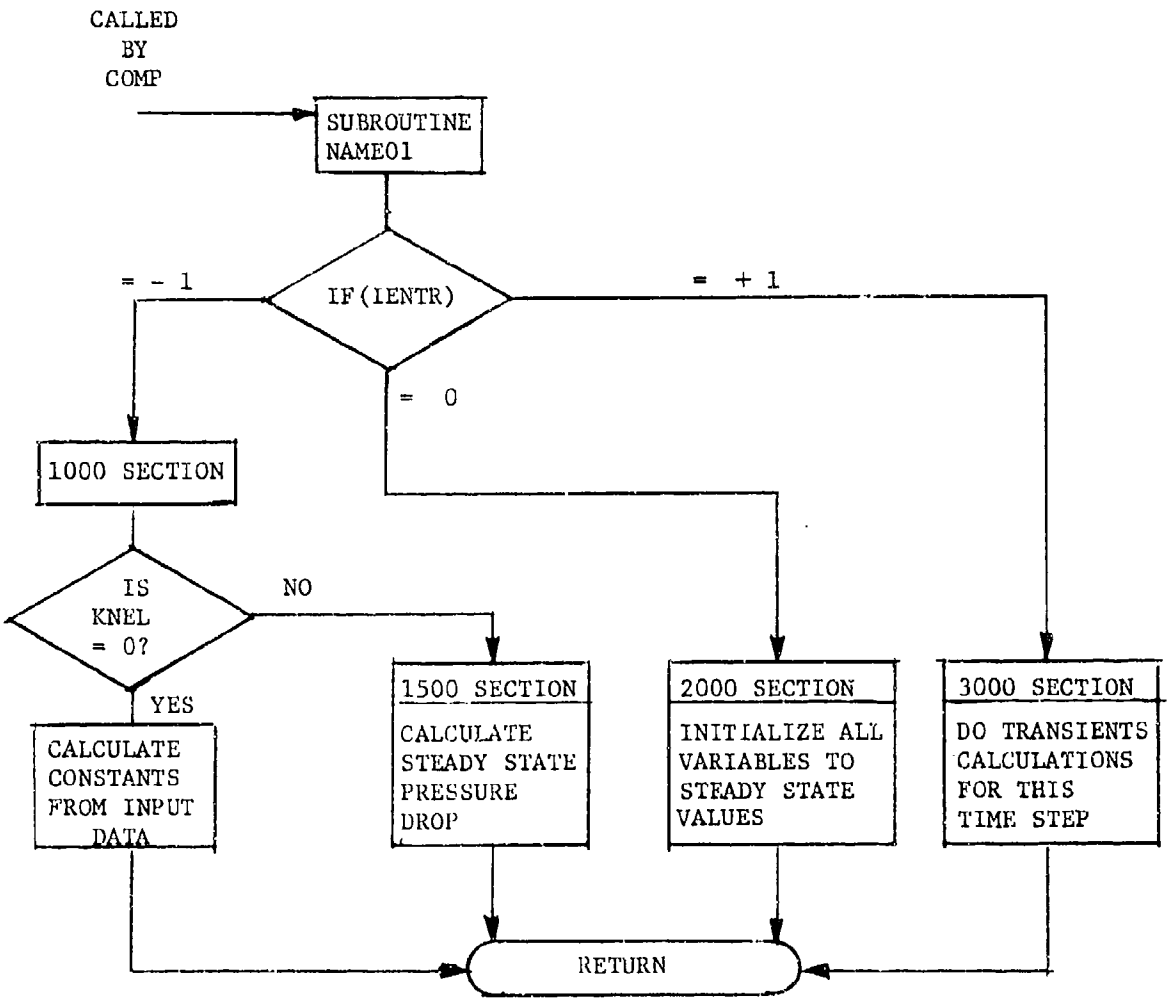



FIGURE 6.1-1

COMPONENT SUBROUTINE ORGANIZATION

## 6.1 SUBROUTINE COMP

Subroutine COMP which is called by HYTR, reads and prints all component input data, sorts out connection data for all components, and calls each individual component in the order in which it was input

### 6.1.1 Math Model

Not applicable.

### 6.1.2 Assumptions

Not applicable.

### 6.1.3 Computation Methods

Section 1000

A component Integer card is read and its data is printed. Connection data for the component is sorted out and stored. The real data cards for the component are then read and printed, if any exists. Next, the addresses of the component's real data, temporary, double precision and integer variables are established. Finally, the individual component subroutine is called passing as arguments its starting address in the real data, temporary, double precision and integer arrays. This process is repeated until all input components have been called.

Section 2000

This section consists of a DO loop that ranges from 1 to the number of input components. Within the loop a component group type is isolated by taking its type number, dividing by 10 and forcing truncation (due to the use of integers). This truncated value is then used in a computed GO TO statement to direct control to a statement or section that calls the specific component. If there is more than one component of that group type, specific component isolation is accomplished by subtracting the group component type number from the individual component type number and using the

resulting value in a computed GO TO statement. This GO TO statement then directs control to a statement that calls the component. COMP isolates and calls each component in a simple and straightforward manner aiding in the running of the overall program since every component has to be called each time step in the transient calculations.

6.1.4 Approximations

Not applicable.

6.1.5 Limitations

Not applicable.

### 6.1.6 Variable Names

| Variable | Description | Units |
|---|---|---|
| I | Counter | |
| J | Counter | |
| KK | Dummy Variable for L4 | |
| KTYPE | Dummy Variable for LTYPE( ) | |
| LDATAC | Number of Element Real Data Cards | |
| L1 | Data value 1 | |
| L4 | Data value 1 | |
| N | Counter | |
| NCI | Number of Last Active Connection | |
| NDATAC | LDATAC/1000 | |
| NLIM | Number of Real Data Fields | |
| NN | Dummy Variable Representing Maximum Number of Component Connections | |
| NTYPE | Group Type Number | |
| NX | Data Value 2 | |
| N1 | Dummy Variable Representing Max Number of Real Data Fields for a Given Component | |
| N2 | Dummy Variable Representing Max Number of Temporary Variables for a Given Component | |
| N3 | Dummy Variable Representing Max Number of Double Precision Variables for a Given Component | |
| N4 | Dummy Variable Representing Max Number of Integer Variables for a Given Component | |

### 6.1.7 Subroutine Listing

```
      SUBROUTINE COMP
C *** REVISED AUGUST 5, 1975 ***
      DOUBLE PRECISION DD
      COMMON NTELPL,NTOLPL,IPT,IPOINT,NPTS,INEL,KNEL,NTOPL,NLPLT(61,3),
     1 POLEG(90,12),ND(150,10)
      COMMON/SUB/PARM(150,9),PM(1500),QM(1500),P(300),Q(300),C(300)
     1,Z(300),RHO(20),S2ORHO(20),VISC(20),BULK(20),TEMP(20),PVAP(20)
     2,ATPRES,T,DELT,TFINAL,PLTDEL,PI,TITLE(20),LEGN,ICON
     3,KTEMP(99),LSTART(150),NLPT(150),LTYPE(99),NC(99),INX,INZ
     4,INV,ISTEP,NLINE,NEL,IND,IENTR,MNLINE,MNEL,MNLEG,MNNODE,MNPLOT
     5,MNLPTS,MDS
      COMMON/COMPD/D(4500),L(1500),LE(99,4)
      DIMENSION DD(1400),LT(10,150),LC(1)
      EQUIVALENCE (L(1),LT(1,1)),(D(1),DD(1)),(C(1),LC(1))
      DATA L1,L4,NX/1,1,2/
      IND=1
      IF(IENTR) 1000,2000,2000
 1000 CONTINUE
      DO 1001 I=1,10
      DO 1002 J=1,150
 1002 ND(J,I)=LT(I,J)
 1001 CONTINUE
 1003 CONTINUE
      NCI=0
C     THIS READ STATEMENT INPUTS THE FOLLOWING DATA
C     I      =INDIVIDUAL COMPONENT NUMBER
C     LTYPE =COMPONENT TYPE
      READ(5,170) I,LTYPE(I),LDATAC,(L(L4-1+J),J=1,12),KTEMP(I)
      WRITE(6,500) I,I,LTYPE(I),LDATAC,(L(L4-1+J),J=1,12),KTEMP(I)
      NDATAC=LDATAC/1000
      LDATAC=LDATAC-NDATAC*1000
      IF(KTEMP(I))11,12,13
   11 KTEMP(I)=10-KTEMP(I)
      GO TO 13
   12 KTEMP(I)=1
   13 CONTINUE
      NLIM=LDATAC*8
C***  LDATAC EQUALS THE NUMBER OF ELEMENT REAL DATA CARDS
      KTYPE=LTYPE(I)
      NC(I)=ND(KTYPE,7)
      IF(NC(I).EQ.0) GO TO 7
      KK=L4
      NN=L4+NC(I)-1
      DO 4 N=KK,NN
      IF(L(N)) 1,2,3
    1 L(N)=-L(N)*2-1
      LC(L(N))=0
      NCI=NCI+1
      GO TO 4
    2 L(N)=MNLINE*2-ND(KTYPE,9)
```

6.1.7 (Continued)

```
      GO TO 4
    3 L(N)=L(N)*2
      LC(L(N))=0
      NCI=NCI+1
    4 CONTINUE
    5 IF(L(NN).LT.MNLINE*2-1) GO TO 6
      NC(I)=NC(I)-1
      NN=NN-1
      GO TO 5
    6 CONTINUE
      INZ=NCI
      IF(NCI.NE.NC(I).AND.ND(KTYPE,10).NE.0)GO TO 420
      IF(NC(I).LT.ND(KTYPE,8)) NC(I)=ND(KTYPE,8)
    7 N1=ND(KTYPE,1)
      N2=ND(KTYPE,2)
      N3=ND(KTYPE,3)
      N4=ND(KTYPE,4)
      IF(NLIM.EQ.0) GO TO 15
      READ(5,450) (D(L1+N-1),N=1,NLIM)
      IF(IND.NE.I) GO TO 410
      NN=L1
      DO 10 KK=1,LDATAC
      WRITE(6,510)KK,(D(NN+N-1),N=1,3)
      NN=NN+3
   10 CONTINUE
   15 INX=0
      IF (NDATAC.NE.0) GO TO 20
      IF (LDATAC.GT.ND(KTYPE,6)) N1=NLIM
      LE(I,1)=L1
      LE(I,2)=L1+N1
      GO TO 30
   20 CONTINUE
      LE(I,1)=LE(NDATAC,1)
      LE(I,2)=L1
      INX=1
   30 IF(NX*N3.LE.1) GO TO 40
      LE(I,3)=(LE(I,2)+N2+3)/2
      L1=(LE(I,3)+N3)*2+1
      GO TO 50
   40 LE(I,3)=LE(I,2)+N2
      L1=LE(I,3)+N3
   50 LE(I,4)=L4
      L4=L4+N4
      ENTRY COMPE
 2000 CONTINUE
      KTYPE=LTYPE(IND)
      NTYPE =KTYPE/10
      N1=LE(IND,1)
      N2=LE(IND,2)
      N3=LE(IND,3)
```

6.1.7 (Continued)

```
      N4=LE(IND,4)
      GO TO (210,220,230,240,250,260,270,280,290,300),NTYPE
C 360 GO TO 400
  210 CONTINUE
      CALL BRAN11 (D(N1),D(N2),DD(N3),L(N4))
      GO TO  400
  220 CONTINUE
      KTYPE=KTYPE-20
      GO TO (221,222,223,224),KTYPE
  221 CALL VALV21 (D(N1),D(N2),DD(N3),L(N4))
      GO TO 400
  222 CALL VALV22 (D(N1),D(N2),DD(N3),L(N4))
      GO TO 400
  223 CALL VALV23(D(N1),D(N2),DD(N3),L(N4))
      GO TO 400
  224 CALL VALV24(D(N1),D(N2),DD(N3),L(N4))
      GO TO 400
  230 CONTINUE
      KTYPE=KTYPE-30
      GO TO (231,232,233,234),KTYPE
  231 CALL CVAL31 (D(N1),D(N2),DD(N3),L(N4))
      GO TO 400
  232 CALL CREL32 (D(N1),D(N2),DD(N3),L(N4))
      GO TO 400
  233 CALL CVAL33(D(N1),D(N2),DD(N3),L(N4))
      GO TO 400
  234 CALL CVAL34(D(N1),D(N2),DD(N3),L(N4))
      GO TO 400
  240 CONTINUE
      CALL RLST41 (D(N1),D(N2),DD(N3),L(N4))
      GO TO 400
  250 CONTINUE
      KTYPE=KTYPE-50
      GO TO (251,252,253,254,400),KTYPE
  251 CALL PUMP51 (D(N1),D(N2),DD(N3),L(N4))
      GO TO 400
  252 CALL PUMP52 (D(N1),D(N2),DD(N3),L(N4))
      GO TO 400
  253 CALL PUMP53 (D(N1),D(N2),DD(N3),L(N4))
      GO TO 400
  254 CALL PUMP54 (D(N1),D(N2),DD(N3),L(N4))
      GO TO 400
  260 CONTINUE
      KTYPE=KTYPE-50
      GO TO (261,262,263),KTYPE
  261 CALL RSVR61 (D(N1),D(N2),DD(N3),L(N4))
      GO TO 400
  262 CALL RSVR62 (D(N1),D(N2),DD(N3),L(N4))
      GO TO 400
  263 CALL RSVR63 (D(N1),D(N2),DD(N3),L(N4))
```

6.1.7 (Continued)

```
      GO TO 400
  270 KTYPE=KTYPE-70
      GO TO (271,272,400),KTYPE
  271 CALL ACUM71 (D(N1),D(N2),DD(N3),L(N4))
      GO TO 400
  272 CALL ACUM72 (D(N1),D(N2),DD(N3),L(N4))
      GO TO 400
  280 CONTINUE
      KTYPE=KTYPE-80
      GO TO (281,282,283,400),KTYPE
  281 CALL FILT81 (D(N1),D(N2),DD(N3),L(N4))
      GO TO 400
  282 CALL FILT82 (D(N1),D(N2),DD(N3),L(N4))
      GO TO 400
  283 CALL FILT83(D(N1),D(N2),DD(N3),L(N4))
      GO TO 400
  290 CONTINUE
      KTYPE=KTYPE-90
      GO TO (291,292,293,294,295,296,297,298,299),KTYPE
  291 CALL TEST91 (D(N1),D(N2),DD(N3),L(N4))
C291  CONTINUE
      GO TO 400
  292 CALL CAD92 (D(N1),D(N2),DD(N3),L(N4))
      GO TO 400
  293 CALL CAD93 (D(N1),D(N2),DD(N3),L(N4))
      GO TO 400
  294 CONTINUE
      GO TO 400
  295 CALL APU95(D(N1),D(N2),DD(N3),L(N4))
      GO TO 400
  296 GO TO 400
  297 GO TO 400
  298 CALL CAD98(D(N1),D(N2),DD(N3),L(N4))
      GO TO 400
  299 CALL CAD99(D(N1),D(N2),DD(N3),L(N4))
      GO TO 400
  300 CONTINUE
      KTYPE=KTYPE-100
      GO TO (301,302,303,304,305,306,307,400),KTYPE
  301 CALL ACT101 (D(N1),D(N2),DD(N3),L(N4))
      GO TO 400
  302 CALL ACT102 (D(N1),D(N2),DD(N3),L(N4))
      GO TO 400
  303 CALL ACT103 (D(N1),D(N2),DD(N3),L(N4))
      GO TO 400
  304 CALL ACT104 (D(N1),D(N2),DD(N3),L(N4))
      GO TO 400
  305 CALL ACT105 (D(N1),D(N2),DD(N3),L(N4))
      GO TO 400
  306 CALL ACT106 (D(N1),D(N2),DD(N3),L(N4))
```

**6.1.7 (Continued)**

```
      GO TO 400
  307 CALL ACT107 (D(N1),D(N2),DD(N3),L(N4))
  400 CONTINUE
      IF(INEL.NE.0.OR.IND.GE.NEL) RETURN
      IND=IND+1
      IF(IENTR) 1003,2000,2000
  410 WRITE (6,180)
      WRITE(6,999)
  999 FORMAT(10X,31HPROGRAM STOP IN SUBROUTINE COMP)
      STOP 6001
  420 WRITE(6,190)IND
  190 FORMAT(5X,42HTHERE ARE MISSING CONNECTIONS IN COMP NO   ,I5)
      WRITE(6,999)
      STOP 6001
  170 FORMAT (16I5)
  180 FORMAT ( 35H THE ELEMENT CARDS ARE OUT OF ORDER  )
  450 FORMAT (8E10.0)
  500 FORMAT(1H0,5X,6HCOMP#,I5,2X,12HINTEGER DATA,2X,16I5,)
  510 FORMAT(/,5X,15HREAL DATA CARD # ,I5,2X,8E13.4)
      END
```

## 6.11 SUBROUTINE BRAN11

The branch subroutine describes a simple lossless junction in the system, which can have any of the configurations shown in Figure 6.11-1.

The program is generalized so that there can be any combination of upstream and downstream connections.

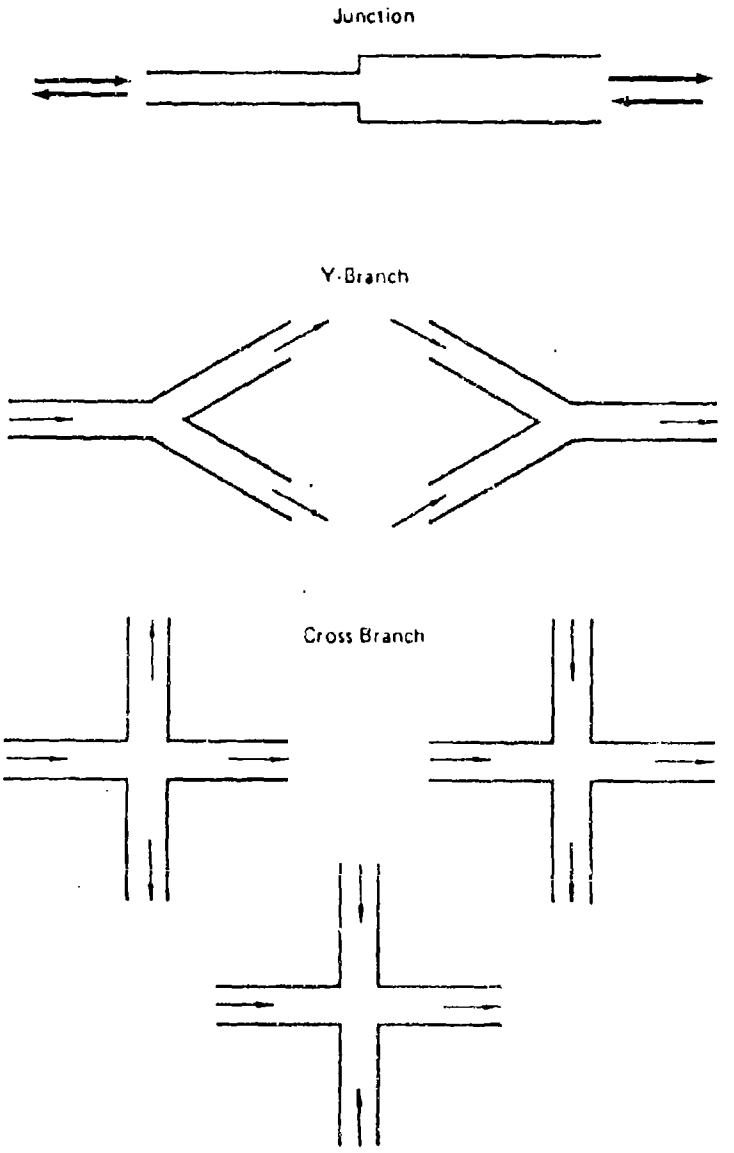



FIGURE 6.11-1
BRANCH CONFIGURATIONS

### 6.11.1 Math Model

Starting with the basic line equations

$$P + Z(I) \times Q(I) - C(I) = 0 \quad . \tag{1}$$

where C(I) is the Ith line characteristic

Z(I) is the line characteristic

and Q(I) is the flow out of the line into the branch junction (positive flow is into the junction) .

Rearranging (1) we get

$$Q(I) = PX/Z(I) + C(I)/Z(I) \tag{2}$$

$$\text{If } \sum Q(I) = 0 \tag{3}$$

I = 1,N.

$$PX^* \sum 1/Z(I) = \sum_{I=1,\ N} C(I)/Z(I) \tag{4}$$

$$\text{Let } DT(3) \sum_{I=1,\ N} 1/Z(I) \tag{5}$$

$$CN = \sum_{I=1,\ N} C(I)/Z(I) \tag{6}$$

$$PX = CN/DT(3) \tag{7}$$

which is the solution for the pressure at the flow junction.

$$Q(I) = (C(I) - PX/Z(I) \tag{8}$$

for cavitation conditions .

If (PX.LT.PVAP) or IF (DT(VCAV)GT.0) PX = PVAP

Q(1) = (C(1) - PVAP)/Z(1)

QNEW = Net flow away from the junction

DT(VCAV) = D(VCAV) - QNEW

DT(QCAV) = QNEW

6.11.2 Assumptions

The math model does not incorporate any of the losses which normally occur at junctions which have changes in diameter, flow direction, or flow division.

Line losses due to steady state friction are incorporated into the values of C(I) but are not corrected for the average flow in that line segment. This will tend to give rise to inaccurate pressure losses during flow acceleration, but under steady or pseudo steady conditions there will be no error.

6.11.3 Computation Method

Section 1000

This section calculates the variable D(3) for each connection.

Section 2000

Section 2000 initializes DT(VCAV) and DT(QCAV) to zero.

Section 3000

This section retrieves values for Z(I) @ C(I), and sums C(I)/Z(I). From this PX is calculated and then the pressures and flows are returned to the line ends in P(I) and Q(I) if the pressure is greater than the vapor pressure or if there is no cavitation bubble present signified by DT(VCAV)=0.0.

If cavitation condition exists, the flows are calculated using PVAP at the junction instead of PX. The net flow at the junction, QNEW is then calculated and added to the old cavity volume DT(VCAV).

The values of DT(QCAV) and DT(VCAV) are stored for use next time and the cavitation model will continue to control the pressure at the junction until the pressure rises again and a series of negative QNEWS refill the cavity, and then the simulation returns to normal.

6.11.4 Approximations

The only computation approximation is the use of pseudo Z and C for an unassigned connection # which is less than the number of connections to

the branch such as blanked off connection will be most unusual in practice and the error will not be noticed.

6.11.5 Limitations

The limitations of this subroutine are due to the pressure drop errors. It is recommended that a series of special type branch subroutines be added so that these losses can be taken into account. This particular subroutine being fast and simple and should be kept until the other subroutines are added. Additional losses can be simulated by adding a pseudo 90 degree elbow or bend, to the appropriate line.

6.11.6 Variable Names

| NAME | DESCRIPTION | DIMENSIONS |
|---|---|---|
| CN | Temporary Variable | CIS |
| I | Integer Counter | |
| N | Integer Counter | |
| NCI | Number of Line Connections | |
| PX | Branch Pressure | PSI |
| DT(QCAV) | Previous cavity flow | CIS |
| QNEW | Net cavity flow toward the junction | CIS |
| DT(VCAV) | Cavity Volume | IN**3 |

For variables in common refer to Para. 3.3.

## 6.11.7 Subrourtine Listing

```
      SUBROUTINE BRAN11 (D,DT,DD,L)
C *** REVISED AUGUST 5, 1975 ***
      DOUBLE PRECISION DD
      DIMENSION D(1),DT(3),DD(1),L(1)
      COMMON NTELPL,NTOLPL,IPT,IPOINT,NPTS,INEL,KNEL,NTOPL,NLPLT(61,3),
     1 PQLEG(90,12)
      COMMON/SUB/PARM(150,9),PM(1500),QM(1500),P(300),Q(300),C(300)
     1,Z(300),RHO(20),S2ORHO(20),VISC(20),BULK(20),TEMP(20),PVAP(20)
     2,ATPRES,T,DELT,TFINAL,PLTDEL,PI,TITLE(20),LEGN,ICON
     3,KTEMP(99),LSTART(150),NLPT(150),LTYPE(99),NC(99),INX,INZ
     4,INV,ISTEP,NLINE,NEL,IND,IENTR,MNLINE,MNLL,MNLEG,MNNODE,MNPLOT
     5,MNLPTS,MDS
      INTEGER VCAV,QCAV
      DATA VCAV,QCAV/1,2/
      IF(IENTR)1000,2000,3000
 1000 CONTINUE
      IF(INEL.NE.0) GO TO 1500
      DT(3)=0.0
      DT(VCAV)=0.0
      NCI=NC(IND)
      DO 1001 I=1,NCI
 1001 DT(3)=DT(3)+1.0/Z(L(I))
      RETURN
 1500 DT(VCAV)=PQLEG(INEL,11)
      RETURN
 2000 DT(VCAV)=0.0
      DT(QCAV)=0.0
      RETURN
 3000 NCI=NC(IND)
      CN=0.0
      DO 3010 I=1,NCI
      CN=C(L(I))/Z(L(I))+CN
 3010 CONTINUE
      PX=CN/DT(3)
C
C *** CHECK TO SEE IF PX IS LT PVAP OR A BUBBLE IS PRESENT
C
      IF (PX.LT.PVAP(KTEMP(IND)).OR.DT(VCAV).GT.0.0)GO TO 3025
C
      DO 3020 I=1,NCI
      P(L(I))=PX
 3020 Q(L(I))=(C(L(I))-PX)/Z(L(I))
      RETURN
 3025 QNEW=0.0
      DO 3030 I=1,NCI
      P(L(I))=PVAP(KTEMP(IND))
      Q(L(I))=(C(L(I))-PVAP(KTEMP(IND)))/Z(L(I))
 3030 QNEW=QNEW+Q(L(I))
C *** CALCULATE THE BUBBLE SIZE
      DT(VCAV)=DT(VCAV)-QNEW
      IF(DT(VCAV).LT.0.0)DT(VCAV)=0.0
      DT(QCAV)=QNEW
      RETURN
      END
```

## 6.21 SUBROUTINE VALV21

VALV21 simulates a simple two-way valve, with an input valve position time history. Its main purpose is in simulations associated with test rigs where a fast or slow, opening or closing valve is used to generate system transients. It can be used to simulate directly operated solenoid valves, or balanced poppet valves, but not those which have displacement characteristics such as two stage solenoid valves and unbalanced poppet valves.

A typical valve is shown in Figure 6.21-1.

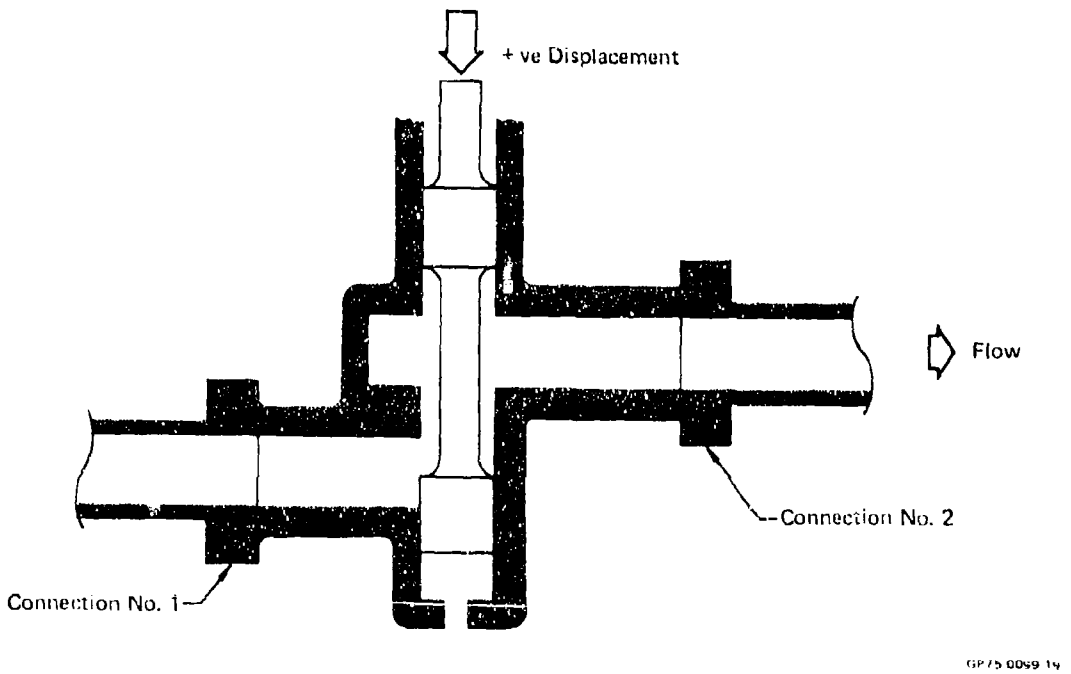



FIGURE 6.21-1
TYPE NO. 21 TWO-WAY VALVE

6.21.1 Math Model

The basic equation for flow through an orifice

$$Q = AREA * Cd * \left(2*(P1-P2)/RHO\right)^{1/2} \quad (1)$$

where AREA = area of the valve orifice $in^2$
Cd=valve discharge coefficient

RHO = fluid density lbs $sec^2/in^4$
Q = flow in CIS
P1 = inlet pressure PSI
P2 = outlet pressure PSI

From the line equations

$$P1 = -Z_1Q_1 + C_1 \quad (2)$$

$$P2 = Z_2Q_2 + C_2 \quad (3)$$

using a +ve flow convention from 1 to 2

where C1 = Con #1 line characteristic
and Z1 = Con #1 line characteristic impedance .

Assuming no flow loss,

$$Q = Q_1 = Q_2 \quad . \quad (4)$$

Substituting 2 and 3 in 1 we get

$$Q^2 = VFAC(-Z_1Q_1 + C_1 - Z_2Q_2 - C_2) \quad (5)$$

where $VFAC = (AREA * Cd)^2 * 2/RHO$ .

Equating the flows we get

$$Q^2 + Q*(Z_1 + Z_2) *VFAC - (C_1 - C_2)*VFAC = 0 \quad (6)$$

$$Q^2 + Q*VB + VC = 0 \quad (7)$$

where

$$VB = (Z_1 + Z_2) \times VFAC \quad (8a)$$

$$VC = (C_2 - C_1) \times VFAC \quad . \quad (8b)$$

The restrictions on the solution are

when $V_c = 0$ $\quad Q = 0$

which gives

```
Q = (-VB + (VB**2-4*VC)**.5)/2
```

6.21.2 Assumptions

The math model assumes a square law characteristic and a constant discharge coefficient for the complete flow range, which in practice is not correct. At very low flows the pressure drop tends toward a linear characteristic, and the discharge coefficient varies.

In addition, most valves have inlet and outlet passages and fittings which add to the pressure drop and hence modify its overall characteristics. These can of course be lumped together to obtain an approximate overall square law but the resulting characteristic would not fit very well over the complete range of flows and valve openings.

To avoid these errors a more complex math model is required, and this should be implemented in another subroutine.

6.21.3 Computation Method

A call to INTERP is made to derive the valve opening from the input data which includes a table of valve positions versus time. A first order interpolation is used in this derivation, second or higher order interpolations will cause unintended valve motion.

Once the valve opening is established, the valve area is calculated, and then the flow.

Since this is a direct calculation there is no difference between the math model and the calculation technique until very small valve openings are reached. To avoid division by zero, valve openings of less than 0.00001 are ignored and the flow is assumed to be zero.

6.21.4 Assumptions - To be added later.

6.21.5 Limitations

The computation is limited to a linear valve area versus position relationship. This apparent limitation can be overcome by inputting a non-linear valve position versus time relationship which can produce any desired area versus time.

The constant discharge coefficient is also a limitation but since the changes in discharge coefficients depend on the particular valve configuration; this limitation is not easily overcome.

If the valve slot width and discharge coefficient are input as 1.0 then the valve position table becomes a table of the product of valve area, times the discharge coefficient, versus time.

The combined effects of area and discharge coefficient can then be inputted, but this will still leave out the effects of changes in the orifice characteristic with differential pressure.

6.21.6 Variable Names

| NAME | DESCRIPTION | DIMENSION |
|---|---|---|
| AREA | SS. VALVE AREA | IN**2 |
| CONST | SS. VALVE CONSTANT | PSI/CIS**2 |
| DELTP | SS. VALVE PRESSURE DROP | PSI |
| DIFF | CHARACTERISTIC DIFFERENCE | PSI |
| IOD | ERROR INDICATOR | |
| L1 | DUMMY VARIABLE | |
| L2 | DUMMY VARIABLE | |
| QX | VALVE FLOW | CIS |
| SGN | SIGN OF FLOW | |
| VALY | VALVE POSITION | IN |
| VFAC | VALVE FLOW CONSTANT | PSI/CIS**2 |
| VYMAG | ABSOLUTE VALUE VALVE POSITION | IN |

FOR VARIABLES IN COMMON REFER TO PARA. 3.3

## 6.21.7 Subroutine Listing

```
      SUBROUTINE VALV21 (D,DT,DD,L)
C *** REVISED AUGUST 5, 1975 ***
      DOUBLE PRECISION DD
      COMMON NTELPL,NTOLPL,IPT,IPOINT,NPTS,INEL,KNEL,NTOPL,NLPLT(61,3),
     1 PQLEG(90,12),LCS(90,10),ILEG(1400),PN(90),QN(90)
      DIMENSION D(24),DT(2),DD(1),L(5)
      COMMON/SUB/PARM(150,9),PM(1500),QM(1500),P(300),Q(300),C(300)
     1,Z(300),RHO(20),S2ORHO(20),VISC(20),BULK(20),TEMP(20),PVAP(20)
     2,ATPRES,T,DELT,TFINAL,PLTDEL,PI,TITLE(20),LEGN,ICON
     3,KTEMP(99),LSTART(150),NLPT(150),LTYPE(99),NC(99),INX,INZ
     4,INV,ISTEP,NLINE,NEL,IND,IENTR,MNLINE,MNEL,MNLEG,MNNODE,MNPLOT
     5,MNLPTS,MDS
      IF (IENTR) 1000,2000,3000
 1000 CONTINUE
      IF(INEL.NE.0) GO TO 1500
      L(3)=9
      L(4)=9+(L(11)+7)/8*8
      D(3)=(D(1)*D(2))**2*2/RHO(KTEMP(IND))
      L(5)=L(11)
      D(4)=(Z(L(1))+Z(L(2)))/2.
      D(5)=1.0/D(4)**2
      VALY=D(L(4))
      IF(VALY.EQ.0.0)GO TO 1100
      DT(1)=RHO(KTEMP(IND))/(2.*(D(1)*VALY*D(2))**2)
      DT(2)=0.0
      RETURN
 1100 DT(1)=0.0
      DT(2)=10.E5
      RETURN
C
C**** STEADY STATE SECTION ****
C
 1500 PQLEG(INEL,8)=PQLEG(INEL,8)+DT(1)
      QX=PQLEG(INEL,1)
      PQLEG(INEL,6)=PQLEG(INEL,6)+DT(2)
      PQLEG(INEL,11)=PQLEG(INEL,11)-(PQLEG(INEL,2)*QX*(DT(2)+QX*DT(1)))
      RETURN
 2000 CONTINUE
      DT(1)=0.0
      DT(2)=0.0
      RETURN
 3000 CONTINUE
      L1=L(1)
      L2=L(2)
      CALL INTERP (T,D(L(3)),D(L(4)),10,L(5),VALY,IOD)
      VYMAG= ABS(VALY)
      DIFF=C(L1)-C(L2)
      IF(DT(1)+DT(2).GT.0.0)GO TO 3300
      IF(VYMAG.LT.0.00001) GO TO 3100
      VFAC=D(3)*VALY**2
```

6.21.7 (Continued)

```
      QX=SIGN(VFAC*D(4)*(SQRT(1+ABS(DIFF)/VFAC*D(5))-1),DIFF)
      P(L1)=C(L1)-Z(L1)*QX
      P(L2)=C(L2)-Z(L2)*(-QX)
      Q(L1)=QX
      Q(L2)=-QX
      GO TO 3200
 3100 CONTINUE
      P(L1)=C(L1)
      P(L2)=C(L2)
      Q(L1)=0.0
      Q(L2)=0.0
 3200 IF(P(L1).GE.PVAP(KTEMP(IND)).AND.P(L2).GE.PVAP(KTEMP(IND)))RETURN
      IF(P(L1).LT.PVAP(KTEMP(IND)).AND.P(L2).LT.PVAP(KTEMP(IND)))
     + GO TO 3230
      IF(P(L1).LT.PVAP(KTEMP(IND)))GO TO 3220
 3210 P2=PVAP(KTEMP(IND))
      ZT=Z(L1)
      P1=C(L1)
      IF(C(L1).GT.PVAP(KTEMP(IND)))GO TO 3500
      P1=PVAP(KTEMP(IND))
      GO TO 3600
 3220 P1=PVAP(KTEMP(IND))
      ZT=Z(L2)
      P2=C(L2)
      IF(C(L2).GT.PVAP(KTEMP(IND)))GO TO 3500
      P2=PVAP(KTEMP(IND))
      GO TO 3600
 3230 IF(P(L1)-P(L2))3210,3400,3220
 3300 IF(DT(1).GT.0.0.AND.DT(2).GT.0.0)GO TO 3400
      IF(DT(1).GT.0.0)GO TO 3220
      GO TO 3210
 3400 P1=PVAP(KTEMP(IND))
      P2=PVAP(KTEMP(IND))
      GO TO 3600
 3500 IF(VYMAG.LT..00001)GO TO 3600
      VFAC=D(3)*VALY**2
      QV=VFAC*ZT/2.0*(SQRT(1.0+4.0*ABS(P1-P2)/(VFAC*ZT**2))-1.0)
      QV=SIGN(QV,P1-P2)
      IF(P1.GT.PVAP(KTEMP(IND)))GO TO 3700
      GO TO 3800
 3600 QV=0.0
      GO TO 3750
 3700 P(L1)=C(L1)-QV*Z(L1)
      Q(L1)=QV
 3750 Q(L2)=(C(L2)-P2)/Z(L2)
      P(L2)=P2
      DT(2)=DT(2)-QV-Q(L2)
      IF(QV.EQ.0.0)GO TO 3850
      RETURN
 3800 P(L2)=C(L2)-QV*Z(L2)
```

6.21.7 (Continued)

```
      Q(L2)=QV
 3850 Q(L1)=(C(L1)-P1)/Z(L1)
      P(L1)=P1
      DT(1)=DT(1)+QV-Q(L1)
      RETURN
      END
```

## 6.22 SUBROUTINE VALV22

Subroutine VALV22 describes a generalized four-way valve which can be a segment of a servo actuator (connected to it by lines) or control any servo or utility type function.

The valve position is derived from input data, tabulated versus time. The actual position is obtained using linear interpolation between the nearest two data inputs.

The valve orifice areas are derived using a variable law which can effectively describe leakage, open center, underlap and overlap conditions, with various pressure gains.

The valve calculations are treated somewhat differently to other components. Pressures at the valve ports are obtained using a matrix technique with a iterative solution. This method can easily be extended to include other flow paths such as the first stage flow for an electro-hydraulic servo valve.

VALV22 will also be expanded in the future to include the effects cavitation at all four ports, since fast openings or closing valves are a common source of cavitation at return and load ports, and even at the supply port under high flow, low pressure conditions.

### 6.22.1 Math Model

CALCULATION OF ORIFICE AREAS

Spool and sleeve type servo valves can have a variety of orifice configurations, the commonest of which are round holes and square or rectangular slots.

Because of radial clearances between the spool and sleeve, there is usually a leakage flow when the orifice is completely covered.

This leakage tends to round the ends of what would otherwise be a linear flow versus spool position characteristic. In order to simplify the flow calculations, we have assumed that the valve area is an equivalent area which allows the orifice equations to be used at all times.

The orifice equation

Q = AREA*Cd*SQRT(2*(P1-P2)/RHO) * SIGN (P1-P2)

is used in the form

QNEW = VA(I)*SQRABS(CALC2(I)-CALC2(N5))

where VA(I) = AREA*Cd*SQRT(2/RHO) * X

and CALC2( ) are the pressures at the valve connections.

To obtain the valve area, for a given valve position, a characteristic curve is generated based on the projected cut-off position, the projected max open position and the max valve area.

The maximum valve area is combined with the discharge coefficient and the SQRT(2/RHO) to give an orifice resistance. The formula used to generate the characteristic curve is

X = (.5 + XT/(1 + ABS(XT*2)**Y)**(1/Y)

where $0 \leq X \leq 1.0$ for all values of XT.

when Y is large ie 64, the characteristic curve is almost a straight line between projected cut-off and projected maximum opening. Family of curves for different values of Y is shown in Figure 6.22-3

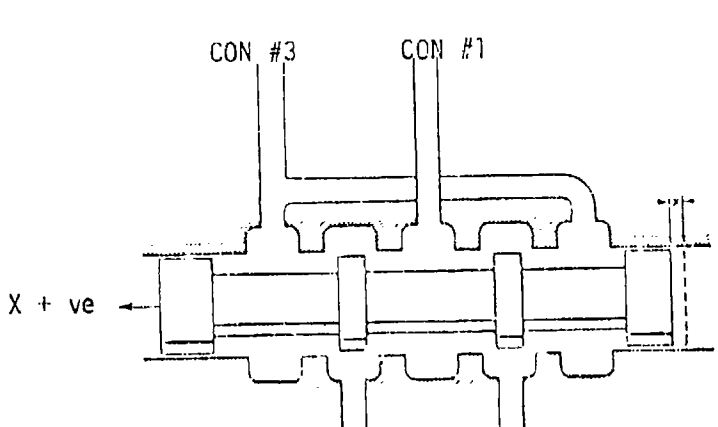



**FIGURE 6.22-1**

TYPE NO. 22 FOUR-WAY VALVE

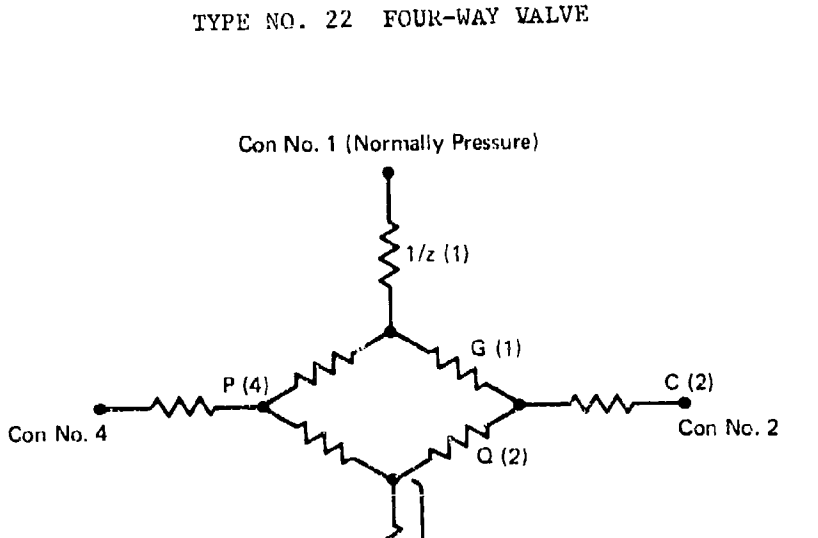



**FIGURE 6.22.2**
**VALV22 MATRIX MODEL**

CALCULATION OF PORT PRESSURES

A pseuedo linear admittance G(I) for each valve port is calculated from the pressure across the port and the flow through it.

G(I) = DELTAP/Q(I)

with all the valve admittances evaluated, the circuit shown in Figure 6.22-3 is solved using a matrix method to obtain the pressures at P(I).

The calculated pressures are then used to obtain updated values of Q(I) and the iteration is repeated until the changes in flows becomes negligible.

6.22.2 Assumptions

The technique used to solve the group of equations does not have any significant problems but may have some stability problems when the cavitation model is added.

The pseudo linearization of the orifice coefficients is valid since it is updated by the latest flow guess.

6.22.3 Computations

1000 Section

A test is made to determine if a port is not dimensioned, which can happen if it is a 3-way or 2-way valve. If the area is zero or XT is zero, XT is set to .0001 to prevent the computation blowing up when XT is used in the denominator.

There is no need for a 2000 section since it is not necessary to initialize any parameters.

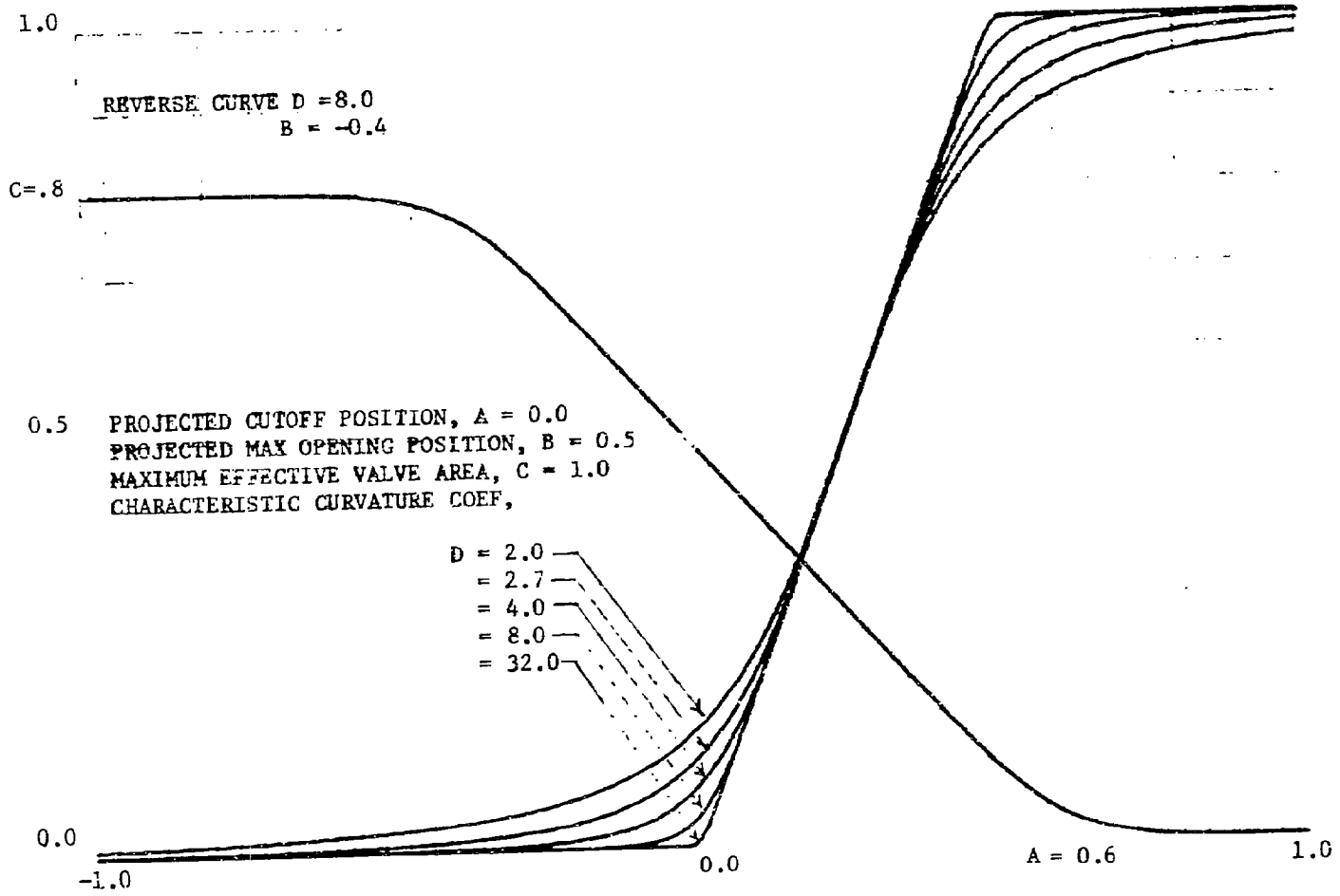



FIGURE 6.22-3
EFFECTIVE VALVE AREA CHARACTERISTICS

1500 SECTION

The steady state section is straight forward, the valve impedance is added to the leg PQLEG (INEL,8) for each call to a particular connection, and the pressure drop across the particular valve port is subtracted from PQLEG (INEL, 11).

Note: When some parts are closed during steady state, it is not essential to have them in a leg as an element. If the connection is not called it will only add a small transient flow when the transient section is started.

3000 SECTION

Figure 6.22-2 shows a simplified flow diagram of the 3000 section.

The first computation is a call to INTERP, to find the valve position.

In the call statement D(L(5)) is the start of the time table, D(L(6)) is the start of the valve position table and L(7) is the number of tabulated values in each of the tables, a first order interpolation is used. XV the valve position at time T is returned.

In the next section a DO loop is used to calculate XT and the value of VA( ).

In the next section QNEW, the new flows, are calculated using the old pressures which were loaded into CALC2( ).

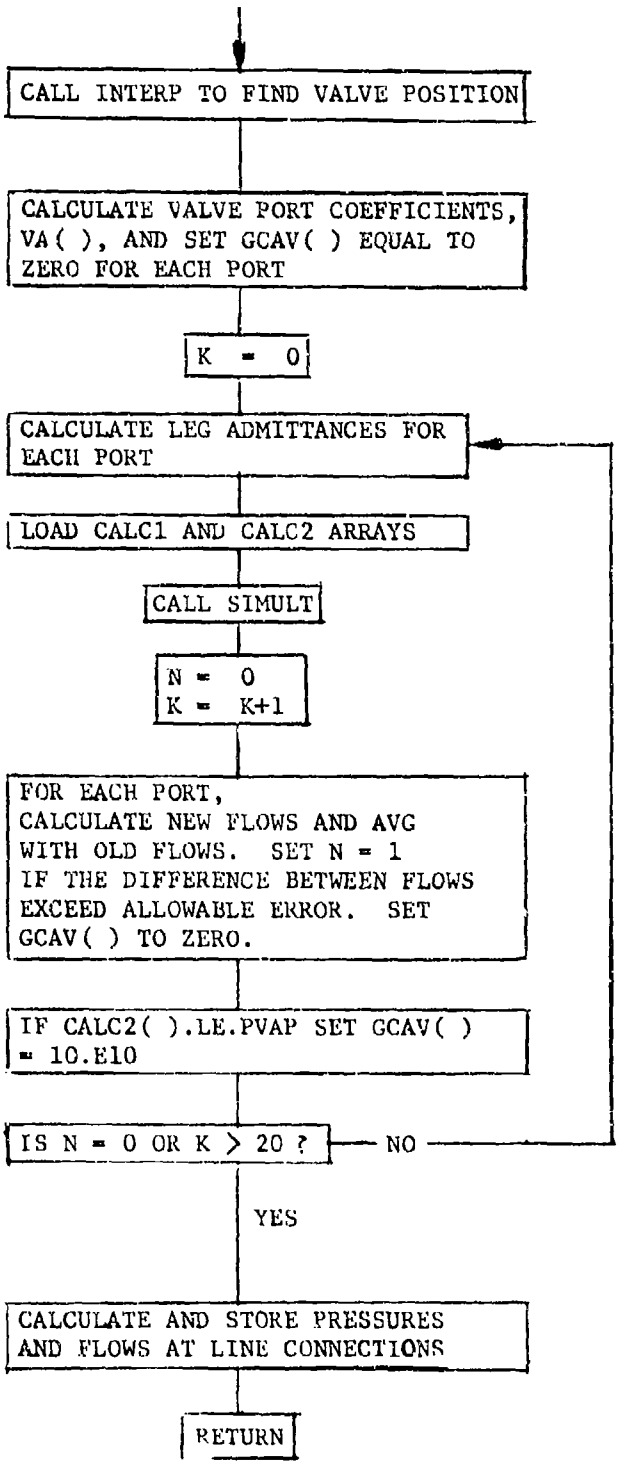
The calculated flow is compared with the previous flow value, if the absolute value of the difference exceeds .01, an error indicator N is set to 1.

Using an average of the new and old flow, G( ) is calculated.

A test is then made for too many iterations or if N = 0, if so the calculation is considered complete. The iteration limit is not normally needed.

If both tests fail, the next section loads the CALC1( ) and CALC2( )

3000 SECTION




VALV22 FLOW DIAGRAM
FIGURE 6.22-4

arrays and calls SIMULT to obtain new values for the pressures at the valve ports and restarts the calculation. The last section stores the flows and pressures at the line junctions before returning.

6.22.4 Approximations - Not applicable.

6.22.5 Limitations

The current limitation of VALV22 is the possible need for a variable orifice coefficient, particularly in the overlap region.

An undesireable feature is the need for up to four nodes at the junctions with the lines when all the parts have a significant flow.

The extra four nodes require a significant amount of computation and extra storage space in the CALC1( ) array of SSDATA.

6.22.6 Variable Names

| NAME | DESCRIPTION | DIMENSIONS |
|---|---|---|
| CALC1( ) | Solution array | |
| CALC2( ) | Solution array | |
| G( ) | Leg admittances | CIS/PST |
| GCAV( ) | Pseudo Cavity Impedance | |
| I | Integer counter | |
| K | Integer counter | |
| LEI | Data address in PARME | |
| N | Integer counter | |

6.22.6 Variable Names (cont'd)

| NAME | DESCRIPTION | DIMENSIONS |
|---|---|---|
| N1 | | |
| N2 | Address variables which are rolled during calculation | |
| N3 | | |
| N4 | | |
| N5 | | |
| PDELTA | Leg pressure drop | PSI |
| QCAV( ) | Cavitation flow (not used) | CIS |
| QNEW | Leg flow | CIS |
| VA( ) | Valve orifice impedance array | $CIS/PSI^{1/2}$ |
| XT | valve position | IN |
| XV | Temporary variable | |

## 6.22.7 Subroutine Listing

```
      SUBROUTINE VALV22 (D,DT,DD,L)
C**** REVISED AUGUST 5, 1975 ****
      DOUBLE PRECISION DD
      COMMON NTELPL,NTOLPL,IPT,IPOINT,NPTS,INEL,KNEL,NTOPL,NLPLT(61,3),
     1 PQLEG(90,12),LCS(90,10),ILEG(1400),PN(90),QN(90)
      COMMON/SUB/PARM(150,9),PM(1500),QM(1500),P(300),Q(300),C(300)
     1,Z(300),RHO(20),S2ORHO(20),VISC(20),BULK(20),TEMP(20),PVA2(20)
     2,ATPRES,T,DELT,TFINAL,PLTDEL,PI,TITLE(20),LEGM,ICON
     3,KTEMP(99),LSTART(150),NLPT(150),LTYPE(99),NC(99),INX,INZ
     4,INV,ISTEP,NLINE,NEL,IND,IENTR,MNLINE,MNEL,MNLEG,MNNODE,MNPLOT
     5,MNLPTS,NDS
      DIMENSION CALC1(4,4),CALC2(4),VA(4),G(4),QCAV(4),GCAV(4)
      DIMENSION D(1),DT(1),DD(1),L(1)
      DATA N1,N2,N3,N4/1,2,3,4/
      DATA CALC2,QCAV/8*0.0/
      IF(IENTR)1000,2000,3000
C *** INITIALIZE COMPONENT CONSTANTS
 1000 CONTINUE
      IF (INEL.NE.0) GO TO 1500
      LEI=((L(11)+7)/8)*8
      L(6)=17+LEI
      L(5)=17
      L(7)=L(11)
      XV=D(L(6))
C
C     SET UP VALVE CONSTANTS FOR STEADY STATE
C
      DO 1004 I=1,4
      N=4*I-3
      D(N+2)=D(N+2)*S2ORHO(KTEMP(IND))*.65
      XT=D(N+1)-D(N)
      D(N)=(D(N)+D(N+1))/2.
      IF(D(N+2).LE.0.0) D(N+2)=.00001
      IF(XT.EQ.0.0) XT=.0001
      D(N+1)=XT
      XT=(XV-D(N))/D(N+1)
      IF(D(N+3).EQ.0.0)D(N+3)=2.0
      DT(I+4)=D(N+2)*(.5+XT/(1.+ABS(XT*2.)**D(N+3))
     1 **(1./D(N+3)))+.0001
 1004 DT(I)=0.0
      RETURN
C
C****STEADY STATE SECTION ****
C
 1500 CONTINUE
      PDELTA=PQLEG(INEL,2)*(PQLEG(INEL,1)/DT(KNEL+4))**2
      PQLEG(INEL,11)=PQLEG(INEL,11)-PDELTA
      PQLEG(INEL,8)=PQLEG(INEL,8)+1.0/DT(KNEL+1)**2
      DT(KNEL)=PQLEG(INEL,1)*PQLEG(INEL,2)
      RETURN
```

```
C
 2000 DO 2002 I=5,8
 2002 DT(I)=0.0
      RETURN
C
C ***  TRANSIENT CALCULATION SECTION
C
 3000 CONTINUE
C *** CALL INTERP TO FIND VALVE POSITION
      CALL INTERP(T,D(L(5)),D(L(6)),10,L(7),XV,N)
C
C *** SET UP VALVE AREAS AND GET VALUES FOR C AND Z
C
      DO 3100 I=1,4
      N=4*I-3
      XT=(XV-D(N))/D(N+1)
      VA(I)=D(N+2)*(.5+XT/(1+ABS(XT*2.)**D(N+3))
     1 **(1.0/D(N+3)))+.00001
      GCAV(I)=0.0
 3100 CONTINUE
      K=0
      GO TO 3240
 3150 N=0
      K=K+1
C
C *** CALCULATE NEW FLOWS AND SET COUNTER FOR TO LARGE AN ERROR
C ***  CALCULATE VALUES FOR ORIFICIES ADMITTANCES
C
      DO 3200 I=1,4
      N5=I+1
      IF(N5.EQ.5) N5=1
      QNEW=VA(I)*SQRABS(CALC2(I)-CALC2(N5))
      DT(I)=DT(I)*.3+QNEW*.7
      IF(ABS(DT(I)-QNEW).GT.0.001/(ABS(QNEW)+1.0)) N=1
      GCAV(I)=0.0
      IF (CALC2(I).LE.PVAP(KTEMP(IND)).OR.DT(I+4).GT.0.0) GCAV(I)=10.E3
 3200 CONTINUE
C
C *** CHECK ERROR INDICATOR OR TOO MANY ITERATIONS
C
      IF(N.EQ.0.OR.K.GT.20) GO TO 3400
 3240 DO 3250 I=1,4
 3250 G(I)=VA(I)**2/(ABS(DT(I)+.00001))
C
C *** LOAD CALC1 AND 2 ARRAYS
C
      DO 3300 I=1,4
      IF (DT(I+4).GT.0.0) GCAV(I)=10.E3
      CALC1(I,N1)=1.0/Z(L(I))+G(N4)+G(N1)+GCAV(I)
      CALC1(I,N2)=-G(N1)
```

**6.22.7 (Continued)**

```
      CALC1(I,N3)=0.0
      CALC1(I,N4)=-G(N4)
      CALC2(I)=C(L(I))/Z(L(I))+PVAP(KTEMP(IND))*GCAV(I)
      N5=N1
      N1=N2
      N2=N3
      N3=N4
      N4=N5
 3300 CONTINUE
C
C *** CALL SIMULT TO CALCULATE VALVE PORT PRESSURES
C
      CALL SIMULT(CALC1,CALC2,4,N5)
      GO TO 3150
 3400 CONTINUE
C
C ***  RETURN PRESSURES AND FLOWS TO THE LINE ADRESSES
C
      DO 3500 I=1,4
      LI=L(I)
      P(LI)=CALC2(I)
      Q(LI)=(C(LI)-CALC2(I))/Z(LI)
      VAP=PVAP(KTEMP(IND))
      IF(P(LI).GT.VAP.AND.DT(I+4).LE.0.0) GO TO 3500
      N5=I+1
      IF(N5.EQ.5) N5=1
      DT(I+4)=DT(I+4)-Q(LI)-DT(I)+DT(N5)
      IF(DT(I+4).LT.0.0) DT(I+4)=0.0
C3450 WRITE(6,2) K,Q(LI),CALC2(I),C(LI),Z(LI),DT(I)
C   2 FORMAT(1X,I5,5E12.5)
 3500 CONTINUE
      RETURN
      END
```

## 6.31 SUBROUTINE CVAL31

Subroutine CVAL31 models a simple undamped check valve shown in Figure 6.31-1, which is typical of the many check valves in use in industry and on aircraft. Although the actual mechanical configurations vary greatly the basic method of operation stays about the same.

The subroutine is not limited, and can be used for any number of check valves, within the limits of the common storage.

### 6.31.1 Math Model

The check valve is assumed to have a variable orifice characteristic between the fully open and fully closed positions.

Reverse flow can take place transiently until the valve closes.

No effort has been made to include the effects of flow forces on the poppet since these are not very well defined theoretically, and depend upon the actual valve geometry.

The model used to calculate the steady state pressure drop assumes a straight line flow/pressure drop characteristic between the cracking pressure and the fully open position. The cracking pressure drop is set equal to the inlet area divided by the spring preload and the slope, DT(5), is set to the change in pressure required to fully open the poppet divided by the flow at that condition which is

DT(4) = D(1)*CV*SQRT(DT(2))*S2ORHO

where D(1) is considered to be the maximum valve area.

The orifice resistance at the fully open position DT(10), is used when the flow exceeds DT(4). Figure 6.31-2 shows graphically how this is done.

In the transient analysis the flow through the valve is calculated using

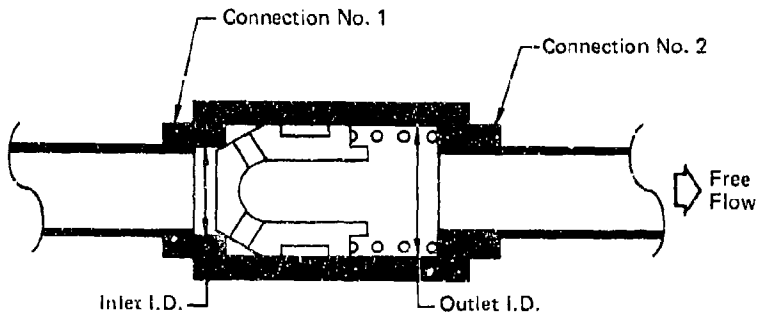



**FIGURE 6.31-1**
**TYPE NO. 31 CHECK VALVE**

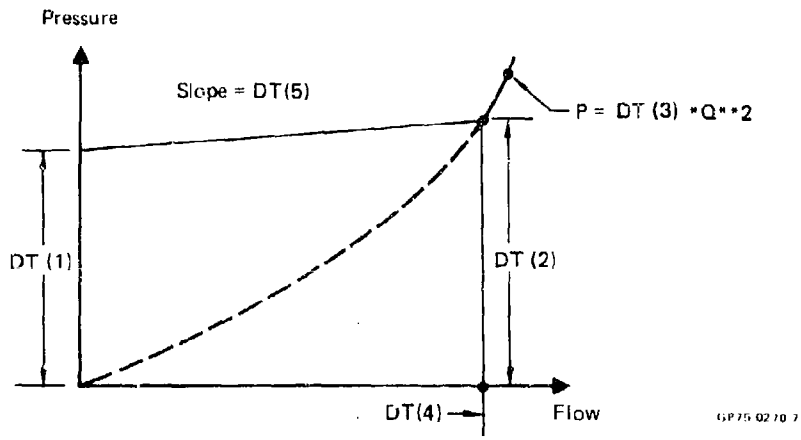



**FIGURE 6.31-2**
**CVAL 31 STEADY STATE PRESSURE DROP CHARACTERISTICS**

the normal valve equations, with the valve orifice area being proportional to the valve displacement.

The valve position is calculated using an iterative procedure.

The flow is first calculated using the last position of the valve, and the latest values of the line characteristics.

QA = SIGN(CA*(-1+SQRT(1 + CB*ABS(CDIF))),CDIF)

Where: CDIF = C(L(1))-C(L(2))

CA = DT(1) * XNEW**2

CB = DT(2)/XNEW**2

DT(1)=CA*(Z(L(1))+Z(L(2))).

DT(2) = 2.0/(CA*(Z(L/1))+Z(L/2)))**2 where the temporary variable CA is

CA = (CV*D(1)/D(5))**2/RHO

With Q known the two pressures can be calculated:

P1 = C(L(1)) - QA*Z(L(1))

P2 = C(L(2)) + QA*Z(L(2))

These values of pressure at either side of the valve are then used to calculate the force on the valve poppet.

F = (P1 - P2) *D(1)-D(6)-D(4)*XNEW-.05*VNEW

Where: D(1) = Poppet area

D(4) = Spring rate

D(6) = Spring preload

The .05 term is a damping factor.

The poppet acceleration:

ANEW = F/D(3)

= FORCE/Poppet mass

The velocity and position:

VNEW = VLST + (ALST + ANEW)/2 * DELT
XNEW = XLST + (VLST + VNEW)/2 * DELT

XNEW, the new calculated position, has to be tested for position limits. A lower limit of 0.00001 inches is used to allow some leakage and to prevent division by zero in the computation.

6.31.2 Limitations

The model does not account for displacement flow due to poppet motion, for the variations in orifice characteristics with poppet position, for secondary pressure drops due to other flow restrictions and for flow forces on the poppet.

The simple integration method which was chosen to save computation time and cost will not give very good results for very fast transients but since the poppet will normally be fully open or closed for the majority of the time, the above limitations were considered to be acceptable for a component that could be used many times in a system.

6.31.3 Computations

1000 Section - In the 1000 section, the steady state valve characteristics are calculated, ready for use in the steady state entry, since simple characteristics are assumed. The computation mainly consists of calculating and storing the temporary variables.

1500 Section - This section is called from LEGCAL via COMPE using either CON #1, if the check valve is connected so that the free flow direction is the same as the positive flow in the leg, or CON #2 if the valve is in backwards.

When the valve is closed,

IENTR = 1 & QS = -1 or IENTR = 2 QS = 1.

The valve impedance is set at 1.0E8, which is essentially open circuit.

When the valve is fully open (IENTR = 1, QS = 1, or IENTR = 2, QS = -1) plus Q > DT(4), the valve orifice impedance DT(10) is added into the Q**2 term of LEGCAL formulae.

With the same basic conditions but with Q > DT(4) the valve characteristics are assumed to be a constant pressure differential, plus a linear flow/pressure gain.

When the flow guess is negative for CON #2 the constant differential becomes a pressure rise.

The three modes of the check valve, closed, partially open and fully open will show up in the leg constants particularly when a negative flow guess is tried, there will be a sudden increase in the formulae 'Q' constant.

2000 Section - In the 2000 section new variables are calculated for use in the transient section in order to cut down on running time.

3000 Section - The calculations for Q, P1 and P2, and PISTON acceleration follow the theory discussed in 6.31.2 above.

An iterative procedure is used to improve the calculation using an iteration limit of six, or an error of less than .001 inches between successive values of XNEW, to terminate the iteration.

Under normal operation, the valve will be either fully open or fully closed, so that there will be no iterations.

The calculated value of XNEW is checked and if it is at a stroke limit, and if the calculated acceleration or velocity are in the limit direction they are set to zero.

These limits are not applied when the acceleration or velocity are away from the end of stroke.

6.31.4 Assumptions

The assumptions made are associated with the valve orifice calculations and the lack of flow force effects. These assumptions speed the computation but can be a source of inaccuracy.

6.31.5 Limitations

The main limitation is the lack of a cavitation simulation which can cause large inaccuracies when the pressures drop below vapor pressure.

6.31.6 Variable Names

| Name | Description | Dimension |
|---|---|---|
| ALST | Last poppet acceleration | IN/SEC**2 |
| ANEW | Latest poppet acceleration | IN/SEC**2 |
| CA | Orifice variable | --- |
| CB | Orifice variable | -- |
| CDIF | Characteristic difference | PSI |
| CV | Discharge coefficient | |

| Name | Description | Dimension |
|---|---|---|
| F | Force on valve poppet | LBS |
| MM | Iteration DO loop counter | |
| P1 | Con #1 pressure | PSI |
| P2 | Con #2 pressure | PSI |
| QA | Valve flow | CIS |
| QS | Leg flow sign | |
| VLST | Last poppet velocity | IN/SEC |
| VNEW | New poppet velocity | IN/SEC |
| XLST | Last poppet position | IN |
| XNEW | New poppet position | IN |
| XPREV | Iteration poppet position | IN |

## 6.31.7 Subroutine Listing

```
      SUBROUTINE CVAL31 (D,DT,DD,L)
C**** REVISED AUGUST 5, 1975 ****
      DIMENSION D(1),DT(1),DD(1),L(1)
      DOUBLE PRECISION DD
      COMMON NTELPL,NTOLPL,IPT,IPOINT,NPTS,INEL,KNEL,NTOPL,NLPLT(61,3),
     1 PQLEG(90,12),LCS(90,10),ILEG(1400),PN(90),QN(90)
      COMMON/SUB/PARM(150,9),PM(1500),QM(1500),P(300),Q(300),C(300)
     1,Z(300),RHO(20),S2ORHO(20),VISC(20),BULK(20),TEMP(20),PVAP(20)
     2,ATPRES,T,DELT,TFINAL,PLTDEL,PI,TITLE(20),LEGN,ICON
     3,KTEMP(99),LSTART(150),NLPT(150),LTYPE(99),NC(99),INX,INZ
     4,INV,ISTEP,NLINE,NEL,IND,IENTR,MNLINE,MNEL,MNLEG,MNNODE,MNPLOT
     5,MNLPTS,MDS
C     D(1)=INTERNAL DIAMETER (INLET)
C     D(2)=INTERNAL DIAMETER (OUTLET) NOT IN USE
C     D(3)=POPPET MASS
C     D(4)=SPRING CONSTANT
C     D(5)=MAX POPPET DISPLACEMENT
C     D(6)=SPRING PRELOAD
C     DT(1)=CONSTANT STORAGE
C     DT(2)=CONSTANT STORAGE
C     DT(3)= STORAGE FOR FULLY OPEN ORIFICE COEF.
C     DT(4)= STORAGE FOR PREVIOUS POPPET VELOCITY
C     DT(5)=STORAGE FOR PREVIOUS POPPET ACCELERATION
C     DT(6)= STORAGE FOR PREVIOUS POPPET POSITION
      IF(IENTR) 1000,2000,3000
 1000 CONTINUE
      IF(INEL)1020,1010,1500
 1010 CV=.65
      D(1)=D(1)**2.*PI/4.
 1020 DT(1)=D(6)/D(1)
      DT(2)=D(4)*D(5)/D(1)+DT(1)
      DT(3)=(RHO(KTEMP(IND))/2.0)/(D(1)*CV)**2
      DT(4)=D(1)*CV*SQRT(DT(2))*S2ORHO(KTEMP(IND))
      DT(5)=(DT(2)-DT(1))/DT(4)
      RETURN
C      SECTION FOR STEADY STATE CALCULATION
 1500 CONTINUE
      QA=PQLEG(INEL,1)
      QS=PQLEG(INEL,2)
      LCS(INEL,6)=QS
      IF(L(3).EQ.1.) GO TO 1600
C     THE VALVE IS CONNECTED CONVENTIONALLY
      IF(L(3).NE.0)GO TO 1000
      IF(QS.EQ.1.) GO TO 1700
      GO TO 1650
C     THE VALVE IS CONNECTED BACKWARDS
 1600 IF(L(3).NE.1)GO TO 1000
      IF(QS.LT.1.0) GO TO 1700
C     THE VALVE IS CLOSED
 1650 PQLEG(INEL,6)=1.0E6+PQLEG(INEL,6)
```

```
      PQLEG(INEL,11)=PQLEG(INEL,11)-QA*QS*1.0E6
      RETURN
 1700 IF (QA.LE.DT(4)) GO TO 1800
C     THE VALVE IS FULLY OPEN
      IF(PQLEG(INEL,3).LT.DT(4)) PQLEG(INEL,3)=DT(4)
      DT(6)=D(5)
      PQLEG(INEL,8)=DT(3)+PQLEG(INEL,8)
      PQLEG(INEL,11)=PQLEG(INEL,11)-QS*QA**2*DT(3)
      RETURN
C     THE FLOW IS LESS THAN THE 'FULL OPEN FLOW'
 1800 PQLEG(INEL,11)=PQLEG(INEL,11)-DT(1)*QS
     1 -QA*QS*DT(5)
      DT(6)=((DT(5)*QA)/DT(2))*D(5)
      IF(PQLEG(INEL,4).GT.DT(4)) PQLEG(INEL,4)=DT(4)
      PQLEG(INEL,5)=PQLEG(INEL,5)-DT(1)*QS
      PQLEG(INEL,6)=PQLEG(INEL,6)+DT(5)
      RETURN
C*****ILLEGAL INPUT DATA TERMINATES PROGRAM*****
 1900 WRITE(6,9980)
 9980 FORMAT(10X,33HPROGRAM STOP IN SUBROUTINE CVAL31)
      STOP 1531
 2000 CONTINUE
      D(2)=D(1)/(D(5)/(1.0+D(5)))
      CA=(CV*D(1)/D(5))**2/RHO(KTYP(IND))
      DT(1)=CA*(Z(L(1))+Z(L(2)))
      DT(2)=2.0/(CA*(Z(L(1))+Z(L(2)))**2)
      DT(4)=0.
      DT(5)=0.
      RETURN
 3000 CONTINUE
      CDIF=C(L(1))-C(L(2))
      XLST=DT(6)
      VLST=DT(4)
      ALST=DT(5)
      XNEW=XLST
      VNEW=VLST
      DO 3100 MM=1,6
      XPREV=XNEW
      IF(XNEW.LE.0.0) GO TO 3050
      XSQ=(XNEW+.0001)**2
      CB=DT(2)/XSQ
      CA=DT(1)*XSQ
      QA=SIGN(CA*(-1+SQRT(1+CB*ABS(CDIF))),CDIF)
      GO TO 3060
 3050 QA=0.0
 3060 P1=C(L(1))-QA*Z(L(1))
      P2=C(L(2))+QA*Z(L(2))
      F=(P1-P2)*D(1)-D(6)-D(4)*XNEW-.05*VNEW
C     F1=.43*(P1-P2)*D(2)*XLST/(1.+XLST)
C     F1=2.*CV*ABS(P1-P2)*D(2)*XLST/(1.+XLST)
```

**6.31.7 (Continued)**

```
C       F=F-F1
        ANEW=F/D(3)
        VNEW=VLST+(ANEW+ALST)/2*DELT
        XNEW=XLST+(VNEW+VLST)/2*DELT
        IF(XNEW.LT.0.00001) XNEW=0.0
        IF(XNEW.GT.D(5)) XNEW=D(5)
        IF(ABS(XNEW-XPREV).LT.0.001) GO TO 3150
 3100 CONTINUE
C       WRITE(6,999)P1,P2,F,QA,Z(L(1)),Z(L(2)),C(L(1)),C(L(2))
 999   FORMAT(10X,8F12.4)
 3150 IF(XNEW.GT.0.0) GO TO 3200
        IF(ANEW.LT.0.0) DT(5)=0.0
        IF(VNEW.LT.0.0) DT(4)=0.0
        GO TO 3500
 3200 IF(XNEW.LT.D(5)) GO TO 3400
        IF(ANEW.GT.0.0) DT(5)=0.0
        IF(VNEW.GT.0.0) DT(4)=0.0
        GO TO 3500
 3400 DT(4)=VNEW
        DT(5)=ANEW
 3500 DT(6)=XNEW
        P(L(1))=P1
        P(L(2))=P2
        Q(L(1))=QA
        Q(L(2))=-QA
        RETURN
        END
```

## 6.32 SUBROUTINE CREL32

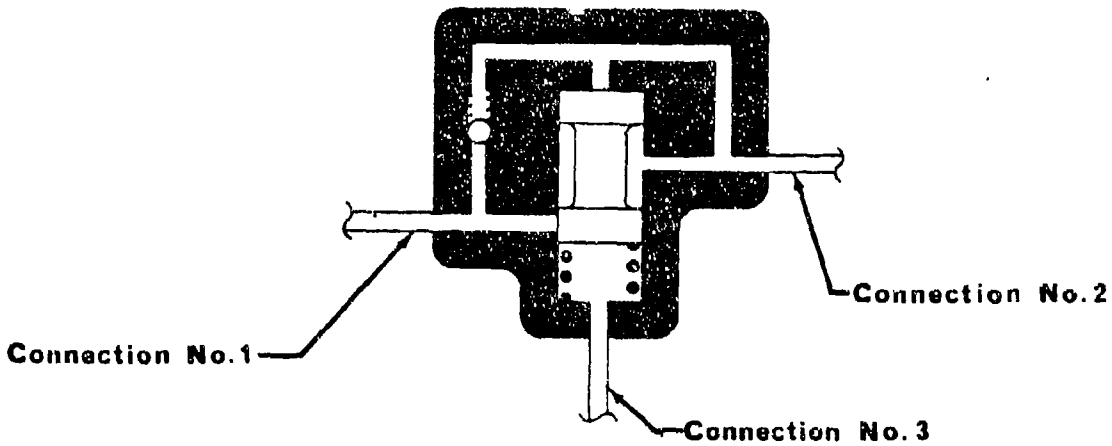



**FIGURE 6.32-1**
**TYPE NO. 32 PRIORITY VALVE**

Subroutine CREL32 models a priority valve shown in Figure 6.32-1. The subroutine models the priority valve as a combination check and relief valve. The check and relief valve are considered to be arranged in parallel between connection #1 and #2. The relief valve cracking pressure is referenced between connections #2 and #3. When the relief valve is closed, flow is allowed in only one direction from connection #1 to connection #2 through the check valve which prevents flow in the opposite direction. When the relief valve is open, flow is allowed in both directions between connections #1 and #2.

6.32.1 Math Model

The priority valve has several modes of operation depending upon the pressure differential between connections #2 and #3 and connections #1 and #2.

1) P2 - P3 > relief valve cracking pressure

The relief valve is open and flow is allowed in either direction between connections #1 and #2.

2) P2 - P3 > relief valve cracking pressure and P2 - P3 > relief valve reseat pressure.

If the relief valve is open, flow is allowed in both directions.

If the relief valve is closed and P1-P2> check valve cracking pressure there is leakage flow through the relief valve.

If the relief valve is closed and P1-P2> check valve cracking pressure there is flow through the check valve from connection #1 to connection #2.

3) P2-P3 < relief valve reseat pressure

P1-P2 < check valve cracking pressure, the relief valve and check valve are closed and there is leakage flow through the relief valve.

P1-P2 > check valve cracking pressure, the relief valve is closed and there is flow through the check valve from connection #1 to connection #2.

For steady state calculations the relief valve is assumed to be open. A linear relationship between flow and pressure is assumed to calculate the pressure drop through the valve.

PQLEG(INEL,6) = D(TZREL) + PQLEG(INEL,6)

Q1 = PQLEG(INEL,1)

PQLEG(INEL,11) = PQLEG(INEL,11) - D(TZREL)*Q1*PQLEG(INEL,2)

where, PQLEG(INEL,6) is the total impedance of the leg.

D(TZREL) is the impedance through the relief valve

QI and PQLEG(INEL,1) is the flow through the leg.

PQLEG(INEL,11) is the upstream pressure of the leg.

In the transient calculations the pressure differential between connections #2 and #3, and #1 and #2 are compared with given cracking and reseat pressures for the relief valve and the check valve to determine which of the flow conditions described above exist at that particular time step. Once the flow condition is established an impedance based on this condition is selected and used to calculate the pressures and flows at connections #1, #2 and #3.

$$Q(L1) = (C(L1) - C(L2))/(Z(L1) + Z(L2) + ZI)$$

$$Q(L2) = -Q(L1)$$

$$Q(L3) = 0.0$$

$$P(L1) = C(L1) - Q(L1) * Z(L1)$$

$$P(L2) = C(L2) - Q(L2) * Z(L2)$$

$$P(L3) = C(C3)$$

where:

Q(L1), Q(L2), Q(L3) are the flows at connections 1, 2 and 3,

Z(L1) and Z(L2) are the impedance at connections 1 and 2,

ZI is the impedance of the valve.

P(L1), P(L2) and P(L3) are the pressures at connections 1, 2 and 3.

6.32.2 Assumptions

The relief valve is assumed to be open during the steady state calculations to allow the flows and pressures to balance with the rest of the system.

The check valve is assumed to open and close instantaneously.

The reseat pressure for the check valve is assumed to be equal to the cracking pressure.

6.32.3 Computations

1500 Section - In the 1500 section the impedance for the valve is added to the impedance for the leg, PQLEG (INEL,6). A new upstream pressure PQLEG,(INEL,11), is calculated using the impedance of the open relief valve.

3000 Section - In the 3000 section an impedance for flow through the valve is selected depending upon the pressure differentials between connections #2 and #3 and connections #1 and #2. This impedance is used to calculate the flows at pressures at each of the connections.

6.32.4 Approximations

None

6.32.5 Limitations

The CREL32 subroutine does not include the effects of inertia or damping on the valve components. Therefore, use of the CREL32 subroutine where these effects may be significant may give erroneous results.

6.32.6 Variable Names

| Variable | Description | Units |
|---|---|---|
| D(CLOS) | Relief Valve Reseat Pressure | PSI |
| DELP1 | Pressure Drop Across Connect. 2 and 3 | PSI |
| DELP2 | Pressure Drop Across Connect. 1 and 2 | PSI |
| D(IPCH) | Check Valve Cracking Pressure | PSI |
| D(IPREL) | Relief Valve Cracking Pressure | PSI |
| D(IZCH) | Check Valve Impedance | PSI/CIS |

| Variable | Description | Units |
|---|---|---|
| D(IZLK) | Relief Valve Leakage Impedance | PSI/CIS |
| D(IZREL) | Relief Valve Impedance | PSI/CIS |
| L1 | Connection No. 1 | - |
| L2 | Connection No. 2 | - |
| L3 | Connection No. 3 | - |
| NPOSCK | Temporary Variable | - |
| NPOS1 | Temporary Variable | - |
| NPOS2 | Temporary Variable | - |
| DT(POSO) | Valve Position Indicator | - |
| DT(P1) | Pressure at connection #1 | PSI |
| DT(P2) | Pressure at connection #2 | PSI |
| DT(P3) | Pressure at connection #3 | PSI |
| ZI | Temporary Variable | - |
| ZV | Impedance of the Valve | PSI/CIS |

## 6.32.7 Subroutine Listing

```
      SUBROUTINE CREL32 (D,DT,DD,L)
C *** REVISED AUGUST 5,1975 ***
      DOUBLE PRECISION DD
      DIMENSION D(6),DT(5),DD(1),L(4)
      COMMON NTELPL,NTOLPL,IPT,IPOINT,NPTS,INEL,KNEL,NTOPL,NLPLT(61,3),
     1 PQLEG(90,12),LCS(90,10),ILEG(1400),PN(90),QN(90),PLX(90)
      COMMON/SUB/PARM(150,9),PM(1500),QM(1500),P(300),Q(300),C(300)
     1,Z(300),RHO(20),S2ORHO(20),VISC(20),BULK(20),TEMP(20),PVAP(20)
     2,ATPRES,T,DELT,TFINAL,PLTDEL,PI,TITLE(20),LEGN,ICON
     3,KTEMP(99),LSTART(150),NLPT(150),LTYPE(99),NC(99),INX,INZ
     4,INV,ISTEP,NLINE,NEL,IND,IENTR,ANLINE,ANEL,ANLEG,ANNODE,ANPLOT
     5,ANLPTS,ADS
      INTEGER POSO,CLOS,P1,P2,P3,ZV
      DATA IPREL/1/,CLOS/2/,IZREL/3/,IZCH/4/,IZLK/5/,IPCH/6/,POSO/1/
     1 ,P1/2/,P2/3/,P3/4/,ZV/5/
      IF(IENTR) 1000,2000,3000
 1000 CONTINUE
      DT(POSO)=1.0
      IF (INEL.NE.0) GO TO 1500
      IF(L(4).EQ.0) GO TO 1060
      IF(INV) 1030,1050,1040
 1030 IF(INV.EQ.-L(4)) GO TO 1045
      GO TO 1050
 1040 IF(INV.EQ.L(4)) GO TO 1050
 1045 L(4)=1
      GO TO 1060
 1050 L(4)=0
 1060 CONTINUE
      RETURN
C
C     STEADY STATE SECTION
 1500 IF (KNEL-2) 1600,1700,1700
 1600 DIMP=D(IZREL)
      IF(L(4).EQ.0) GO TO 1610
      DIMP=D(IZLK)
      DT(POSO)=-1.0
      N=LCS(INEL,3)
      PLX(N)=20.
      QN(N)=D(CLOS)*20.-PQLEG(INEL,1)*PQLEG(INEL,2)
      LCS(INEL,7)=5
 1610 PQLEG(INEL,6)=DIMP+PQLEG(INEL,6)
      DI=PQLEG(INEL,1)
      DT(P1)=PQLEG(INEL,11)
      PQLEG(INEL,11)=PQLEG(INEL,11)-DIMP*DI*PQLEG(INEL,2)
      DT(P2)=PQLEG(INEL,11)
      DT(ZV)=DIMP
      LCS(INEL,7)=5
 1700 RETURN
 2000 CONTINUE
      RETURN
 3000 CONTINUE
```

6.32.7 (Continued)

```
      L1=L(1)
      L2=L(2)
      L3=L(3)
      Q(L3)=0.0
      P(L3)=C(L3)
      DT(P3)=P(L3)
      DELP1=DT(P2)-DT(P3)
      DELP2=DT(P1)-DT(P2)
      NPOS1=DELP1/D(IPREL)
      NPOS2=DELP1/D(CLOS)
      NPOSCK=DELP2/D(IPCH)
      IF(DT(POSO))3100,3200,3300
 3100 IF(NPOS1)3200,3130,3120
 3120 ZI=SQRT(DT(ZV))
      DT(POSO)=1.0
      GO TO 3400
 3130 DT(POSO)=-1.0
      IF(NPOSCK)3140,3140,3150
 3140 ZI=D(IZLK)
      GO TO 3400
 3150 ZI=D(IZCH)
      GO TO 3400
 3200 WRITE(6,999)
  999 FORMAT(10X,33HPROGRAM STOP IN SUBROUTINE CRLL32)

      STOP 32
 3300 IF (NPOS1)3200,3320,3310
 3310 ZI=D(IZRLL)
      DT(POSO)=1.0
      GO TO 3400
 3320 IF(NPOS2)3200,3340,3330
 3330 ZI=SQRT(DT(ZV))
      IF(ZI.LT.2.92)ZI=2.92
      DT(POSO)=1.0
      GO TO 3400
 3340 ZI=DT(ZV)**2.
      DT(POSO)=1.0
      IF(ZI.GT.1100.)DT(POSO)=-1.0
      GO TO 3400
 3400 Q(L1)=(C(L1)-C(L2))/(Z(L1)+Z(L2)+ZI)
      Q(L2)=-Q(L1)
      P(L1)=C(L1)-Q(L1)*Z(L1)
      P(L2)=C(L2)-Q(L2)*Z(L2)
      DT(P1)=P(L1)
      DT(P2)=P(L2)
      DT(P3)=P(L3)
      DT(ZV)=ZI
C     WRITE(6,4010)NPOS1,NPOS2,NPOSCK,DT(POSO),DELP1,DELP2
C4010 FORMAT(2X,6HNPOS1=,I4,2X,6HNPOS2=,I4,2X,7HNPOSCK=,I4,2X,3E12.4)
C     WRITE(6,4000) P(L1),Q(L1),P(L2),Q(L2),ZI,C(L1),C(L2),C(L3)
C4000 FORMAT(1X,8E12.4)
      RETURN
      END
```

## 6.33 SUBROUTINE CVAL33

Subroutine CVAL33 models a simple undamped one-way restrictor shown in Figure 6.33-1, which is typical of the many restrictors in use in industry and on aircraft. Although the actual mechanical configurations vary greatly, the basic method of operation stays about the same.

The subroutine is not limited, and can be used for any number of one-way restrictors, within the limits of the common storage.

### 6.33.1 Math Model

The one-way restrictor is assumed to have a variable orifice characteristic between the fully open and fully closed positions.

Reverse flow can take place transiently through the variable orifice when the valve is closing.

The model used to calculate the steady state pressure drop assumes a straight line flow/pressure drop characteristic between the cracking pressure and the fully open position. The cracking pressure drop is set equal to the inlet area divided by the spring preload. The slope, DT(DPSQ) equals the change in pressure required to fully open the poppet divided by the flow at that condition which is

DT(INITQ) = D(DIAIN)*CV*SQRT(DT(INITDP))*S2ORHO(KTEMP(IND))

where D(DIAIN) is considered to be the maximum valve area.

The orifice resistance at the fully open position DT(LDSA), is used when the flow exceeds DT(INITQ). Figure 6.33-2 shows graphically how this is done.

In the transient analysis the flow through the one-way restrictor is calculated using two orifices in parallel, with the valve orifice area being proportional to the valve displacement.

The valve position is calculated using an iterative procedure.

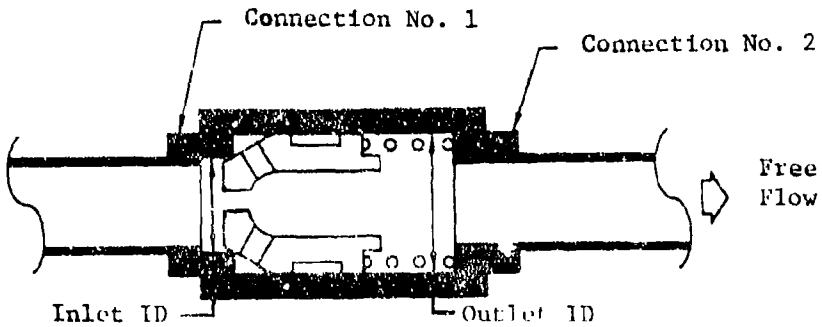



FIGURE 6.33-1

TYPE NO. 33 ONE-WAY RESTRICTOR

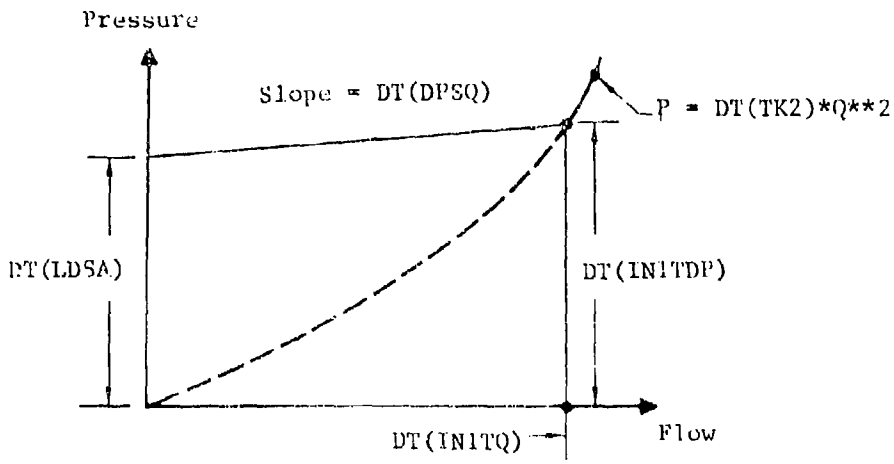



FIGURE 6.33-2

CVAL 33 STEADY STATE PRESSURE DROP CHARACTERISTICS
FOR AN OPEN POPPET

The flow is first calculated using the last position of the valve, and the latest values of the line characteristics.

```
DT(QT) = X*(-1. + SQRT(1. + ABS(Y))*SIGN(1.,Y)
```

Where: X = ((Z(L1) + Z(L2))*CKT**2.)/2.

Y = 4.*(C(L1) - C(L2))/(CKT*(Z(L1) + Z(L2)))**2

CKT is equal to the sum of the two orifice coefficients.

With Q known the two pressures can be calculated:

```
P1 = C(L(1)) - DT(QT)*Z(L(1))
P2 = C(L(2)) + DT(QT)*Z(L(2))
```

The values of pressure at either side of the valve are then used to calculate the force on the valve poppet.

```
F = (P1 - P2) *D(DIAIN) - D(LOAD) - D(SPRING)*XNEW - .05*VNEW
```

Where: D(SPRING) = Spring rate

D(LOAD) = Spring preload

The .05 term is a damping factor.

The axial flow forces on the poppet in the one-way restrictor are equated with the net change of momentum as

```
F1 = 2.*CV*ABS(P1-P2)*DT(ARFAC)
```

When: CV = Valve discharge coefficient (.65 assumed)

$P_1-P_2$ = Pressure drop across the poppet

DT(ARFAC) = Area factor dependent on poppet position

The total force on the poppet is then

```
F = F-F1
```

The poppet acceleration:

```
ANEW = F/D(MASS)
     = FORCE/Poppet mass
```

The velocity and position:

```
VNEW = VLST + (ALST + ANEW)/2 * DELT
XNEW = XLST + (VLST + VNEW)/2 * DELT
```

XNEW, the new calculated position, has to be tested for position limits. A lower limit of 0.001 inches is used to allow some leakage and to prevent division by zero in the computation.

6.33.2 Assumptions

The model does not account for displacement flow due to poppet motion, for the variations in orifice characteristics with poppet position and for secondary pressure drops due to other flow restrictions.

The simplest integration method, which was chosen to save computation time and cost, will not give very good results for very fast transients but since the poppet will normally be fully open or closed for the majority of the time, the above limitations were considered to be acceptable for a component that could be used many times in a system.

6.33.3 Computations

1000 Section - In the 1000 section, the steady state valve characteristics are calculated, ready for use in the steady state entry, since simple characteristics are assumed. The computation mainly consists of calculating and storing the temporary variables.

1500 Section - This section is called from LEGCAL via COMPE. If the one-way restrictor is connected so that the free flow direction is the same as the positive flow in the leg, L(3) equals zero. L(3) equals one if the valve is in backwards. When the valve is closed, the steady state pressure drop comes from the restrictor orifice.

When the one-way restrictor is fully open plus Q > DT(INITQ), the valve orifice impedance DT(TK2) is added into the Q**2 term of the LEGCAL formulae.

With the same basic conditions but with Q $\leq$ DT(INITQ) the valve characteristics are assumed to be a constant pressure differential, plus a linear flow/pressure gain.

When the flow guess is negative for CON #2 the constant differential becomes a pressure rise.

The three modes of the restrictor, closed, partially open and fully open will show up in the leg constants particularly when a negative flow guess is tried, there will be a sudden increase in the formulae 'Q' constant.

2000 Section - In the 2000 section new variables are calculated for use in the transient section in order to cut down on running time.

3000 Section - The calculations for DT(QT), P1 and P2, and poppet acceleration follow the theory discussed in 6.33.2 above.

An iterative procedure is used to improve the calculation. When an error of less than .001 inches between successive values of XNEW is reached the iteration is terminated.

Under normal operation, the one-way restrictor will be either fully open or fully closed, so that there will be no iterations.

The calculated value of XNEW is checked and if it is at a stroke limit, and if the calculated acceleration or velocity are in the limit direction they are set to zero.

These limits are not applied when the acceleration or velocity are away from the end of stroke.

6.33.4 Approximations

The approximations made are associated with the valve orifice calculations. These approximations speed the computation but can be a source of inaccuracy.

### 6.33.5 Limitations

The main limitation is the lack of a cavitation simulation which can cause large inaccuracies when the pressures drop below vapor pressure.

### 6.33.6 Variable Names

| Name | Description | Dimension |
|---|---|---|
| DT(ALST) | Last poppet acceleration | IN/SEC**2 |
| DT(ARFAC) | Area factor | IN**2 |
| ANEW | Latest poppet acceleration | IN/SEC**2 |
| D(LOEF) | Orifice discharge coefficient | - |
| CV | Discharge coefficient | - |
| D(DIAIN) | Inlet diameter | IN |
| D(DIAOT) | Outlet diameter | IN |
| D(DIAOR) | Orifice diameter | IN |
| DIS | Poppet displacement | IN |
| DT(DPSQ) | ΔP divided by flow | PSI/CIS |
| F | Force on valve poppet | LBS |
| FI | Axial flow force on poppet | LBS |
| DT(INITDP) | Initial valve ΔP | PSI |
| DT(INITQ) | Initial flow | CIS |
| DT(K1) | Orifice constant | - |
| DT(K2) | Orifice constant | - |
| DT(LDSA) | Spring preload divided by inlet area | PSI |
| D(LOAD) | Spring payload | LBS |
| D(MASS) | Poppet mass | $\frac{\text{LB-SEC**2}}{\text{IN}}$ |

| Name | Description | Dimension |
|---|---|---|
| D(POPDIS) | Max. poppet displacement | IN |
| P1 | Con #1 pressure | PSI |
| P2 | Con #2 pressure | PSI |
| QA | Valve flow | CIS |
| QS | Leg flow sign | - |
| DT(QT) | Total valve flow | CIS |
| D(SPRING) | Spring constant | LB/IN |
| DT(TK2) | Orifice constant | - |
| DT(VLST) | Last poppet velocity | IN/SEC |
| VNEW | New poppet velocity | IN/SEC |
| DT(XLST) | Last poppet position | IN |
| XNEW | New poppet position | IN |
| XPREV | Iteration poppet position | IN |

## 6.33.7 Subroutine Listing

```
      SUBROUTINE CVAL33 (D,DT,DD,L)
C**** REVISED AUGUST 5, 1975 ****
C     DOUBLE PRECISION DD
      DIMENSION D(1),DT(1),DD(1),L(1)
      COMMON NTELPL,NTOLPL,IPT,IPOINT,NPTS,INEL,KNEL,NTOPL,NLPLT(61,3),
     1 PQLEG(90,12),LCS(90,10),ILEG(1400),PN(90),QN(90)
      COMMON/SUB/PARM(150,9),PM(1500),QM(1500),P(300),Q(300),C(300)
     1,Z(300),RHO(20),S2ORHO(20),VISC(20),BULK(20),TEMP(20),PVAP(20)
     2,ATPRES,T,DELT,TFINAL,PLTDEL,PI,TITLE(20),LEGN,ICON
     3,KTEMP(99),LSTART(150),NLPT(150),LTYPE(99),NC(99),INX,INZ
     4,INV,ISTEP,NLINE,NEL,IND,IENTR,MNLINE,MNEL,MNLEG,MNMODE,MNPLOT
     5,MNLPTS,MDS
      INTEGER DIAIN,DIAOR,SPRING,POPDIS,COEF,DIAOT,DPSQ,AREAC,
     + TK2,QT,QOLD,VLST,ALST,XLST,DELP,SUB
C       **** D - ARRAY ****
      DATA DIAIN/1/,DIAOT/2/,MASS/3/,SPRING/4/,POPDIS/5/,
     + LOAD/6/,DIAOR/7/,COEF/8/
C      **** DT - ARRAY ****
      DATA TK2/1/,QT/2/,QOLD/3/,VLST/4/,ALST/5/,XLST/6/,DELP/7/,
     + K1/8/,K2/9/,INITDP/10/,DPSQ/11/,LOSA/12/,INITQ/13/,
     + AREAC/14/,SUB/15/
C      D(1)=INTERNAL DIAMETER (INLET)
C      D(2)=INTERNAL DIAMETER (OUTLET)
C      D(3)=POPPET MASS
C      D(4)=SPRING CONSTANT
C      D(5)=MAX POPPET DISPLACEMENT
C      D(6)=SPRING PRELOAD
C      D(7)=ORIFICE DIAMETER
C      D(8)=DISCHARGE COEFFICIENT
C      DT(1)= STORAGE FOR PREVIOUS POPPET VELOCITY
C      DT(5)=STORAGE FOR PREVIOUS POPPET ACCELERATION
C      DT(6)= STORAGE FOR PREVIOUS POPPET POSITION
      IF(IENTR) 1000,2000,3000
 1000 CONTINUE
      IF(INEL)1020,1010,1500
 1010 CONTINUE
      CV=.65
      D(DIAIN)=D(DIAIN)**2.*PI/4.
      D(DIAOT)=D(DIAOT)**2.*PI/4.
      D(DIAOR)=D(DIAOR)**2.*PI/4.
      DT(SUB)=D(DIAOT)-D(DIAOR)
 1020 DT(K1)=(2./RHO(KTEMP(IND)))**.5*D(DIAOR)*D(COEF)
      DT(K2)=(2./RHO(KTEMP(IND)))**.5*D(DIAIN)*CV
      DT(TK2)=(RHO(KTEMP(IND))/2.)/(D(DIAIN)*CV)**2
      DT(AREAC)=(RHO(KTEMP(IND))/2.)/(D(DIAOR)*D(COEF))**2
      DT(LOSA)=D(LOAD)/D(DIAIN)
      DT(INITDP)=D(SPRING)*D(POPDIS)/D(DIAIN)+DT(LOSA)
      DT(INITQ)=D(DIAIN)*CV*SQRT(DT(INITDP))*S2ORHO(KTEMP(IND))
      DT(DPSQ)=D(SPRING)*D(POPDIS)/(D(DIAIN)*DT(INITDP))
      RETURN
```

**6.33.7 (Continued)**

```
C       SECTION FOR STEADY STATE CALCULATION
 1500 CONTINUE
      QA=POLEG(INEL,1)
      QS=POLEG(INEL,2)
      DT(QOLD)=QA*QS
      LCS(INEL,6)=QS
      IF(L(3).EQ.1)GO TO 1600
C     THE VALVE IS CONNECTED CONVENTIONALY
      IF(L(3).NE.0)GO TO 1900
      IF(QS.EQ.1.) GO TO 1700
      GO TO 1650
C     THE VALVE IS CONNECTED BACKWARDS
 1600 IF(L(3).NE.1)GO TO 1900
      IF(QS.LT.1.0) GO TO 1700
C     THE VALVE IS CLOSED
 1650 POLEG(INEL,8)=DT(AREAC)+POLEG(INEL,8)
      POLEG(INEL,11)=POLEG(INEL,11)-QS*QA**2*DT(AREAC)
      DT(XLST)=0.0
      RETURN
 1700 IF (QA.LE.DT(INITQ)) GO TO 1800
C     THE VALVE IS FULLY OPEN
      IF(POLEG(INEL,3).LT.DT(INITQ)) POLEG(INEL,3)=DT(INITQ)
      DT(XLST)=D(POPDIS)
      POLEG(INEL,8)=DT(TK2)+POLEG(INEL,8)
      POLEG(INEL,11)=POLEG(INEL,11)-QS*QA**2*DT(TK2)
      RETURN
C     THE FLOW IS LESS THAN THE 'FULL OPEN FLOW'
 1800 POLEG(INEL,11)=POLEG(INEL,11)-DT(LOSA)*QS
     1 -QA*QS*DT(OPSQ)
      DT(XLST)=((DT(OPSQ)*QA)/DT(INITQ2))*D(POPDIS)
      IF(POLEG(INEL,4).GT.DT(INITQ)) POLEG(INEL,4)=DT(INITQ)
      POLEG(INEL,5)=POLEG(INEL,5)-DT(LOSA)*QS
      POLEG(INEL,6)=POLEG(INEL,6)+DT(OPSQ)
      RETURN
C*****ILLEGAL INPUT DATA TERMINATES PROGRAM*****
 1900 WRITE(6,999)
  999 FORMAT(10X,33HPROGRAM STOP IN SUBROUTINE CVAL33)
      STOP 1531
 2000 CONTINUE
      DT(DELP)=(D(DIAIN)-D(DIAOR))/(D(POPDIS)/(1.+D(POPDIS)))
      DT(K2)=(2./RHO(ITEMP(IND)))**.5*D(COEF)
      DT(VLST)=0.0
      DT(ALST)=0.0
      RETURN
 3000 CONTINUE
      ITER1=1
      L1=L(1)
      L2=L(2)
      XNEW=DT(XLST)
      VNEW=DT(VLST)
```

**6.33.7 (Continued)**

```
      ANEW=DT(ALST)
      DIS=XNEW/D(POPDIS)
      DT(ARFAC)=DIS*DT(SUB)
      DT(TK2)=DT(K2)*DT(ARFAC)
      XPREV=XNEW
      CKT=DT(K1)+DT(TK2)
      X=((Z(L1)+Z(L2))*CKT**2.)/2.
      Y=4.*(C(L1)-C(L2))/(CKT*(Z(L1)+Z(L2)))**2.
      SPAC=1.+ABS(Y)
      DT(QT)=X*(-1.+SQRT(SPAC))*SIGN(1.,Y)
 3060 CONTINUE
      P1=C(L1)-DT(QT)*Z(L1)
      P2=C(L2)+DT(QT)*Z(L2)
      F=(P1-P2)*D(DMAIN)-D(LOAD)-D(SPRING)*XNEW-.05*VNEW
      F1=2.*CV*ABS(P1-P2)*DT(ARFAC)
C     F1=2.*CV*ABS(P1-P2)*DT(DELP)*DT(XLST)/(1.+DT(XLST))
      F=F-F1
      ANEW=F/D(MASS)
      VNEW=DT(VLST)+(ANEW+DT(ALST))/2*DELT
      XNEW=DT(XLST)+(VNEW+DT(VLST))/2*DELT
      IF(XNEW.LT.0.0) XNEW=0.0
      IF(XNEW.GT.D(POPDIS)) XNEW=D(POPDIS)
      IF(ABS(XNEW-XPREV).LT.0.001) GO TO 3150
      IF(ITER1.GE.25)WRITE(6,910)XNEW,XPREV
  910 FORMAT(10X,*XNEW =*,2F40.10)
      IF(ITER1.EQ.25)GO TO 3150
      ITER1=ITER1+1
      XPREV=XNEW
      DIS=XNEW/D(POPDIS)
      DT(ARFAC)=DIS*DT(SUB)
      GO TO 3060
 3150 IF(XNEW.GT.0.0) GO TO 3200
      IF(ANEW.LT.0.0) DT(ALST)=0.0
      IF(VNEW.LT.0.0) DT(VLST)=0.0
      GO TO 3500
 3200 IF(XNEW.LT.D(POPDIS)) GO TO 3400
      IF(ANEW.GT.0.0) DT(ALST)=0.0
      IF(VNEW.GT.0.0) DT(VLST)=0.0
      GO TO 3500
 3400 DT(VLST)=VNEW
      DT(ALST)=ANEW
 3500 DT(XLST)=XNEW
      P(L1)=P1
      P(L2)=P2
      C(L1)=DT(QT)
      C(L2)=-DT(QT)
      RETURN
      END
```

## 6.34 SUBROUTINE CVAL34

Subroutine CVAL34 models a two stage relief valve of the type shown in Figure 6.34-1. This is a high response device used to limit pressure surges and to compensate for slow pump pressure controls. The subroutine can be used for any number of relief valves within the limits of the common storage.

### 6.34-1 Math Model

The relief valve is assumed to have a variable orifice characteristic between the fully open and fully closed positions. The effects of flow forces on the poppet are not included since these are not very well defined theoretically and depend upon the actual valve geometry.

In the steady state section the relief valve is assumed to be closed with no pilot flow.

In the transient analysis the flow through the valve is computed with the normal valve equations. The poppet position is predicted from the previous time step and is used to compute the valve orifice area.

The relief flow is first calculated using the predicted position of the valve, and the latest values of the line characteristics.

QI = SIGN(CA*(SQRT(1.+CB*ABS(CDIFF))-1., CDIFF)

where: CDIFF = C(L1) - C(L2)

CA = ((ARRELF*CV)**2*(Z(L1)+Z(L2))/RHO(K TEMP(IND))

CB = 2.*RHO(KTEMP(IND))/((ARRELF*CV)**2*(Z(L1)+Z(L2))**2)

CV = Discharge Coefficient

ARRELF = Valve orifice area open to return pressure

If the predicted poppet position is zero, the flow calculation is bypassed and the element pressures are set to the up and downstream characteristic pressures.

Connection No. 1 Connection No. 2

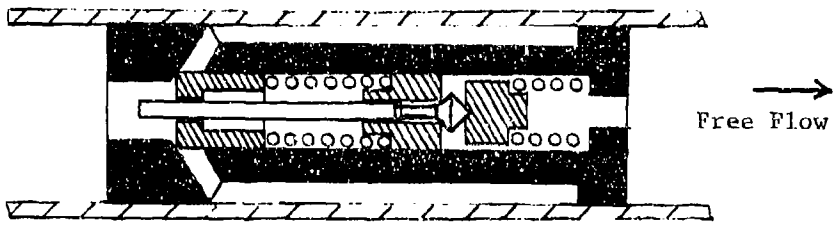



FIGURE 6.34-1

TYPE NO. 34 TWO STAGE RELIEF VALVE

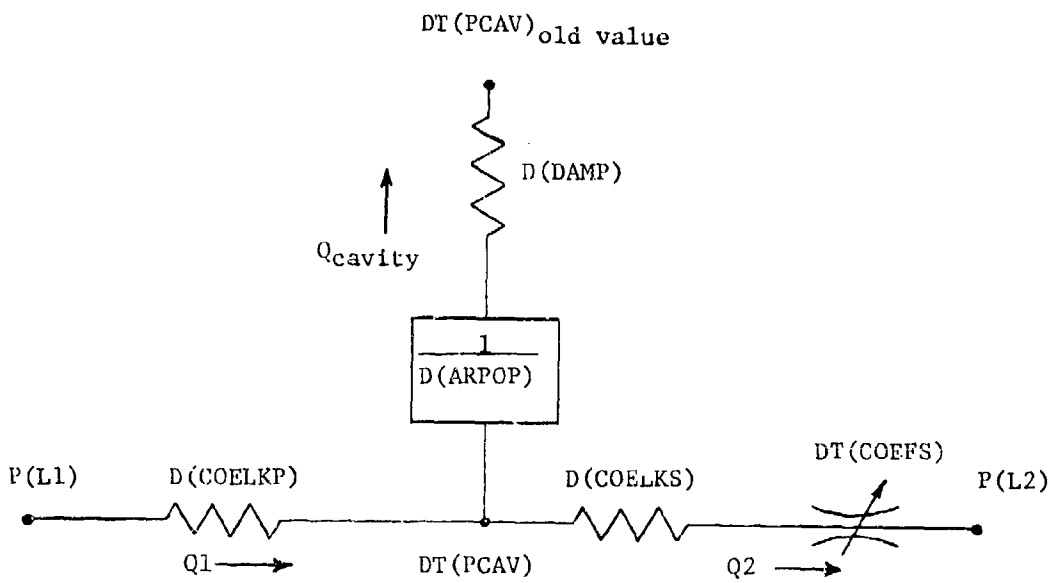



FIGURE 6.34-2

SCHEMATIC DIAGRAM FOR TYPE NO. 34 TWO STAGE RELIEF VALVE

With QI known the two pressures can be calculated:

$$P(L1) = C(L1) - Z(L1) * QI$$

$$P(L2) = C(L2) - Z(L2) * QI$$

When the valve is open or the difference in P(L1) - P(L2) is greater than the relief pressure, the latest position of the poppet is computed.

A force balance on the poppet yields equation (1).

$$FORCE_{poppet} = (P(L1) - DT(PCAV))*D(ARPOP) - DT(PXPOS)*D(KSPOP) - D(KZPOP) \quad (1)$$

where

DT(PCAV) - Cavity pressure behind the poppet

D(ARPOP) - Poppet area

DT(PXPOS) - Predicted poppet position

D(KSPOP) - Spring constant for poppet spring

D(KZPOP) - Poppet spring preload

An equivalent circuit schematic in Figure 6.34-2 can be drawn modeling the operation of the two stage relief valve.

The force on the poppet in Equation (1) can be set to the velocity of the poppet times a damping factor.

$$FORCE_{poppet} = VNEW * D(DAMP) = \frac{Q_{cavity}}{D(ARPOP)} * D(DAMP) \quad (2)$$

Equating the right hand side of equation (2) to equation (1) yields an expression for flow in the relief valve cavity-$Q_{cavity}$.

$$Q_{cavity} = (P(L1) - DT(PCAV))* \frac{D(ARPOP)^2}{D(DAMP)} - DT(PXPOS)*D(KSPOP)*\frac{D(ARPOP)}{D(DAMP)} - D(KZPOP) * \frac{D(ARPOP)}{D(DAMP)} \quad (3)$$

The leakage flow (Q1) from the high pressure side of the poppet to the cavity is

$$Q1 = \frac{P(L1) - DT(PCAV)}{D(COELKP)} \tag{4}$$

where

D(COELKP) = Pin leakage coefficient at poppet

The pressure drop from the cavity to return pressure is:

$$DT(PCAV) - P(L2) = \frac{Q2}{D(COELKS)} + \frac{Q2^2}{DT(COEFS)} \tag{5}$$

where

Q2 - Flow from cavity to return through the seat

D(COELKS) - Pin leakage coefficient at seat

DT(COEFS) - Constant based on open orifice area of pin and seat

The flow balance from Figure 6.34-2 gives

$$Q2 = Q1 - Q_{cavity} \tag{6}$$

Equations (3) and (4) are substituted into (6) to yield an equation with two unknowns - Q2 and DT(PCAV). Equation (5) gives DT(PCAV) in terms of Q2 and this substitution yields the following equation:

$$Q2^2 * \frac{DT(COEF)}{DT(COEFS)} + Q2\left(\frac{DT(COEF)}{D(COELKS)} + 1.\right) -$$

$$[DT(COEF)*(P(L1)-P(L2) + \frac{D(ARPOP)}{D(DAMP)}*(DT(PXPOS)*D(KSPOP)+D(KZPOP))] = 0 \tag{7}$$

After solving for Q2 in equation (7) the pressure in the cavity DT(PCAV) is determined from equation (5). Q1 can now be solved for in equation (4) and the flow into the cavity ($Q_{cavity}$) is easily computed. The poppet velocity is

$$VNEW = (Q1-Q2)/D(ARPOP) \tag{8}$$

The position is determined by a simple integration

$$XNEW = DT(XPOS) + (VNEW+DT(VLST))/2.*DELT \tag{9}$$

where

DT(XPOS) - previous poppet position

DT(VLST) - previous poppet velocity

DELT - Time step

XNEW, the new calculated position, is then tested to determine if the poppet is at a limit. A predicted position is computed for the next time step and the calculation is finished.

6.34.2 Assumptions

It is assumed that there are no variations in orifice characteristics with poppet position, and the flow forces on the poppet are negligible.

The simple integration method was chosen to save time and cost. Some errors will be obtained for very fast transients but this is considered acceptable for a component that could be used many times in a system.

6.34.3 Computations

1000 Section - In the 1000 section, temporary variables are calculated and stored in the D and DT arrays.

1500 Section - In the 1500 section the steady state values used in the computation of leg flow are computed and returned to the PQLEG array.

2000 Section - The poppet position and velocity are initialized in the 2000 section.

3000 Section - The calculations for Q, P(L1), P(L2) and poppet velocity follow the theory discussed in 6.31.2 above.

The velocity of the poppet and hence its position is computed if the difference in P(L1) and P(L2) is greater than the relief pressure or the predicted poppet position is greater than zero.

The calculated value of XNEW is checked to determine if it is at a stroke limit, and if the calculated velocity is in the limit direction. Should these conditions occur the limit subroutine will set the velocity (VNEW) and position (XNEW) to zero. These limits do not apply away from the end of the stroke.

From the computed value of XNEW a new poppet position is then predicted for the next time step.

6.34.4 Approximations

The approximations made are associated with the valve orifice calculations and the absence of flow force effects. These approximations speed the computation but are a source of inaccuracy.

6.34.5 Limitations

The main limitation is the lack of a cavitation simulation. This can yield erroneous results whenever the pressures drop below vapor pressure.

6.34.6 Variable Names

| NAME | DESCRIPTION | DIMENSION |
|---|---|---|
| ALPHA | Dummy variable | -- |
| D(ARPIN) | Pin rod end area | IN**2 |
| D(ARPOP) | Poppet area | IN**2 |
| ARRELF | Compu d relief area | IN**2 |
| D(ARSRP) | Area of seat relief port | IN**2 |
| BETA | Dummy variable | -- |
| CA | Dummy variable | -- |
| CB | Dummy variable | -- |
| CDIFF | Difference in characteristic pressures | PSI |

| NAME | DESCRIPTION | DIMENSION |
|---|---|---|
| CHI | Dummy variable | -- |
| DT(COEF) | Dummy variable | -- |
| DT(COEFS) | Dummy variable | -- |
| D(COELKP) | Pin leakage coefficient at poppet | CIS/PSI |
| D(COELKS) | Pin leakage coefficient at seat | CIS/PSI |
| CV | Discharge coefficient | -- |
| D(DAMP) | Poppet damping factor | LB/IN/SEC |
| DT(DIA1) | Computed relief diameter | IN |
| DT(DIA2) | Computed seat relief diameter | IN |
| DT(GAMMA) | Angle of relief flow relative to poppet | DEG |
| D(KSPIN) | Pin spring constant | LB/IN |
| D(KSPOP) | Poppet spring constant | LB/IN |
| D(KZPIN) | Pin spring preload | LB |
| D(KZPOP) | Poppet spring preload | LB |
| DT(PCAV) | Pressure in poppet cavity | PSI |
| DT(PXPOS) | Predicted poppet position | IN |
| Q1 | Relief flow | CIS |
| Q1 | Leakage flow through poppet and pin | CIS |
| Q2 | Leakage flow through seat and pin | CIS |
| DT(VLST) | Previous poppet velocity | IN/SEC |
| VNEW | Poppet velocity | IN/SEC |
| D(XMAX) | Maximum poppet travel | IN |
| XNEW | Latest poppet position | IN |
| DT(XPOS) | Poppet position | IN |
| XPOSIN | Pin position | IN |

## 6.34.7 Subroutine Listing

```
      SUBROUTINE CVAL34 (D,DT,DD,L)
C *** REVISED AUGUST 5,1976 ***
      DIMENSION D(1),DT(1),DD(1),L(1)
      COMMON NTELPL,NTOLPL,IPT,IPOINT,NPTS,INEL,KNEL,NTOPL,NLPLT(61,3),
     1 PQLEG(90,12),LCS(90,10),ILEG(1400),PN(90),QN(90)
      COMMON/SUB/PARM(150,9),PH(1500),QM(1500),P(300),Q(300),C(300)
     1,Z(300),RHO(20),S2ORHO(20),VISC(20),BULK(20),TEMP(20),PVAP(20)
     2,ATPRES,T,DELT,TFINAL,PLTDEL,PI,TITLE(20),LEGN,ICON
     3,KTEMP(99),LSTART(150),NLPT(150),LTYPE(99),NC(99),INX,INZ
     4,INV,ISTEP,NLINE,NEL,IND,IENTR,MNLINE,MNEL,MNLEG,MNNODE,MNPLOT
     5,MNLPTS,MDS
      INTEGER ARPOP,XMAX,PRELF,COELKP,COELKS,COEF,COEFS,PCAV,QCAV,
     + ARSRP,ARPIN,DIA1,DIA2,PXPOS,XPOS,VLST,DAMP,GAMMA,COLFA
C       **** D - ARRAY INPUT VARIABLES ****
      DATA ARPOP/1/,XMAX/2/,KSPOP/3/,KZPOP/4/,PRELF/5/,COELKP/6/,
     + COELKS/7/,ARSRP/8/,ARPIN/9/,KSPIN/10/,DAMP/11/,
     + GAMMA/12/
C       **** DT - COMPUTED VARIABLES ****
      DATA DIA1/1/,DIA2/2/,COEF/3/,PXPOS/4/,XPOS/5/,VLST/6/,
     + INPPSI/7/,COEFS/8/,PCAV/9/,COLFA/10/,QCAV/11/
      IF(IENTR) 1000,2000,3000
 1000 CONTINUE
      IF (INEL.NE.0) GO TO 1500
      DT(DIA1)=D(ARPOP)
      D(ARPOP)=(D(ARPOP)**2)*PI/4.
      DT(DIA2)=D(ARSRP)
      D(ARSRP)=(D(ARSRP)**2)*PI/4.
      D(ARPIN)=(D(ARPIN)**2)*PI/4.
      D(GAMMA)=D(GAMMA)*PI/180.
      D(GAMMA)=SIN(D(GAMMA))
      DT(INPPSI)=D(ARPIN)/D(KSPIN)
      CV=.65
      RETURN
C     STEADY STATE SECTION
 1500 CONTINUE
      IF(KNEL.NE.1)RETURN
      PQLEG(INEL,6)=3.0E6+PQLEG(INEL,6)
      QI=PQLEG(INEL,1)
      PQLEG(INEL,11)=PQLEG(INEL,11)-3.0E6*QI*PQLEG(INEL,2)
      RETURN
 2000 CONTINUE
      DT(QCAV)=0.0
      DT(COLFS)=0.0
      DT(PCAV)=P(L(1))
      DT(PXPOS)=0.0
      DT(XPOS)=0.0
      ASIGN=0.0
      DT(VLST)=0.0
      DT(COEF)=1./D(COELKP)-D(ARPOP)**2/D(DAMP)
      DT(COLFA)=D(ARPOP)/D(DAMP)
```

**6.34.7 (Continued)**

```
      WRITE(6,900)DT(COEF),DT(COLFA)
      RETURN
 3000 CONTINUE
      L1=L(1)
      L2=L(2)
      QI=0.0
      IF(DT(QCAV).GT.0.0)GO TO  3110
      IF(DT(PXPOS).EQ.0.0)GO TO 3100
      ARRELF=(DT(DIA1)+D(GAMMA)*DT(PXPOS)*2.)**2*PI/4.-D(ARPOP)
      CA=((ARRELF*CV)**2*(Z(L1)+Z(L2)))/RHO(KTEMP(IND))
      CB=2.*RHO(KTEMP(IND))/((ARRELF*CV)**2*(Z(L1)+Z(L2))**2)
      CDIFF=C(L1)-C(L2)
      QI=SIGN(CA*(SQRT(1.+CB*ABS(CDIFF))-1.),CDIFF)
 3100 CONTINUE
      P(L1)=C(L1)-Z(L1)*QI
      P(L2)=C(L2)+Z(L2)*QI
      Q(L1)=QI
      Q(L2)=-QI
      P2=P(L2)
      IF(P2.GT.PVAP(KTEMP(IND)))GO TO 3150
 3110 IF(DT(PXPOS).EQ.0.0)GO TO 3120
      ARRELF=(DT(DIA1)+D(GAMMA)*DT(PXPOS)*2.)**2*PI/4.-D(ARPOP)
      CA=((ARRELF*CV)**2*(Z(L1)+Z(L2)))/RHO(KTEMP(IND))
      CB=2.*RHO(KTEMP(IND))/((ARRELF*CV)**2*(Z(L1)+Z(L2))**2)
      CDIFF=C(L1)-C(L2)
      QI=SIGN(CA*(SQRT(1.+CB*ABS(CDIFF))-1.),CDIFF)
 3120 CONTINUE
      P(L1)=C(L1)-Z(L1)*QI
      P(L2)=PVAP(KTEMP(IND))
      P2=P(L2)
      Q(L1)=QI
      Q(L2)=(C(L2)-P(L2))/Z(L2)
      DT(QCAV)=DT(QCAV)-(Q(L2)+QI)
      IF(DT(QCAV).LT.0.0)DT(QCAV)=0.0
 3150 CONTINUE
      WRITE(6,900)P(L1),P(L2),D(PRELF),C(L1),C(L2)
  900 FORMAT(1X,10E12.5)
      IF((P(L1)-P2).LT.D(PRELF).AND.DT(PXPOS).EQ.0.0)RETURN
      XPOSIN=(P(L1)-P2-D(PRELF))*DT(INPPSI)
      IF(XPOSIN.LE.0.0)GO TO 3200
      ARRELF=(DT(DIA2)-D(GAMMA)*XPOSIN*2.)**2*PI/4.
      ARRELF=D(ARSRP)-ARRELF
      DT(COEFS)=(2.*(CV*ARRELF)**2)/RHO(KTEMP(IND))
      ALPHA=DT(COLF)/DT(COEFS)
      BETA=1.+DT(COLF)/D(COLLKS)
      CHI=(DT(COEFA))*(DT(PXPOS)*D(KSPOP)+D(KSPOP))
      CHI=CHI+DT(COLF)*(P(L1)-P2)
      CHI=-CHI
      Q2=(-BETA+SQRT(BETA**2-4.*ALPHA*ABS(CHI)))/(2.*ALPHA)
      Q2=SIGN(Q2,CHI)
```

**6.34.7 (Continued)**

```
      GO TO 3300
 3200 Q2=0.0
 3300 CONTINUE
      DT(PCAV)=Q2/D(COELKS)+Q2**2/DT(COEFS)+P2
      Q1=(P(L1)-DT(PCAV))/D(COELKP)
      VNEW=(Q2-Q1)/D(ARPOP)
      XNEW=DT(XPOS)+(VNEW+DT(VLST))/2.*DELT
      CALL XLIMIT(XNEW,VNEW,ASIGN,0.0,D(XMAX))
      DT(VLST)=VNEW
      DT(PXPOS)=2.*XNEW-DT(XPOS)
      IF(XNEW.EQ.0.0)DT(PXPOS)=0.0
      DT(XPOS)=XNEW
      WRITE(6,910)T
  910 FORMAT(10X,* DATA CALCULATED AT TIME =*,F12.5)
      WRITE(6,900)XNEW,VNEW,Q1,Q2,ALPHA,BLTA,CHI,XPOSIN,ARRELF,CI
      RETURN
      END
```

## 6.41 SUBROUTINE REST41

REST41 simulates a fixed, two-way, orifice restrictor with two connections. The coefficient of discharge is assumed the same for flow in either direction so that the unit can be installed backwards i.e., either end can be called connection #1.

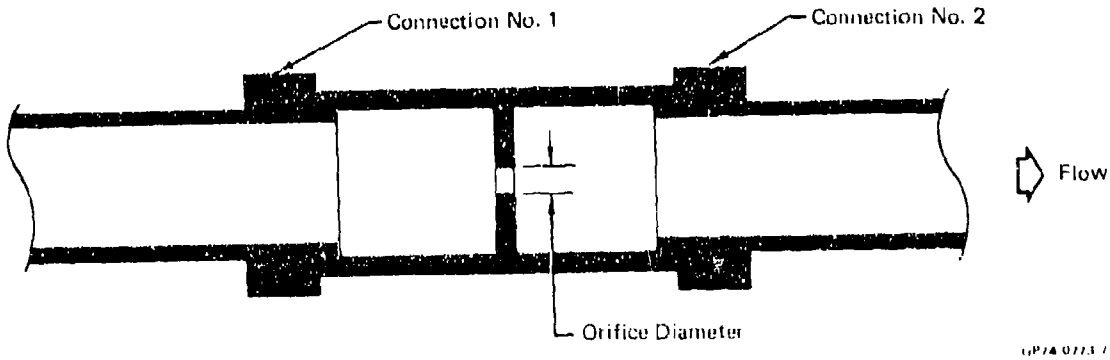



FIGURE 6.41-1
TYPE NO. 41 ORIFICE RESTRICTOR

### 6.41.1 Math Model

The basic equation for flow through an orifice is

$$Q = AV*CV*(2*(P1-P2)/RHO(\ ))^{\frac{1}{2}} \quad (1)$$

where AV = area of the valve orifice $in^2$
CV = discharge coefficient

RHO = fluid density in lbs $sec^2/in^4$
Q = flow in CIS
P1 = Inlet pressure in PSI
P2 = Outlet pressure in PSI.

From the line equations

$$P1 = -Z_1Q_1+C_1 \quad (2)$$

$$P2 = Z_2Q_2+C_2 \quad (3)$$

using a +ve flow convention from P1 to P2

where $C_1$ = Con #1 line characteristic
$Z_1$ = Con #1 line characteristic impedance
$C_2$ = Con #2 line characteristic
$Z_2$ = Con #2 line characteristic impedance.

Assuming no flow loss (4)

$$Q = Q_1 = Q_2$$

Substituting equations (2) and (3) in (1)

$$Q^2 = (-Z_1Q+C_1-Z_2Q-C_2)*CV^2*AV^2*2/RHO(\ ) \quad (5)$$

The solution to this equation is

$$Q = SIGN(CA*(SQRT(1.+CB*ABS(CDIFF))-1), CDIFF)$$

where $CA = (Z_1+Z_2)*CV^2*AV^2*2/RHO(\ )$

$CDIFF = C_1 - C_2$

$CB = 2*RHO( )/((AV^2*CV^2)*(Z1+Z2)^2)$.

P1 and P2 are then calculated using the Q value.

$$P1 = C_1 - Z_1*Q \quad (8)$$

$$P2 = C_2 + Z_2*Q \quad (9)$$

6.41.2 Assumptions

1. It is assumed that the restrictor does not have any ancillary parts and that the oil volume is sufficiently small so that integration is not required.
2. The discharge coefficient is considered the same in either flow direction.

6.41.3 Computation Methods

SECTION 1000

Input orifice diameter, D(1), is converted to area

D(1) = D(1)**2*P1/4.

The steady state orifice equation constant is calculated using the formula:

D(3) = RHO( )/((D(2)*D(1))**2*2.)

SECTION 1500

Leg constant for $Q^2$ is updated and pressure at outlet connection is calculated and stored.

SECTION 2000

Constants for the dynamic orifice equation are calculated.

D(3) = ((D(1)*D(2)**2*(Z(L(1))+Z(L(2))))/RHO( )

D(4) = Z*RHO( )/(((D(1)*D(2))**2)*(Z(L(1)) + Z(L(2)))**2)

SECTION 3000

Flow is calculated using the equation.

QI = SIGN(D(3)*(SQRT(1.+D(4)*ABS(CDIFF))-1), CDIFF)

This flow is used to calculate pressures P1 and P2.

P(L(1)) = C(L(1))-Z(L(1))*QI

P(L(2)) = C(L(2)) + QI*Z(L(2))

6.41.4 Approximations

None

6.41.5 Limitations

Subroutine is limited to fixed two way restrictors having the same discharge coefficient for flow in either direction.

6.41.6 Variable Names

| NAME | DESCRIPTION | DIMENSIONS |
|---|---|---|
| CDIFF | Characteristic Difference | PSI |
| D(1) | Orifice Diameter | IN |
| D(2) | Orifice Discharge Coefficient | -- |
| D(3), D(4) | Orifice Equation Constants | -- |
| QI | Flow | CIS |

## 6.41.7 Subroutine Listing

```
      SUBROUTINE REST41 (D,DT,DD,L)
C *** REVISED AUGUST 5,1975 ***
      DOUBLE PRECISION DD
      DIMENSION D(4),DT(1),DD(1),L(2)
      COMMON NTELPL,NTOLPL,IPT,IPOINT,NPTS,INEL,KNEL,NTOPL,NLPLT(61,3),
     1 POLEG(90,12),LCS(90,10),ILEG(1400),PN(90),QN(90)
      COMMON/SUB/PARM(150,9),PM(1500),QM(1500),P(300),Q(300),C(300)
     1,Z(300),RHO(20),S2ORHO(20),VISC(20),BULK(20),TEMP(20),PVAP(20)
     2,ATPRES,T,DELT,TFINAL,PLTDEL,PI,TITLE(20),LEGN,ICON
     3,KTEMP(99),LSTART(150),NLPT(150),LTYPE(99),NC(99),INX,INZ
     4,INV,ISTEP,NLINE,NEL,IND,IENTR,MNLINL,MNEL,MNLEG,MNNODL,MNPLOT
     5,MNLPTS,MDS
      IF(IENTR) 1000,2000,3000
 1000 CONTINUE
      IF (INEL.NE.0) GO TO 1500
      D(1)=D(1)**2*PI/4.
      D(3)=RHO(KTEMP(IND))/((D(2)*D(1))**2*2.)
      RETURN
C
C     STEADY STATE SECTION
 1500 CONTINUE
      POLEG(INEL,8)=D(3)+POLEG(INEL,8)
      QI=POLEG(INEL,1)
      POLEG(INEL,11)=POLEG(INEL,11)-D(3)*QI**2*POLEG(INEL,2)
      RETURN
 2000 CONTINUE
      D(3)=((D(1)*D(2))**2*(Z(L(1))+Z(L(2))))/RHO(KTEMP(IND))
      D(4)=2*RHO(KTEMP(IND))/(((D(1)*D(2))**2)*(Z(L(1))+Z(L(2)))**2)
      RETURN
 3000 CONTINUE
      CDIFF=C(L(1))-C(L(2))
      QI=SIGN(D(3)*(SQRT(1.+D(4)*ABS(CDIFF))-1),CDIFF)
      P(L(1))=C(L(1))-Z(L(1))*QI
      P(L(2))=C(L(2))+QI*Z(L(2))
      Q(L(1))=QI
      Q(L(2))=-QI
      RETURN
      END
```

## 6.51 SUBROUTINE PUMP51

Subroutine PUMP51 was set up to model an F15 pump, which is basically a simple in line piston pump, though it incorporates many mechanical refinements and sophisticated design features. The model is intended for use by system designers and is primarily aimed at the study of pump system stability under dynamic loading conditions.

The model incorporates the effects of case drain dynamics, since the F15 pump output pressure is referenced to case and the actuator discharges to the case, and displaces case volume when it is moving. The treatment of leakage and damping characteristics are rudimentary.

The dynamics of the hanger are complex. For the model the effects of the dynamic forces on the actuator are included. These forces push the pump to maximum flow, the hanger spring provides this force on start-up.

In addition the hanger offset creates a negative flow at the pump inlet and outlet when the hanger is moving toward maximum flow, and this has a destabilizing effect, when the hanger response is very fast.

Some of the hanger forces are oscillatory but no attempt has been made to describe this effect, except that the magnitude is sufficient to keep the hanger in motion, so the effects of static friction is ignored. This is an assumption that helps the simulation by keeping the integration of the hanger velocity a continuous function, between its mechanical stops.

The compensator valve dynamics are a significant part of the model. The forces on the valve are a combination of the outlet pressure force pushing against the case pressure and spring forces, with damping and flow forces acting in either direction.

The damping and flow forces are estimates using classical formulae.

Under certain conditions, the pump compensator valve responds to the oscillatory pressures caused by the pump pulsations. While no attempt has been made to describe how these are generated, provisions have been added so that their effect on the compensator can be determined.

The compensator valve on the F15 is overlapped, but under certain conditions the pulsations can cause the valve to dither, reducing the effect of the overlap and changing the response characteristics of the pump.

The damping and flow forces are included because of the fast response of this particular valve.

6.51.1 Math Model

A simplified diagram of the pressure regulated variable displacement pump is shown in Figure 6.51-1.

An equivalent circuit schematic diagram for the pump model is shown in Figure 6.51-2.

Pump Displacement Flow

For the pump inlet the displacement flow is computed as follows:

QPUMP = D(DISP)*DT(PRPM)*(DISACT)

if D(DISAMI)<DT(DISACT)<D(DISAM)

or QPUMP = QINLET + QCASIN

if PINLET< PMIN

Actuator Pressure

The actuator pressure is based on the contributions of the spring force, case pressure, outlet pressure and pump rpm, plus the reaction force due to velocity damping which is generated when the hanger is moving. The input data establishing actuator pressure vs pump operating conditions is modified to give a simpler algorithm for the transient calculations.

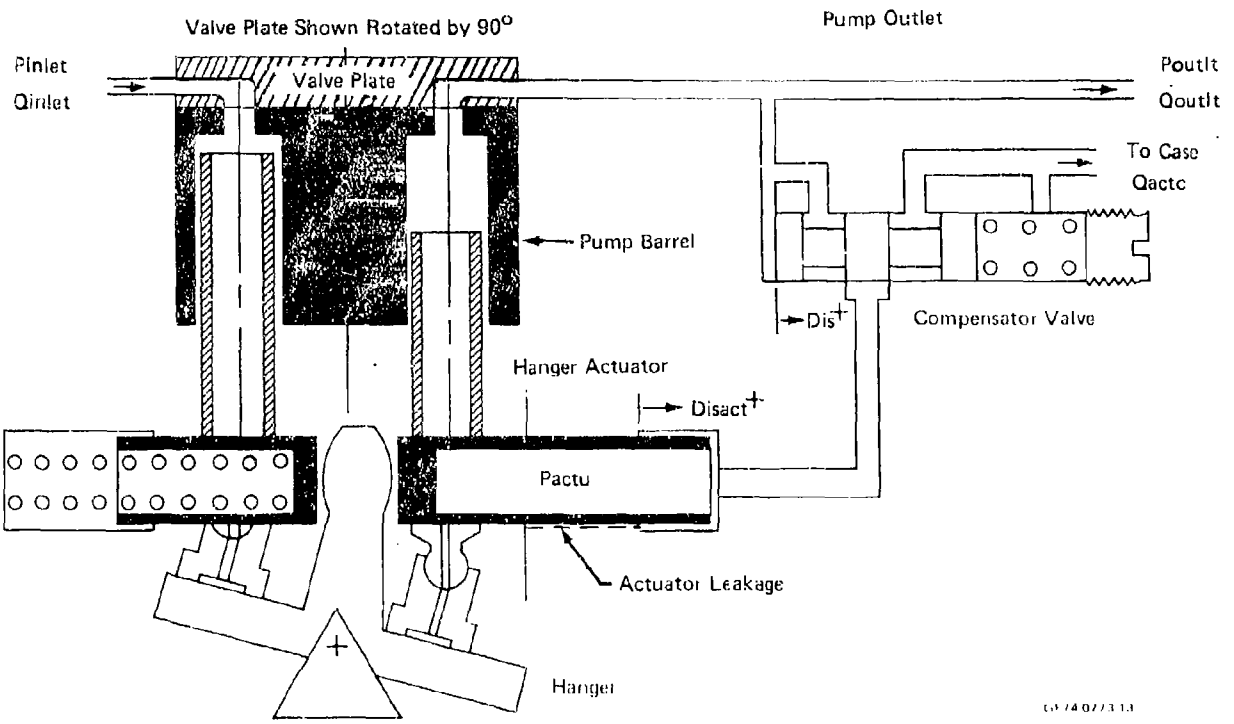
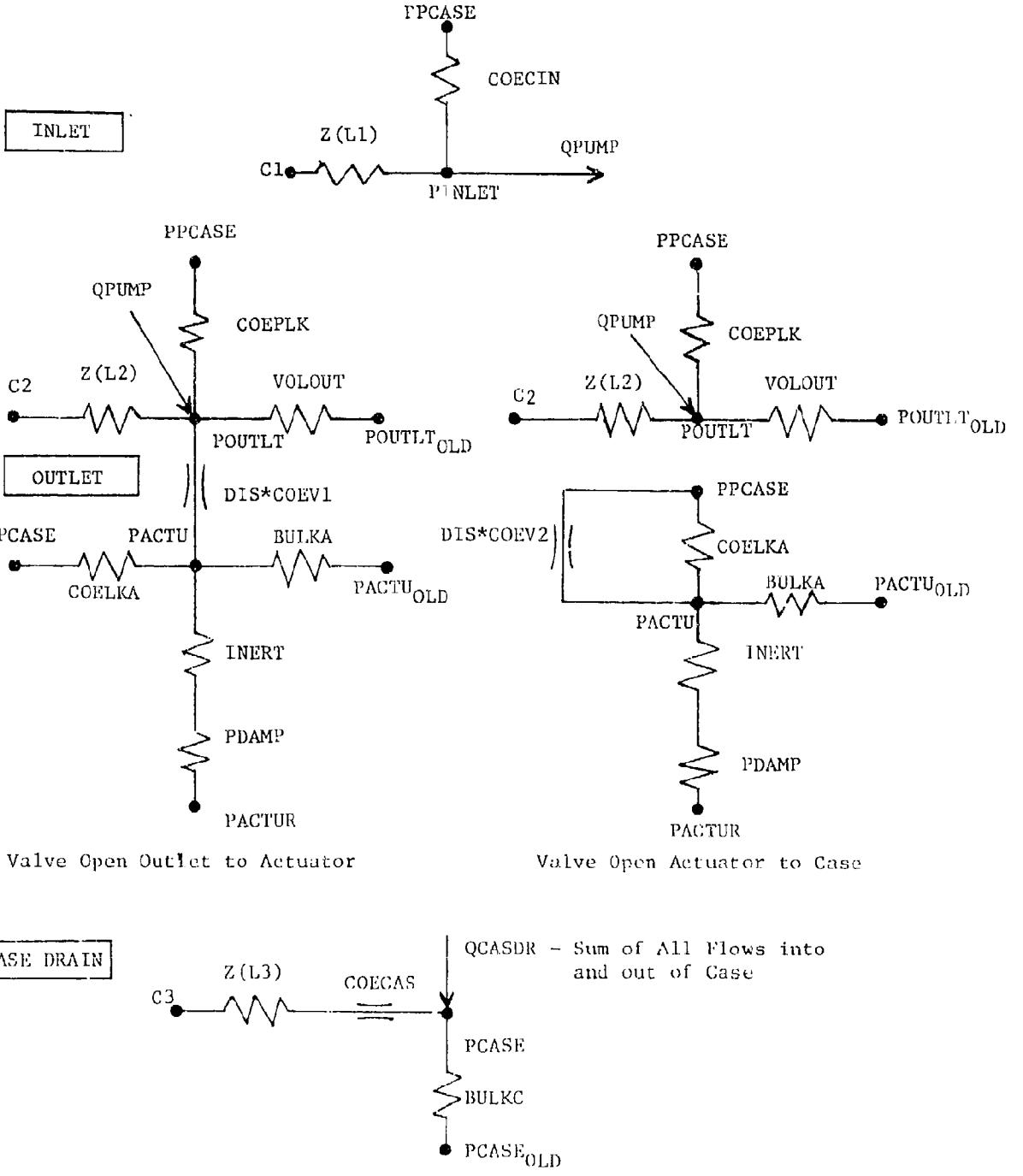



**FIGURE 6.51-1**
**TYPE NO. 51 PRESSURE REGULATED VARIABLE DISPLACEMENT PUMP**




Valve Open Outlet to Actuator

Valve Open Actuator to Case




SCHEMATIC DIAGRAM FOR F-15 PUMP MODEL

The input data gives the actuator pressure due to the spring force at maximum flow, D(PSPRIM) and at zero flow, (D(PSPRIZ). P(PZRPM) is input as the pressure at zero actuator displacement and 3600 rpm. It is then modified to give the pressure at zero rpm by subtracting from it the slope of the pressure versus rpm curve, which is input as D(PSLRPM).

Using the F-15 pump experimental data and the actuator pressure predictions generated by the HSFR program, a formula was derived which related actuator pressure to the output pressure/bulk modulus ratio.

The test data collected at different fluid temperatures showed a need for temperature correction which was obtained via the bulk modulus. A correction factor DT(BULKO)/DT(POUTLT) is used where

DT(BULKO) = BULK(KTEMP(1ND))*D(DPVOA)/223000

D(DPVOA) is the reference pressure and 223000 is the reference bulk modulus of MIL-H-5606B at 3000 psi and 130$^{O}$F.

The reference actuator pressure for this time step is then computed as

```
PACTUR = D(PSPRIZ)+PDISA*D(PSRIM) - VOLD*D(INERT)
       + PDISA*(D(PACCP)*DT(PRPM)**2 + D(PDISAC))
       + DT(PCASE) + (D(PZRPM) + D(PSLRPM)*DT(PRPM))
       *DT(BULKO)/DT(POUTLT)
```

where

PDISA = PREDICTED ACTUATOR POSITION BASED ON THE PREVIOUS POSITION AND VELOCITY
(=DT(DISACT)+DT(VELACT)*DELT)

VOLD = ACTUATOR VELOCITY (IN/SEC)

D(PACCP) = D(PACCP)/(D(DISA )*3600**2)

D(PACCP) IS INPUT AS THE PRESSURE DUE TO PISTON ACCELERATION AT 3600 RPM AND MAXIMUM STROKE

D(PSPRIM) = (D(PSPRIM)-D(PSPRIZ))/D(DISAM)

D(PSPRIM) IS INPUT AS ACTUATOR PRESSURE DUE TO SPRING FORCE AT MAXIMUM PUMP DISPLACEMENT

D(PDISAC) = (D(PDISAC)-D(PZRPM))/D(DISAM)

D(PDISAC) IS INPUT AS ACTUATOR PRESSURE AT 3600 RPM AND MAXIMUM PUMP DISPLACEMENT

D(PZRPM) = D(PSRPM)-D(PSLRPM)*3600

D(PZRPM) IS INPUT AS ACTUATOR PRESSURE AT 3600 RPM AND ZERO PUMP DISPLACEMENT

The actuator pressure and damping characteristics are important variables since they govern how fast the pump goes from zero to full flow. No easy way exists to obtain these values accurately from purely dimensional data. The HSFR program is able to get within 30% of the measured actuator pressure and gives reasonable predictions of the variations with temperature and rpm. Measurements of actuator pressure and damping characteristics require a complex set of instrumentation and analysis of the data, to extract the variables would require inspired judgement. Therefore a reasonable initial actuator pressure D(PZRPM) which includes estimated values for the contributions of the spring is 800 psi. Other contributions to actuator pressure are input as zero.

Compensator Valve

The compensator valve position is assumed to be directly proportional to the differential pressure between outlet and case. The lag, due to the valve damping and hanger inertia is included in the computation of the actuator pressure as shown in the INERT and DAMP terms of Figure 6.51-2.

The valve spring rate D(INPPSI) is input in lbs/in and converted to in/psi,

D(INPPSI) = D(ARCOM)/D(INPPSI)

The differential pressure at which the valve opens from outlet to actuator D(DPVOA) is used to determine the valve position. Two pressures POUTMX and POUTMI are derived, POUTMX is the maximum outlet pressure that can be obtained assuming zero flow from the outlet to the actuator.

POUTMX = (C2/Z(L2) + QPUMP + DT(PCASE)*D(COEPLK))*ZOUT

where

ZOUT = Z(L2)/(1.0 + Z(L2)*D(COEPLK) + Z(L2)*D(VOLOUT))

QPUMP = DT(QINLET) + QCASIN + QOSIN

The ZOUT term includes the impedance of the outlet volume.

The minimum pressure, POMTMI, which is the outlet pressure when the valve is just about to open is computed as:

POUTMI = D(DPVOA) + DT(PPCASE)

where DT(PPCASE) is the predicted case pressure for the current time step.

The actual outlet pressure lies between these limits and is obtained by iteration. The valve orifice area is calculated for any position of the valve and the inlet port. The valve displacement for

POUTMX>POUTMI is

DIS = (DT(POUTLT) - POUTMI)*D(INPPSI)/(D(FLOFRC)*(DT(POUTLT) DT(PACTU)) +1.0)

where

D(FLOFRC) = 2*D(WIDTH)*D(INPPSI)*(D(COEVL1)**2)*.358/D(ARCOM)

The orifice area is easily computed since the F-15 pump has square metering ports.

When the valve is within the overlap region, the valve flow is set to zero. No allowance is made for leakage due to diametrical clearance within the valve.

Since the F-15 pump is essentially a closed loop servo, a gain term was calculated as:

GAIN = 0.7/(1.+20.*DT(COEV1)*D(INPPSI)*Z(L(2)))

where

DT(COEV1) = D(COEVL1)*S2ORHO(KTEMP(IND))D(WIDTH)

For the initial guess of outlet pressure DT(POUTLT) is computed at DT(ITUP)*POUTMX + DT(MITUP)*POUTMI. (DT(ITUP)=GAIN, DT(MITUP)=1.-GAIN) This pressure is used to compute the valve position, area, and flow, DT(QACTU), into the actuator. The actuator flow is then used to recompute the pressure

TPOUT = POUTMX - DT(QACTU)*ZOUT

A check is made to see if the recomputed flow is within .05 psi of the pressure valve and if it is not, the outlet pressure is updated by

DT(POUTLT) = DT(POUTLT)*DT(MITUP) + TPOUT*DT(ITUP)

The choice of the DT(MITUP)/DT(ITUP) ratio and the initial guess ratio of DT(ITUP)/DT(MITUP) was made using an actual closed loop gain for the F-15 pump. For other pumps, adjustments of these ratios could reduce the number of iterations required to achieve a balance. The tolerance of .05 psi could also be increased with little penalty.

Flow From Actuator Piston to Case

When POUTMI>POUTMX the valve is either closed or open from actuator to case. POUTMI is changed to

POUTMI = D(DPVAC) + DT(PPCASE)

and if POUTMI is still greater than POUTMX, the valve is open allowing flow out of the actuator so that the pump flow, QPUMP increases. The valve

displacement area and flow (DT(QACTC)) are recalculated. Since the flow does not affect outlet pressure, no iteration is necessary.

The actuator leakage to case is assumed to be laminar since the passage is small around the actuator barrel. The actuator leakage is computed as

QACTLK = COELKA*(DT(PACTU) - DT(PPCASE))

with all the actuator flows known, the actuator velocity is calculated

DT(VELACT) = - QNET/D(ARACT)

where

QNET = DT(QACTU) - DT(QACTC) - QACTLK - (DT(PACTU) - PACTUO)*DT(BULKA)

The difference in the new and old actuator pressures times DT(BULKA) accounts for the compressible fluid flow in the actuator. The new actuator position is then

DT(DISACT) = DT(DISACT) + (VOLD+DT(VELACT))*DELT/2.

A check is made to determine if the actuator is at the stroke limits. If it is the actuator flow must be recalculated. If the actuator is at maximum stroke (fully retracted with the pump at full stroke) then the actuator pressure drops to case pressure and DT(QACTC) is set to zero.

If the actuator is at minimum stroke (with the pump outlet flow negative), then DT(QACTU) and DT(PACTU) have to be recalculated so that the actuator leakage flow can be determined.

Pump Outlet Pressure and Flow

After the actuator pressure flows and valve outlet pressure DT(POUTLT) are computed, pump outlet flow is calculated as:

Q(L2) = -(QPUMP-QPLEAK-DT(QACTU)-(DT(POUTLT)-OPOUT)*D(VOLOUT))

Actual pump output pressure is then

P(L2) = C(L2)-Q(L2)*Z(L2)

Pump Case Outlet Pressure and Flow

The case outlet flow is determined using the schematic of Figure 6.51-2 to write a second degree equation for case drain flow:

DT(COECAS)*Q(L3)**2+(Z(L3)+DT(BULKC))*Q(L3)

-(DT(PCASE)+QCASDR*DT(BULKC)-C(L3))=0

where DT(PCASE) is the previous time step value of case pressure.

The above equation is solved for Q(L3) and the case pressure is computed as

DT(PCASE)=DT(PCASE)+(QCASDR+Q(L3))*DT(BULKC)

The outlet case pressure is then

P(L3) = C(L3)-Q(L3)*Z(L3)

6.51.2 Assumptions

The assumptions in a model of this nature are almost too many to enumerate. By its very nature the pump is a complex piece of equipment with multiple leak paths across the port plate, down the side of the piston and out of the shoes. This whole set of leak paths have been linearized and assumed to be constant for a constant output pressure, which is no doubt rather hard to accept. The alternative would be go into very detailed calculations with the leakage dependent on the piston load, hanger angle, RPM and anything else one could add. Unfortunately this too would probably be inaccurate so instead of an inaccurate complex leakage model we choose a simple leakage model, which could be improved when more data is available from the verification tests.

The forces on the hanger are not taken into account as it rotates only the hanger inertia. Flow and leakage are all treated as though the pump had a continuous output rather than the individual pumping pistons.

A model which includes the dynamics of each individual piston would by necessity be considerably more complex and consume much more computer time, but is is an alternative to what we have done here.

In all the calculations the bulk modulus is treated as a constant for the high pressure (output) side and as a different constant for the low pressure (inlet) side, the elastic expansion of the volume cavities is included.

Friction effects have not been included primarily because of the cost of putting them in, but in actual fact the forces on the hanger have an oscillatory content which tend to keep it in motion. The pulsations of the outlet also tend to keep the valve in motion so that friction effects would not normally be significant.

6.51.3 Computation

The pump subroutine is set up using the HYTRAN program commons, plus the D, DT, DD, and L arrays.

1000 SECTION

In the 1000 section the constants are initialized where desirable for more efficient computation.

The remaining part of Section 1000 deals with the calculations of the steady state pump characteristics.

In order to balance the pump at some steady state condition, it is first necessary to establish what the pump characteristics are, over the maximum range of pump flow.

To help in this it was assumed that these characteristics could be approximated by a straight line interpolation between the pump conditions at maximum and minimum flows which correspond to zero and maximum actuator displacement respectively.

A chain of interdependent calculations are needed to derive the maximum and minimum conditions. We have to establish:

DT(PACTU) = Actuator pressure

DT(QACTC) = Flow from actuator to case

DT(QACTU) = Flow from outlet to actuator through the valve

ACASDR = Case drain flow

QACTLK = Flow into the actuator due to leakage

DT(DISVLV) = Valve displacement

DT(POUTLT) = Valve chamber pressure which is the same as outlet pressure

The actuator pressure is dependent on the outlet pressure due to the pumping offset, and the valve flow is dependent on the difference between outlet pressure and actuator pressure.

The initial flows (QACTU and QACTC) are computed at the appropriate valve positions and this information is used in the steady state portion of the program.

1500 SECTION

Steady State Calculations

The pump which has three connections, has a node located at the inlet.

The leg which has the inlet connection receives the pump inlet pressure from the steady state routine.

For the leg which has the outlet connection as its first element, a value of the output flow ratio is computed as

$$DT(DISVLV) = \frac{QGUESS-DT(QOUTLT)}{DT(QINLET)-DT(QOUTLT)}$$

The ratio is calculated to determine the percentage of actuator flow that is actual leakage flow (QACTLK) into the pump case.

The output pressure is determined by the computed maximum outlet pressure at maximum valve displacement DT(POUTM) plus the pressure drop from case to inlet, DT(DELP13).

The outlet pressure is added to PQLEG(INEL, 5) and the output impedance (.00001) is added to PQLPG(INEL, 6).

DT(POUTLT) is initialized to DT(POUTM) + DT(DELP13)1

And the outlet pressure PQLEG (INEL, 11) is also increased by DT(POUTM) + DT(DELP13).

LCS(INCL, 7) is set to 5 which means that the LEG formulae must be recalculated for every iteration because of the variation in inlet pressure.

The call for CON #3, the case drain, first gets the value of the flow guess for the case drain flow and then calculates the actuator leakage based

on the outlet flow ratio.

QACTLK = DT(DISVLV)*(DT(QACTC)-DT(QACTU))+DT(QACTU)

The pressure rise from inlet to case, DT(DELP13) is then calculated using the sum of the leakage flows divided by the coefficient of case to inlet leakage, D(COECIN). Since DT(DELP13) is based on a case drain flow, QCASDR/D(COECIN) is added along with DT(DELP13) to PQLEG (INEL, 5) for the constant pressure rise temperature.

1/D(COECIN), the case drain impedance is added to PQLEG(INEL, 6) and LCS (INEL, 7) is set to 5 so that the leg must be recalled each iteration since the leakage is dependent on theoutlet pressure and/or updat d DT(PINLET) is required by the other CON #2 calculation.

PQLEG (INEL, 11) is increased by DT(DELP13) and DT(PCASE) is initialized to PQLEG (INEL, 11).

A test at the start of both the case drain and outlet calculations checks to see if INX = 1 which can only be true if CON 2 and 3 are the first or only elements in this leg.

The calculation method was two interdependent pressure rises in two separate legs which is unfortunately necessary since the variables are not easily separated.

2000 SECTION

With the completion of the steady state calculations, where DT(PCASE), DT(POUTLT) and DT(PINLET) are initialized and a value for DT(DISVLV) is calculated, the pump state variables can be initialized, ready for the transient simulation.

The ratio DT(DISVLV) is used to initialize DT(DISACT) and DT(QACTU). Actuator velocity, DT(VELACT), valve displacement, DT(DISVLV), and DT(QACTC) are set to zero. DT(POUTM) is set equal to thepump outlet pressure, DT(POUTLT).

3000 SECTION

The 3000 section starts with a computation predicting the actuator displacement for the current time step. This value is usedin the computation of the actuator pressure.

The pump inlet pressure is determined from the input impedance, Z(L1), a pump inlet to case coefficient and pump flow as shown in the equivalent schematic diagram of the pump inlet model, Figure 6.51-2.

The next step is to determine the minimum (POUTMI) and maximum (POUTMX) pump output pressure range. The compensator valve displacement is then computed using an iterative technique as explained in the math model section. Once the valve position is known the actuator pressure and flow is computed along with the pump outlet pressure and flow.

The position of the valve, determines whether the actuator flow is going from the outlet to the actuator or from the actuator to case. The two equivalent circuit schematics used in the solution process are shown in Figure 6.51-2.

Figure 6.51-2 also shows the schematic for computing the case drain output pressure and flow. QCASDR is the sum of all flows into and out of the case:

QCASDR = QPLEAK + QACTLK + DT(QACTC) - D(ARACT)*DT(VELACT)

- QCASIN - QOSIN*2.

DT(BULKC) is the impedance of the case. It is computed as the fluid bulk modulus times DELT divided by the case volume.

The final values of outlet and case drain pressures and flows are passed to the appropriate P and Q arrays. The pump output horsepower is computed as:

POWER = -Q(L2)*(P(L2)-P(L1))/6600.0

and is added to the previous value in DT(PPOWER).

A case pressure is computed using a simple integration.

DT(PPCASE)=2*DT(PCASE)-DT(PPCASE)

where

DT(PCASE)-LATEST CASE PRESSURE

DT(PPCASE)-PREDICTED CASE PRESSURE

If the pump inlet pressure is less than the minimum pump inlet pressure, a message is printed that gives the time at which pump cavitation occurs.

### 6.51.4 Approximations

The approximation used in the program is the rather gross linearization between maximum and minimum flows used in the steady state calculations.

The remaining calculations follow the math model which is itself a large approximation.

### 6.51.5 Limitations

The current pump subroutine does not attempt to describe the true cavitation effects that can be caused by improper filling of the pistons. The effect on hanger angle and RPM vary greatly from pump to pump.

However, it is also a condition which the designer should avoid, by improving the pump inlet supply system, to prevent the inlet pressure dropping to the point where cavitation effects are a concern.

A current limitation, is the correct steady state prediction of pump outlet pressure, when the system flow exceeds the pump capacity. The transient section will limit the flows, but flow limitation is not included in the steady state section.

6.51.6 Variable Names

| Name | Description | Dimension |
|---|---|---|
| A | Leakage coefficient from actuator to case | PSI/CIS**2 |
| ALPHA | Computation constant | -- |
| D(ARACT) | Actuator area | IN**2 |
| D(ARCOM) | Compensator valve area | IN**2 |
| B,BETA | Computation constants | -- |
| DT(BULKA) | Impedance of actuator volume | PSI/CIS |
| DT(BULKC) | Impedance of case volume | PSI/CIS |
| DT(BULKO) | Oil bulk modulus | PSI |
| C,CHI | Computation constants | -- |
| D(COEALK) | Coefficient of actuator leakage at zero pump displacement | CIS/PSI |
| D(COEALM) | Coefficient of actuator leakage at maximum pump displacement | CIS/PSI |
| DT(COEALS) | Dummy variable | -- |
| DT(COEALZ) | Dummy variable | -- |
| DT(COECAS) | Constant term use to determine case outlet pressure drop | PSI/CIS**2 |
| D(COECIN) | Coefficient of flow from case to inlet | CIS/PSI |
| COELKA | Dummy variable | -- |
| D(COEOSO) | Coefficient of outlet flow due to actuator motion | CIS/(IN/SEC) |
| D(COEPLK) | Coefficient of pump leakage (outlet to case) | CIS/PSI |
| D(COEVL1) | Discharge coefficient outlet to actuator | -- |
| D(COEVL2) | Discharge coefficient actuator to case | -- |
| DT(COEV1) | Constant term used to determine outlet to actuator pressure drop | PSI/CIS**2 |

| Name | Description | Dimension |
|---|---|---|
| DT(COEV2) | Constant term used to determine actuator to case pressure drop | PSI/CIS**2 |
| CONA, CONB | Dummy variables | -- |
| C1 | Inlet characteristic pressure | PSI |
| C2 | Outlet characteristic pressure | PSI |
| C3 | Case drain characteristic pressure | PSI |
| DT(DELP13) | Pressure drop from case to inlet | PSI |
| DT(DELP23) | Pressure drop from outlet to case | PSI |
| DIS | Dummy variable | -- |
| DT(DISACT) | Actuator displacement | IN |
| D(DISAM) | Actuator Position at maximum pump displacement | IN |
| D(DISAMI) | Actuator position at minimum pump displacement | IN |
| D(DISP) | Theoretical maximum pump displacement changed to IN**3/IN/RPM | IN**3/REV |
| DT(DISVLV) | Valve displacement | IN |
| D(DISVM) | Maximum valve displacement | IN |
| DPDAMP | Dummy variable | -- |
| D(DPVAC) | Pressure at which valve is open from outlet to actuator | PSI |
| D(DPVOA) | Pressure at which valve starts to open from outlet to actuator | PSI |
| DP1, DP2 | Dummy variables | -- |
| D(FLOFRC) | Flow force on valve spool | -- |
| GAIN | Closed loop pump gain | -- |
| D(INERT) | Hanger inertia | LBS-SEC**2/IN |
| D(INPPSI) | Spring rate of spool changed to IN/PSI | LB/IN |
| DT(ITUP ),DT(MITUP) | Iteration constants | -- |

| Name | Description | Dimension |
|---|---|---|
| OPOUT | Previous value of outlet pressure | PSI |
| D(PACCP) | Actuator pressure due to piston acceleration at 3600 rpm & maximum pump displacement | PSI |
| DT(PACTU) | Actuator pressure | PSI |
| PACTUO | Previous value of actuator pressure | PSI |
| PACTUR | Reference actuator pressure | PSI |
| DT(PCASE) | Case pressure | PSI |
| D(PDAMP) | Hanger damping | PSI/IN/SEC |
| PDISA | Predicted actuator displacement | IN |
| D(PDISAC) | Pressure at 3600 RPM and maximum pump displacement: changed to rate of change of pressure with actuator position | PSI |
| DT(PINLET) | Inlet pressure | PSI |
| D(PINMIM) | Minimum inlet pressure | PSI |
| DT(POUTLT) | Outlet pressure | PSI |
| POUTM,POUTMI POUTMX | Dummy variables | -- |
| POWER | Pump output horsepower | HP |
| DT(PPCASE) | Predicted case pressure | PSI |
| DT(PPOWER) | Cumulative output horsepower | HP |
| DT(PRPM) | Pump operating speed | RPM |
| D(PSLRPM) | Slope of pressure vs RPM curve | PSI/RPM |
| D(PSPEED) | Pump operating speed | RPM |
| D(PSPRIM) | Actuator pressure due to spring force at maximum pump displacement adjusted to slope PSI/IN | PSI |
| D(PSPRIZ) | Actuator pressure due to spring force at zero pump displacement | PSI |
| D(PZRPM) | Actuator pressure inputted at 3600 RPM and zero pump displacement; adjusted to zero RPM | PSI |

| Name | Description | Dimension |
|---|---|---|
| DT(QACTC) | Actuator flow with valve is open from actuator to case | CIS |
| QACTLK | Leakage flow from actuator to case | CIS |
| DT(QACTU) | Actuator flow when valve is open from outlet to actuator | CIS |
| QCASDR | Sum of all flows into and out of the case | CIS |
| QCASIN | Flow from case to inlet | CIS |
| DT(QINLET) | Inlet flow | CIS |
| QNET | Net actuator flow | CIS |
| QOSIN | Outlet flow due to actuator motion | CIS |
| DT(QOUTLT) | Outlet flow | CIS |
| QPLEAK | Pump leakage flow | CIS |
| QPUMP | Pump flow | CIS |
| TPOUT | Dummy variable | -- |
| DT(VELACT) | Actuator velocity | IN/SEC |
| D(VLVOL) | Valve overlap | IN |
| D(VOLACT) | Actuator volume | IN**3 |
| D(VOLCAS) | Case volume | IN**3 |
| VOLD | Previous actuator velocity | IN/SEC |
| D(VOLOUT) | Outlet volume | IN**3 |
| D(WIDTH) | Slot width | IN |
| ZOUT | Outlet impedance | PSI/CIS |

## 6.51.7 Subroutine Listing

```
      SUBROUTINE PUMP51 (D,DT,DD,L)
C **** REVISED MARCH 25, 1976 ****
C          X VERSION OF THE YPUMP54 SUBROUTINE  7 APR 76
      COMMON NTELPL,NTOLPL,IPT,IPOINT,NPTS,INEL,KNEL,NTOPL,NLPLT(61,3),
     1 PQLEG(90,12),LCS(90,10),ILEG(1400),PN(90),QN(90)
      COMMON/SUB/PARM(150,9),PM(1500),QM(1500),P(300),Q(300),C(300)
     1,Z(300),RHO(20),S2ORHO(20),VISC(20),BULK(20),TEMP(20),PVAP(20)
     2,ATPRES,T,DELT,TFINAL,PLTDEL,PI,TITLE(20),LEGN,ICON
     3,KTEMP(99),LSTART(150),NLPT(150),LTYPE(99),NC(99),INX,INZ
     4,INV,ISTEP,NLINE,NEL,IND,IENTR,MNLINE,MNEL,MNLEG,MNNODE,MNPLOT
     5,MNLPTS,MDS
      DIMENSION D(37),DT(43),DD(1),L(10)
      INTEGER ARCOS,WIDTH,FLOFRC,COEVL1,COEVL2,ARACT,PSPRIZ,PSPRIA,
     1 PACCP,PZRPM,PSLRPM,PDISAC,PDAMP,DISP,DPVOA,DPVAC,
     2 DISAM,DISAMI,COEALK,COEALM,COEPLK,COECIN,VOLCAS,PINLIN,
     3 PSPEED,PRPM,PPOWER,FLAG,QACTU,QACTC,PACTU,POUTLT,
     4 PCASE,PINLET,COEV1,COEV2,BULKC,VLLACT,DISACT,DISVLV,
     5 COEALZ,COEALS,DELP13,DELP23,COLCAS,POUTM,BULKO,COLOSO
     6 ,QINLET,QOUTLT,DISVM,VLVOL,VOLACT,BULKA,VOLOUT,PPCASE
C      D( ) ARRAY *****
      DATA DPVOA/1/,INPPSI/2/,ARCOS/3/,WIDTH/4/,FLOFRC/5/,VLVOL/6/,
     1 COEVL1/7/,COEVL2/8/,ARACT/9/,PSPRIZ/10/,PSPRIA/11/,
     2 PACCP/12/,PZRPM/13/,PDISAC/14/,PSLRPM/15/,PDAMP/16/,
     3 DISP/17/,DISAM/18/,DISAMI/19/,COEALK/20/,COEALM/21/,
     4 COEPLK/22/,COECIN/23/,VOLCAS/24/,PINLIN/25/,PSPEED/26/,
     5 COLOSO/27/,DISVM/28/,DPVAC/29/,INLRT/30/,VOLACT/31/,VOLOUT/32/
C      DT( ) ARRAY *****
      DATA PRPM/1/,PPOWER/2/,FLAG/3/,QACTU/4/,QACTC/5/,
     1 PACTU/7/,POUTLT/8/,PCASE/9/,PINLET/10/,COEV1/11/,COEV2/12/,
     2 BULKC/13/,VLLACT/14/,DISACT/15/,DISVLV/16/,KO/17/,COEALZ/18/,
     3 COEALS/19/,COLCAS/20/,POUTM/21/,DELP13/22/,DELP23/23/,BULKO/24/
     4 ,QINLET/25/,QOUTLT/26/,ITOP/27/,MITOP/28/,BULKA/29/,PPCASE/30/
C
C
      IF(IENTR) 1000,2000,3000
C *** 1000 SECTION
 1000 CONTINUE
      IF (INEL.NE.0) GO TO 1500
      DO 1001 I=1,29
 1001 DT(I)=0.0
      N=KTEMP(IND)
      IF(N.LT.11) N=N+10
C      .03043=VISC FOR MIL-H-83282 AT 100 F
      POWER=VISC(N)/.03043
      DT(BULKC)=BULK(N)*DELT/D(VOLCAS)
      DT(BULKA)=D(VOLACT)/(BULK(N)*DELT)
      D(VOLOUT)=D(VOLOUT)/(BULK(N)*DELT)
      DT(COEV1)=D(COEVL1)*S2ORHO(KTEMP(IND))*D(WIDTH)
      DT(COEV2)=D(COEVL2)*S2ORHO(N)*D(WIDTH)
      DT(COLCAS)=1.4142135/((.023962*S2ORHO(KTEMP(IND)))**2)
```

6.51.7 (Continued)

```
      DT(COEALZ)=D(COEALK)*POWER
      DT(COEALS)=D(COEALM)*POWER
      D(INERT)=D(INERT)/(DELT*D(ARACT))
      IF(D(FLOFRC).NE.0.0)GO TO 1250
      DT(COEALS)=(DT(COEALS)-DT(COEALZ))/D(DISAM)
      D(INPPSI)=D(ARCOM)/D(INPPSI)
      D(FLOFRC)=2.*D(WIDTH)*D(INPPSI)*(D(COEVL1)**2)*.359/D(ARCOM)
      D(DPVAC)=D(DPVOA)-D(VLVOL)/D(INPPSI)
      D(DISP)=D(DISP)/60./D(DISAM)
      D(PSPRIM)=(D(PSPRIM)-D(PSPRIZ))/D(DISAM)
      D(PACCP)=D(PACCP)/(D(DISAM)*3600.**2)
      D(PDISAC)=(D(PDISAC)-D(PZRPM))/D(DISAM)
      D(PZRPM)=D(PZRPM)-D(PSLRPM)*3600.
 1250 DT(PRPM)=D(PSPEED)
      DT(BULKO)=BULK(KTEMP(IND))*D(DPVOA)/232000.
C***  THIS SECTION CALCULATES THE STEADY STATE CHARCTERISTICS
C     OF THE PUMP OUTPUT PRESSURE VERSUS FLOW.
      COEFKA=DT(COEALZ)
      DT(POUTLT)=D(DPVOA)+2.0
 1260 DT(QACTC)=DT(QACTU)
 1270 DT(PACTU)=D(PSPRIZ)+DT(DISACT)*D(PSPRIM)+DT(PCASE)
     1 +DT(DISACT)*(D(PACCP)*DT(PRPM)**2+D(PDISAC))
     2 +(D(PZRPM)+D(PSLRPM)*DT(PRPM))*DT(BULKO)/DT(POUTLT)
      IF(IENTR.EQ.0)RETURN
      DT(QACTU)=COEFKA*DT(PACTU)
      DT(DISACT)=D(DISAM)
      COEFKA=DT(COEALZ)+DT(DISACT)*DT(COEALS)
      IF(DT(QACTC).EQ.0.)GO TO 1260
      DT(POUTM)=DT(POUTLT)
      DT(QINLET)=D(DISP)*DT(PRPM)*D(DISAM)-DT(POUTM)*D(COEPLK)
     1 -DT(QACTU)
      DT(QOUTLT)=-(D(COEPLK))*DT(POUTLT)-DT(QACTC)
      DT(DELP23)=-DT(QOUTLT)/(DT(QINLET)-DT(QOUTLT))
      DT(DISVLV)=.5
      DT(DELP13)=10.
      GAIN=.7/(1.+20.*DT(COEVL)*D(INPPSI)*Z(L(2)))
      DT(ITUP)=GAIN
      DT(ITDP)=1.-GAIN
      RETURN
C
C      STEADY STATE CALCULATION SECTION
C     IND=COMPONENT #,KNLL=CONNECTION #,INLL=LEG #
C     CON #1=INLET,CON #2=OUTLET,CON #3=CASE DRAIN
C     THE INLET IS A NODAL POINT IN THE SYSTEM
C
 1500 IF(KNLL-2)1510,1530,1520
 1510 DT(PINLET)=POLEG(INLL,11)
      GO TO 1600
 1520 IF(INX.NE.1) GO TO 1700
      PCASDR=POLEG(INLL,1)*POLEG(INLL,2)
```

6.51.7 (Continued)

```
      DT(6)=QCASDR
      QACTLK=DT(DISVLV)*(DT(QACTC)-DT(QACTU))+DT(QACTU)
      DT(DELP13)=(DT(POUTn)*D(COEPLK)+QACTLK-QCASDR)/D(COECIN)
      PQLEG(INEL,5)=PQLEG(INEL,5)+DT(DELP13)+QCASDR/D(COECIN)
      PQLEG(INEL,6)=PQLEG(INEL,6)+1.0/D(COECIN)
      LCS(INEL,7)=5
      PQLEG(INEL,11)=PQLEG(INEL,11)+DT(DELP13)
      DT(PCASL)=PQLEG(INEL,11)
      PQLEG(INEL,8)=PQLEG(INEL,8)+DT(COECAS)
      PQLEG(INEL,11)=PQLEG(INEL,11)-DT(COECAS)*(QCASDR**2)
      GO TO 1600
 1530 IF(INK.NE.1) GO TO 1700
      DT(DISVLV)=PQLEG(INEL,1)*PQLEG(INEL,2)-DT(QOUTLT)
      DT(DISVLV)=DT(DISVLV)/(DT(QINLET)-DT(QOUTLT))
      IF(DT(DISVLV).LT.0.0) DT(DISVLV)=0.0
      IF(DT(DISVLV).GT.1.0) DT(DISVLV)=1.0
      PQLEG(INEL,5)=PQLEG(INEL,5)+DT(POUTn)+DT(DELP13)
      PQLEG(INEL,6)=PQLEG(INEL,6)+0.00001
      PQLEG(INEL,11)=PQLEG(INEL,11)+DT(POUTn)+DT(DELP13)
      DT(POUTLT)=PQLEG(INEL,11)
      LCS(INEL,7)=5
 1600 RETURN
 1700 WRITE(6,1800) IND,KNEL,INEL
 1800 FORMAT(5X,46H CALL SEQUENCE ERROR DETECTED IN COMPONENT NO  ,
     1 I5,14H CONNECTION NO  ,I5,7H LEG NO  ,I5)
      WRITE(6,943)
  943 FORMAT(10X,33HPROGRAM STOP IN SUBROUTINE PUMP51)

      STOP 6054
C *** 2000 SECTION
 2000 CONTINUE
      DT(DISACT)=DT(DISVLV)*D(DISAn)
      DT(QACTC)=DT(DISVLV)*(DT(QACTC)-DT(QACTU))+DT(QACTU)
      DT(QACTC)=0.0
      DT(DISVLV)=0.0
      DT(PPCASL)=DT(PCASL)
      GO TO 1270
C
C *** 3000 SECTION
 3000 CONTINUE
      ICOUNT=0
C
C     CALCULATE TRANSIENT RESPONSE OF PUMP
C
      POWER=0.0
      L1=L(1)
      L2=L(2)
      L3=L(3)
      C1=C(L1)
      C2=C(L2)
```

6.51.7 (Continued)

```
      C3=C(L3)
      VOLD=DT(VELACT)
      DPDA.P=D(PDA.P)
      DPDA.P=D(ARACT)/(DPDA.P+D(INLRT))
      DT(QACTU)=0.0
      DT(QACTC)=0.0
      PACTUO=DT(PACTU)
      PDISA=DT(DISACT)+DT(VELACT)*DELT
      PACTUR=D(PSPRIZ)+PDISA*D(PSPRIM)+VOLD*D(INERT)
     1 +PDISA*(D(PACCP)*DT(PRPM)**2+D(PDISAC))+DT(PPCASE)
     2 +(D(PSRPM)+D(PSLRPM)*DT(PRPM))*DT(BULKO)/DT(POUTLT)
      QOSIN=-VOLD*D(COLOSO)
      QPUMP=PDISA*D(DISP)*DT(PRPM)+QOSIN
      DT(PINLET)=(C1/Z(L1)+DT(PPCASE)*D(COLCIN)-QPUMP)
     1  /(1.0/Z(L1)+D(COLCIN))
      IF(DT(PINLET).LT.D(PINMIN)) DT(PINLET)=D(PINMIN)
      DT(QINLET)=(C1-DT(PINLET))/Z(L1)
      QCASIN=(DT(PPCASE)-DT(PINLET))*D(COLCIN)
      QPUMP=DT(QINLET)+QCASIN+QOSIN
      COLLKA=DT(COLALZ)+PDISA*DT(COLALS)
      ZOUT=Z(L2)/(1.0+Z(L2)*D(COLPLK)+Z(L2)*D(VOLOUT))
      QPOUT=DT(POUTLT)
      B=1./(DPDA.P+DT(BULKA)+COLLKA)
      DP1=(DT(PPCASE)*COLLKA+PACTUO*DT(BULKA)+PACTUR*DPDA.P)*B
      DP2=B*(PACTUR*DPDA.P+PACTUO*DT(BULKA)+DT(PPCASE)*(COLLKA-1./B))
3200  POUT.X=(C2/Z(L2)+QPUMP+DT(PPCASE)*D(COLPLK)+D(VOLOUT)*QPOUT)*ZOUT
      POUT.I=D(DPVOA)+DT(PPCASE)
      IF(POUT.X.LE.POUT.I) GO TO 3220
      DT(POUTLT)=POUT.X*DT(.ITUP)+POUT.I*DT(.ITUP)
3210  DIS=(DT(POUTLT)-POUT.I)*D(INPRSI)
      DIS=DIS/(D(FLOFRC)*(DT(POUTLT)-DT(PACTU))+1.0)
      IF(DIS.GT.D(DISVA)) DIS=D(DISVA)
3215  IF(POWER.EQ.0.0) GO TO 3216
      CONA=(DIS*DT(COLV1))**2/(2.*COLLKA)
      CONB=2.*ABS(DT(POUTLT)-DT(PPCASE))*COLLKA/CONA
      DT(QACTU)=CONA*(SQRT(1.0+CONB)-1.0)
      WRITE(6,997)CONA,CONB,COLLKA,DT(PPCASE),DT(COLV1),DT(POUTLT)
997   FORMAT(3X,6E12.5)
      GO TO 3217
3216  CONTINUE
      A=DIS*DT(COLV1)
      DT(QACTU)=-3*A**2/2.+A/2.*SQRT((A*3)**2+4*(DT(POUTLT)-DP1))
      DT(PACTU)=DT(POUTLT)-(DT(QACTU)/A)**2
3217  TPOUT=POUT.X-DT(QACTU)*ZOUT
      IF(ABS(TPOUT-DT(POUTLT)).LT.0.05) GO TO 3230
      DT(POUTLT)=DT(POUTLT)*DT(.ITUP)+TPOUT*DT(ITUP)
      ICOUNT=ICOUNT+1
      IF(ICOUNT.EQ.25)WRITE(6,999)ICOUNT
      IF(ICOUNT.EQ.25)WRITE(6,998)DT(POUTLT),TPOUT,POUT.X,POUT.I
998   FORMAT(10X,4F20.5)
```

**6.51.7 (Continued)**

```
      IF(ICOUNT.EQ.25)GO TO 3230
  999 FORMAT(10X,13HEXCEEDED ITER,I10)
      GO TO 3210
C
C     FLOW FROM ACTUATOR PISTON TO CASE
 3220 POUTMI=D(DPVAC)+DT(PPCASE)
      IF(POUTMX.GE.POUTMI) GO TO 3230
      DIS=(POUTMI-POUTMX)*D(INPPSI)
      DIS=DIS/((DT(PACTU)-DT(PPCASE))*D(FLOFRC)+1.0)
      IF(DIS.GT.D(DISVM))DIS=D(DISVM)
 3226 CONTINUE
      A=DIS*DT(COEV2)
      DT(QACTC)=-B*A**2/2.+A/2.*SQRT((A*B)**2+4*DP2)
      DT(PACTU)=DT(PPCASE)+(DT(QACTC)/A)**2
 3230 IF(POWER.NE.0.0) GO TO 3250
C
C     TEST PISTON DISPLACEMENT AGAINST MAXIMUM STROKE
      QACTLK=(DT(PACTU)-DT(PPCASE))*COELKA
      QNET=DT(QACTU)-QACTLK-DT(QACTC)-(DT(PACTU)-PACTUO)*DT(BULKA)
      DT(VELACT)=-QNET/D(ARACT)
      DT(DISACT)=DT(DISACT)+(VOLO+DT(VELACT))/2.*DELT
      CALL XLIMIT(DT(DISACT),DT(VLLACT),POWER,D(DISAMI),D(DISAM))
      IF(POWER.EQ.0.0) GO TO 3300
      IF(DT(PINLET).LE.D(PINMIN)) GO TO 3240
      QPUMP=D(DISP)*DT(PRPM)*DT(DISACT)
      QOSIN=0.0
 3240 IF(POWER.EQ.-1.0) GO TO 3200
      DT(QACTC)=0.0
      DT(PACTU)=DT(PPCASE)
      QACTLK=0.0
      GO TO 3300
 3250 DT(PACTU)=DT(PPCASE)+DT(QACTU)/COELKA
      QACTLK=DT(QACTU)
 3300 P(L1)=DT(PINLET)
      Q(L1)=DT(QINLET)
      QPLLAK=(DT(POUTLT)-DT(PPCASE))*D(COEPLK)
      Q(L2)=-(QPUMP-QPLLAK-DT(QACTU)-(DT(POUTLT)-OPOUT)*D(VOLOUT))
      DT(DISVLV)=(DT(POUTLT)-DT(PPCASE)-D(DPVOA))*D(INPPSI)
      DT(POUTLT)=C(L2)-Q(L2)*Z(L2)
      P(L2)=DT(POUTLT)
      QCASDR=QPLLAK+QACTLK+DT(QACTC)-D(ARACT)*DT(VELACT)-QCASIN-QOSIN*2.
      DT(6)=QCASDR
      ALPHA=DT(COECAS)
      BETA=Z(L3)+DT(BULKC)
      CHI=DT(PCASE)+QCASDR*DT(BULKC)-C(L3)
      Q(L3)=(-BETA+SQRT(BETA**2+4.*ALPHA*ABS(CHI)))/(2.*ALPHA)
      Q(L3)=SIGN(Q(L3),-CHI)
      DT(PPCASE)=DT(PCASE)
      DT(PCASE)=DT(PCASE)+(QCASDR+Q(L3))*DT(BULKC)
      P(L3)=C(L3)-Q(L3)*Z(L3)
```

6.51.7 (Continued)

```
      DT(PPCASE)=2.*DT(PCASE)-DT(PPCASE)
C
      POWER=-Q(L2)*(P(L2)-P(L1))/6600.0
      DT(PPOWER)=POWER+DT(PPOWER)
      IF(DT(PINLET).LE.D(PINMIN)) WRITE(6,3310) T
      RETURN
 3310 FORMAT(2X,36H******** PUMP CAVITATION ONSET AT T= ,E12.5)
      END
```

## 6.54 SUBROUTINE PUMP54

Subroutine PUMP54 was set up to model the space shuttle (F-14) pump, which is basically a simple in line piston pump, though it incorporates many mechanical refinements and sophisticated design features. This model is intended for use by system designers and is primarily aimed at the study of pump system stability under dynamic loading conditions.

The model incorporates the effects of case drain dynamics, since the shuttle pump output pressure is referenced to case and the actuator discharges to the case, and displaces case volume when it is moving.

The treatment of leakage and damping characteristics are rudimentary.

The dynamics of the swash plate, yoke or hanger, are complex. For the model the effects of the dynamic forces on the actuator are included. These forces push the pump to maximum flow, the hanger spring provides this force on startup.

In addition the hanger offset creates a negative flow at the pump inlet and outlet when the hanger is moving toward maximum flow, and this has a destabilizing effect, when the swash response is very fast.

Some of the hanger forces are oscillatory but no attempt has been made to describe this effect, except that the magnitude is sufficient to keep the hanger in motion, so we have ignored the effects of static friction. This is an assumption that helps the simulation by keeping the integration of the hanger velocity a continuous function, between its mechanical stops.

The compensator valve flow characteristics are a significant part of the model. The forces on the valve are a combination of the outlet pressure forces pushing against the case pressure and spring forces, with damping and flow forces acting in either direction.

The damping and flow forces are not included because of the low flow levels of this particular valve.

6.54.1 Math Model

A simplified diagram of the pressure regulated variable displacement pump is shown in Figure 6.54-1.

An equivalent circuit schematic diagram for the pump model is shown in Figure 6.54-2.

Pump Displacement Flow

For the pump inlet the displacement flow is computed as follows:

QPUMP = D(DISP)*DT(PRPM)*DT(DISACT)

if D(DISAMI)< DT(DISACT)< D(DISAM)

or QPUMP = QINLET + QCASIN

if PINLET≤PINMIN

Actuator Pressure

The actuator pressure is based on the contributions of the spring force, case pressure, outlet pressure and pump rpm, plus the reaction force due to velocity damping which is generated when the hanger is moving. The input data establishing actuator pressure for pump operating conditions is modified to give a simpler algorithm for the transient calculations.

The input data gives the actuator pressure due to the spring force at maximum flow, D(PSPRIM) and at zero flow, D(PSPRIZ). P(PZRPM) is input as the pressure at zero actuator displacement and 3600 rpm. It is then modified to give the pressure at zero rpm by subtracting from it the slope of the pressure versus rpm curve, which is input as D(PSLRPM).

Using the F-15 pump experimental data and the actuator pressure predictions generated by the HSFR program, a formula was derived which related actuator pressure to the output pressure/bulk modulus ratio.

The test data collected at different fluid temperatures showed a need for temperature correction which was obtained via the bulk modulus. A

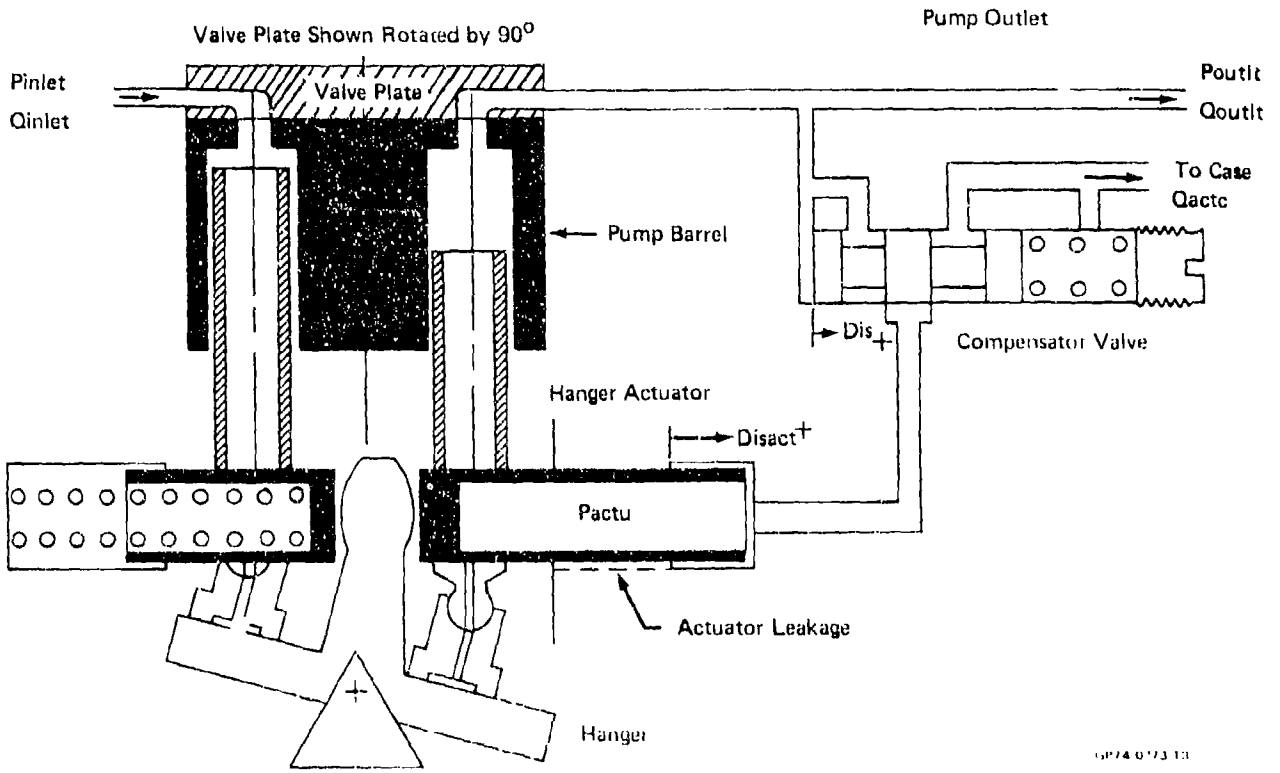
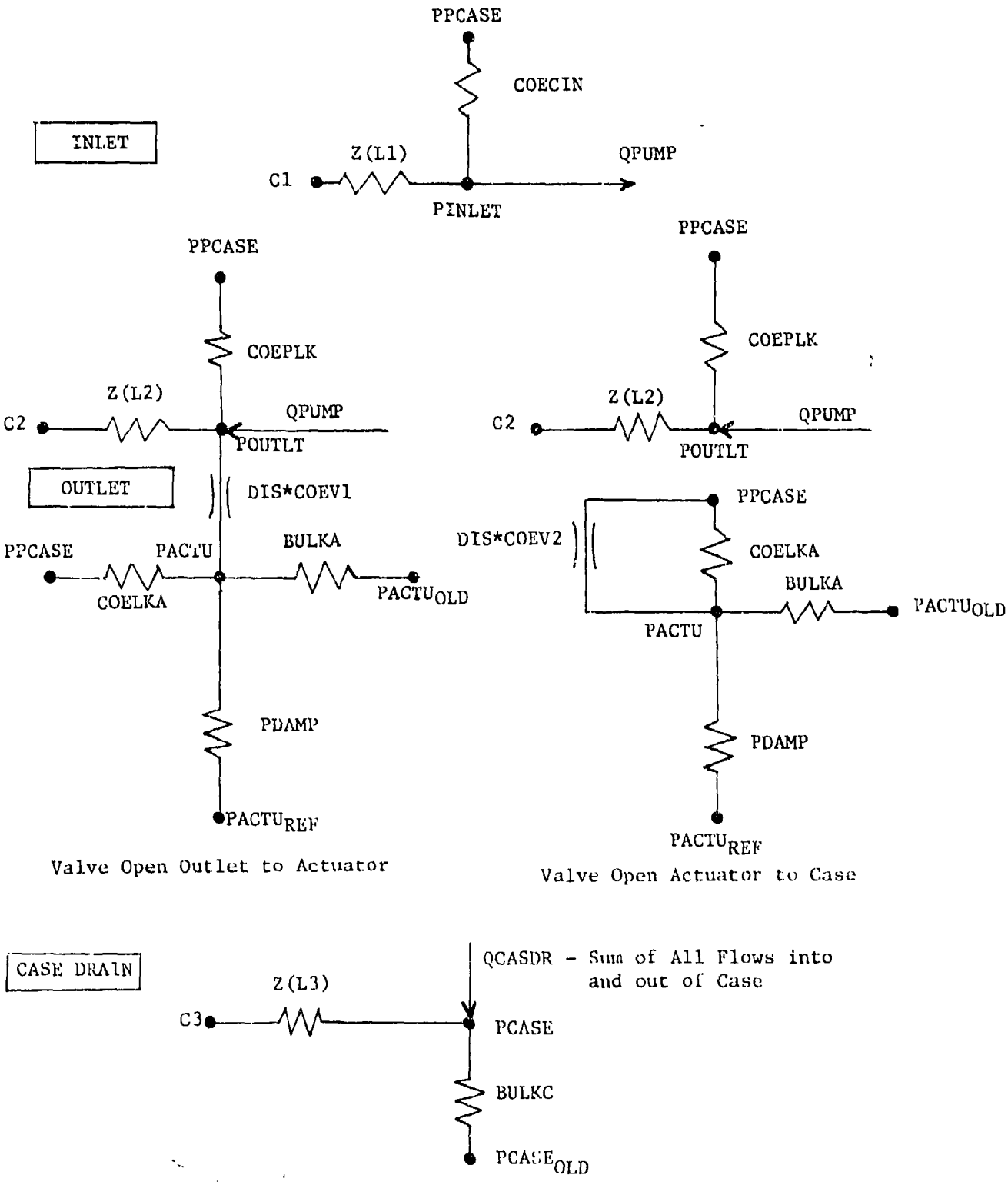



Figure 6.54-1

TYPE NO. 54 PRESSURE REGULATED VARIABLE DISPLACEMENT PUMP




Valve Open Outlet to Actuator

Valve Open Actuator to Case

SCHEMATIC DIAGRAM FOR THE ORBITER PUMP MODEL
FIGURE 6.54-2

correction factor DT(BULKO)/DT(POUTLT) is used where

DT(BULKO) = BULK(KTEMP(IND))*D(DPVOA)/223000

D(DPVOA) is the reference pressure and 223000 is the reference bulk modulus of MIL-H-5606B at 3000 psi and 130°F.

The actuator pressure is then computed as

```
DT(PACTU) = D(PSPRIZ)+PDISA*D(PSRIM) - VOLD*D(PDAMP)
          + PDISA*(D(PACCP)*DT(PRPM)**2 + D(PDISAC))
          + DT(PCASE) + (D(PZRPM) + D(PSLRPM)*DT(PRPM))
          *DT(BULKO)/DT(POUTLT)
```

where

PDISA = PREDICTED ACTUATOR POSITION BASED ON THE PREVIOUS POSITION AND VELOCITY

(=DT(DISACT) + DT(VELACT)*DELT)

VOLD = ACTUATOR VELOCITY (IN/SEC)

D(PACCP) = D(PACCP)/(D(DISAM)*3600**2)

D(PACCP) IS INPUT AS THE PRESSURE DUE TO PISTON ACCELERATION AT 3600 RPM AND MAXIMUM STROKE

D(PSPRIM) = (D(PSPRIM)-D(PSPRIZ))/D(DISAM)

D(PSPRIM) IS INPUT AS ACTUATOR PRESSURE DUE TO SPRING FORCE AT MAXIMUM PUMP DISPLACEMENT

D(PDISAC) = (D(PDISAC)-D(PZRPM))/D(DISAM)

D(PDISAC) IS INPUT AS ACTUATOR PRESSURE AT 3600 RPM AND MAXIMUM PUMP DISPLACEMENT

D(PZRPM) = D(PZRPM)-D(PSLRPM)*3600

D(PZRPM) IS INPUT AS ACTUATOR PRESSURE AT 3600 RPM AND ZERO PUMP DISPLACEMENT

The actuator pressure and damping characteristics are important variables since they govern how fast the pump goes from zero to full flow. No easy way exists to obtain these values accurately from purely dimensional data. The HSFR program is able to get within 30% of the measured actuator pressure and gives reasonable predictions of the variations with temperature and rpm. Measurements of actuator pressure and damping characteristics require a complex set of instrumentation and analysis of the data to extract the variables would require inspired judgement. Therefore a reasonable initial actuator pressure D(PZRPM) which includes estimated values for the contributions of the spring is 800 psi. Other contributions to actuator pressure are input as zero.

Compensator Valve

The compensator valve position is assumed to be directly proportional to the differential pressure between outlet and case. The lag, due to the valve damping and inertia, is very small compared to the response of the hanger and is ignored.

The valve spring rate D(INPPSI) is input in lbs/in and converted to in/psi,

D(INPPSI) = D(ARCOM)/D(INPPSI)

The differential pressure at which the valve opens from outlet to actuator D(DPVOA) is used to determine the valve position. Two pressures POUTMX and POUTM1 are derived, POUTMX is the maximum outlet pressure that can be obtained assuming zero flow from the outlet to the actuator.

POUTMX = (C2/Z(L2) + QPUMP + DT(PCASE)*D(COEPLK))*ZOUT

where

ZOUT = Z(L2)/(1.0 + Z(L2)*D(COEPLK))

QPUMP = DT(QINLET) + QCASIN + QOSIN

The minimum pressure, POMTMI, which is the outlet pressure when the valve is just about to open is computed as:

POUTMI = D(DPVOA) + DT(PCASE)

The actual outlet pressure lies between these limits and is obtained by iteration. The valve orifice area is calculated using the trigonometric formula for round port orifices. The valve displacement for

POUTMX>POUTMI is

DIS = (DT(POUTLT) - POUTMI)*D(INPPSI)

The orifice area is computed as

AREA = D(SRAD)*ACOS(B/D(RAD)) - B*SQRT(D(SRAD) - B**2)

where B = D(RAD) - DIS

D(RAD) = RADIUS OF METERING HOLE

D(SRAD) = D(RAD)**2

This area calculation works within the limits of the port diameter. For the computation, two ports diametrically opposite are assumed. When the valve is within the overlap region, the valve flow is set to zero. No allowance is made for leakage due to diametrical clearance within the valve.

For the initial guess of outlet pressure DT(POUTLT) is computed at .75*POUTMX + .25 POUTMI. This pressure is used to compute the valve position, area, and flow into the actuator DT(QACTU). The actuator flow is then used to recompute the pressure.

TPOUT = POUTMX - DT(QACTU)*ZOUT

A check is made to see if the recomputed flow is within .05 psi of the pressure valve and if it is not, the outlet pressure is updated by

DT(POUTLT) = DT(POUTLT)*.15 + TPOUT*.85

The choice of the .15/.85 ratio and the initial guess ratio of .75/.25 was made using the actual area, displacement and load impedance data which is applicable to its installation in the space shuttle. For other installations adjustments of these ratios could reduce the number of iterations required to achieve a balance. The tolerance of .05 psi could also be increased with little penalty.

Flow From Actuator Piston to Case

When POUTMI>POUTMX the valve is either closed or open from actuator to case. POUTMI is changed to

POUTMI = D(DPVAC) + DT(PCASE)

and if POUTMI is still greater than POUTMX, the valve is open allowing flow out of the actuator so that the pump flow QPUMP increases. The valve displacement area and flow (DT(QACTC)) are recalculated. Since the flow does not affect outlet pressure, no iteration is necessary.

The actuator leakage to case is assumed to be laminar since the passage is small around the actuator barrel. The actuator leakage is computed as

QACTLK = COELKA*(DT(PACTU) - DT(PCASE))

with all the actuator flows known, the actuator velocity is calculated

DT(VELACT) = -QNET/D(ARACT)

where

QNET = DT(QACTU) - DT(QACTC) - QACTLK

The new actuator position is then

DT(DISACT) = DT(DISACT) + (VOLD+DT(VELACT))*DELT/2.

A check is made to determine if the actuator is at the stroke limits. If it is the actuator flow must be recalculated. If the actuator is at maximum stroke (fully retracted with the pump at full stroke) then the actuator pressure drops to case pressure and DT(QACTC) is set to zero.

If the actuator is at minimum stroke (with the pump outlet flow negative), then DT(QACTU) and DT(PACTU) have to be recalculated so that the actuator leakage flow can be determined.

Pump Outlet Pressure and Flow

After the actuator pressure, flows and valve outlet pressure DT(POUTLT) are computed, pump outlet flow is calculated as:

Q(L2) = -(QPUMP) - QPLEAK - DT(QACTU))

actual pump output pressure is then

P(L2) = C(L2) - Q(L2)*Z(L2)

Pump Case Outlet Pressure and Flow

The case outlet flow is determined using the schematic of Figure 6.54-2 to write a linear equation in flow:

QCASDR = (QCASDR*DT(BULKC)-C3+DT(PCASE))/(Z(L3)+DT(BULKC))

Q(L3) = -QCASDR

The outlet case pressure is

P(L3) = C(L3)+QCASDR*Z(L3)

6.54.2 Assumptions

The assumptions in a model of this nature are almost too many to enumerate. By its very nature the pump is a complex piece of equipment with multiple leak paths across the port plate, down the side of the piston and out of the shoes. This whole set of leak paths have been linearized and assumed to be constant for a constant output pressure, which is no doubt rather hard to accept. The alternative would be go into very detailed calculations with the leakage dependent on the piston load, swash angle, RPM and anything else one could add. Unfortunately this too would probably be inaccurate so instead of an inaccurate complex leakage model we choose a simple leakage model, which could be improved when more data is available from the verification tests.

The forces on the swash plate are not taken into account as it rotates. Flow and leakage, are all treated as though the pump had a continuous output rather than the individual pumping pistons.

A model which includes the dynamics of each individual piston would by necessity be considerably more complex and consume much more computer time, but is is an alternative to what we have done here.

In all the calculations the bulk modulus is treated as a constant for the high pressure (output) side and as a different constant for the low pressure (inlet) side, the elastic expansion of the volume cavities is not included though a correction factor can be included in the model by increasing inputted volumes above their actual values.

Friction effects have not been included primarily because of the cost of putting them in, but in actual fact the forces on the swash plate have an

oscillatory content which tend to keep it in motion. The pulsations of the outlet also tend to keep the valve in motion so that friction effects would not normally be significant.

6.54.3 Computation

The pump subroutine is set up using the HYTRAN program commons, plus the D, DT, DD, and L arrays.

1000 SECTION

In the 1000 section the constants are initialized where desirable for more efficient computation. The variables are initialized depending on whether the pump is in a pressurized or depressurized operating mode.

The remaining part of Section 1000 deals with the calculations of the steady state pump characteristics.

In order to balance the pump at some steady state condition, it is first necessary to establish what the pump characteristics are, over the maximum range of pump flow.

To help in this it was assumed that these characteristics could be approximated by a straight line interpolation between the pump conditions at maximum and minimum flows which correspond to zero and maximum actuator displacement respectively.

A chain of interdependent calculations are needed to derive the maximum and minimum conditions. We have to establish:

DT(PACTU) = Actuator pressure

DT(QACTC) = Flow from actuator to case

DT(QACTU) = Flow from outlet to actuator through the valve

QCASDR = Case drain flow

QACTLK = Flow into the actuator due to leakage

DT(DISVLV) = Valve displacement

DT(POUTLT) = Valve chamber pressure which is the same as outlet pressure

The actuator pressure is dependent on the outlet pressure due to the pumping offset, and the valve flow is dependent on the difference between

outlet pressure and actuator pressure.

The initial flows (QACTU and QACTC) are computed at the appropriate valve positions and this information is used in the steady state portion of the program.

1500 SECTION

Steady State Calculations

The pump which has three connections, has a node located at the inlet.

The leg which has the inlet connection receives the pump inlet pressure from the steady state routine.

For the leg which has the outlet connection as its first element, a value of the output flow ratio is computed as

$$DT(DISVLV) = \frac{QGUESS-DT(QOUTLT)}{DT(QINLET)-DT(QOUTLT)}$$

The ratio is calculated to determine the percentage of actuator flow that is actual leakage flow (QACTLK) into the pump case.

The output pressure is determined by the computed maximum outlet pressure at maximum valve displacement DT(POUTM) plus the pressure drop from case to inlet, DT(DELP13).

The outlet pressure is added to PQLEG(INEL, 5) and the output impedance (.00001) is added to PQLPG(INEL, 6).

DT(POUTLT) is initialized to DT(POUTM) + DT(DELP13).

And the outlet pressure PQLEG (INEL, 11) is also increased by DT(POUTM) + DT(DELP13).

LCS(INCL, 7) is set to 5 which means that the LEG formulae must be recalculated for every iteration because of the variation in inlet pressure.

The call for CON #3, the case drain, first gets the value of the flow guess for the case drain flow and then calculates the actuator leakage based

on the outlet flow ratio.

QACTLK = DT(DISVLV)*(DT(QACTC)-DT(QACTU))+DT(QACTU)

The pressure rise from inlet to case, DT(DELP13) is then calculated using the sum of the leakage flows divided by the coefficient of case to inlet leakage, D(COECIN). Since DT(DELP13) is based on a case drain flow, QCASDR/D(COECIN) is added along with DT(DELP13) to PQLEG(INEL, 5) for the constant pressure rise temperature.

1/D(COECIN), the case drain impedance is added to PQLEG(INEL, 6) and LCS (INEL, 7) is set to 5 so that the leg must be recalled each iteration since the leakage is dependent on the outlet pressure and/or updated DT(PINLET) is required by the other CON #2 calculation.

PQLEG (INEL, 11) is increased by DT(DELP13) and DT(PCASE) is initialized to PQLEG (INEL, 11).

A test at the start of both the case drain and outlet calculations checks to see if INX = 1 which can only be true if CON 2 and 3 are the first or only elements in this leg.

The calculation method was two interdependent pressure rises in two separate legs which is unfortunately necessary since the variables are not easily separated.

2000 SECTION

With the completion of the steady state calculations, where DT(PCASE), DT(POUTLT) and DT(PINLET) are initialized and a value for DT(DISVLV) is calculated, the pump state variables can be initialized, ready for the transient simulation.

The ratio DT(DISVLV) is used to initialize DT(DISACT) and DT(QACTU). Actuator velocity, DT(VELACT), valve displacement, DT(DISVLV), and DT(QACTC) are set to zero. DT(POUTM) is set equal to the pump outlet pressure, DT(POUTLT).

3000 SECTION

The 3000 section starts with a computation predicting the actuator displacement for the current time step. This value is used in the computation of the actuator pressure.

The pump inlet pressure is determined from the input impedance, Z(L1), a pump inlet to case coefficient and pump flow as shown in the equivalent schematic diagram of the pump inlet model, Figure 6.54-2.

The next step is to determine the minimum (POUTMI) and maximum (POUTMX) pump output pressure range. The compensator valve displacement is then computed using an iterative technique as explained in the math model section. Once the valve position is known the actuator pressure and flow is computed along with the pump outlet pressure and flow.

The position of the valve, determines whether the actuator flow is going from the outlet to the actuator or from the actuator to case. The two equivalent circuit schematics used in the solution process are shown in Figure 6.54-2.

Figure 6.54-2 also shows the schematic for computing the case drain output pressure and flow. QCASDR is the sum of all flows into and out of the case:

```
QCASDR = QPLEAK + QACTLK + DT(QACTC) - D(ARACT)*DT(VELACT)
         - QCASIN - QOSIN*2.
```

DT(BULKC) is the impedance of the case. It is computed as the fluid bulk modulus times DELT divided by the case volume.

The final values of outlet and case drain pressures and flows are passed to the appropriate P and Q arrays. The pump output horsepower is computed as:

POWER = - Q(L2)*(P(L2)-P(L1))/6600.0

and is added to the previous value in DT(PPOWER).

Should the pump be depressurized at this time step the above calculations are bypassed and the pump output pressure is set to 800 + DT(PCASE) and the appropriate outlet and case drain pressures and flows are computed.

If the pump inlet pressure is less than the minimum pump inlet pressure, a message is printed that gives the time at which pump cavitation occurs.

### 6.54.4 Approximations

The approximations used in the program, are the formula used to generate the compensator valve flow areas, and the rather gross linearization between maximum and minimum flows used in the steady state calculations.

The remaining calculations follow the math model which is itself a large approximation.

### 6.54.5 Limitations

The current pump subroutine does not attempt to describe the true cavitation effects that can be caused by improper filling of the pistons. The effect on swash angle and RPM vary greatly from pump to pump.

However, it is also a condition which the designer should avoid, by improving the pump inlet supply system, to prevent the inlet pressure dropping to the point where cavitation effects are a concern.

A current limitation, is the correct steady state prediction of pump outlet pressure, when the system flow exceeds the pump capacity. The transient section will limit the flows, but flow limitation is not included in the steady state section.

6.54.6 Variable Names

| Name | Description | Dimension |
|---|---|---|
| D(ARACT) | Actuator area | IN**2 |
| D(ARCOM) | Compensator valve area | IN**2 |
| AREA | Valve opening area | IN**2 |
| BULK | Impedance of actuator volume | PSI/CIS |
| DT(BULKC) | Impedance of case volume | PSI/CIS |
| DT(BULKO) | Oil bulk modulus | PSI |
| C | Dummy variable | -- |
| D(COEALK) | Coefficient of actuator leakage at zero pump displacement | CIS/PSI |
| D(COEALM) | Coefficient of actuator leakage at maximum pump displacement | CIS/PSI |
| DT(COEALS) | Dummy variable | -- |
| DT(COEALZ) | Dummy variable | -- |
| D(COECIN) | Coefficient of flow from case to inlet | CIS/PSI |
| COELKA | Dummy variable | -- |
| D(COEOSO) | Coefficient of outlet flow due to actuator motion | CIS/(IN/SEC) |
| D(COEPLK) | Coefficient of pump leakage (outlet to case) | CIS/PSI |
| D(COEVL1) | Discharge coefficient outlet to actuator | -- |
| D(COEVL2) | Discharge coefficient actuator to case | -- |
| DT(COEV1) | Constant term used to determine outlet to actuator pressure drop | PSI/CIS**2 |
| DT(COEV2) | Constant term used to determine actuator to case pressure drop | PSI/CIS**2 |
| CONA,CONB | Dummy variables | -- |
| C1 | Inlet characteristic pressure | PSI |
| C2 | Outlet characteristic pressure | PSI |

| Name | Description | Dimension |
|---|---|---|
| C3 | Case drain characteristic pressure | PSI |
| DT(DELP13) | Pressure drop from case to inlet | PSI |
| DT(DELP23) | Pressure drop from outlet to case | PSI |
| DIS | Dummy variable | -- |
| DT(DISACT) | Actuator displacement | IN |
| D(DISAM) | Actuator position at maximum pump displacement | IN |
| D(DISAMI) | Actuator position at minimum pump displacement | IN |
| D(DISP) | Theoretical maximum pump displacement changed to IN**3/IN/RPM | IN**3/REV |
| DT(DISVLV) | Valve displacement | IN |
| D(DISVM) | Maximum valve displacement | IN |
| DPDAMP | Dummy variable | -- |
| D(DPVAC) | Pressure at which valve is open from outlet to actuator | PSI |
| D(DPVOA) | Pressure at which valve starts to open from outlet to actuator | PSI |
| D(INPPSI) | Spring rate of spool changed to IN/PSI | LB/IN |
| DT(ISYS) | System number | -- |
| D(KTIME) | Pressurization/depressurization time | SEC |
| D(PACCP) | Actuator pressure due to piston acceleration at 3600 RPM and maximum pump displacement | PSI |
| DT(PACTU) | Actuator pressure | PSI |
| DT(PCASE) | Case pressure | PSI |
| D(PDAMP) | Hanger damping | PSI/IN/SEC |
| PDISA | Predicted actuator displacement | IN |

| Name | Description | Dimension |
|---|---|---|
| D(PDISAC) | Pressure at 3600 RPM and maximum pump displacement: changed to rate of change of pressure with actuator position | PSI |
| DT(PINLET) | Inlet pressure | PSI |
| D(PINMIM) | Minimum inlet pressure | PSI |
| DT(POUTLT) | Outlet pressure | PSI |
| POUTM,POUTM1 POUTMX,POUTZ | Dummy variables | -- |
| POWER | Pump output horsepower | HP |
| DT(PPOWER) | Cumulative output horsepower | HP |
| DT(PRPM) | Pump operating speed | RPM |
| D(PSLRPM) | Slope of pressure vs RPM curve | PSI/RPM |
| D(PSPEED) | Pump operating speed | RPM |
| D(PSPRIM) | Actuator pressure due to spring force at maximum pump displacement adjusted to slope PSI/IN | PSI |
| D(PSPRIZ) | Actuator pressure due to spring force at zero pump displacement | PSI |
| D(PZRPM) | Actuator pressure inputted at 3600 RPM and zero pump displacement; adjusted to zero RPM | PSI |
| DT(QACTC) | Actuator flow with valve is open from actuator to case | CIS |
| QACTLK | Leakage flow from actuator to case | CIS |
| DT(QACTU) | Actuator flow when valve is open from outlet to actuator | CIS |
| QCASDR | Sum of all flows into and out of the case | CIS |
| QCASIN | Flow from case to inlet | CIS |
| DT(QINLET) | Inlet flow | CIS |
| QNET | Net actuator flow | CIS |

| Name | Description | Dimension |
|---|---|---|
| QOSIN | Outlet flow due to actuator motion | CIS |
| DT(QOUTLT) | Outlet flow | CIS |
| QPLEAK | Pump leakage flow | CIS |
| QPUMP | Pump flow | CIS |
| D(RAD) | Radius of valve port | IN |
| D(SRAD) | Initial radius squared | IN**2 |
| TPOUT | Dummy variable | -- |
| DT(VELACT) | Actuator velocity | IN/SEC |
| D(VLVOL) | Valve overlap | IN |
| D(VOLACT) | Actuator volume | IN**3 |
| D(VOLCAS) | Case volume | IN**3 |
| VOLD | Previous actuator velocity | IN/SEC |
| ZOUT | Outlet impedance | PSI/CIS |

## 6.54.7 Subroutine Listing

```
      SUBROUTINE PUMP54 (D,DT,DD,L)
C **** REVISED JUNE 25, 1976 ****
      DIMENSION D(37),DT(43),DD(1),L(10)
      COMMON NTELPL,NTOLPL,IPT,IPOINT,NPTS,INEL,KNEL,NTOPL,NLPLT(61,3),
     1 PQLEG(90,12),LCS(90,10),ILEG(1400),PN(90),QN(90)
      COMMON/SUB/PARM(150,9),PM(1500),QM(1500),P(300),Q(300),C(300)
     1,Z(300),RHO(20),S2ORHO(20),VISC(20),BULK(20),TEMP(20),PVAP(20)
     2,ATPRES,T,DELT,TFINAL,PLTDEL,PI,TITLE(20),LEGN,ICON
     3,KTEMP(99),LSTART(150),NLPT(150),LTYPE(99),NC(99),INX,INZ
     4,INV,ISTEP,NLINE,NEL,IND,IENTR,MNLINE,MNEL,MNLEG,MNNODE,MNPLOT
     5,MNLPTS,MDS
      INTEGER ARCON,RAD,SRAD,COEVL1,COEVL2,ARACT,PSPRIZ,PSPRIM,
     1 PACCP,PZRPM,PSLRPM,PDISAC,PDAMP,DISP,DPVOA,DPVAC,
     2 DISAM,DISAMI,COEALK,COEALM,COEPLK,COECIN,VOLCAS,PINMIN,
     3 PSPEED,PRPM,PPOWER,QACTU,QACTC,PACTU,POUTLT,
     4 PCASE,PINLET,COEV1,COEV2,BULKC,VELACT,DISACT,DISVLV,
     5 COEALZ,COEALS,DELP13,POUTZ,POUTM,BULKO,COLOSO
     6 ,QINLET,QOUTLT,VLVOL
C      D( ) ARRAY *****
      DATA DPVOA/1/,INPPSI/2/,ARCON/3/,RAD/4/,SRAD/5/,VLVOL/6/,
     1 COEVL1/7/,COEVL2/8/,ARACT/9/,PSPRIZ/10/,PSPRIM/11/,
     2 PACCP/12/,PZRPM/13/,PDISAC/14/,PSLRPM/15/,PDAMP/16/,
     3 DISP/17/,DISAM/18/,DISAMI/19/,COEALK/20/,COEALM/21/,
     4 COEPLK/22/,COECIN/23/,VOLCAS/24/,PINMIN/25/,PSPEED/26/,
     5 COLOSO/27/,DPVAC/29/,KTEMP/29/
C      DT( ) ARRAY *****
      DATA PRPM/1/,PPOWER/2/,QACTU/4/,QACTC/5/,
     1 PACTU/7/,POUTLT/8/,PCASE/9/,PINLET/10/,COEV1/11/,COEV2/12/,
     2 BULKC/13/,VELACT/14/,DISACT/15/,DISVLV/16/,KQ/17/,COEALZ/18/,
     3 COEALS/19/,POUTZ/20/,POUTM/21/,DELP13/22/,BULKO/23/
     4 ,QINLET/24/,QOUTLT/25/,ISYS/4/
C
C
      IF(IENTR) 1000,2000,3000
C *** 1000 SECTION
 1000 CONTINUE
      IF (INEL.NE.0) GO TO 1500
      DT(PPOWER)=0.0
      N=KTEMP(IND)
      IF(N.LT.11) N=N+10
C      .0248=VISC OF MIL-H-56060 AT 100 F
      POWER=VISC(N)/.0248
      DT(BULKC)=BULK(N)*DELT/D(VOLCAS)
      DT(COEV1)=2.0*D(COEVL1)*S2ORHO(KTEMP(IND))
      DT(COEV2)=2.0*D(COEVL2)*S2ORHO(N)
      DT(COEALZ)=D(COEALK)*POWER
      DT(COEALS)=D(COEALM)*POWER
      DT(COEALS)=(DT(COEALS)-DT(COEALZ))/D(DISAM)
      DT(6)=0.0
      IF(D(SRAD).NE.0.0) GO TO 1250
```

```
      D(SRAD)=D(RAD)**2
      D(INPPSI)=D(ARCOM)/D(INPPSI)
      D(DPVAC)=D(DPVOA)-D(VLVOL)/D(INPPSI)
      D(DISP)=D(DISP)/60./D(DISAM)
      D(PSPRIM)=(D(PSPRIM)-D(PSPRIZ))/D(DISAM)
      D(PACCP)=D(PACCP)/(D(DISAM)*3600.**2)
      D(PDISAC)=(D(PDISAC)-D(PZRPM))/D(DISAM)
      D(PZRPM)=D(PZRPM)-D(PSLRPM)*3600.
 1250 DT(PRPM)=D(PSPEED)
      DT(BULKO)=BULK(KTEMP(IND))*D(DPVOA)/223000.
C***  THIS SECTION CALCULATES THE STEADY STATE CHARCTERISTICS
C
      DT(POUTLT)=D(DPVOA)+55.0
      IF(D(KTIME).EQ.0.0) D(KTIML)=10000.
      IF(INV) 1020,1040,1030
 1020 IF(L(ISYS).NE.-INV) GO TO 1040
      GO TO 1035
 1030 IF(L(ISYS).EQ.INV) GO TO 1040
 1035 D(KTIME)=-D(KTIME)
 1040 IF(D(KTIME).LT.0.0)DT(POUTLT)=800.
      L(ISYS)=0
      IF(D(KTIME).LT.0.0) L(ISYS)=2
      DT(POUTM)=DT(POUTLT)
      DT(VELACT)=DT(POUTM)*D(CODPLR)
      DT(QINLET)=D(DISP)*DT(PRPM)*D(DISAM)-DT(VELACT)
      DT(QOUTLT)=-DT(VELACT)
      DT(QACTC)=0.0
      DT(QACTU)=0.0
      DT(DISVLV)=.5
      DT(PCASE)=0.0
      DT(DELP13)=10.0
      IF(D(KTIML).LT.0.0) RETURN
      DT(DISACT)=0.0
      COELKA=DT(COLALZ)
 1270 CONTINUE
      DT(PACTU)=DT(PCASE)+D(PSPRIZ)+DT(DISACT)*D(PSPRIM)
     1 +DT(DISACT)*(D(PACCP)*DT(PRPM)**2+D(PDISAC))
     2 +(D(PZRPM)+D(PSLRPM)*DT(PRPM))*DT(BULKO)/DT(POUTM)
      IF(IGNTR.EQ.0) RETURN
      DT(QACTU)=DT(PACTU)*COELKA
      IF(DT(QACTC).NE.0.0) GO TO 1295
      DT(QACTC)=DT(QACTU)
      DT(DISACT)=D(DISAM)
      COELKA=DT(COLALZ)+DT(DISACT)*DT(COLALS)
      GO TO 1270
 1295 DT(QINLET)=DT(QINLET)-DT(QACTU)
      DT(QOUTLT)=DT(QOUTLT)-DT(QACTC)
      RETURN
C
C       STEADY STATE CALCULATION SECTION
```

**6.54.7 (Continued)**

```
C      IND=COMPONENT #,KNEL=CONNECTION #,INEL=LEG #
C      CON #1=INLET,CON #2=OUTLET,CON #3=CASE DRAIN
C      THE INLET IS A NODAL POINT IN THE SYSTEM
C
 1500 IF(KNEL-2)1510,1530,1520
 1510 DT(PINLET)=PQLEG(INEL,11)
      GO TO 1600
 1520 IF(INX.NE.1) GO TO 1700
      QCASDR=PQLEG(INEL,1)*PQLEG(INEL,2)
      QACTLK=DT(DISVLV)*(DT(QACTC)-DT(QACTU))+DT(QACTU)
      DT(DELP13)=(DT(VELACT)+QACTLK-QCASDR)/D(COLCIN)
      PQLEG(INEL,5)=PQLEG(INEL,5)+DT(DELP13)+QCASDR/D(COLCIN)
      PQLEG(INEL,6)=PQLEG(INEL,6)+1.0/D(COLCIN)
      LCS(INEL,7)=5
      PQLEG(INEL,11)=PQLEG(INEL,11)+DT(DELP13)
      DT(PCASE)=PQLEG(INEL,11)
      GO TO 1600
 1530 IF(INX.NE.1) GO TO 1700
      DT(DISVLV)=PQLEG(INEL,1)*PQLEG(INEL,2)-DT(QOUTLT)
      DT(DISVLV)=DT(DISVLV)/(DT(QINLET)-DT(QOUTLT))
      IF(DT(DISVLV).LT.0.0) DT(DISVLV)=0.0
      IF(DT(DISVLV).GT.1.0) DT(DISVLV)=1.0
      PQLEG(INEL,5)=PQLEG(INEL,5)+DT(POUTLT)+DT(DELP13)
      PQLEG(INEL,6)=PQLEG(INEL,6)+.00001
      PQLEG(INEL,11)=PQLEG(INEL,11)+DT(POUTLT)+DT(DELP13)
      DT(POUTLT)=PQLEG(INEL,11)
      LCS(INEL,7)=5
 1600 RETURN
 1700 WRITE(6,1800) IND,KNEL,INEL
 1800 FORMAT(5X,46H CALL SEQUENCE ERROR DETECTED IN COMPONENT NO  ,
     1 I5,14H CONNECTION NO  ,I5,7H LEG NO  ,I5)
      WRITE(6,999)
  999 FORMAT(10X,33HPROGRAM STOP IN SUBROUTINE PUMP54
      STOP 6054
C *** 2000 SECTION
 2000 CONTINUE
      DT(DISACT)=DT(DISVLV)*D(DISAR)
      DT(VELACT)=0.0
      DT(QACTU)=DT(DISVLV)*(DT(QACTC)-DT(QACTU))+DT(QACTU)
      DT(QACTC)=0.0
      DT(DISVLV)=0.0
      DT(POUTLT)=DT(POUTLT)
      GO TO 1270
C
C *** 3000 SECTION
 3000 CONTINUE
C
C     CALCULATE TRANSIENT RESPONSE OF PUMP
C
      POWPMP=0.0
```

6.54.7 (Continued)



```
      L1=L(1)
      L2=L(2)
      L3=L(3)
      C1=C(L1)
      C2=C(L2)
      C3=C(L3)
      VOLD=DT(VELACT)
      DT(QACTU)=0.0
      DT(QACTC)=0.0
      D(KTIME)=D(KTIME)+DELT*L(ISYS)
      IF(T.GT.D(KTIME)) GO TO 4000
      PDISA=DT(DISACT)+DT(VELACT)*DELT
      DT(PACTU)=D(PSPRIZ)+PDISA*D(PSPRIA)-VOLD*D(PDAMP)
     1 +PDISA*(D(PACCP)*DT(PRPM)**2+D(PDISAC))+DT(PCASE)
     2 +(D(PZERO)+D(PSLOPE)*DT(PRPM))*DT(BULKO)/DT(POUTLT)
      QOSIN=-VOLD*D(COEOSO)
      QPUMP=PDISA*D(DISP)*DT(PRPM)+QOSIN
      DT(PINLET)=(C1/Z(L1)+DT(PCASE)*D(COECIN)-QPUMP)
     1  /(1.0/Z(L1)+D(COECIN))
      IF(DT(PINLET).LT.D(PINMIN)) DT(PINLET)=D(PINMIN)
      DT(QINLET)=(C1-DT(PINLET))/Z(L1)
      QCASIN=(DT(PCASE)-DT(PINLET))*D(COECIN)
      QPUMP=DT(QINLET)+QCASIN+QOSIN
      COELKA=DT(COLALZ)+PDISA*DT(COLALS)
      ZOUT=Z(L2)/(1.0+Z(L2)*D(COEPLK))
 3200 POUTMX=(C2/Z(L2)+QPUMP+DT(PCASE)*D(COEPLK))*ZOUT
      POUTMI=D(OPVOA)+DT(PCASE)
      IF(POUTMX.LT.POUTMI) GO TO 3220
      DT(POUTLT)=POUTMX*.75+POUTMI*.25
 3210 DIS=(DT(POUTLT)-POUTMI)*D(INPPSI)
      B=D(RAD)-DIS
      IF(B.LT.-D(RAD))GO TO 3214
      AREA=D(SRAD)*ACOS(B/D(RAD))-B*SQRT(D(SRAD)-B**2)
      GO TO 3215
 3214 AREA=D(SRAD)*PI
 3215 IF(POWER.EQ.0.0) GO TO 3216
      CONA=(AREA*DT(COLV1))**2/(2.*COLLKA)
      CONB=2*ABS(DT(POUTLT)-DT(PCASE))*COLLKA/CONA
      DT(QACTU)=CONA*(SQRT(1.0+CONB)-1.0)
      GO TO 3217
 3216 DT(QACTU)=DT(COLV1)*AREA*SQRT(DT(POUTLT)-DT(PACTU))
 3217 TPOUT=POUTMX-DT(QACTU)*ZOUT
      IF(ABS(TPOUT-DT(POUTLT)).LT.0.05) GO TO 3230
      DT(POUTLT)=DT(POUTLT)*.15+TPOUT*.85
      GO TO 3210
C
C     FLOW FROM ACTUATOR PISTON TO CASE
 3220 POUTMI=D(OPVAC)+DT(PCASE)
      IF(POUTMX.GE.POUTMI) GO TO 3230
      DIS=(POUTMI-POUTMX)*D(INPPSI)
```

**6.54.7 (Continued)**

```
      B=D(RAD)-DIS
      IF(B.LE.-D(RAD)) GO TO 3225
      AREA=D(SRAD)*ACOS(B/D(RAD))-B*SQRT(D(SRAD)-B**2)
      GO TO 3226
 3225 AREA=D(SRAD)*PI
 3226 DT(QACTC)=DT(COEV2)*AREA*SQRABS(DT(PACTU)-DT(PCASE))
 3230 IF(POWER.NE.0.0) GO TO 3250
C
C     TEST PISTON DISPLACEMENT AGAINST MAXIMUM STROKE
      QACTLK=(DT(PACTU)-DT(PCASE))*COLLKA
      QNET=DT(QACTU)-QACTLK-DT(QACTC)
      DT(VELACT)=-QNET/D(ARACT)
      DT(DISACT)=DT(DISACT)+(VOLD+DT(VELACT))/2.*DELT
      CALL XLIMIT(DT(DISACT),DT(VELACT),POWER,D(DISMIN),D(DISMAX))
      IF(POWER.EQ.0.0) GO TO 3300
      IF(DT(PINLET).LE.D(PINMIN)) GO TO 3240
      QPUMP=D(DISP)*DT(PRPM)*DT(DISACT)
      QOSIN=0.0
 3240 IF(POWER.EQ.-1.0) GO TO 3200
      DT(QACTC)=0.0
      DT(PACTU)=DT(PCASE)
      QACTLK=0.0
      GO TO 3300
 3250 DT(PACTU)=DT(PCASE)+DT(QACTU)/COLLKA
      QACTLK=DT(QACTU)
 3300 QPLEAK=(DT(POUTLT)-DT(PCASE))*D(COEPLK)
      Q(L2)=-(QPUMP-QPLEAK-DT(QACTU))
      DT(DISVLV)=DIS
      QCASDR=QPLEAK+QACTLK+DT(QACTC)-D(ARACT)*DT(VELACT)-QCASIN-QOSIN*2.
      QCASDR=(QCASDR*DT(BULKC)-C3+DT(PCASE))/(Z(L3)+DT(BULKC))
      DT(6)=AREA
 3400 P(L1)=DT(PINLET)
      Q(L1)=DT(QINLET)
      DT(POUTLT)=C(L2)-Q(L2)*Z(L2)
      P(L2)=DT(POUTLT)
      Q(L3)=-QCASDR
      DT(PCASE)=C(L3)+QCASDR*Z(L3)
      P(L3)=DT(PCASE)
C
      POWER=-Q(L2)*(P(L2)-P(L1))/6600.0
      DT(PPOWER)=POWER+DT(PPOWER)
      IF(DT(PINLET).LE.D(PINMIN)) WRITE(6,3310) T
      RETURN
C
C     DEPRESSURIZED CALCULATIONS
 4000 QPUMP=D(DISP)*D(DISMAX)*DT(PRPM)
      QCASDR=800.*D(COEPLK)-(DT(PCASE)-DT(PINLET))*D(COLCIN)
      DT(POUTLT)=DT(PCASE)+800.
      IF(C2.GT.DT(POUTLT)) DT(POUTLT)=C2
      Q(L2)=(C2-DT(POUTLT))/Z(L2)
```

6.54.7 (Continued)

```
      IF(Q(L2).LT.-QPUMP) Q(L2)=-QPUMP
      DT(QINLET)=QCASDR-Q(L2)
      DT(PINLET)=C1-DT(QINLET)*Z(L1)
      DT(DISACT)=D(DISAM)*(800.*D(COEPLK)-Q(L2))/QPUMP
      DT(PACTU)=0.0
      DT(VELACT)=0.0
      GO TO 3400
 3310 FORMAT(2X,36H******** PUMP CAVITATION ONSET AT T= ,E12.5)
      END
```

## 6.61 SUBROUTINE RSVR61

RSVR61 is a simulation of a hypothetical constant pressure reservoir that can be used in test simulation work. The input pressure is maintained without fluctuation while the flow(s) are adjusted to the line requirements. A maximum of four connections can be used.

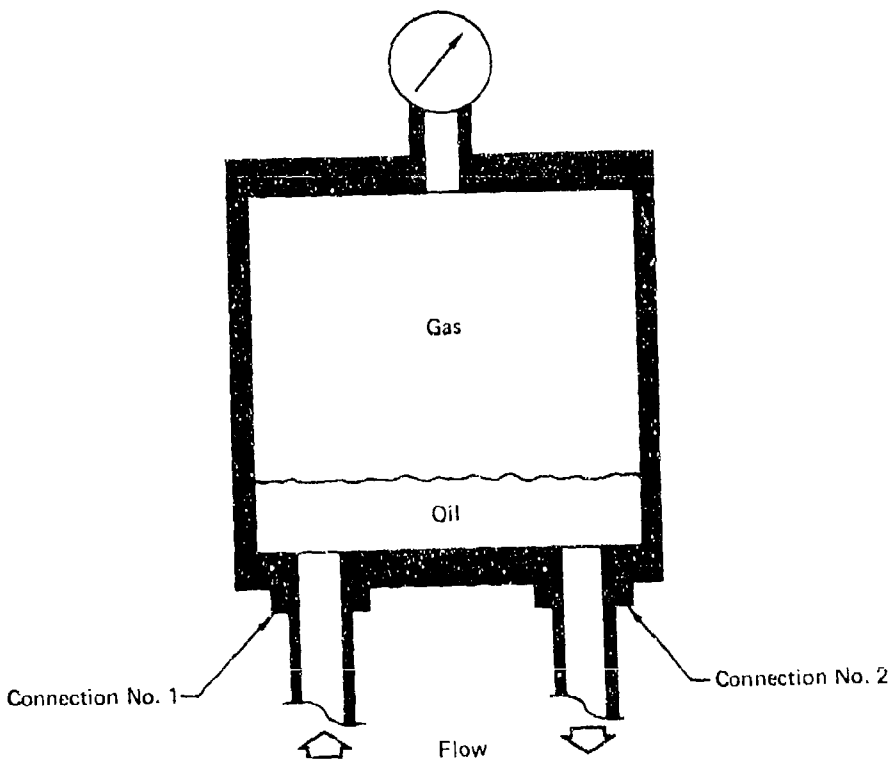



**FIGURE 6.61-1**
**TYPE NO. 61 CONSTANT PRESSURE RESERVOIR**

6.61.1 Math Model

SECTION 3000

Flow at each active connection based upon the input constant pressure, PRES, is calculated using the algebraic equation

$$Q(N) = (C(N)-D(1))/Z(N).$$

6.61.2 Assumptions

The reservoir is assumed to have an infinitely large gas volume so that pressure remains unchanged.

6.61.3 Computation Methods

SECTION 1000

Counter, L(5), and steady state flow variable, D(2), are set to zero. Variable, INV is stored in L(6).

SECTION 1500

Section 1500 sums flows into and/or out of the reservoir as the entry is called for each active connection. Counter, L(5), is incremented by 1 each time the entry is made until the counter is equal to the number of active connections, L(6) The counter is then reset to zero. Each entry determines if the flow is into or out of the reservoir and makes any necessary adjustments in the flow sign. The flow is added to the old net flow

$$1600\ D(2) = D(2) + QR$$

once the total net flow has been determined, QN(N) is calculated

$$QN(N) = D(1)*20. - D(2)$$

Net flow is then reset to zero.

$$D(2) = 0.0$$

SECTION 3000

Flow at each active connection is calculated using a DO loop. The flow is calculated for each connection and stored in the Q array. Input pressure, D(1), is stored in the P array.

$$Q(N) = (C(N)-D(1))/Z(N)$$

$$P(N) = D(1)$$

6.61.4 Approximations

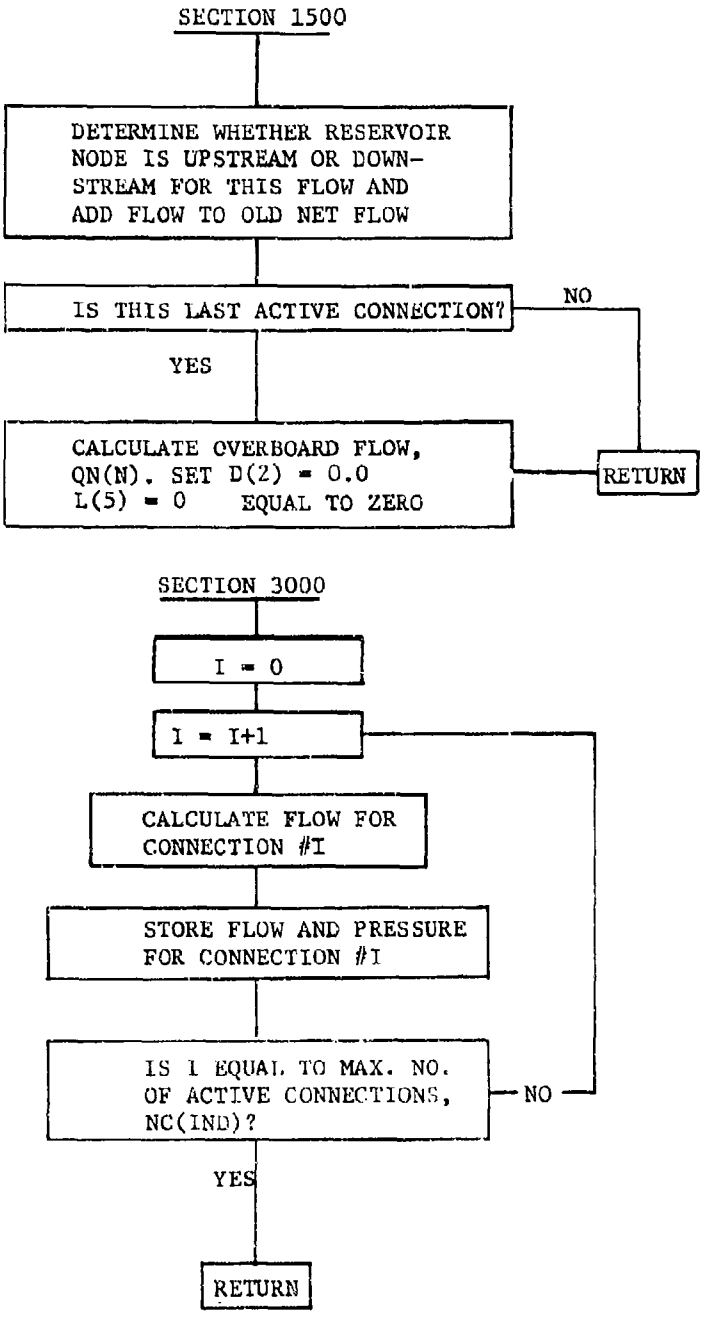
Not applicable.

6.61.5 Limitations

No applicable.

6.61.6 Variable Names

| VARIABLE | DESCRIPTION | UNITS |
|---|---|---|
| D(1) | Reservoir Pressure | PSIA |
| I | Counter | -- |
| N | Dummy Variable | -- |
| NCI | Dummy Variable | -- |
| QA | Dummy Variable | -- |
| QR | Steady State Flow | CIS |
| QS | Dummy Variable | -- |



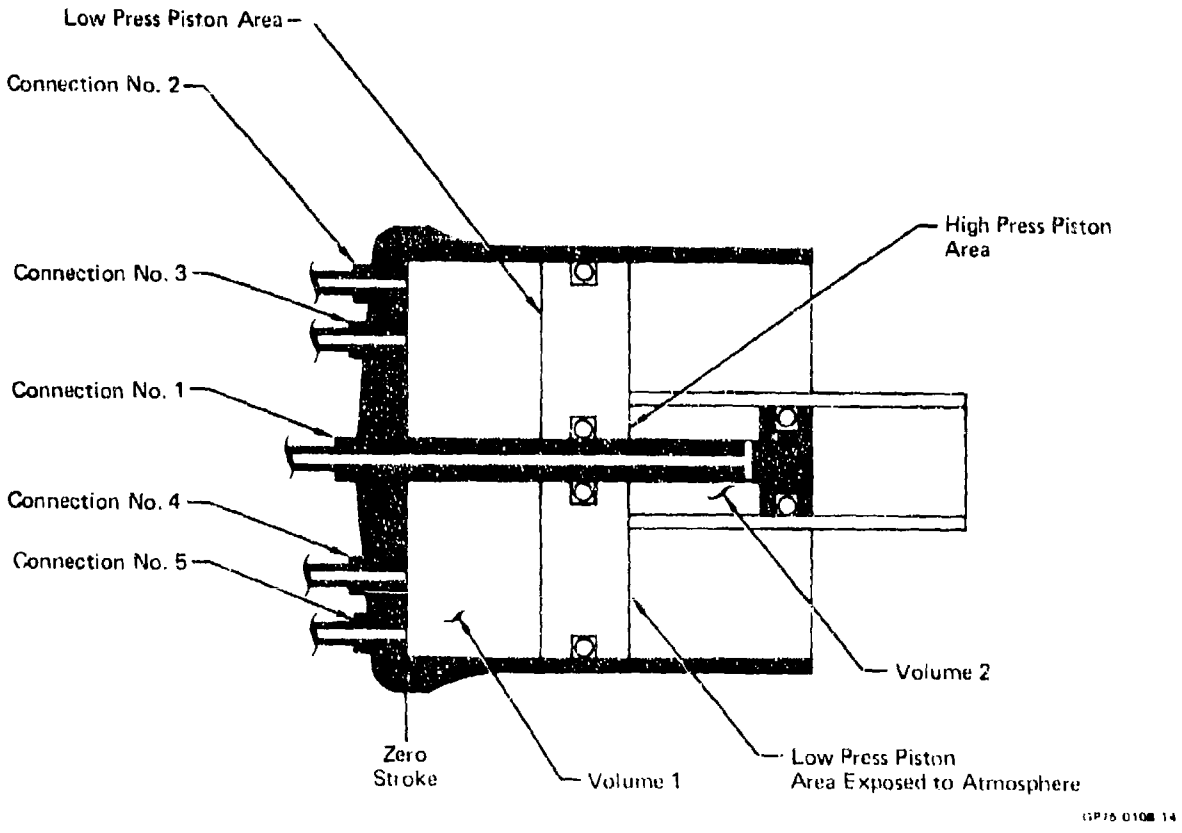

STEADY STATE AND 3000 SECTION CALCULATIONS
FIGURE 6.61-2

## 6.61.7 Subroutine Listing

```
      SUBROUTINE RSVR61 (D,DT,DD,L)
C *** REVISED AUGUST 5,1975 ***
      DOUBLE PRECISION DD
      COMMON NTELPL,NTOLPL,IPT,IPOINT,NPTS,INEL,KNEL,NTOPL,NLPLT(61,3),
     1 POLLG(90,12),LCS(90,10),ILOG(1400),PN(90),QN(90),PLX(90)
      COMMON/SUB/PARM(150,9),PM(1500),QM(1500),P(300),Q(300),C(300)
     1,Z(300),KBC(20),S2ORHO(20),VISC(20),BULK(20),TEMP(20),PVAP(20)
     2,ATPRLG,T,DELT,TFINAL,PLTDEL,PI,TITLE(20),LEGN,ICON
     3,NTEMP(99),LSTART(150),NLPT(150),LTYPE(99),NC(99),INX,INZ
     4,INV,ISTEP,NLINE,NEL,IND,IENTR,MNLINL,MNLL,MNLEG,MNNODE,MNPLOT
     5,MNLPTS,MDS
      DIMENSION D(1),DT(1),DD(1),L(1)
      IF(IENTR) 1000,2000,3000
 1000 CONTINUE
      IF(INEL.NE.0)GO TO 1500
      D(2)=0.0
      L(11)=0
      L(12)=INZ
      RETURN
 1500 CONTINUE
      L(11)=L(11)+1
      LCS(INLL,7)=5
      QA=POLLG(INLL,1)
      QB=POLLG(INLL,2)
      QR=QB*QA
      N=LCS(INLL,3)
      IF(INX.NE.1) GO TO 1600
      QR=-QR
      N=LCS(INLL,2)
 1600 D(2)=D(2)+QR
      IF(L(11).NE.L(12))RETURN
      QN(N)=D(1)*20.-D(2)
      PLX(N)=20.0
      D(2)=0.0
      L(11)=0
      RETURN
 2000 CONTINUE
      RETURN
 3000 CONTINUE
      NCI=NC(IND)
      DO 3100 I=1,NCI
      M=L(I)
      D(M)=(C(M)-D(1))/Z(M)
 3100 P(M)=D(1)
      RETURN
      END
```

## 6.62 SUBROUTINE RSVR62

RSVR62 subroutine is a simulation of a bootstrap reservoir. The subroutine can accommodate up to 4 low pressure lines along with the high pressure (bootstrap) line. The high pressure source does not have to be part of the low pressure system and can be completely independent.




TYPE NO. 62 BOOTSTRAP RESERVOIR

FIGURE 6.62-1

6.62.1 Math Model

Section 1500

The steady state high and low reservoir pressures are determined as follows.

A sign convention is established such that flow into the low pressure end is positive. A pseudo leg (see Figure 6.62-2) that terminates at low pressure node N is established. The pressure at the external end of this leg is set as the average of the pressure at node N PN(N) and pressure calculated for node N using the piston area ratio times pressure at node M, PMN where:

PMN = PN(M)*DT(NAREAR)+DT(EXPRES)

Any difference in PMN and PN(N) will produce flow in the pseudo leg and hence an unbalanced system. As LEGCAL balances the flows at all system nodes, the pseudo leg flow is forced to zero which in turn forces PMN and PN(N) pressures to be equal.

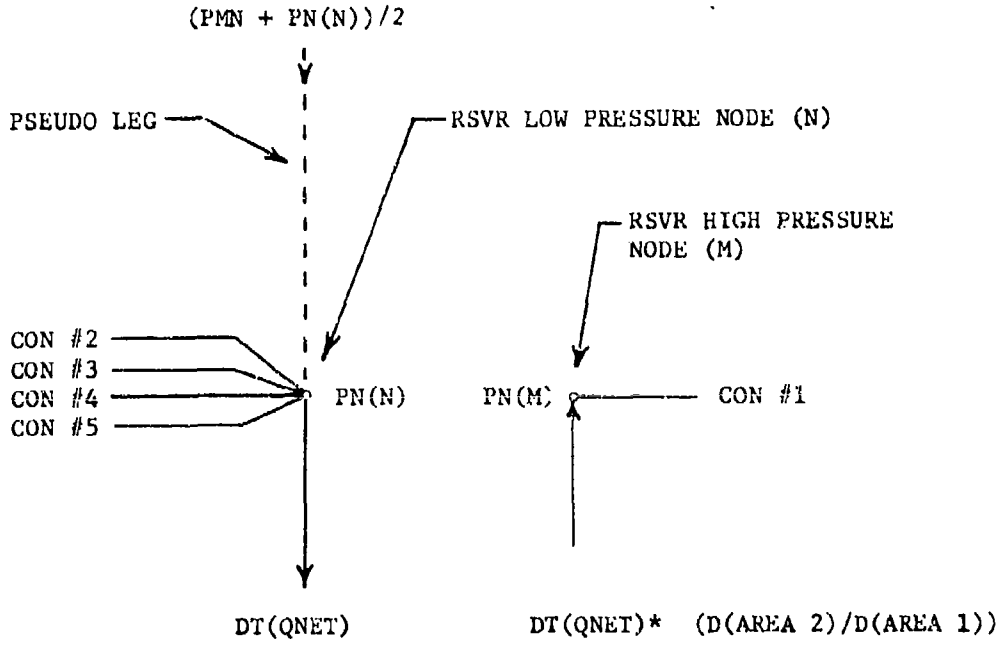



Figure 6.62-2

RSVR62 STEADY STATE ORGANIZATION

Section 3000

The high pressure line impedance may be expressed in terms of the low pressure impedances through an easily derived expression.

DT(NZ1)=Z(L(1))*DT(NAREAR)**2

Therefore the total impedance of the reservoir volume is simply the sum of DT(NZ1) and the impedances of the lines connected to the low pressure side. The high pressure characteristic is also modified to a low pressure characteristic.

CL1=C(L1)*DT(NAREAR)+DT(EXPRES)

Since there is no flow between the high and low pressure areas, volume and pressure is:

DT(P2)=CN/DT(NGZ)

where:

CN=CL1/DT(NZ1)+C(L(J))/Z(L(1))

The net flow into or out of the reservoir is then:

Q1=(DT(P2)-CL1)/DT(NZ1)

The flow out the high pressure side is simply:

Q(L1)=-Q1*DT(NAREAR)

and the pressure is:

P(L1)=C(L1)-Q(L1)*Z(L1)

The net change in flow volume into or out of volume 1 is computed and a test is made to determine if the piston is at a limit condition.

The flow at each active connection is then calculated using the algebraic equation:

$$Q(LI)=(C(LI)-P(LI))/Z(LI)$$

where

Q(LI) = fluid flow

P(LI) = fluid pressure

C(LI) = characteristic

Z(LI) = characteristic impedance

LI = connection number

6.62.2 Assumptions

The seal friction is zero.

6.62.3 Computation Methods

Section 1000

Variables DT(1QV), DT(MINIQV), DT(MAXIQV), DT(QNET) are initialized. The pistor force generated by atmospheric pressure is calculated using:

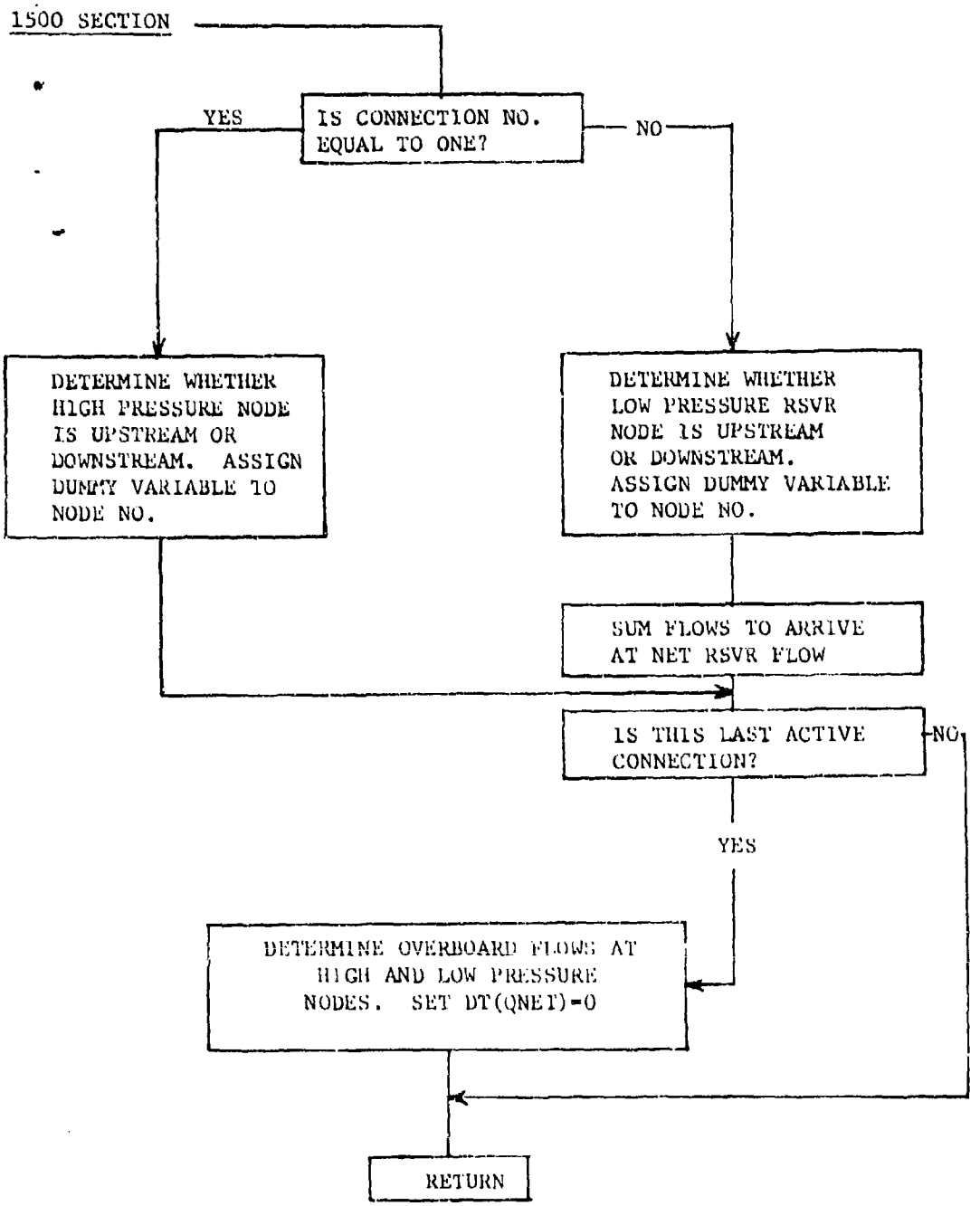
DT(EXPRES)=ATPRES*(D(AREAZ)-D(AREA1))/D(AREAZ)

Section 1500

This section sums the flows into and/or out of the low pressure chamber as the entry is called for each active connection. It also determines overboard flow at the high pressure node (M) and low pressure node (N).

Section 2000

DT(QNET) is initialized and the volume 1 impedance is computed.




RSVR62 STEADY STATE FLOW DIAGRAM
Figure 6.62-3

Section 3000

The characteristics are then calculated and subsequently used to calculate the flows.

CL1=CL1/DT(NZ1)+C(LCI))/Z(LCI))

Q1=(DT(PZ)-CL1)/DT(NZ1)

Note: These equations are used to calculate high pressure flow.

Next, the flow is summed using the new flows.

Tests are then performed to determine piston position and direction of travel. If the piston is on a stop and the motion is in the direction of the stop, velocity is set to zero.

Computations are then performed to determine the appropriate pressures and flows. The net flow, DT(QNET) is then updated for the next time step.

6.62.4 Approximations

The elasticity of the reservoir walls and the resulting change in volume for any piston movement is ignored as well as the compressibility effects of the oil.

6.62.5 Limitations

1. RSVR62 is limited to four low pressure connection and one high pressure connection.

6.62.6 Variable Names

| Variable | Description | Dimension |
|---|---|---|
| D(AREA1) | Low Pressure Piston Area | IN**2 |
| D(AREA2) | High Pressure Piston Area | IN**2 |
| CL1 | High Pressure Characteristic | PSI |
| CN | Characteristic of Low Pressure Connection | PSI |
| DT(EXPRES) | Atmospheric Pressure Contribution to Piston | PSI |
| DT(IQV) | Integral of Flow in Volume 1 | CIS |
| DT(MAX1QV) | Integral of Flow in Volume 1 at Maximum Piston Stroke | CIS |
| DT(MINIQU) | Integral of Flow in Volume 1 at Minimum Piston Stroke | CIS |
| DT(NAREAR) | Piston Area Ratio | -- |
| NC1 | Number of Connections | -- |
| DT(NGZ) | Conductance of Volume 1 | CIS/PSI |
| DT(NZ1) | Equivalent Volume 1 Impedance of the High Pressure Volume | $\frac{PSI}{CIS}$ |
| DT(QNET) | Net Flow in Volume 1 | CIS |
| D(STROKE) | Maximum Piston Stroke | IN |
| D(VOL1) | Reservoir Oil Volume | IN**3 |
| D(VOL2) | Bootstrap Oil Volume | IN**3 |

## 6.62.7 Subroutine Listing

```
      SUBROUTINE RSVR62 (D,DT,DD,L)
C**** REVISED JANUARY 24, 1976 ****
      DOUBLE PRECISION DD
      COMMON NTELPL,NTOLPL,IPT,IPOINT,NPTS,INEL,KNEL,NTOPL,NLPLT(61,3),
     1 PQLEG(90,12),LCS(90,10),ILEG(1400),PN(90),QN(90),PLX(90)
      COMMON/SUB/PARM(150,9),P1(1500),QM(1500),P(300),Q(300),C(300)
     1,Z(300),RHO(20),S2ORHO(20),VISC(20),BULK(20),TEMP(20),PVAP(20)
     2,ATPRES,T,DELT,TFINAL,PLTDEL,PI,TITLE(20),LEGN,ICON
     3,KTEMP(99),LSTART(150),NLPT(150),LTYPE(99),NC(99),INX,INZ
     4,INV,ISTEP,NLINE,NEL,IND,IENTR,MNLINE,MNEL,MNLEG,MNNODE,MNPLOT
     5,MNLPTS,MDS
      DIMENSION D(6),DT(10),DD(1),L(7)
      INTEGER AREA1,AREA2,VOL1,VOL2,STROKE,QNET,EXPRES,P2,DUM
      DATA AREA1/1/,AREA2/2/,VOL1/3/,VOL2/4/,STROKE/5/,INPOS/6/
     1,QNET/1/,EXPRES/2/,MINIQV/3/
     2 ,MAXIQV/4/,IQV/5/,NAREAR/6/,NZ1/7/,NC2/8/,P2/9/,DUM/10/
      IF(IENTR) 1000,2000,3000
 1000 CONTINUE
      IF(INEL.NE.0)GO TO 1500
      DT(IQV)=(D(INPOS)*D(AREA2)+D(VOL2))*2.0/DELT
      DT(MINIQV)=D(VOL2)*2.0/DELT
      DT(MAXIQV)=D(STROKE)*D(AREA2)*2.0/DELT+DT(MINIQV)
      L(7)=0
      DT(QNET)=0.0
      DT(AREAR)=D(AREA1)/D(AREA2)
      DT(EXPRES)=ATPRES*(D(AREA2)-D(AREA1))/D(AREA2)
      RETURN
C
C*****STEADY STATE SECTION*****
C
 1500 CONTINUE
C
C     M IS THE BOOTSTRAP NODE, N IS THE LOW PRESSURE NODE
      L(7)=L(7)+1
      LCS(INEL,7)=5
      QA=PQLEG(INEL,1)
      QS=PQLEG(INEL,2)
      IF(KNEL.NE.1) GO TO 1600
      M=LCS(INEL,3)
      IF(INX.NE.1) GO TO 1300
      N=LCS(INEL,2)
      GO TO 1800
 1600 QR=QA*QS
      N=LCS(INEL,3)
      IF(LIX.NE.1) GO TO 1700
      QR=-QR
      N=LCS(INEL,2)
 1700 DT(QNET)=DT(QNET)+QR
 1300 IF(L(7).NE.NC(IND)) RETURN
      IF(N.EQ.M) WRITE(6,1900)
```

**6.62.7 (Continued)**

```
      L(7)=0
      DT(NZ1)=DT(QNET)
      IF(PN(M).EQ.0.0) PN(M)=3000.
      IF(PN(N).EQ.0.0) PN(N)=PN(M)*DT(NAREAR)+DT(EXPRES)
      PMN=PN(M)*DT(NAREAR)+DT(EXPRES)
      QN(N)=((PMN+PN(N))*20.)/2.-DT(QNET)
      QN(M)=DT(QNET)*DT(NAREAR)
      DT(P2)=PN(N)
      PLX(N)=20.
      DT(QNET)=0.0
 1900 FORMAT(5X,45HRSVR62 REQUIRES TWO NODES FOR BOOTSTRAP FLOW    )
      RETURN
 2000 CONTINUE
      DT(QNET)=DT(NZ1)
      DT(NZ1)=Z(L(1))*DT(NAREAR)**2
      DT(NGZ)=1.0/DT(NZ1)
      NCI=NC(IND)
      DO 2100 I=2,NCI
 2100 DT(NGZ)=DT(NGZ)+1.0/Z(L(I))
      RETURN
 3000 CONTINUE
      NCI=NC(IND)
      L1=L(1)
      CL1=C(L1)*DT(NAREAR)+DT(EXPRES)
      CN=CL1/DT(NZ1)
      DO 3100 I=2,NCI
 3100 CN=CN+C(L(I))/Z(L(I))
      DT(P2)=CN/DT(NGZ)
      Q1=(DT(P2)-CL1)/DT(NZ1)
      Q(L1)=-Q1*DT(NAREAR)
      P(L1)=C(L1)-Q(L1)*Z(L1)
      DT(IQV)=DT(IQV)+DT(QNET)+Q1
      CALL XLIMIT(DT(IQV),Q1,CN,DT(MINIQV),DT(MAXIQV))
      IF(CN.EQ.0.0.AND.Q1.NE.0.0) GO TO 3500
      CN=0.0
      ZN=0.0
      DO 3400 I=2,NCI
      LI=L(I)
      ZN=ZN+1.0/Z(LI)
 3400 CN=CN+C(LI)/Z(LI)
      DT(P2)=CN/ZN
      P(L1)=C(L1)
      Q(L1)=0.0
 3500 CONTINUE
      DO 3600 I=2,NCI
      LI=L(I)
      P(LI)=DT(P2)
 3600 Q(LI)=(C(LI)-P(LI))/Z(LI)
      DT(QNET)=Q1
      DT(DUM)=DT(IQV)*DELT/2.
      RETURN
      END
```

## 6.71 SUBROUTINE ACUM71

Subroutine ACUM71 models a simple gas charged piston type accumulator that can be used as a system accumulator or as a source of supply for transient simulations.

When used as a system accumulator, the initial volume of oil in the accumulator is determined by the steady state pressure.

When used as a source of supply a constant pressure node must be used to determine the initial volume of oil in the accumulator.

Two connections are provided, both of which are assumed to be at the same pressure. When a single connection is used, the other is blanked off automatically.

Since it is basically a passive device, its response is entirely dependent on line flow and pressure changes.

No attempt has been made to include the true gas dynamics which involve not only a varying gas constant, but also the heat transfer characteristics between the gas and accumulator walls. Figure 6.71-1 shows a typical accumulator layout.

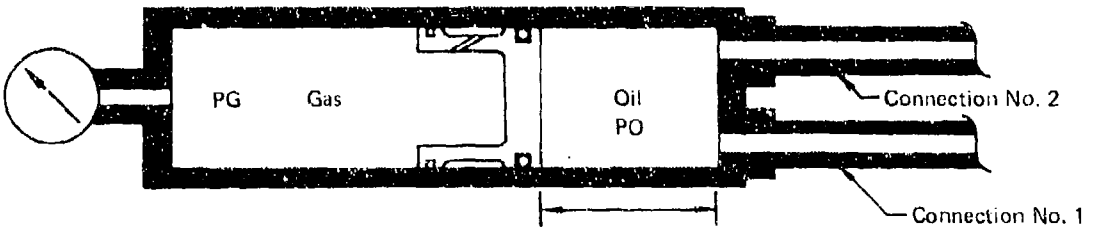



**FIGURE 6.71-1**
**TYPE NO. 71 FREE PISTON ACCUMULATOR**

GP74 0773 1

## 6.71 Math Model

The ACUM71 model calculates transient pressures and flows based on three different conditions within the accumulator.

The calculation of transient pressures and flows in the ACUM71 model depends upon whether the accumulator is empty, full or within the working range.

Conductances are calculated for the lines connected to the accumulator and volume of oil in the accumulator.

$G_{line} = 1/Z(L1) + 1/Z(L2)$

$G_{vol} = VOLO/(BULK*DELT)$

where,

$G_{line}$ is the conductance of the lines to the accumulator,

$G_{vol}$ is the conductance of the accumulator oil volume,

Z is the impedance of the line,

BULK is the bulk modulus of the oil,

VOLO is the volume of the oil,

DELT is the calculation time step.

If the accumulator is empty the pressure of the oil in the accumulator is given by:

$PO = (Q1 + Q2)/(G_{line})$

where,

PO is the pressure of the oil in the accumulator,

Q1 and Q2 are the flows into connections 1 and 2 respectively.

If the accumulator is full the pressure of the oil in the accumulator is given by:

$$PO = (Q1+Q2+POA/G_{vol})/(G_{line}+G_{vol})$$

where,

POA is equal to the current pressure of the oil in the accumulator.

If the accumulator is within its working range the change in the gas pressure is,

$$DPG = (PG*(Q1+Q2)/(MAVOLG-TVOLO + M1VOLO)$$

where,

DPG is the rate of change of the gas pressure,

PG is the gas pressure,

MAVOLG is the maximum gas volume,

TVOLO is the current oil volume,

M1VOLO is the minimum oil volume.

The pressure of the gas is given by

$$PG = PG+(DPGS+DPG)*DELT/2$$

where,

DPGS is the rate of change of the gas pressure calculated in the previous time step. The flows and pressures at the connections are then calculated using PO.

$$P(L1) = PO$$

$$Q(L1) = (C-PO)/Z$$

where,

P is the pressure at the connection,

Q is the flow at the connection,

C is the characteristic line pressure,

Z is the characteristic line impedance.

6.71.2 Assumptions

The true gas dynamics and the frictional and inertial effects of the piston have been ignored in this simulation to avoid complicating what is essentially a simple component model. Entry and exit pressure drops have also been ignored.

6.71.3 Computation

SECTION 1000

The precharge pressure, inputed for 100°F, is corrected for the working pressure of the fluid using the ideal gas formula.

```
DT(PRESPS) = D(PRESP)*(TEMP(KTEMP(1ND))+460.)/520
```

SECTION 1500

The pressure of the oil in the accumulator, DT(PO), is set equal to the upstream pressure of the leg.

```
DT(PO)= PQLEG(1NEL, 11)
```

SECTION 2000

The conductance of the two lines is calculated.

```
DT(NZ) = 1.0/Z(L(1))+L.0(Z(L(2))
```

The steady state volume of oil in the accumulator is then calculated.

```
DT(1VOLO) = D(M1VOLO)+(D(MAVOLG)-(DT(PRESPS)*D(MAVOLG))/DT(PG)
```

If the steady state oil pressure is less than the precharge pressure, then the steady state volume is set equal to the minimum oil volume and the gas pressure, DT(PG), is set equal to the precharge pressure.

The impedance of the maximum oil volume is then calculated and, after conversion to a conductance, is summed with the line conductances.

```
     1F(DT(PRESPS).LT.DT(PG))GO TO 2100
     DT(1VOLO) = D(M1VOLO)
     DT(PG) = DT(PRESPS)
2100 DT(KBULK) = BULK(KTEMP(1ND))*DT(NDELT)/D(MAVOLO)
     DT(NVZ) = DT(NZ)+1.0/DT(KBULK)
```

SECTION 3000

The net flow into the accumulator is calculated using the oil pressure calculated in the previous time step. The volume of oil in the accumulator is then calculated.

```
Q(L1) = (C(L1)-DT(PO))/Z(L1)
Q(L2) = (C(L2)-DT(PO))/Z(L2)
TQV = Q(L1)+Q(L2)
TVOLO = DT(IVOLO)+(DT(IQV)+TQV)*DT(NDELT)
```

A test is made to determine if the volume of oil in the accumulator is less than the minimum oil volume, D(M1VOLO), greater than the maximum oil volume, D(MAVOLO), or within the working range of the accumulator.

If the volume is less than the minimum oil volume the volume is set equal to D(M1VOLO), the change in the gas pressure and the net oil flow are set to zero and the pressure of the oil is calculated.

```
TVOLO = D(M1VOLO)
TDPG = 0.0
TQV = 0.0
DT(PO) = (C(L1)/Z(L1)+C(L2))/DT(NZ)
```

If the volume of oil is greater than the maximum oil volume the volume is set equal to D(MAVOLO), the change in oil pressure and volume are set to zero, and the oil pressure is calculated.

```
DT(PO) = C(L1)/Z(L1)+C(L2)/Z(L2)+DT(PO)/DT(KBULK)
DT(PO) = DT(PO)/DT(NVZ)
```

If the volume of oil is within the working range of the accumulator the change in gas pressure due to the change in oil volume is calculated, a new gas pressure is calculated and the oil pressure is set equal to the gas pressure.

```
    TDPG = DT(PG)*TQV/(D(MAVOLG)-TVOLO+D(M1VOLO))
    DT(PG) = DT(PG)+(DT(DPG)+TDPG)*DT(NDELT)
3250 DT(PO) = DT(PG)
```

The pressure and flows at the accumulator connections are then calculated using the oil pressure calculated above.

6.71.4 Approximations

Leaving out static friction from the simulation gives poor low amplitude results but helps the simulation by avoiding the nonlinear effects. Leaving out the inertial effects of the piston may reduce the accuracy of the results at very high frequencies but reduces the cost of using the accumulator subroutine.

6.71.5 Limitations

The current model can be used for rapid small amplitude discharge transients where heat transfer is not a problem.

For very small amplitudes where frictional effects become significant, or very high frequencies where inertia effects may become significant this model should not be used.

### 6.71.6 Variable Names

| Name | Description | Dimensions |
|---|---|---|
| DT(DPG) | Rate of change of gas pressure | PSI/sec |
| DT(DPO) | Rate of change of oil pressure | PSI/sec |
| DT(IQV) | Rate of change of oil volume | CIS |
| DT(IVOLO) | Volume of oil in accumulator | $in^3$ |
| DT(KBULK) | Conductance of oil volume | CIS/PSI |
| L1 | Dummy variable | - |
| L2 | Dummy variable | - |
| D(MAVOLG) | Maximum gas volume | $in^3$ |
| D(MAVOLO) | Maximum oil volume | $in^3$ |
| D(MIVOLG) | Minimum gas volume | $in^3$ |
| DT(NDELT) | Calculation time step | sec |
| DT(NVZ) | Sum of line and volume conductance | CIS/PSI |
| DT(NZ) | Sum of line conductances | CIS/PSI |
| DT(PG) | Gas pressure | PSI |
| DT(PO) | Oil pressure | PSI |
| D(PRESP) | Precharge gas pressure | PSI |
| DT(PRESPS) | Corrected precharge gas pressure | PSI |
| TDPG | Temporary rate of change of gas pressure | PSI/sec |
| TQV | Temporary value for rate of change of oil volume | CIS |
| TVOLO | Temporary volume of oil | $in^3$ |

## 6.71.7 Subroutine Listing

```
      SUBROUTINE ACUM71 (D,DT,DD,L)
C**** REVISED DECEMBER 5, 1975 ****
C     A SIMPLE ACCUMULATOR MODEL WHICH NEGLECTS PISTON FRICTION
C     PISTON MASS, AND THE TRUE GAS DYNAMICS
      DOUBLE PRECISION DD
      COMMON NTELPL,NTOLPL,IPT,IPOINT,NPTS,INLL,KNLL,NTOPL,NLPLT(61,3),
     1 PQLEG(90,12),LCS(90,10),ILEG(1400),PN(90),QN(90)
      COMMON/SUB/PARM(150,9),PM(1500),QM(1500),P(300),Q(300),C(300)
     1,Z(300),RHO(20),S2ORHO(20),VISC(20),BULK(20),TEMP(20),PVAP(20)
     2,ATPRES,T,DELT,TFINAL,PLTDEL,PI,TITLE(20),LEGN,ICON
     3,KTEMP(99),LSTART(150),NLPT(150),LTYPE(99),NC(99),INX,INZ
     4,INV,ISTOP,NLINE,NLL,IND,IENTR,MNLINE,MNLL,MNLLG,MSNODE,MAPLOT
     5,MNLPTS,MDS
      DIMENSION DD(1),L(2),D(5),DT(10)
      INTEGER PRESP,PRLSPS,DPO,PO,DPG,PG
      DATA MIVOLO/1/,MAVOLO/2/,MIVOLG/3/,PRESP/4/,
     1 PRLSPS/1/,NDELT/2/,DPO/3/,PG/4/,DPG/5/,PG/6/,
     2 IQV/7/,IVOLO/8/,NZ/9/,KBULK/10/,NVZ/11/
      IF(IENTR) 1000,2000,3000
 1000 CONTINUE
      IF(INLL.NE.0)GO TO 1500
C     IDEAL GAS RELATIONSHIP IS USED TO ADJUST PRECHARGE PRESS
      DT(PRESPS)=D(PRESP)*(TEMP(KTEMP(IND))+460.)/520.
      RETURN
 1500 IF(KNLL.EQ.2)GO TO 1600
      DT(IQV)=PQLEG(INLL,1)*PQLEG(INLL,2)
      DT(PO)=PQLEG(INLL,11)
      RETURN
 1600 DT(DPO)=PQLEG(INLL,11)
      RETURN
 2000 CONTINUE
      DT(NZ)=1.0/Z(L(1))+1.0/Z(L(2))
      IF(NC(IND).EQ.1) GO TO 2050
      DT(IQV)=0.0
      DT(PO)=(DT(PO)+DT(DPO))/2.0
 2050 DT(DPO)=0.0
      DT(DPG)=0.0
      DT(NDELT)=0.5*DELT
      DT(PG)=DT(PO)
      D(MAVOLG)=D(MAVOLO)-D(MIVOLO)+D(MIVOLG)
      DT(IVOLO)=D(MIVOLO)+(D(MAVOLG)-(DT(PRESPS)*D(MAVOLG))/DT(PG))
      IF(DT(PRLSPS).LT.DT(PG)) GO TO 2100
      DT(IVOLO)=D(MIVOLO)
      DT(PG)=DT(PRESPS)
 2100 DT(KBULK)=BULK(KTEMP(IND))*DT(NDELT)/D(MAVOLO)
      DT(NVZ)=DT(NZ)+1.0/DT(KBULK)
      RETURN
 3000 CONTINUE
      L1=L(1)
      L2=L(2)
```

6.71.7 (Continued)

```
      Q(L1)=(C(L1)-DT(PO))/Z(L1)
      Q(L2)=(C(L2)-DT(PO))/Z(L2)
      TQV=Q(L1)+Q(L2)
      TVOLO=DT(IVOLO)+(DT(IQV)+TQV)*DT(NDELT)
      IF(TVOLO.GT.D(MIVOLO).AND.DT(PO).GE.DT(PG)) GO TO 3100
C
C     THE ACCUMULATOR IS EMPTY
      TVOLO=D(MIVOLO)
      TDPG=0.0
      TQV=0.0
      DT(PO)=(C(L1)/Z(L1)+C(L2)/Z(L2))/DT(NZ)
      IF(DT(PO).LE.DT(PG)) GO TO 3300
      TVOLO=D(MIVOLO)+.001
      GO TO 3250
 3100 IF(TVOLO.LT.D(MAVOLO)) GO TO 3200
C
C     THE ACCUMULATOR IS FULL
      TVOLO=D(MAVOLO)
      TQV=0.0
      TDPG=0.0
      DT(PO)=C(L1)/Z(L1)+C(L2)/Z(L2)+DT(PO)/DT(KBULK)
      DT(PO)=DT(PO)/DT(NZ)
      IF(DT(PO).GT.DT(PG)) GO TO 3300
      TVOLO=DT(MAVOLO)-.001
      GO TO 3250
C
C     THE ACCUMULATOR IS WITHIN IT'S WORKING RANGE
 3200 TDPG=DT(PG)*TQV/(D(MAVOLG)-TVOLO+D(MIVOLO))
      DT(PG)=DT(PG)+(DT(DPG)+TDPG)*DT(NDELT)
 3250 DT(PO)=DT(PG)
 3300 P(L1)=DT(PO)
      Q(L1)=(C(L1)-DT(PO))/Z(L1)
      Q(L2)=(C(L2)-DT(PO))/Z(L2)
      P(L2)=DT(PO)
      DT(DPO)=Q(L1)+Q(L2)
      DT(IQV)=TQV
      DT(IVOLO)=TVOLO
      DT(DPG)=TDPG
      RETURN
      END
```

## 6.81 SUBROUTINE FILT81

FILT81 is a simulation of an inline, non-bypass filter with no moving parts, using standard cleanable elements.

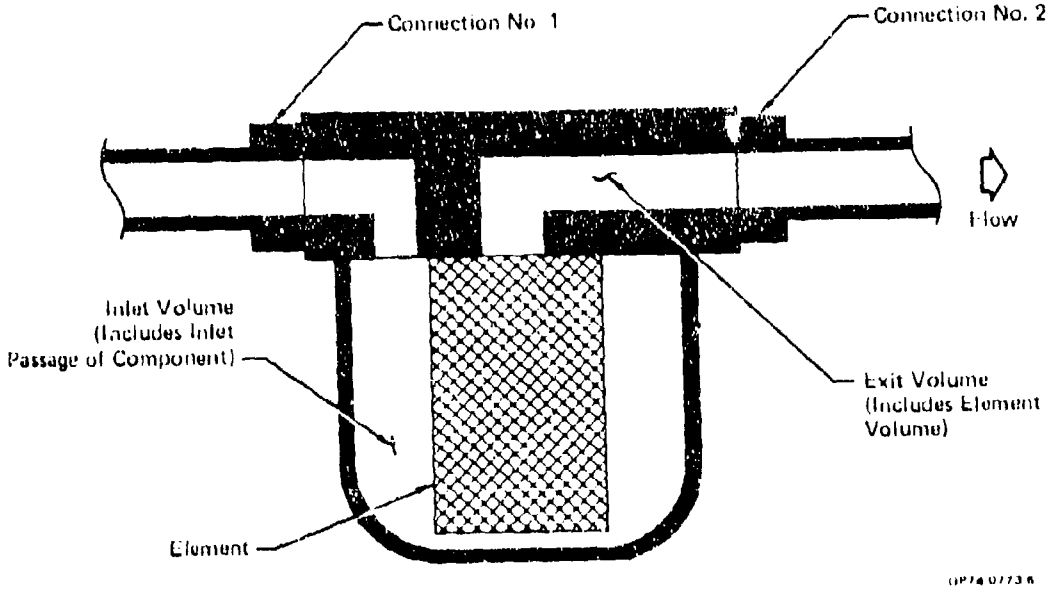



**FIGURE 6.81-1**
**TYPE NO. 81 F-4 TYPE IN-LINE FILTER**

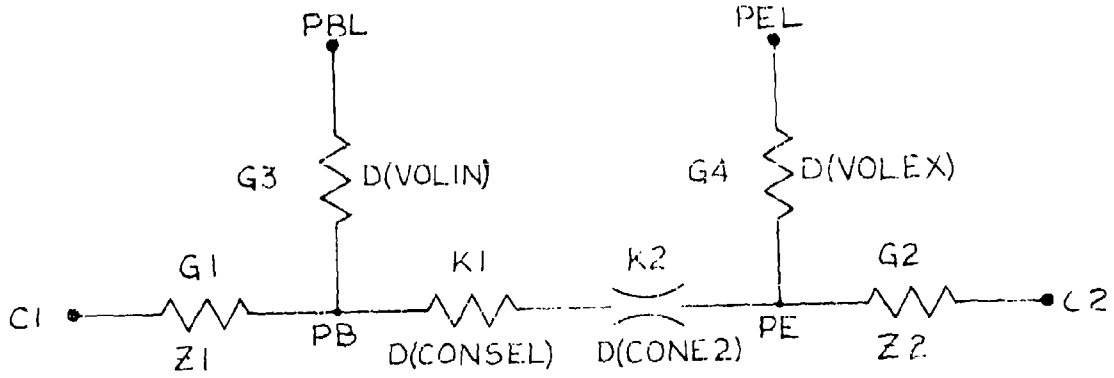



FIGURE 6.81-2
FILT81 DIAGRAM

6.81.1 Math Model

Subroutine FILT81 uses a simple model to describe the pressure and flow characteristics of a filter under transient conditions. FILT81 Diagram, figure 6.81.2, schematically describes the model where PB and PE, the bowl and element pressures, are to be determined. The past values of the bowl and element pressures, PBL and PEL, are shown at the ends of pseudo passages along with their conductances respectively.

The conductances G1 and G2 are calculated using the equation

$$G_{vol} = VOL/(BULK*DELT)$$

where

$G_{vol}$ = conductance
BULK = fluid bulk modulus
VOL = volume of oil in bowl or element section
DELT = time step of program

The conductances of the lines connected to the filter is simply the reciprocal of the line impedance

$$G_{line} = 1/Z_{line}$$

where

$Z_{line}$ = impedance of the line.

With all the conductances calculated, flows are summed about PB and PE resulting in the following equations.

$$(C1-PB)*G1+(PBL-PB)*G3 - QE = 0 \qquad 6.81.1$$

$$QE + (PEL-PE)*G4 - (PE-C2)*G2 = 0 \qquad 6.81.2$$

where C1 & C2 the line characteristic pressure
QE the flow across the element

The pressure drop between PB and PE is

$$PB - PE = QE^2*K2 + QE*K1$$

where K1 is the linear pressure drop coefficient
K2 is the $Q^2$ pressure drop coefficient

This equation is solved for QE in terms of PB and PE using the quadratic formula. The result is substituted into equations 6.81.1 and 6.81.2. The substitution for QE results in a set of two equations and two unknowns, PB and PE. The equations are solved simultaneously to give PB and PE.

The inlet and outlet flows, Q1 and Q2, are then calculated.

$$Q1 = (C1 - PB)/Z1$$

$$Q2 = (C2 - PE)/Z2$$

6.81.2 Assumptions - Not applicable.

6.81.3 Computational Methods

SECTION 1000

Pseudo conductances for the bowl and element sections are calculated and the filter element constant, D(CONSEL), is adjusted for fluid other than MIL-H-5606 and/or temperature other than 100°F.

D(VOL1N) = D(VOL1N)/DELT/BULK(L1)
D(VOLEX) = D(VOLEX)/DELT/BULK(L1)
D(CONSEL) = VISC(L1)*RHO(L1)/(.028*8.2E-5)*D(CONSEL)

SECTION 1500

The leg constants for Q and $Q^2$ are updated and the outlet pressure is calculated and stored. The bowl and element pressures, DT(PBL) and DT(PEL), are initialized.

SECTION 2000

Pseudo conductances DT(GIN), DT(GOUT) and DT(GTOT) are calculated for use in the transient calculation section.

```
DT(GIM) = 1.0/Z(L(1)) + D(VOLIN)
DT(GOUT) = 1.0/Z(L(2)) + D(VOLEX)
DT(GTOT) = (DT(GIN)+DT(GOUT)/(DT/GIN)*DT(GOUT))
DT(GTOT) = DT(GTOT) + D(CONSEL)
```

SECTION 3000

The values of DT(PBL) and DT(PEL) are calculated.

```
DPIN = (C(L1)/Z(L1) + D(VOLIN)*DT(PBL))/DT(GIN)
DPOUT = (C(L2)/Z(L2) + D(VOLEX)*DT(PEL))/DT(GOUT)
QE = SQRT(DT(GTOT)**2.+4.*D(CONE2)*ABS(DPIN-DPOUT)
QE = ((QE-DT(GTOT))/(2.*D(CONE2)),DPIN-DPOUT)
DT(PBL) = DPIN - QE/DT(GIN)
DT(PEZ) = DPOUT + QE/DT(GOUT)
```

The pressures and flows at each of the connections are then calculated using DT(PBL) and DT(PEL)

```
Q(L1) =  (C(L1) - DT(PBL))/Z(L1)
P(L1) = DT(PBL)
Q(L2) = (C(L2) - DT(PEL)/Z(L2)
P(L2) = DT(PEL)
```

6.81.4 Approximations

Non-linear pressure drop effects are assumed to be proportional to the $Q^2$ term.

6.81.5 Limitations - Not applicable.

6.81.6 Variable Names

| Variable | Description | Units |
|---|---|---|
| D(CONE2) | $Q^2$ element coefficient | $PSI/CIS^2$ |
| D(CONSEL) | Q element coefficient | PSI/CIS |
| DPIN | Intermediate calculation | -- |
| DPOUT | Intermediate calculation | -- |
| DT(GIN) | Sum of Inlet Conductances | CIS/PSI |
| DT(GOUT) | Sum of Outlet Conductances | CIS/PSI |
| DT(GTOT) | Intermediate Calculation | -- |
| L1 | Dummy Variable | -- |

| | | |
|---|---|---|
| L2 | Dummy Variable | -- |
| DT(PBL) | Bowl Pressure | PSI |
| DT(PEL) | Element Pressure | PSI |
| QE | Flow Across Element | CIS |
| D(VOLEX) | Exit Volume | $IN^3$ |
| D(VOLIN) | Inlet Volume | $IN^3$ |

## 6.81.7 Subroutine Listing

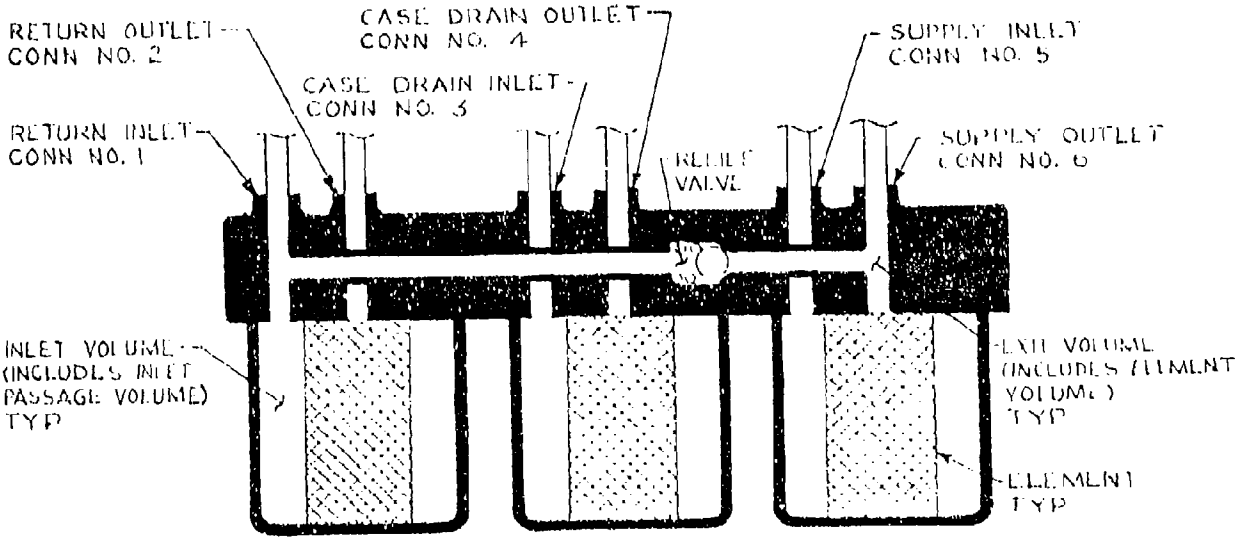
```
      SUBROUTINE FILT81 (D,DT,DD,L)
C**** REVISED JAY 1, 1976 ****
      COMMON NTELPL,NTOLPL,IPT,IPOINT,APTS,INEL,KNEL,NTOPL,NLPLT(61,3),
     1 POLEG(90,12),LCS(90,10),ILEG(1400),PN(90),QN(90)
      COMMON/SUB/PARA(150,9),PA(1500),QA(1500),P(300),Q(300),C(300)
     1,Z(300),RHO(20),S2ORHO(20),VISC(20),BULK(20),TEMP(20),PVAP(20)
     2,ATPRES,T,DELT,TFINAL,PLTDEL,PI,TITLE(20),LEGN,ICON
     3,NTEMP(99),LSTART(150),NLPT(150),LTYPE(99),KC(99),INX,INZ
     4,INV,ISTEP,NLINE,NEL,IND,IENTR,ANLINE,ANEL,ANLEG,ANMODE,ANPLOT
     5,ANLPTS,ADS
      DIMENSION D(1),DT(1),DD(1),L(1)
      INTEGER VOLIN,VOLEX,CONSEL,PBL,PEL,GTOT,GIN,GOUT,CON&2
C**** D ARRAY VARIABLES ****
      DATA VOLIN/1/,VOLEX/2/,CONSEL/3/,CON&2/4/
C**** DT ARRAY VARIABLES ****
      DATA PBL/1/,PEL/2/,GTOT/3/,GIN/4/,GOUT/5/
      IF(IENTR) 1000,2000,3000
C     SECTION 1000 ESTABLISHES CONSTANTS FOR USE IN FUTURE CALCULATIONS
 1000 CONTINUE
      IF (INEL.NE.0) GO TO 1500
C     SET CONSTANTS FOR INTEGRATION OF PRESSURES
      L1=NTEMP(IND)
      D(VOLIN)=D(VOLIN)/DELT/BULK(L1)
      D(VOLEX)=D(VOLEX)/DELT/BULK(L1)
C     ADJUST INPUTTED CONSTANTS FOR FLUID OTHER THAN MIL-H-5606 AND/OR
C       TEMPERATURES OTHER THAN 100 DEG F
      D(CONSEL)=VISC(L1)*RHO(L1)/(.029*3.2E-5)*D(CONSEL)
      IF(D(CONSEL).EQ.0.0) D(CONSEL)=.0001
      RETURN
 1500 CONTINUE
      IF(NEL-2)1510,1610,1610
 1510 QA=POLEG(INEL,1)
      QS=POLEG(INEL,2)
      POLEG(INEL,6)=POLEG(INEL,6)+D(CONSEL)
      POLEG(INEL,8)=POLEG(INEL,8)+D(CON&2)
      DT(PBL)=POLEG(INEL,11)
      DT(PEL)=DT(PBL)-QA*QS*(D(CONSEL)+QA*D(CON&2))
      POLEG(INEL,11)=DT(PEL)
 1610 RETURN
 2000 CONTINUE
      DT(GIN)=1.0/Z(L(1))+D(VOLIN)
      DT(GOUT)=1.0/Z(L(2))+D(VOLEX)
      DT(GTOT)=(DT(GIN)+DT(GOUT))/(DT(GIN)*DT(GOUT))
      DT(GTOT)=DT(GTOT)+D(CONSEL)
      RETURN
C
C     SECT 3000 CALCULATES AND STORES INTERNAL AND CONDUCTION PRESS'S
C     AND FLOWS
 3000 CONTINUE
      L1=L(1)
```

**6.81.7 (Continued)**

```
L2=L(2)
DPIN=(C(L1)/Z(L1)+D(VOLIN)*DT(PBL))/DT(GIN)
DPOUT=(C(L2)/Z(L2)+D(VOLEX)*DT(PEL))/DT(GOUT)
QE=SQRT(DT(GTOT)**2.+4.*D(CONL2)*ABS(DPIN-DPOUT))
QE=SIGN((QE-DT(GTOT))/(2.*D(CONL2)),DPIN-DPOUT)
DT(PBL)=DPIN-QE/DT(GIN)
DT(PEL)=DPOUT+QE/DT(GOUT)
C(L1)=(C(L1)-DT(PBL))/Z(L1)
P(L1)=DT(PBL)
C(L2)=(C(L2)-DT(PEL))/Z(L2)
P(L2)=DT(PEL)
RETURN
END
```

6.82 SUBROUTINE FILT82

FILT82 is a simulation of a three element filter manifold incorporating a relief valve between the supply and return filters. Inlet and outlet connections are specified and an incorrect installation will cause a message to be printed.




FILTER MANIFOLD

FIGURE 6.82 1

6.82.1 Math Model

The FILT82 model calculates the transient pressure and flow demands of a three element filter manifold incorporating a relief valve. Figure 6.82.2 schematically describes the filter manifold where the bowl and element pressures of each filter, PB and PE, are to be determined. The past valves of the bowl and element pressures, PBS and PES, are shown at the ends of pseudo passages along with their conductances, G. The conductance of the pseudo passage is calculated using the equation:

$$G_{Vol} = VOL/(BULK*DELT)$$

where

$G_{Vol}$ = conductance of the volume, CIS/PSI,

VOL = volume, $IN^3$,

BULK = fluid bulk modulus, PSI,

DELT = calculation time interval, SEC.

The conductances of the lines connected to the filter manifold is simply the reciprocal of the line impedance, Z.

$$G_{line} = 1/Z_{line}$$

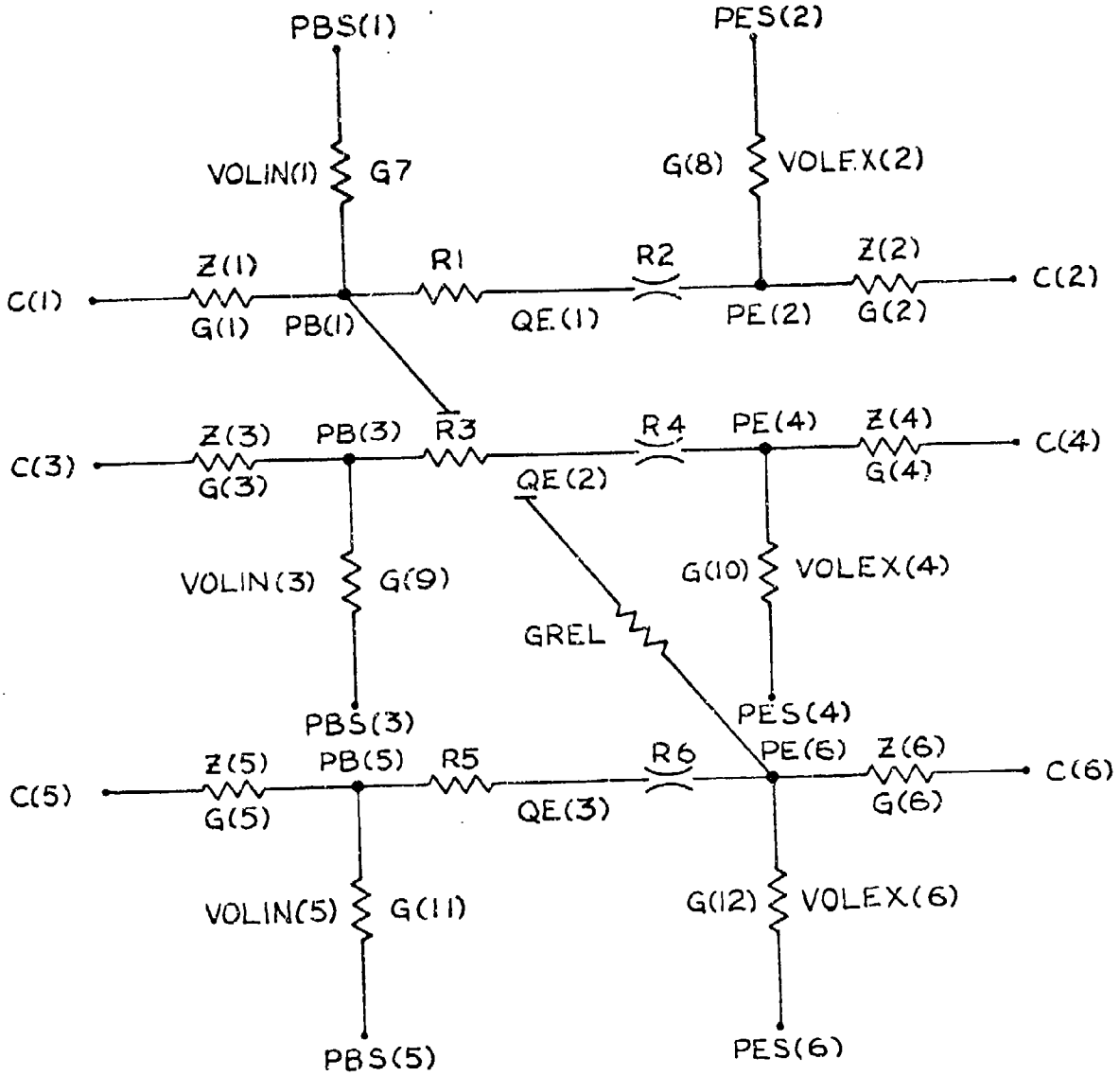
where

$G_{line}$ = conductance of the line, CIS/PSI,

$Z_{line}$ = impedance of the line, PSI/CIS.

Initially, the relief valve is assumed to be closed and a solution technique similar to that used in FILT81 is used to solve for the bowl and element pressures of each filter.

After all the line and volume conductances have been calculated, the flows are summed about PB and PE resulting in the following equations.




FILT82 DIAGRAM

FIGURE 6.82-2

$$(C(1)-PB)*G(1)+(PBS-PB)*G(3)-QE=0 \qquad 6.82.1$$

$$QE+(PES-PE)*G(4)-(PE-C(2))*G(2)=0 \qquad 6.82.2$$

where

C = the line characteristic pressures

QE = the flow across the element.

The pressure drop between PB and PE is

$$PB-PE=QE^2*R2+QE*R1 \qquad 6.82.3$$

where

R1 = linear pressure drop coefficient

R2 = $Q^2$ pressure drop coefficient.

Equation 6.82.3 is solved for QE in terms of PB and PE using the quadratic formula. The result is substituted into equations 6.82.1 and 6.82.2. The substitution for QE results in a set of two equations and two unknowns, PB and PE. The equations are solved simultaneously to give PB and PE.

After the bowl and element pressures for each filter have been calculated a test is made to determine if the relief valve is open. If PE(6)-PB(1) is less than the relief pressure, the relief valve is closed and the pressures and flows at the connections to the filters may be immediately calculated.

$$Q(N)=(C(N)-PB(N))/Z(N)$$

$$P(N)=PB(N)$$

$$Q(M)=(C(M)-PE(M))/Z(M)$$

$$P(M)=PE(M)$$

where

N = 1, 3 or 5

M = 2, 4 or 6

If PE(6)-PB(1) is greater than the relief pressure the relief valve is open and the bowl and element pressures for the supply and return filters are recalculated.

An equivalent conductance, GEQ, is calculated for the flow across the filter elements of the supply and return filters.

GEQ(1) = QE(1)/(PB(1)-PE(1))

GEQ(2) = QE(3)/(PB(5)-PE(6))

With the relief valve open there is a flow from PE(6) to PB(1). The flow through the relief valve plus all other flows are summed about PB(1), PE(2), PB(5) and PE(6) to give the following equations.

$$(C(1)-PB(1))*G(1)+(PBS(1)-PB(1))*G(7)+(PE(6)-PB(1))*GREL$$
$$-(PB(1)-PE(2))*GEQ(1)=0$$

$$(PB(1)-PE(2))*GEQ(1)+(PES(2)-PE(2))*G(8)-(PE(2)-C(2))*G(2)=0$$

$$(C(5)-PB(5))*G(5)+(PBS(5)-PB(5))*G(11)-(PB(5)-PE(6))*GEQ(2)=0$$

$$(PB(5)-PE(6))*GEQ(2)+(PES(6)-PE(6))*G(12)-(PE(6)-PB(1))*GREL$$
$$-(PE(6)-C(6))*G(6)=0$$

This results in a set of four equations with four unknowns, PB(1), PE(2), PB(5) and PE(6). These equations are solved simultaneously to give the bowl and element pressures in the supply and return filters with the relief valve open.

The bowl and element pressures for the supply and return filters are used to calculate the flows and pressures at the connections to the filters.

Q(1)=(C(1)-PB(1))/Z(1)

P(1)=PE(1)

Q(2)=(C(2)-PE(2))/Z(2)

P(2)=PE(2)

Q(5)=(C(5)-PB(5))/Z(5)

P(5)=PB(5)

Q(6)=(C(6)-PE(6))/Z(6)

6.82.2 Assumptions

It is assumed that a linearized equivalent conductance may be used to calculate the pressure drop across the element when the relief valve is open. The relief valve is assumed to open instantaneously.

6.82.3 Computational Methods

SECTION 1000

A DO loop is used to calculate pseudo conductances for the bowl and element volumes for each filter. The filter element constant for each filter, D(CONSEL+F), is adjusted for fluids other than MIL-H-5606 and/or temperatures other than 100°F.

```
      DO 1200 K=1, 3
      F=(K-1)*4
      H=(K-1)*8
      D(VOLIN+F)=D(VOLIN+F)/(BULK(KI)*DELT)
      D(VOLEX+F)=D(VOLEX+F)/(BULK(KI)*DELT)
      D(CONSEL+F)=VISC(KI)*RHO(KI)/(.028*8.2E-5)*D(CONSEL+F)
1200  CONTINUE
```

SECTION 1500

A test is made to insure the filter is properly connected. If a connection is found to be in error a message is printed and the run continues. Only calls to connection 1, 3 and 5 are allowed.

For each filter the filter element constants D(CONSEL+F) and D(CONEZ+F) are added to the Q and $Q^2$ steady state flow coefficients.

respectively. The pressure drop across the filter is subtracted from the upstream pressure for the leg.

SECTION 2000

A do loop is used to calculate the following constants for each filter. Pseudo conductances are calculated for use in the transient calculation section.

```
DO 2020 I=1, 3
M=2*I-1
N=M+1
L1=L(M)
L2=L(N)
F=(I-1)*4
H=(I-1)*8
DT(GIN+H)=1.0/Z(L1)+D(VOLIN+F)
DT(GOUT+H)=1.0/Z(L2)+D(VOLEX+F)
DT(GTOT+H)=(DT(GIN+H)+DT(GOUT+H))/(DT/GIN+H)*DT(GOUT+H))
DT(GTOT+H)=0.5*(DT/GTOT+H)+D(CONSEL+F))/D(COMEZ+F)
```

SECTION 3000

The values of DT(PBL+H) and DT(PEL+H) are calculated for each filter using a do loop.

```
DO 3060 I=1, 3
M=2*I-1
N=M+L
L1=L(M)
L2=(L(N)
F=(I-1)*4
H=(I-1)*8
```

```
DPIN=(C(L1)/Z(L1)+D(VOLIM+F)*DT(PRESSB+H))/DT(GIN+H)
DPOUT=(C(L2)/Z(2)+D(VOLEX+F)*DT(PRESSE+H))/DT(GOUT+H)
DT(QE+H)=SQRT(DT(GTOT+H)**2.+ABS(DPIN-DPOUT)/D(CONEZ+F))
DT(QE+H)=SIGN(DT(QE+A)-DT(GTOT+H), DPIN-DPOUT)
DT(PBL+H)=DPIN--DT(QE+H)/DT(GIN+H)
DT(PEL+H)=DPOUT+DT(QE+H)/DT(GOUT+H)
```

The pressures and flows at the connections are then calculated and post values of DT(PBL+H) and DT(PEL+H) are saved.

```
Q(L1)=(C(L1)-DT(PBL+H))/Z(L1)
P(L1)=DT(PBL+H)
Q(L2)=(C(L2)-DT(PEL+H))/Z(L2)
P(L2)=DT(PEL+H)
DT(PRESSB+H)=DT(PBL+H)
DT(PRESSE+H)=DT(PEL+H)
```

The difference between the supply element pressure, DT(PEL+16), and the return bowl pressure, DT(PBL), is then compared with the relief pressure, PREL. If the difference is less than the relief pressure the subroutine returns execution to the calling program. If the difference is greater than PREL the bowl and element pressures for the supply and return filters are recomputed.

The conductances of the lines are computed.

```
G1=1./Z(L1)
G2=1./Z(L2)
G7=1./Z(L6)
G9=1./Z(L6)
```

The A array is zeroed. The A and B arrays are then loaded for the matrix solution using SIMULT.

```
A(1,1)=G1+D(VOLIM)+GREL+GEQ1
A(1,2)=-GEQ1
A(1,4)=-GREL
A(2,1)=-GEQL
A(2,2)=GEQ1+D(VOLEX)+G3
A(3,3)=G7+D(VOLIN+8)+GEQ2
A(3,4)=-GEQ2
A(4,1)=-GREL
A(4,3)=-GEQ2
A(4,4)=GEQ2+D(VOLEX+8)+GREL+G9
B(1)=C(L1)*G1+DT(PBL)*DT(VOLIN)-PREL*GREL
B(2)=DT(PEL)*D(VOLEX)+C(L2)*G3
B(3)=C(L5)*G)+DT(PBL+16)*D(VOLIN+8)
B(4)=DT(PEL+16)*D(VOLEX+8)+PREL*GREL+C(L6)*G9
CALL SIMULT (A, B, 4, J)
```

The bowl and element pressures are returned through B and are saved. The flows and pressures at the connections to the supply and return filters are then recalculated using the newly computed bowl and element pressures.

```
DT(PRESSB)=B(1)
DT(PRESSE)=B(2)
DT(PRESSB+16)=B(3)
DT(PRESSE+16)=B(4)
Q(L1)=(C(L1)-B(1))/Z(L1)
Q(L2)=(C(L2)-B(2))/Z(L2)
Q(L5)=(C(L5)-B(3))/Z(L5)
Q(L6)=(C(L6)-B(4))/Z(L6)
```

P(L1)=B(1)

P(L2)=B(2)

P(L5)=B(3)

P(L6)=B(4)

6.82.4 Approximations

Nonlinear pressure drop effects are assumed to be proportional to the $Q^2$ term.

6.82.5 Limitations

The pressure and flow calculations with the relief valve open are approximations the accuracy of which will decrease as the duration of the relief valve opening increases.

6.82.6 Variable Names

| Variable | Description | Units |
|---|---|---|
| D(CONE2) | Nonlinear Element Constant | $PSI/CIS^{1/2}$ |
| D(CONSEL+F) | Linear Element Constant | PSI/CIS |
| DPIN | Intermediate Calculation | - |
| DPOUT | Intermediate Calculation | - |
| F | Increment | - |
| GEQ1 | Return Filter Equivalent Conductance | CIS/PSI |
| GEQ2 | Supply Filter Equivalent Conductance | CIS/PSI |
| DT(GIN+H) | Sum of Inlet Conductances | CIS/PSI |
| DT(GOUT+H) | Sum of Outlet Conductances | CIS/PSI |
| GREL | Relief Valve Conductance | CIS/PSI |
| DT(GTOT) | Intermediate Calculation | - |
| G1 | Line Conductance | CIS/PSI |
| G3 | Line Conductance | CIS/PSI |
| G7 | Line Conductance | CIS/PSI |

| | | |
|---|---|---|
| G9 | Line Conductance | CIS/PSI |
| H | Increment | - |
| I | Counter | - |
| I1 | Counter | - |
| J | Counter | - |
| J1 | Counter | - |
| K | Counter | - |
| L1 | Dummy Variable | - |
| L2 | Dummy Variable | - |
| L3 | Dummy Variable | - |
| L4 | Dummy Variable | - |
| L5 | Dummy Variable | - |
| L6 | Dummy Variable | - |
| M | Dummy Variable | - |
| N | Dummy Variable | - |
| DT(PBL+H) | Filter Bowl Pressure | PSI |
| DT(PEL+H) | Filter Element Pressure | PSI |
| PREL | Relief Pressure | PSI |
| DT(PRESSB-H) | Filter Bowl Pressure | PSI |
| DT(PRESSE+H) | Filter Element Pressure | PSI |
| DT(QE=H) | Flow Across Filter Element | CIS |
| DT(VOLEX+F) | Exit Volume | $IN^3$ |
| DT(VOLIN+F) | Inlet Volume | $IN^3$ |

## 6.82.7 Subroutine Listing

```
      SUBROUTINE FILT82 (D,DT,DD,L)
C**** REVISED FEB 5, 1976 *****
      COMMON NTSLPL,NTOLPL,IPT,IPOINT,NPTS,INEL,KNEL,NTOPL,NLPLT(51,3),
     1 PQLEG(90,12),LCS(90,10),ILEG(1400),PN(90),QN(90)
      COMMON/SUB/PARM(150,3),PM(1500),QM(1500),P(300),Q(300),C(300)
     1,Z(300),RHO(20),S20RHO(20),VISC(20),BULK(20),TEMP(20),PVAP(20)
     2,ATPRES,T,DELT,TFINAL,PLTDEL,PI,TITLE(20),LEGN,ICOS
     3,KTEMP(99),LSTART(150),NLPT(150),LTYPE(99),NC(99),IMX,IMS
     4,INV,ISTEP,NLINE,NLL,IND,IENTR,MNLINE,MNEL,MNLEG,MNNODL,MNPLOT
     5,MNLPTS,MDS
      DIMENSION D(24),DT(12),DD(1),L(8),A(4,4),B(4)
      INTEGER VOLIN,VOLEX,CONSEL,CONE2,GIN,GOUT,GTOT,PBL,PEL,QE,
     1PRESSB,PRESSE,F,S
C ***** D ARRAY VARIABLES *******
      DATA VOLIN/1/,VOLEX/2/,CONSEL/3/,CONE2/4/
C **** DT ARRAY VARIABLES ******
      DATA GIN/1/,GOUT/2/,PRESSB/3/,PRESSE/4/,GTOT/5/,PBL/6/,PEL/7/
     1,QE/8/
      IF(IENTR) 1000,2000,3000
C     SECTION 1000 ESTABLISHES CONSTANTS FOR USE IN FUTURE CALCULATIONS
 1000 CONTINUE
   50 IF (INEL.NE.0) GO TO 1500
C      SET RELIEF VALVE PRESSURE
      PELL=3050.
C      SET RELIEF VALVE ADMITTANCE
      CNEL=.6
      KI=KTEMP(ING)
      DO 1200 K=1,3
      F=(K-1)*4
      S=(K-1)*8
C     SET CONSTANTS FOR INTEGRATION OF PRESSURES
      D(VOLIN+S)=D(VOLIN+S)/(BULK(KI)*DELT)
      D(VOLEX+S)=D(VOLEX+S)/(BULK(KI)*DELT)
C     ADJUST INPUTTED CONSTANTS FOR FLUID OTHER THAN MIL-H-5606 AND/OR
C       TEMPERATURES OTHER THAN 100 DEG F
      D(CONSEL+F)=VISC(KI)*RHO(KI)/(.028*9.2E-5)
     1*D(CONSEL+F)
      IF(D(CONSEL+F).LE.0.0) D(CONSEL+F)=.0001
 1200 CONTINUE
      RETURN
 1500 F=0
      GO TO(1550,1800,1520,1800,1510,1800),KNEL
 1510 F=F+4
 1520 F=F+4
 1550 CONTINUE
      QA=PQLEG(INEL,1)
      QS=PQLEG(INEL,2)
      PQLEG(INEL,6)=PQLEG(INEL,6)+D(CONSEL+F)
      PQLEG(INEL,8)=PQLEG(INEL,8)+D(CONE2+F)
      PQLEG(INEL,11)=PQLEG(INEL,11)-QA*QS*D(CONSEL+F)-
     1 QS*QA**2.*D(CONE2+F)
      RETURN
 1800 CONTINUE
```

**6.82.7 (Continued)**

```
      WRITE(6,1301) IND,INEL,KNLL
 1901 FORMAT(2X,7HCOMP NO,I4,31H CALLED INCORRECTLY FROM LEG NO,
     1 I4,7H CON NO,I4)
      RETURN
C
C     SECT 2000 DETERMINES VALIDITY OF COMPONENT CONNECTIONS AND
C     DIRECTION OF FLOW. INITIAL PRESSURES IN ELEMENT AND IN BOWL AS
C     WELL AS INITIAL FLOW ACROSS ELEMENT ARE CALCULATED AND STORED
 2000 CONTINUE
      DO 2020 I=1,3
      J=2*I-1
      N=J+1
      L1=L(J)
      L2=L(N)
      F=(I-1)*4
      H=(I-1)*8
C     SET DUMMY VARIABLE
C     STORE BOWL PRESS, ELEMENT PRESS AND FLOW, AND PASSAGE DELTA P'S
C      D(CONSBL+F)=1./D(CONSLL+F)
      DT(PRESSB+H)=P(L1)
      DT(PRESSL+H)=P(L2)
      DT(GIN+H)=1.0/Z(L1)+D(VOLIN+F)
      DT(GOUT+H)=1.0/Z(L2)+D(VOLEX+F)
      DT(GTOT+H)=(DT(GIN+H)+DT(GOUT+H))/(DT(GIN+H)*DT(GOUT+H))
      DT(GTOT+H)=0.5*(DT(GTOT+H)+D(CONSBL+F))/D(CONL2+F)
      DT(PBL+H)=DT(PRESSB+H)
      DT(PEL+H)=DT(PRESSL+H)
      DT(QL+H)=0.0
 2020 CONTINUE
      RETURN
C
C     SECT 3000 CALCULATES AND STORES INTERNAL AND CONNECTION PRESS'S
C     AND FLOWS
 3000 CONTINUE
      DO 3060 I=1,3
      J=2*I-1
      N=J+1
      L1=L(J)
      L2=L(N)
      F=(I-1)*4
      H=(I-1)*8
 3052 CONTINUE
      DPIN=(C(L1)/Z(L1)+D(VOLIN+F)*DT(PRESSB+H))/DT(GIN+H)
      DPOUT=(C(L2)/Z(L2)+D(VOLEX+F)*DT(PRESSL+H))/DT(GOUT+H)
      DT(QL+H)=SQRT(DT(GTOT+H)**2.+ABS(DPIN-DPOUT)/D(CONL2+F))
      DT(QL+H)=SIGN(DT(QL+H)-DT(GTOT+H),DPIN-DPOUT)
      DT(PBL+H)=DPIN-DT(QL+H)/DT(GIN+H)
      DT(PEL+H)=DPOUT+DT(QL+H)/DT(GOUT+H)
      C(L1)=(C(L1)-DT(PBL+H))/Z(L1)
      P(L1)=DT(PBL+H)
      C(L2)=(C(L2)-DT(PEL+H))/Z(L2)
      P(L2)=DT(PEL+H)
      DT(PRESSB+H)=DT(PBL+H)
```

6.82.7 (Continued)

```
      DT(PRESSE+H)=DT(PEL+H)
 3060 CONTINUE
      IF(DT(PEL+16)-DT(PBL).LT.PREL)GO TO 3200
      L1=L(1)
      L2=L(2)
      L3=L(3)
      L4=L(4)
      L5=L(5)
      L6=L(6)
      GEQ1=DT(QE)/(DT(PBL)-DT(PEL))
      GEQ2=DT(QE+16)/(DT(PBL+16)-DT(PEL+15))
 3152 G1=1./Z(L1)
      G3=1./Z(L2)
      G7=1./Z(L5)
      G8=1./Z(L6)
      DO 20 I1=1,4
      DO 10 J1=1,4
   10 A(J1,I1)=0.0
      B(I1)=0.0
   20 CONTINUE
      A(1,1)=-G1-D(VOLIN)-GREL-GEQ1
      A(1,2)=GEQ1
      A(1,4)=GREL
      A(2,1)=GEQ1
      A(2,2)=-GEQ1-D(VOLEX)-G3
      A(3,3)=-G7-D(VOLIN+8)-GEQ2
      A(3,4)=GEQ2
      A(4,1)=GREL
      A(4,3)=GEQ2
      A(4,4)=-GEQ2-D(VOLEX+8)-GREL-G8
      B(1)=-C(L1)*G1-DT(PBL)*DT(VOLIN)+PREL*GREL
      B(2)=-DT(PEL)*D(VOLEX)-C(L2)*G3
      B(3)=-C(L5)*G7-DT(PBL+16)*D(VOLIN+8)
      B(4)=-DT(PEL+16)*D(VOLEX+8)-PREL*GREL-C(L6)*G8
      CALL SIMULT (A,B,4,1)
      DT(PRESSB)=B(1)
      DT(PRESSE)=B(2)
      DT(PRESSB+16)=B(3)
      DT(PRESSE+16)=B(4)
      Q(L1)=(C(L1)-B(1))/Z(L1)
      Q(L2)=(C(L2)-B(2))/Z(L2)
      Q(L5)=(C(L5)-B(3))/Z(L5)
      Q(L6)=(C(L6)-B(4))/Z(L6)
      P(L1)=B(1)
      P(L2)=B(2)
      P(L5)=B(3)
      P(L6)=B(4)
 3200 CONTINUE
      RETURN
      END
```

## 6.83 SUBROUTINE FILT83

FILT83 is a simulation of an inline, bypass type, multiconnection filter. One inlet and two outlet connections are used.

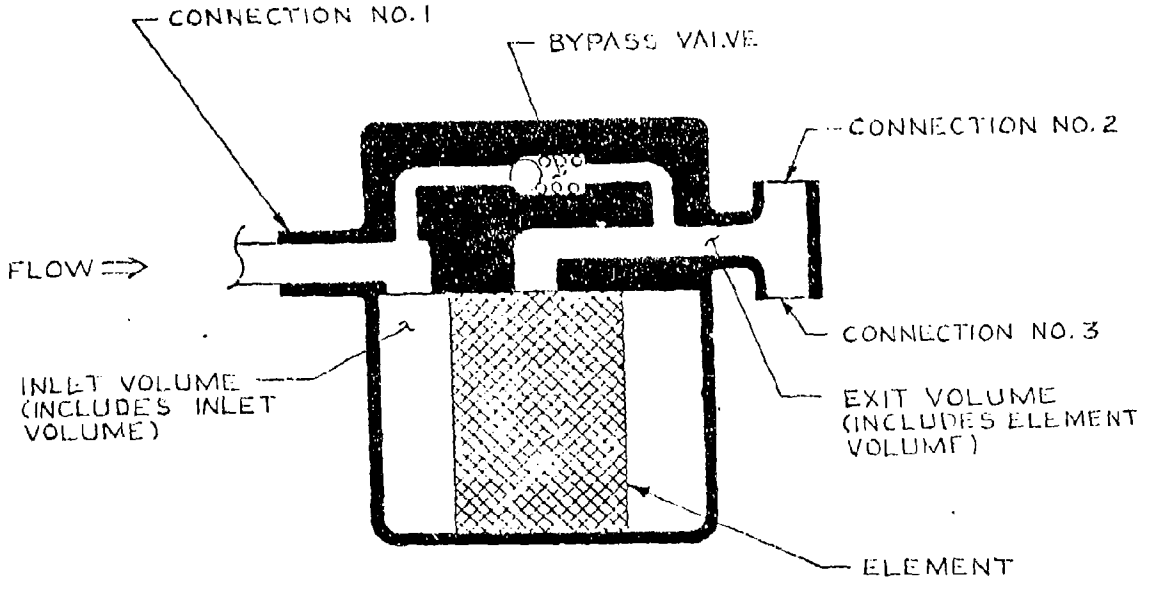



Figure 6.83-1
TYPE NO. 83 INLINE FILTER

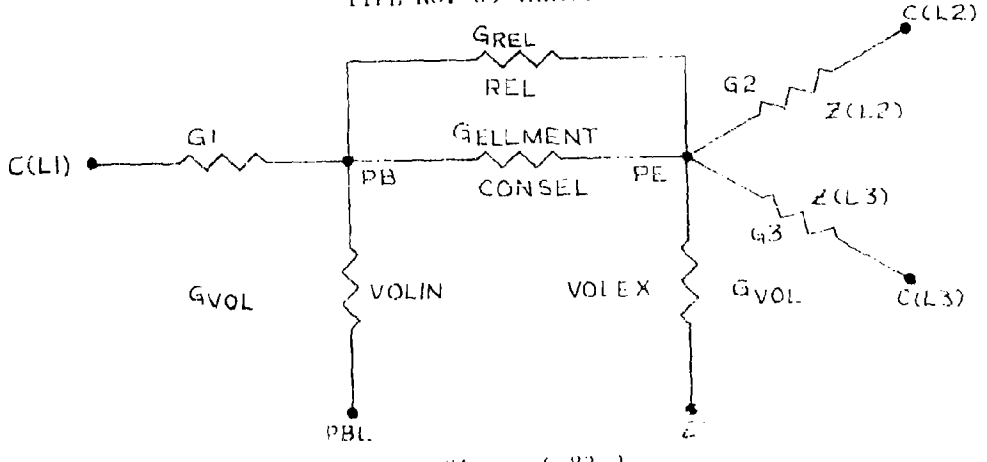



Figure 6.83-2
FILT83 CIRCUIT DIAGRAM

6.83.1 Math Model

FILT83 calculates the bowl and element pressures under transient conditions. FILT83 DIAGRAM, Figure 6.83-2, schematically describes the model where PB and PE, the bowl and element pressures, are to be determined. The past values of the bowl and element pressures, PBL and PEL, are shown at the ends of pseudo passages along with their conductances, D(VOLIN) and D(VOLEX), respectively. These conductances are calculated using the equation:

$$G_{vol} = VOL/(BULK*DELT)$$

where $G_{vol}$ = Conductance of the volume
BULK = fluid bulk modulus
VOL = volume of oil in bowl or element section
DELT = time step of program

The conductance of the element is the reciprocal of the input element constant.

$$G_{ele} = 1./D(CONSEL)$$

The conductance of the relief valve is the reciprocal of the input relief valve constant

$$G_{rel} = 1./D(REL)$$

The conductances of the three lines connected to the filter is simply the reciprocal of the individual line impedances.

$$G_{line} = 1./Z_{line}$$

With all the conductances calculated, flows are summed about PB and PE resulting in the following equations.

$$G1(PB-C(L1)) + G_{volin}(PB-PBL) + G_{ele}(PB-PE) + G_{rel}(PB-PE) = 0$$

$$G_{rel}(PE-PB)+G_{ele}(PE-PB) + G_{volex}(PE-PEL)+G2(PE-L(L2)) + G3(PE-UL3)) = 0$$

If the difference between the previously calculated values of PB and PE is less than the cracking pressure of the relief valve, $G_{rel}$ is set equal to zero. The above equations are then solved to yield the pressures PB and PE. The values of PB and PE are then set equal to PBL and PEL respectively. The inlet and outlet flows are then calculated

$$Q(L1) = (C(L1) - PB)/Z(L1)$$

$$Q(L2) = (C(L2) - PE)/Z(L2)$$

$$Q(L3) = (C(L3) - PE)/Z(L3)$$

6.83.2 Assumptions

The pressure losses at the inlet and outlet are assumed to be negligible and are, therefore, not included in the simulation. The relief valve is assumed to open and close instantaneously.

6.83.3 Computational Methods

SECTION 1000

Pseudo impedances for the bowl and element sections are calculated using

D(VOLIN) = BULK(KTEMP(1ND))/D(VOLIN)*DELT

D(VOLEX) = BULK(KTEMP(1ND))/D(VOLEX)*DELT

The filter element constant is adjusted for fluid other than MIL-H-5606 and/or temperature other than 100°F.

SECTION 1500

The leg constant for Q is updated and the outlet pressure is calculated and stored. The bowl and element pressures are initialized.

SECTION 2000

The impedances are converted to conductances

```
D(VOLEX) =  1./D(VOLEX)
D(VOLIN) =  1./D(VOLIN)
D(CONSEL) = 1./D(CONSEL)
D(REL)   =  1./D(REL)
```

The constants DT(G1), DT(G2), DT(G3) are evaluated

```
DT(G1) =  1./Z(L1) + D(VOLIN, + D(CONSEL)
DT(G2) =  -D(CONSEL)
DT(G3) =  -D(CONSEL) + D(VOLEX) + 1./Z(L2) + 1./Z(L3)
```

SECTION 3000

The difference between the previous values of the bowl and element pressure, DT(PBL) and DT(PEL), is compared to the relief pressure. If the relief pressure is exceeded DRELF, the relief valve conductance, is set equal to the input value for the relief valve conductance, D(REL). If the relief pressure is not exceeded DRELF is set to 0.0.

```
DRELF =  0.0
IF(DT(PBI) - DT(PEL).GT.D(RELPR))DRELF=D(REL)
```

The bowl and element pressures, DT(PBL) and DT(PEL) are then calculated and stored

```
DG1 =  DT(G1) + DRELF
DG2 =  DT(G2) - DRELF
DG3 =  DT(G3) + DRELF
R1 = DT(PB2)*D(VOLIN)+C(L1)/Z(L1)+D(RELPR)*DRELF
R2 = DT(PEL)*D(VOLEX)+C(L2)/Z(L2)+C(L3)/Z(L3)-D(RELPR)*DRELF
DT(PEL)=(R2*DG1-R1*DG2)/(DG3*DG1-DG2**2.)
DT(PBL)=(R1*DG3-R2*DGL)/(DG1*DG3-DG2**2.)
```

The pressures and flows at each of the connections are then calculated and stored.

6.83.4 Approximations

The relationship between pressure drop and flow across the filter is assumed to be linear.

### 6.83.5 Limitations

The accuracy of the model will decrease at flow rates where the effects of the pressure losses of inlet and outlet passages become significant.

### 6.83.6 Variable Names

| Variable | Description | Units |
|---|---|---|
| D(CONSEL) | Element Constant | PSI/CIS |
| DG1 | Dummy Variable | -- |
| DG2 | Dummy Variable | -- |
| DG3 | Dummy Variable | -- |
| DT(G1) | Sum of Inlet Conductances | CIS/PSI |
| DT(G2) | Element Constant | PSI/CIS |
| DT(G3) | Sum of Outlet Conductances | CIS/PSI |
| L1 | Dummy Variable | -- |
| L2 | Dummy Variable | -- |
| L3 | Dummy Variable | -- |
| DT(PBL) | Bowl Pressure | PSI |
| DT(PEL) | Element Pressure | PSI |
| R1 | Intermediate Calculation | -- |
| R2 | Intermediate Calculation | -- |
| D(REL) | Relief Valve Constant | PSI/CIS |
| D(RELPR) | Relief Pressure | PSI |
| D(VOLEX) | Exit Volume | $IN^3$ |
| D(VOLIN) | Inlet Volume | $IN^3$ |

## 6.83.7 Subroutine Listing

```
      SUBROUTINE FILT83 (D,DT,DD,L)
C**** REVISED MAY 1, 1976 ****
      COMMON NTOLPL,NTOL2L,IPT,IPOINT,NPTS,INLL,KNLL,NTOPL,NLPLT(61,3),
     1 POLEG(90,12),LCS(90,10),ILEG(1400),PN(90),QN(90)
      COMMON/SUB/PARM(150,9),PM(1500),QM(1500),P(300),Q(300),C(300)
     1,Z(300),RHO(20),S2ORHO(20),VISC(20),BULK(20),TEMP(20),PVAP(20)
     2,ATPRES,T,DELT,TFINAL,PLTDEL,PI,TITLE(20),LEGN,ICON
     3,KTEMP(99),LSTART(150),NLPT(150),LTYPE(99),NC(99),IMX,IMZ
     4,INV,ISTEP,NLINL,NEL,IND,IENTR,MNLINE,MNEL,MNLEG,MNNODE,MNPLOT
     5,MNLPTS,MDS
      DIMENSION D(1),DT(1),DD(1),L(1)
      INTEGER VOLIN,VOLEX,CONSEL,PBL,PEL,G1,G2,G3,REL,RELPR
C**** D ARRAY VARIABLES ****
      DATA VOLIN/1/,VOLEX/2/,CONSEL/3/,REL/4/,RELPR/5/
C**** DT ARRAY VARIABLES ****
      DATA PBL/1/,PEL/2/,G1/3/,G2/4/,G3/5/
      IF(IENTR) 1000,2000,3000
C     SECTION 1000 ESTABLISHES CONSTANTS FOR USE IN FUTURE CALCULATIONS
 1000 CONTINUE
      IF (INEL.NE.0) GO TO 1500
C     SET CONSTANTS FOR INTEGRATION OF PRESSURES
      D(VOLIN)=BULK(KTEMP(IND))/D(VOLIN)*DELT
      D(VOLEX)=BULK(KTEMP(IND))/D(VOLEX)*DELT
C     ADJUST INPUTTED CONSTANTS FOR FLUID OTHER THAN MIL-H-5606 AND/OR
C       TEMPERATURES OTHER THAN 100 DEG F
      D(CONSEL)=VISC(KTEMP(IND))*RHO(KTEMP(IND))/(.023*8.2E-5)*D(CONSEL)
      IF(D(CONSEL).LE.0.0) D(CONSEL)=.0001
      RETURN
 1500 CONTINUE
      IF(KNEL-2)1510,1610,1610
 1510 DA=POLEG(INEL,1)
      DS=POLEG(INEL,2)
      POLEG(INEL,6)=POLEG(INEL,6)+D(CONSEL)
      DT(PBL)=POLEG(INEL,11)
      POLEG(INEL,11)=POLEG(INEL,11)-DA*DS*D(CONSEL)
      DT(PEL)=POLEG(INEL,11)
 1610 RETURN
 2000 CONTINUE
      D(VOLEX)=1./D(VOLEX)
      D(VOLIN)=1./D(VOLIN)
      D(CONSEL)=1./D(CONSEL)
      D(REL)=1./D(REL)
      DT(G1)=1./Z(L(1))+D(VOLIN)+D(CONSEL)
      DT(G2)=-D(CONSEL)
      DT(G3)=D(CONSEL)+D(VOLEX)+1./Z(L(2))+1./Z(L(3))
      RETURN
C
C     SECT 3000 CALCULATES AND STORES INTERNAL AND CONNECTION PRESS'S
C     AND FLOWS
 3000 CONTINUE
```

**6.83-7 (Continued)**

```
      L1=L(1)
      L2=L(2)
      L3=L(3)
      DRELF=0.0
      IF(DT(PBL)-DT(PEL).GT.D(RELPR)) DRELF=D(REL)
      DG1=DT(G1)+DRELF
      DG2=DT(G2)-DRELF
      DG3=DT(G3)+DRELF
      R1=DT(PBL)*D(VOLIN)+C(L1)/Z(L1)+D(RELPR)*DRELF
      R2=DT(PEL)*D(VOLEX)+C(L2)/Z(L2)+C(L3)/Z(L3)-D(RELPR)*DRELF
      DT(PEL)=(R2*DG1-R1*DG2)/(DG3*DG1-DG2**2.)
      DT(PBL)=(R1*DG3-R2*DG2)/(DG1*DG3-DG2**2.)
      Q(L1)=(C(L1)-DT(PBL))/Z(L1)
      P(L1)=DT(PBL)
      Q(L2)=(C(L2)-DT(PEL))/Z(L2)
      P(L2)=DT(PEL)
      Q(L3)=(C(L3)-DT(PEL))/Z(L3)
      P(L3)=DT(PEL)
      RETURN
      END
```

## 6.92 SUBROUTINE CAD92

Subroutine CAD92 is a dummy input subroutine which can be used in the place of a guidance and control subroutine, to input position commands to the elevon actuator subroutine.

The data is inputted in a tabular form, and accessed via a first order table lookup routine. The data input, described in Volume I, contains the elevon actuator component number (L(1)), the number of data points in the tables (inputted as L(11) and stored as L(5)).

The table of time values, starts in D(9) and can occupy 8 or more spaces in groups of 8, this table is followed by the table of input commands which is also stored in groups of 8.

The first 8 spaces in D( ) are reserved for hinge moment data, which will be added later.

### 6.92.1 Subroutine Listing

```
      SUBROUTINE CAD92 (D,DT,DD,L)
C *** REVISED SEPTEMBER 1975 ***
      DOUBLE PRECISION DD
      INTEGER TLUD
      COMMON NTELPL,NTOLPL,IPT,IPOINT,NPTS,INEL,KNEL,NTOPL,NLPLT(61,3),
     1 PQLEG(90,12),LCS(90,10),ILEG(1400),PN(90),QN(90),VSTORE(6150)
      DIMENSION D(24),DT(1),DD(1),L(5)
      COMMON/SUB/PARM(150,9),PM(1500),QM(1500),P(300),Q(300),C(300)
     1,Z(300),RHO(20),S2ORHO(20),VISC(20),BULK(20),TEMP(20),PVAP(20)
     2,ATPRES,T,DELT,TFINAL,PLTDEL,PI,TITLE(20),LEGN,ICON
     3,KTEMP(99),LSTART(150),NLPT(150),LTYPE(99),NC(99),INX,INZ
     4,INV,ISTEP,NLINE,NEL,IND,IENTR,MNLINE,MNEL,MNLEG,MNNODE,MNPLOT
     5,MNLPTS,MDS
      COMMON /COMPD/ XD(4500),LX(1500),LEX(99,4)
      DATA TLUD/1/
      IF (IENTR) 1000,2000,3000
 1000 CONTINUE
      DT(TLUD)=0.0
      L(3)=9
      L(4)=9+(L(11)+7)/8*8
      L(5)=L(11)
      NCOMP=L(1)
      L(2)=LEX(NCOMP,2)
      IF(NCOMP.GT.IND) GO TO 1500
      XD(LEX(NCOMP,2))=D(L(4))
      GO TO 3300
 1500 CONTINUE
      WRITE(6,1600) NCOMP,IND
 1600 FORMAT(2X,14H COMPONENT NO,I5,
     1 42H SHOULD HAVE A LOWER NO THAN COMPONENT NO,I5)
      RETURN
 2000 CONTINUE
      GO TO 3100
 3000 CONTINUE
      IF(T-DT(TLUD).LT.0.040) GO TO 3400
      DT(TLUD)=T
 3100 CALL INTERP (T,D(L(3)),D(L(4)),10,L(5),VALY,IOD)
      NCOMP=L(1)
      XD(L(2))=VALY
 3300 XD(L(2)+3)=D(1)
      XD(L(2)+4)=D(2)
 3400 RETURN
      END
```

## 6.93 SUBROUTINE CAD93

Subroutine CAD93 is a dummy input subroutine which can be used in the place of a pump subroutine, to input torque loads to the APU subroutine.

The data is inputted in a tabular form, and accessed via a first order table lookup routine. The data input, described in Volume I, contains the number of data points in the tables (inputted as L(11) and stored as L(5)).

The table of time values, starts in D(9) and can occupy 8 or more spaces in groups of 8, this table is followed by the table of torque data which is also stored in groups of 8.

The first 8 spaces in D( ) are reserved for oscillatory torque data, which will be added later.

## 6.93.1 Subroutine Listing

```
      SUBROUTINE CAD93 (D,DT,DD,L)
C *** REVISED SEPTEMBER 1975 ***
      DOUBLE PRECISION DD
      DIMENSION D(1),DT(1),DD(1),L(1)
      COMMON NTELPL,NTOLPL,IPT,IPOINT,NPTS,INEL,KNEL,NTOPL,NLPLT(61,3),
     1 PQLEG(90,12),LCS(90,10),ILEG(1400),PN(90),QN(90),VSTORE(6150)
      COMMON/SUB/PARM(150,9),PM(1500),QM(1500),P(300),Q(300),C(300)
     1,Z(300),RHO(20),S2ORHO(20),VISC(20),BULK(20),TEMP(20),PVAP(20)
     2,ATPRES,T,DELT,TFINAL,PLTDEL,PI,TITLE(20),LEGN,ICON
     3,KTEMP(99),LSTART(150),NLPT(150),LTYPE(99),NC(99),INX,INZ
     4,INV,ISTEP,NLINE,NEL,IND,IENTR,MNLINE,MNEL,MNLEG,MNNODE,MNPLOT
     5,MNLPTS,MDS
      COMMON /COMPD/ XD(4500),LX(1500),LEX(99,4)
      IF (IENTR) 1000,2000,3000
 1000 CONTINUE
      L(3)=9
      L(4)=9+(L(11)+7)/8*8
      L(5)=L(11)
      DT(1)=D(3)
      RETURN
 2000 CONTINUE
      RETURN
 3000 CONTINUE
      CALL INTERP (T,D(L(3)),D(L(4)),10,L(5),VALY,IOD)
      DT(2)=VALY+DT(1)*D(2)+D(1)
      RETURN
      END
```

## 6.95 SUBROUTINE APU95

The APU95 subroutine is a generalized subroutine set up to model the auxiliary power units of the space shuttle. The function of the model is to return values of RPM to the pump subroutine. The value of RPM is determined by burning the liquid fuel and looking at the loads of the gear box, hydraulic pump and fuel pump.

### 6.95.1 Math Model

The block diagram of the APU analytical model is shown in Figure 6.95-1. An ON-OFF fuel controller is used to maintain the speed of the turbine between an upper speed - RPMH and a lower speed - RPML. The liquid fuel is converted to gaseous state in the hydrazing gas generator to drive the turbine. In each cycle, after fuel cut off, the turbine speed reduces rapidly due to the loads reflected by the gear box, hydraulic pump, and fuel pump. At sea level the APU system cycles off and on approximately 1 cps with zero flow from the hydraulic pump, but cycling increases to approximately 3 cps with maximum flow from the pump. The functions and the nomenclature of the APU model are in Table 6.95-1.

Fuel Pump Calculations of Flow Rate, Change in Pressure and Horsepower

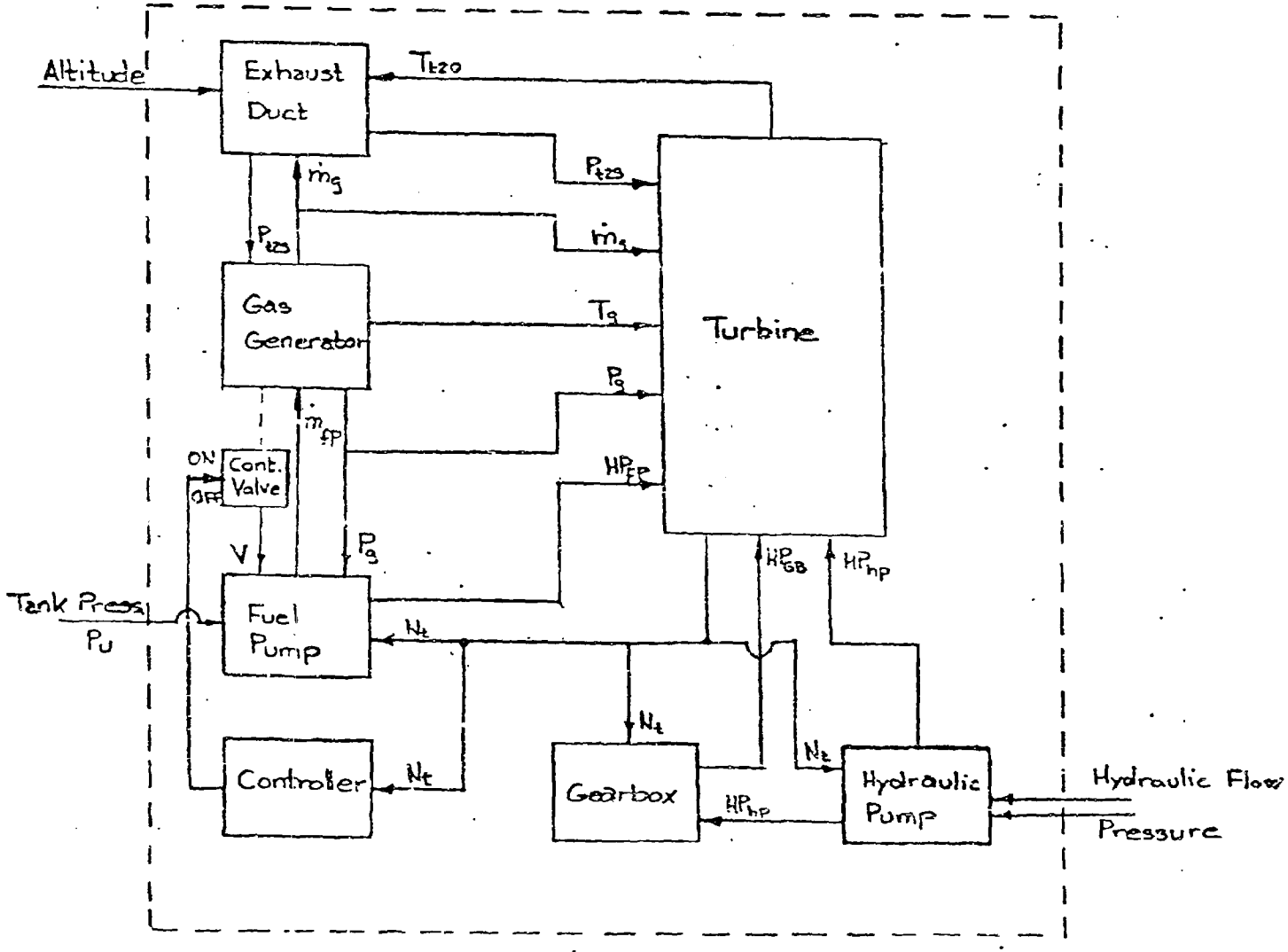
$$\dot{m}_{FP} = 3.0267 \times 10^{-6} N_T (V)(SW)$$

$$\Delta P_{FP} = P_G - P_U + 105.28 \dot{m}_{FP} \qquad (400 \leq P_U \leq 80)$$

$$HP_{FP} = 7.0042 \times 10^{-3} \dot{m}_{FP} \Delta P_{FP}$$

FIGURE 6.95-1

# BLOCK DIAGRAM OF THE APU ANALYTICAL MODEL



| VARIABLE NAME | DESCRIPTION | DIMENSIONS |
|---|---|---|
| $C_o$ | - Gas Spouting Velocity | ft/sec |
| $H_{AD}$ | - Adiabatic Head | $Btu/lb_m$ |
| HP | - Horsepower | HP |
| $\dot{m}$ | - Mass Flowrate | lb/sec |
| $\eta_T$ | - Turbine Efficiency | -- |
| N | - Rotating Speed | RPM |
| P | - Pressure | psia |
| $\rho$ | - Density | $lbm/ft^3$ |
| $R_e$ | - Reynolds Number | |
| $R_e^*$ | - Machine Reynolds Number | |
| T | - Temperature | $^oR$ |
| $\tau$ | - Torque | in-lb |
| U | - Tip Velocity | ft/sec |
| V | - Control Valve Area Factor | |

| SUBSCRIPT NAME | DESCRIPTION | SUBSCRIPT NAME | DESCRIPTION |
|---|---|---|---|
| d | Disk | U | Ullage |
| fP | Fuel Pump | 0 | Total |
| g | Gas Generator | 1 | Inlet |
| GB | Gearbox | 2 | Outlet |
| hp | Hydraulic Pump | | |
| S | Static | | |

TABLE 6.95-1

FUNCTIONS AND NOMENCLATURE OF THE
APU ANALYTICAL MODEL

Controller

If $(N_T \geq RPMH)$ SW = 0.0

If $(N_T < RPML)$ SW = 1.0

Control Valve Position

$$V = \left[1 - \frac{1.0}{e^{(\Delta T/0.01)}}\right] \qquad \text{(Opening Transient)}$$

$$V = \left[\frac{1.0}{e^{(\Delta T/0.01)}}\right] \qquad \text{(Closing Transient)}$$

Gas Generator

Subsonic Flow: $1 > \frac{P_{T2S}}{P_G} \geq .55$

$$\dot{m}_G = \frac{0.046513}{\sqrt{T_G}} P_G \left(\frac{P_{T2S}}{P_G}\right)^{(0.78431)} \sqrt{\left[1 - \left(\frac{P_{T2S}}{P_G}\right)^{(0.21569)}\right]}$$

Sonic Flow: $\frac{P_{T2S}}{P_G} < .55$

$$\dot{m}_C = 0.01012 \frac{P_G}{\sqrt{T_G}}$$

$$\left(\frac{dP_G}{dt}\right) = 117.903\ T_G(\dot{m}_{FP} - \dot{m}_G) \qquad T_G = 2160°R$$

$$P_G = \int \left(\frac{dP_G}{dt}\right) dt$$

Exhaust Duct

Space:

$$P_{T2S} = 1.0688\ \dot{m}_G (T_{T20})^{(0.49938)} - 0.50097\ (\dot{m}_G)^2 (T_{T20})^{(0.48268)}$$

Sea Level:

$$P_{T2S} = 14.7 + 2.2086 \times 10^{-3}\ (\dot{m}_G)\ (T_{T20})^{(0.79309)}$$

$$+ 0.0142\ (\dot{m}_G)^2\ (T_{T20})^{(1.0257)}$$

20,000 FT:

$$P_{T2S} = 6.75 + 4.5162 \times 10^{-6}\ (\dot{m}_G)\ (T_{T20})^{1.7809}$$

$$+ 0.16848\ (\dot{m}_G)\ (T_{T20})^{0.78269}$$

Turbine

$$\frac{dN_T}{dt} = \frac{2.9033 \times 10^7}{N_T} \left(1.3432\ \dot{m}_G\ H_{AD}\ \eta_T - HP_{FP} - HP_{GB} - D_F - HP_{HP}\right)$$

$$H_{AD} = 0.70856\ T_G \left[1 - \frac{P_{T2S}}{P_G}^{(0.21569)}\right]$$

$$C_o = 188.375 \sqrt{T_G \left[1 - \frac{P_{T2S}}{P_G}^{(0.21569)}\right]}$$

$$U = 0.023998\ N_T$$

$$\eta_T = f(U/C_o,\ P_G)$$

Efficiency Data From EFF vs $U/C_o$ Curves

$$T_{T20} = T_G\ (1 - \eta_T) + \eta_T\ T_G\ \frac{P_{T2S}}{P_G}^{(0.21569)}$$

$$D_F = 8.3900 \times 10^{-7}\ C_D\ U^3 \rho$$

$$C_D = \frac{1541.89}{Red}$$

$$C_D = \frac{2.1341}{(Red)^{1/2}}$$

$$C_D = \frac{0.20017}{(Red)^{1/4}}$$

$$C_D = \frac{0.058832}{(Red)^{1/5}}$$

Maximum Value

$$R_E^* = 79.94\ N_T \rho$$

$$R_{E_D} = 0.89236\ R_E^*$$

$$\rho = 1.21165\ \frac{P_{T2S}}{T_{T20}}$$

$$N_T = \int \left(\frac{dNt}{dt}\right)\ dt$$

$$HP_{GB} = f\ (N_T,\ HP_{HP})$$

Gearbox Loss Data from Input Data Curve

6.95.2 Assumptions

TBD

6.95.3 Computations

1000 Section

In the 1000 section the steady state APU characteristics are stored in the temporary variable array(DT). The controller for the fuel pump is initially turned off and the valve area factor (V) is set to 1.

2000 Section

In the 2000 section variables are calculated for use in the transient section in order to decrease running time.

3000 Section

This section of the APU95 subroutine computes the output RPM of the APU through a gearbox and feeds this value to the appropriate pump in the simulated hydraulic system.

On transfer to the 3000 section the previous pump RPM in the DT array is divided by the gearbox ratio to obtain an APU RPM. The fuel pump controller is set to maintain the speed of the turbine between the upper DT(RPMH) and lower DT(RPML) RPM values. The fuel pump horsepower is computed from the given RPM and controller position as follows:

EMDOFP = 3.0267E-6* RPM*V

DELPFP = DT(PG)-DT(PU)+105.28*EMDOFP

HPFP = 7.0042E-3*EMDOFP*DFLPFP

The torque of the pump is also computed externally and returned to the APU95 subroutine. The APU subroutine converts the torque to net horsepower and uses this value along with the APU RPM to obtain the horsepower loss through the gearbox.

The next step in the computations is to determine the flow characteristics in the gas generator, and then calculate the flow rate, gas inlet pressure and change in gas inlet pressure.

Knowing the gas inlet pressure and computing the $U/C_o$ value, the APU efficiency is obtained from a table lookup. The APU efficiency vs $U/C_o$ curves were input with the APU data.

After determining the exhaust duct conditions the differential change in RPM (DRPM) is calculated. The actual value of APU RPM is determined as follows:

RPM = RPM + DELT*(DRPM+DT(DRPMO))/2.0

where

RPM - The initial entry RPM

DELT - Transient time step

DRPM - Computed differential change in RPM for this time step

DRPMO - Previous differential change in RPM

The new value of RPM is multiplied by the gearbox ratio and the value is returned to the calling subroutine.

6.95.4 Approximations- TBD

6.95.5 Limitations - TBD

6.95.6 Variable Names

| Variable | Description | Dimensions |
|---|---|---|
| CD | Maximum value of variables | |
| CDO | Array of temporary variables | |
| CED | Array of input constants | |
| CO | Gas spouting velocity | ft/sec |
| CV | Delay variable of value position | |
| DELPFP | Pressure drop across fuel pump | psi |

| Variable | Description | Dimensions |
|---|---|---|
| DF | Temporary Variable | |
| DPG | Rate of Change of Gas pressure | |
| DPGO | Previous value of DPG | |
| DPGSAV | Previous value of DPGO | |
| DRPM | Rate of change of RPM | |
| DRPMO | Previous value of DRPM | |
| DT | Delta time step | sec |
| DTC | Delay time | sec |
| EFF | Efficiency of APU | |
| EMDOTG | Mass flow rate - gas generator | lb/sec |
| EMDOFP | Mass flow rate - fuel pump | lb/sec |
| GBR | Gear box ratio | |
| HPFP | Fuel pump H.P. | |
| HPGB | Gear box HP loss | |
| HPHP | Horsepower hydraulic pump | |
| HAD | A diabatic head | btu/lbs mass |
| I | Temporary integer variable | |
| IE | Integer variable | |
| ISW | Controller indicator 1 = on - opening transient<br>2 = off - closing transient | |
| NAGB | Constants for gear box data | |
| NEXTR | Temporary variable | |
| PEXO | External pressure | psi |
| PG | Gas pressure | psi |
| PGO | Initial value of PG | psi |
| PGSAV | Previous value of PG | psi |

| Variable | Description | Dimension |
|---|---|---|
| POPTPG | Temporary Variable | |
| PTPG | Temporary variable | |
| PT2S | Turbine exhaust pressure | psi |
| PT2SO | Previous value of PT2S | |
| PU | Fuel tank pressure | psi |
| PUO | Initial fuel tank pressure | psi |
| RED | Temp variable | |
| RES | Temp variable | |
| RESTAR | Temp variable | |
| RHO | Temp variable | |
| RPM | Turbine RPM | rpm |
| RPMH | RPM high limit for valve closing | rpm |
| RPML | RPM low limit for valve opening | rpm |
| RPMO | Initial RPM | rpm |
| RTRED | SQRT or RED | |
| T | Time | sec |
| TCVO | Time at Valve closure | sec |
| TFAPU | Time to fail APU | sec |
| TG | Gas temperature | $^{o}R$ |
| TGO | Initial gas temp | |
| TGPOW | Temperature | |
| TT2O | Outlet gas temp | $^{o}R$ |
| U | Tip velocity | ft/sec |
| UCO | U divided by CO | |
| V | Control valve area factor | |
| ZEM | Temporary Variable | |

## 6.95.7 Subroutine Listing

```
      SUBROUTINE APU95(D,DT,DD,L)
C                                        REVISED 21 AUG '75
C
C     AUXILIARY POWER UNIT  (APU)
C
      INTEGER PT2SO,PGO,TGO,PUO,RPMH,RPML,
     + CED1,CED2,CED3,CED4,TFAPU,GBR,
     + PT2S,PG,TLUD,DPGO,DRPMO,DPG,DPGSAV,TCVO,PGSAV
      COMMON/SUB/PARM(150,9),PM(1500),QM(1500),P(300),Q(300),C(300)
     1,Z(300),RHO(20),S2ORHO(20),VISC(20),BULK(20),TEMP(20),PVAP(20)
     2,ATPRES,T,DELT,TFINAL,PLTDEL,PI,TITLE(20),LEGN,ICON
     3,KTEMP(99),LSTART(150),NLPT(150),LTYPE(99),NC(99),INX,INZ
     4,INV,ISTEP,NLINE,NEL,IND,IENTR,MNLINE,MNEL,MNLEG,MNNODE,MNPLOT
     5,MNLPTS,MDS
      COMMON /COMPD/DARRAY(4500),LT(1500),LE(99,4)
      DIMENSION D(112),DT(15),DD(1),L(7)
      DIMENSION NAGB(2),XGB(8),YGB(15),XA(2),CDO(4)
      DIMENSION TABX(4),TABY(4)
      DATA NAGB/5,3/,
     + XGB/20.,40.,60.,80.,180.,60000.,72000.,80000./,
     + YGB/2.37,2.62,2.85,3.11,4.43,3.35,3.63,3.91,4.19,5.68,
     + 4.17,4.46,4.77,6.08,6.72/
      DATA PT2SO/1/,PGO/2/,TGO/3/,PUO/4/,RPMH/5/,RPML/6/,
     + CED1/7/,CED2/8/,CED3/9/,CED4/10/,TFAPU/11/,GBR/12/,
     + PT2S/1/,PG/2/,TLUD/3/,DPGO/4/,DRPMO/5/,DPG/6/,
     + DPGSAV/7/,PGSAV/8/,TCVO/9/
      DATA TABX/-20.,60.,90.,140./,TABY/3.,7.6,10.8,20.3/
      IF(IENTR)1000,2000,3000
 1000 CONTINUE
      DT(10)=0.
      DT(11)=0.
      DT(12)=0.
      DT(13)=2
      DT(14)=1
      DT(15)=7200.
C
C     INITIALIZATION
C
      ISW=7
      ITER=0
      IF(D(TFAPU).EQ.0.)D(TFAPU)=10E6
      DT(PT2S)=D(PT2SO)
      DT(PG) = D(PGO)
      DT(TLUD)=0.0
      TG=D(TGO)
      PU=D(PUO)
      ADELT=DELT*1.0
      V = 1.0
C     -----
      L(ISW) = 2
```

6.95.7 (Continued)

```
C     -----
      DT(DPGO) = 0.0
      DT(DRPMO) = 0.0
      DT(TCVO) = -1.0
      RETURN
 2000 CONTINUE
      RETURN
 3000 CONTINUE
      ITER=ITER+1
      IF(T.EQ.0.0.OR.T.EQ.TFINAL)GO TO 201
      IF((T-DT(TLUD)).LT.ADELT*.99)RETURN
  201 DT(TLUD)=T
      L5=L(2)+L(3)+17
      N2=LE(L(1),2)
C      N2 - LOCATION OF FIRST DT VARIABLE IN PUMP
      RPM=DARRAY(N2)/D(GBR)
C      OUTPUT HORSEPOWER IS IN POSITION N2+1 OF DT
      CHP=ITER
      ITER=0
      X=DARRAY(N2+1)/CHP
      CALL INTERP(X,TABX,TABY,10,4,Y,IERR)
      HPHP=X+Y
C
C *** CONTROLLER
C
C     L(ISW) = 1     CONTROLLER ON    (OPENING TRANSIENT)
C     L(ISW) = 2     CONTROLLER OFF   (CLOSING TRANSIENT)
C
C
C     ----- TEST FOR FAILURE OF FUEL PUMP
C
      IF(T.LT.D(TFAPU)) GO TO 55
      DT(TCVO) = T
      V = 0.0
      GO TO 130
   55 CONTINUE
      LISW=L(ISW)
      GO TO(60,70),LISW
C
C     ----- CONTROLLER ON
C
   60 CONTINUE
      IF(RPM.LT.D(RPMH)) GO TO 90
      L(ISW) = 2
      GO TO 80
C
C     ----- CONTROLLER OFF
C
   70 CONTINUE
      IF(RPM.GE.D(RPML)) GO TO 90
```

6.95.7 (Continued)

```
          L(ISW) = 1
       80 CONTINUE
          DT(TCVO) = T
       90 CONTINUE
          DTC = T-DT(TCVO)
          CV = 1.0/EXP(DTC/0.01)
C
C *** CONTROL VALVE
C
          LISW=L(ISW)
          GO TO(110,120),LISW
      110 CONTINUE
          V = 1.0-CV
          GO TO 130
      120 CONTINUE
          V = CV
      130 CONTINUE
C
C *** FUEL PUMP
C
          EMDOFP = 3.0267E-6*RPM*V
          DELPFP = DT(PG)-PU+105.28*EMDOFP
          HPFP = 7.0042E-3*EMDOFP*DELPFP
C
C *** GEAR BOX
C
          XA(1) = HPHP
          XA(2) = RPM
          CALL LUCUP(3,NAGB,XGB,YGB,XA,HPGB,IE,NEXTR)
C
C *** GAS GENERATOR
C
          PTPG = DT(PT2S)/DT(PG)
          POPTPG = (PTPG)**0.21569
          ZEM = DT(PG)/SQRT(TG)
          IF(PTPG.LT.0.55) GO TO 150
C
C     ----- SUBSONIC FLOW
C
          EMDOTG = 0.046513*ZEM*SQRT(1.0-POPTPG)*(PTPG**0.78431)
          GO TO 160
      150 CONTINUE
C
C     ----- SONIC FLOW
C
          EMDOTG = 0.01012*ZEM
      160 CONTINUE
          DT(DPG) = 117.903*TG*(EMDOFP-EMDOTG)
          DT(DPGSAV) = DT(DPGO)
          DT(PGSAV) =DT(PG)
```

6.95.7 (Continued)

```
      DT(PG) = DT(PG)+ADELT*(DT(DPG)+DT(DPGO))/2.0
      DT(DPGO)=DT(DPG)
C
C *** TURBINE
C
      TGPOW = TG*(1.0-POPTPG)
      HAD = 0.70856*TGPOW
      CO = 188.375*SQRT(TGPOW)
      U = 0.023998*RPM
      UCO = U/CO
C     -----
      IF(DT(PG).LE.DT(PT2S)) UCO=0.325
      IF(CO.LE.0.0) UCO=0.325
      IF(UCO.GT.0.325) UCO=0.325
C     -----
      XA(1) = UCO
      XA(2) = DT(PG)
      CALL LUCUP(3,L(2),D(17),D(L5),XA,EFF,IE,NEXTR)
C
C     ----- EXHAUST DUCT CONDITIONS  -  PRESSURE AND TEMPERATURE
C
      TT20 = TG*(1.0-EFF)+EFF*TG*POPTPG
      DT(PT2S) = D(CED1)*EMDOTG*(TT20**D(CED2))+D(CED3)*EMDOTG*EMDOTG
     1    *(TT20**D(CED4))+ATPRES
      IF(DT(PG).GE.DT(PT2S)) GO TO 180
C     DT(DPGO) = (DT(PT2S)-DT(PGSAV))*2.0/ADELT-DT(DPGSAV)
      DT(DPGO)=(DT(PT2S)-DT(PGSAV))/ADELT
      DT(PG) = DT(PT2S)+.5
  180 CONTINUE
      DENS = 1.21165*DT(PT2S)/TT20
      RESTAR = 791.94*RPM*DENS
      RED = 0.89236*RESTAR
      CDO(1) = 1541.89/RED
      RTRED = SQRT(RED)
      CDO(2) = 2.1341/RTRED
      CDO(3) = 0.20017/SQRT(RTRED)
      CDO(4) = 0.058832/(RED**0.20)
      CD = CDO(1)
      DO 200 I=2,4
      IF(CDO(I).GT.CD) CD=CDO(I)
  200 CONTINUE
      DF = 8.3900E-7*CD*U*U*U*DENS
      DRPM = 2.9033E7*(1.3432*EMDOTG*HAD*EFF-HPFP-HPGB-DF-HPHP)/RPM
      RPM = RPM+ADELT*(DRPM+DT(DRPMO))/2.0
      DT(DRPMO) = DRPM
      DT(10)=HPFP
      DT(11)=HPGB
      DT(12)=HPHP
      DT(13)=LISW
      DT(14)=V
```

6.95.7 (Continued)

```
      DT(15)=RPM/10.
      DARRAY(N2)=RPM*D(GBR)
      DARRAY(N2+1)=0.0
      RETURN
      END
```

## 6.98 SUBROUTINE CAD98

Subroutine CAD98 is a dummy input subroutine which can be used in the place of a guidance and control subroutine, to input position commands to the SSME engine control (E.C.) actuators. Subroutine CAD98 will supply commands to up to nine actuators. The number of actuators to be used is input in L(10).

The data is inputted in a tabular form, and accessed via a first order table lookup routine. The data input, described in Volume I, contains the E.C. actuator component numbers (L(1) through L(9)), the number of data points in the tables (inputted as L(11) and stored as L(12)), and the number of actuators to be command (L(10)).

The table of time values, starts in D(1) and can occupy 8 or more spaces in groups of 8, this table is followed by the table of input commands which is also stored in groups of 8.

### 6.98.1 MATH MODEL

Not applicable.

### 6.98.2 ASSUMPTIONS

Not applicable.

### 6.98.3 COMPUTATION METHOD

SECTION 1000

The number of data points to be used by the 3000 section is computed. The initial command is calculated.

SECTION 1500

None.

SECTION 2000

None.

SECTION 3000

Commands to be supplied to the actuators are calculated.

6.98.4 APPROXIMATIONS

Not applicable.

6.98.5 LIMITATION

No load data is supplied to the actuators.

6.98.6 VARIABLES

| Variable | Description | Dimensions |
|---|---|---|
| NCOMP | Component Number of Actuator to be Commanded | - |
| DT(TLUD) | Calculation Interval | SEC |

## 6.98.7 Subroutine Listing

```
      SUBROUTINE CAD98 (D,DT,DD,L)
C *** REVISED DECEMBER 1975 ***
      DOUBLE PRECISION DD
      INTEGER TLUD
      COMMON NTELPL,NTOLPL,IPT,IPOINT,NPTS,INEL,KNEL,NTOPL,NLPLT(61,3),
     1 PQLEG(90,12),LCS(90,10),ILEG(1400),PN(90),QN(90),VSTORE(6150)
      DIMENSION D(1),DT(1),DD(1),L(1)
      COMMON/SUB/PARM(150,9),PM(1500),QM(1500),P(300),Q(300),C(300)
     1,Z(300),RHO(20),S2ORHO(20),VISC(20),BULK(20),TEMP(20),PVAP(20)
     2,ATPRES,T,DELT,TFINAL,PLTDEL,PI,TITLE(20),LEGN,ICON
     3,KTEMP(99),LSTART(150),NLPT(150),LTYPE(99),NC(99),INX,INZ
     4,INV,ISTEP,NLINE,NEL,IND,IENTR,MNLINE,MNEL,MNLEG,MNNODE,MNPLOT
     5,MNLPTS,MDS
      COMMON /COMPD/ XD(4500),LX(1500),LEX(99,4)
      DATA TLUD/1/
      IF (IENTR) 1000,2000,3000
 1000 CONTINUE
      DT(TLUD)=0.0
      NO=L(10)
      L(12)=1
      L(13)=1
      DO 1200 J=1,NO
      NCOMP=L(J)
      L(12)=L(12)+(L(11)+7)/8*8
      IF(NCOMP.GT.IND) GO TO 1500
 1200 XD(LEX(NCOMP,2))=D(L(12))
      RETURN
 1500 CONTINUE
      WRITE(6,1600) NCOMP,IND
 1600 FORMAT(2X,14H COMPONENT NO,I5,
     1 42H SHOULD HAVE A LOWER NO THAN COMPONENT NO,I5)
      RETURN
 2000 CONTINUE
      RETURN
 3000 CONTINUE
      J=L(10)
      IF((T-DT(TLUD)).LT.0.020) GO TO 3300
      J=L(10)
      DT(TLUD)=T
 3100 L(12)=1
      DO 3200 I=1,J
      L(12)=L(12)+(L(11)+7)/8*8
      CALL INTERP (T,D(L(13)),D(L(12)),10,L(11),VALY,IOD)
      NCOMP=L(I)
 3200 XD(LEX(NCOMP,2))=VALY
 3300 RETURN
      END
```

## 6.99 SUBROUTINE CAD99

CAD99 is a special type of component subroutine whose purpose is to provide the necessary interface between HYTRAN and the six-degree-of-freedom ( SDF) programs. There are two interface routines. CAD99D is used with the SDF descent program and CAD99A is used with the SDF ascent program.

### 6.99.1 Math Model

Not applicable.

### 6.99.2 Assumptions

Not applicable.

### 6.99.3 Computation Method

#### 1000 Section

The units conversions are input in this section and the beginning address of each actuator in the XD array is computed. The SDF program is called and the HYTRAN actuator commands are initialized to the required trim positions. Hinge moment slopes are initialized to zero and hinge moment intercepts are initialized to the hinge moments computed by the SDF program. No hinge moment data is available from the SDF ascent program.

#### 2000 Section

Not applicable.

#### 3000 Section

This section transfers actuator and surface positions from HYTRAN to the SDF program and it transfers actuator commands from SDF to the HYTRAN program. In addition hinge moment slopes and intercepts are computed from SDF decent data and transferred to HYTRAN. Data is transferred at flight control computation intervals (DTFCS).

6.99.4 Approximations

Not applicable.

6.99.5 Limitations

Not applicable.

6.99.6 Variable Names

| Variable | Description | Dimensions |
|---|---|---|
| XD(BF) | Body Flap Rate Command | |
| XD(BF+1) | Body Flap Position | deg |
| XD(BF+3) | Body Flap Hinge Moment Intercept | in-lb |
| XD(BF+4) | Body Flap Hinge Moment Slope | lb |
| DBI | Inboard Elevon Degree Bias | deg |
| DBO | Outboard Elevon Degree Bias | deg |
| DTFCS | Flight Control Computation Interval | sec |
| DTVEI | Inboard Elevon Conversion Factor | v/deg |
| DTVEO | Outboard Elevon Conversion Factor | v/deg |
| DTVBF | Body Flap Conversion Factor | v/deg |
| DTVR | Rudder Conversion Factor | v/deg |
| DTVSB | Speedbrake Conversion Factor | v/deg |
| DRESET | 1 = reset, 0 = not reset | |
| L(1) | Left Outboard Elevon Component Number | - |
| L(2) | Left Inboard Elevon Component Number | - |
| L(3) | Right Outboard Elevon Component Number | - |
| L(4) | Right Inboard Elevon Component Number | - |
| L(5) | Rudder/Speedbrake Component Number | - |
| L(6) | Body Flap Component Number | - |
| L(7) | Engine 1 Pitch TVC Component Number | - |

| Variable | Description | Dimensions |
|---|---|---|
| L(8) | Engine 1 Yaw TVC Component Number | -- |
| L(9) | Engine 2 Pitch TVC Component Number | -- |
| L(10) | Engine 2 Yaw TVC Component Number | -- |
| L(11) | Engine 3 Pitch TVC Component Number | -- |
| L(12) | Engine 3 Yaw TVC Component Number | -- |
| XD(LIBE) | Left Inboard Elevon Command | v |
| XD(LIBE+3) | Left Inboard Elevon Hinge Moment Intercept | in-lb |
| XD(LIBE+4) | Left Inboard Elevon Hinge Moment Slope | lb |
| XD(LIBE+37) | Left Inboard Elevon Position | deg |
| XD(LOBE) | Left Outboard Elevon Command | v |
| XD(LOBE+3) | Left Outboard Elevon Hinge Moment Intercept | in-lb |
| XD(LOBE+4) | Left Outboard Elevon Hinge Moment Slope | lb |
| XD(LOBE+37) | Left Outboard Elevon Position | deg |
| XD(N1P) | Engine 1 TVC Pitch Command | v |
| XD(N1P+37) | Engine 1 TVC Pitch Position | in |
| XD(N1Y) | Engine 1 TVC Yaw Command | v |
| XD(N1Y+37) | Engine 1 TVC Yaw Position | in |
| XD(N2P) | Engine 2 TVC Pitch Command | v |
| XD(N2P+37) | Engine 2 TVC Pitch Position | in |
| XD(N2Y) | Engine 2 TVC Yaw Command | v |
| XD(N2Y+37) | Engine 2 TVC Yaw Position | in |
| XD(N3P) | Engine 3 TVC Pitch Command | v |
| XD(N3P+37) | Engine 3 TVC Pitch Position | in |
| XD(N3Y) | Engine 3 TVC Yaw Command | v |
| XD(N3Y+37) | Engine 3 TVC Yaw Position | In |

| Variable | Description | Dimension |
|---|---|---|
| XD(R1BE) | Right Inboard Elevon Command | v |
| XD(R1BE+3) | Right Inboard Elevon Hinge Moment Intercept | in-lb |
| XD(R1BE+4) | Right Inboard Elevon Hinge Moment Slope | lb |
| XD(R1BE+37) | Right Inboard Elevon Position | deg |
| RITDEI | Inboard Elevon Conversion Factor | deg/in |
| RITDEO | Outboard Elevon Conversion Factor | deg/in |
| RITDT | TVC Actuator Conversion Factor | deg/in |
| RITVP | TVC Actuator Conversion Factor (Pitch) | v/in |
| RITVY | TVC Actuator Conversion Factor (Yaw) | v/in |
| XD(ROBE) | Right Outboard Elevon Command | v |
| XD(ROBE+3) | Right Outboard Elevon Hinge Moment Intercept | in-lb |
| XD(ROBE+4) | Right Outboard Elevon Hinge Moment Slope | lb |
| XD(ROBE+37) | Right Outboard Elevon Position | deg |
| XD(RSB) | Rudder Command | v |
| XD(RSB+1) | Speedbrake Command | v |
| XD(RSB+3) | Left Rudder Panel Hinge Moment Intercept | in-lb |
| XD(RSB+4) | Left Rudder Panel Hinge Moment Slope | lb |
| XD(RSB+5) | Right Rudder Panel Hinge Moment Intercept | in-lb |
| XD(RSB+6) | Right Rudder Panel Hinge Moment Slope | lb |
| XD(RSB+7) | Left Rudder Panel Position | deg |
| XD(RSB+8) | Right Rudder Panel Position | deg |
| RTD | Radian to Degree Conversion | deg/rad |
| TFT | Time in Flight at Beginning of Run | sec |
| VBI | Inboard Elevon Bias | v |
| VBO | Outboard Elevon Bias | v |

## 6.99.7 Subroutine Listing

```
      SUBROUTINE CAD99 (D,DT,DD,L)
C
C   HYDRAULIC SYSTEM TRANSIENT ANALYSIS (HYTRAN) PROGRAM AND
C   SIX DEGREE OF FREEDOM ASCENT (SDFASC) PROGRAM
C   COMMUNICATION ROUTINE
      INTEGER ROBE,RIBE,RSB,BF
C
C   DECLARATIVE STATEMENTS ASSOCIATED WITH HYTRAN
C
      DOUBLE PRECISION DD
      DIMENSION D(1),DT(1),DD(1),L(1)
      COMMON/SUB/PARM(150,9),PM(1500),QM(1500),P(300),Q(300),C(300)
     1,Z(300),RHO(20),S2ORHO(20),VISC(20),BULK(20),TEMP(20),PVAP(20)
     2,ATPRES,T,DELT,TFINAL,PLTDEL,PI,TITLE(20),LEGN,ICON
     3,KTEMP(99),LSTART(150),NLPT(150),LTYPE(99),NC(99),INX,INZ
     4,INV,ISTEP,NLINE,NEL,IND,IENTR,MNLINE,MNEL,MNLEG,MNNODE,MNPLOT
     5,MNLPTS,MDS
      COMMON /COMPD/ XD(4500),LX(1500),LEX(99,4)
C
C   DECLARATIVE STATEMENTS ASSOCIATED WITH SDFASC
C
      COMMON /FARRAY/ FC(200)
      EQUIVALENCE
     . (DTFCS  ,FC(  2)),(IRESET ,FC(  6)),(RTD  ,FC(7))
     . ,(TFT   ,FC(1))
C
      IF(IENTR) 1000,2000,3000
 1000 CONTINUE
      IRESET = 1
      TPAST = T
      DTVEO=1.173/6.7246
      VBO=7.61*1.173/6.7246
      VBI=7.538*.683/3.92
      DBO=-7.61
      DBI=-7.538
      RITDT=8.499/4.413
      RITVY=4.59/4.413
      RITVP=4.59/5.449
      DTVEI=.683/3.92
      RITDEO=6.7246
      RITDEI=3.92
C***DETERMINE THE LOCATION OF THE ACTUATOR DT ARRAYS
C   IN THE XD ARRAY.
C   L(I)=COMPONENT NUMBER OF THE ACTUATOR
      LOBE=LEX(L(1),2)
      LIBE=LEX(L(2),2)
      ROBE=LEX(L(3),2)
      RIBE=LEX(L(4),2)
      RSB=LEX(L(5),2)
      BF=LEX(L(6),2)
```

6.99.7 (Continued)

```
      N1P=LEX(L(7),2)
      N1Y=LEX(L(8),2)
      N2P=LEX(L(9),2)
      N2Y=LEX(L(10),2)
      N3P=LEX(L(11),2)
      N3Y=LEX(L(12),2)
      CALL SDFASC
      TFO=TFT
      CALL FDUMP(FC,1,200)
      XD(LOBE)=FC(148)*DTVEO+VBO
      XD(LIBE)=FC(146)*DTVEI+VBI
      XD(ROBE)=FC(149)*DTVEO+VBO
      XD(RIBE)=FC(147)*DTVEI+VBI
      XD(N1P)=FC(166)*RITVP/RITDT
      XD(N2P)=FC(167)*RITVP/RITDT
      XD(N3P)=FC(168)*RITVP/RITDT
      XD(N1Y)=FC(171)*RITVY/RITDT
      XD(N2Y)=FC(172)*RITVY/RITDT
      XD(N3Y)=FC(173)*RITVY/RITDT
      DT(1)=FC(186)
      DT(2)=FC(185)
      DT(3)=RITDT*FC(191)
      DT(4)=RITDT*FC(192)
      DT(5)=RITDT*FC(193)
      DT(6)=RITDT*FC(196)
      DT(7)=RITDT*FC(197)
      DT(8)=RITDT*FC(198)
      DT(9)=FC(148)
      DT(10)=FC(146)
      DT(11)=FC(149)
      DT(12)=FC(147)
      DT(13)=FC(166)
      DT(14)=FC(167)
      DT(15)=FC(168)
      DT(16)=FC(171)
      DT(17)=FC(172)
      DT(18)=FC(173)
      DT(19)=FC(71)*RTD
      DT(20)=FC(72)*RTD
      DT(21)=FC(79)*RTD
      DT(22)=FC(80)*RTD
      DT(23)=FC(81)*RTD
      DT(24)=FC(181)
      DT(25)=FC(132)
      RETURN
 2000 CONTINUE
      RETURN
 3000 CONTINUE
      IRESET = 0
C***INTERFACE PARAMETERS TO BE TRANSFERRED TO SDFASC
```

```
C***TVC POSITIONS
      FC(166)=XD(N1P+37)*RITDT
      FC(167)=XD(N2P+37)*RITDT
      FC(168)=XD(N3P+37)*RITDT
      FC(171)=XD(N1Y+37)*RITDT
      FC(172)=XD(N2Y+37)*RITDT
      FC(173)=XD(N3Y+37)*RITDT
C***ELEVON POSITIONS
      FC(148)=XD(LOBE+37)*RITDEO+DBO
      FC(146)=XD(LIBE+37)*RITDEI+DBI
      FC(149)=XD(ROBE+37)*RITDEO+DBO
      FC(147)=XD(RIBE+37)*RITDEI+DBI
      IF((T-TPAST).LT.DTFCS)GO TO 3200
      TPAST=T
      TFT=T+TFO
      CALL SDFASC
C***INTERFACE PARAMETERS TO BE TRANSFERRED TO HYTRAN
C***TVC COMMANDS
      XD(N1P)=FC(191)*RITVP
      XD(N2P)=FC(192)*RITVP
      XD(N3P)=FC(193)*RITVP
      XD(N1Y)=FC(196)*RITVY
      XD(N2Y)=FC(197)*RITVY
      XD(N3Y)=FC(198)*RITVY
C***ELEVON COMMANDS
      XD(LOBE)=FC(186)*DTVEO+VBO
      XD(LIBE)=FC(185)*DTVEI+VBI
      XD(ROBE)=FC(186)*DTVEO+VBO
      XD(RIBE)=FC(185)*DTVEI+VBI
C***RSB & BF CMDS.
      XD(RSB)=0.0
      XD(RSB+1)=0.0
      XD(BF)=0.0
C***ELEV. OUTBOARD CMDS.
      DT(1)=FC(186)
C***ELEV. INBOARD CMDS.
      DT(2)=FC(185)
C***TVC PITCH CMDS.
      DT(3)=RITDT*FC(191)
      DT(4)=RITDT*FC(192)
      DT(5)=RITDT*FC(193)
C***TVC YAW CMDS.
      DT(6)=RITDT*FC(196)
      DT(7)=RITDT*FC(197)
      DT(8)=RITDT*FC(198)
      CALL FDUMP(FC,1,200)
 3200 CONTINUE
C***ELEV. POS. LO,LI,RO,RI
      DT(9)=FC(148)
```

6.99.7 (Continued)

```
      DT(10)=FC(146)
      DT(11)=FC(149)
      DT(12)=FC(147)
C***TVC PITCH POS.
      DT(13)=FC(166)
      DT(14)=FC(167)
      DT(15)=FC(168)
C***TVC YAW POS.
      DT(16)=FC(171)
      DT(17)=FC(172)
      DT(18)=FC(173)
C***ALPHA
      DT(19)=FC(71)*RTD
C***BETA
      DT(20)=FC(72)*RTD
C***BODY RATES
      DT(21)=FC(79)*RTD
      DT(22)=FC(80)*RTD
      DT(23)=FC(81)*RTD
C***BODY ACCELERATIONS
      DT(24)=FC(181)
      DT(25)=FC(182)
C
C
      RETURN
      END
```

6.99.7 (Continued)

```
      SUBROUTINE CAD99 (D,DT,DD,L)
C
C   HYDRAULIC SYSTEM TRANSIENT ANALYSIS (HYTRAN) PROGRAM AND
C   SIX DEGREE OF FREEDOM DESCENT (SDFDES) PROGRAM
C   COMMUNICATION ROUTINE
      INTEGER ROBE,RIBE,RSB,BF
C
C   DECLARATIVE STATEMENTS ASSOCIATED WITH HYTRAN
C
      DOUBLE PRECISION DD
      DIMENSION D(1),DT(26),DD(1),L(6)
      COMMON/SUB/PARM(150,9),PM(1500),QM(1500),P(300),Q(300),C(300)
     1,Z(300),RHO(20),S2ORHO(20),VISC(20),BULK(20),TEMP(20),PVAP(20)
     2,ATPRES,T,DELT,TFINAL,PLTDEL,PI,TITLE(20),LEGN,ICON
     3,KTEMP(99),LSTART(150),NLPT(150),LTYPE(99),NC(99),INX,INZ
     4,INV,ISTEP,NLINE,NEL,IND,IENTR,MNLINE,MNEL,MNLEG,MNNODE,MNPLOT
     5,MNLPTS,MDS
      COMMON /COMPD/ XD(4500),LX(1500),LEX(99,4)
C
C   DECLARATIVE STATEMENTS ASSOCIATED WITH SDFDES
C
      COMMON /FARRAY/ FC(200)
      EQUIVALENCE
     . (DTFCS  ,FC(  2)),(IRESET ,FC(  6)),(RTD  ,FC(7))
     . ,(TFT  ,FC(1))
C
      IF(IENTR) 1000,2000,3000
 1000 CONTINUE
      IRESET = 1
      TPAST = T
      DTVEO=1.173/6.7246
      DTVEI=.683/3.92
      RITDEO=6.7246
      VBO=7.61*1.173/6.7246
      VBI=7.538*.683/3.92
      DBO=-7.61
      DBI=-7.538
      RITDEI=3.92
      DTVR=.184
      DTVSB=.0937
      DTVBF=.18
C***DETERMINE THE LOCATION OF THE ACTUATOR DT ARRAYS
C   IN THE XD ARRAY.
C   L(I)=COMPONENT NUMBER OF THE ACTUATOR
      LOBE=LEX(L(1),2)
      LIBE=LEX(L(2),2)
      ROBE=LEX(L(3),2)
      RIBE=LEX(L(4),2)
      RSB=LEX(L(5),2)
      BF=LEX(L(6),2)
```

6.99.7 (Continued)

```
      CALL SDFDES
      TFO=TFT
      CALL FDUMP(FC,1,200)
C***INITIALIZE COMMANDS
      XD(LOBE)=FC(172)*DTVEO+VBO
      XD(LIBE)=FC(172)*DTVEI+VBI
      XD(ROBE)=FC(171)*DTVEO+VBO
      XD(RIBE)=FC(171)*DTVEI+VBI
      XD(RSB)=FC(174)*DTVR
      XD(RSB+1)=FC(173)*DTVSB
      XD(BF)=0.0
      XD(BF+1)=FC(144)
C***INITIALIZE H.M.
      XD(LOBE+3)=FC(132)
      XD(LIBE+3)=FC(130)
      XD(ROBE+3)=FC(133)
      XD(RIBE+3)=FC(131)
      XD(LOBE+4)=0.0
      XD(LIBE+4)=0.0
      XD(ROBE+4)=0.0
      XD(RIBE+4)=0.0
      XD(RSB+3)=FC(126)
      XD(RSB+5)=FC(127)
      XD(RSB+4)=0.0
      XD(RSB+6)=0.0
      XD(BF+3)=FC(134)
      XD(BF+4)=0.0
      DT(1)=FC(172)
      DT(2)=FC(171)
      DT(3)=FC(148)
      DT(4)=FC(146)
      DT(5)=FC(149)
      DT(6)=FC(147)
      DT(7)=FC(174)
      DT(8)=FC(173)
      DT(11)=FC(169)
      DT(13)=FC(71)*RTD
      DT(14)=FC(72)*RTD
      DT(15)=FC(79)*RTD
      DT(16)=FC(80)*RTD
      DT(17)=FC(81)*RTD
      DT(18)=FC(193)
      DT(19)=FC(194)
      DT(9)=FC(136)
      DT(10)=FC(137)
      DT(12)=FC(143)
      DT(18)=FC(145)
      DT(19)=FC(144)
      DT(20)=FC(126)
      DT(21)=FC(127)
```

6.99.7 (Continued)

```
      DT(22)=FC(134)
      DT(23)=FC(132)
      DT(24)=FC(130)
      DT(25)=FC(133)
      DT(26)=FC(131)
      RETURN
 2000 CONTINUE
      RETURN
 3000 CONTINUE
      IRESET = 0
C***INTERFACE PARAMETERS TO BE TRANSFERRED TO SDFDES
C***ELEVON POSITIONS
      FC(148)=XD(LOBE+37)*RITDEO+DBO
      FC(146)=XD(LIBE+37)*RITDEI+DBI
      FC(149)=XD(ROBE+37)*RITDEO+DBO
      FC(147)=XD(RIBE+37)*RITDEI+DBI
      FC(195)=.25*(FC(146)+FC(147)+FC(148)+FC(149))
C***RUDDER/SB POSITIONS
      FC(143)=(XD(RSB+7)+XD(RSB+8))/2.
      FC(145)=-(XD(RSB+8)-XD(RSB+7))
C***BODYFLAP POSITION
      FC(144)=XD(BF+1)
      IF((T-TPAST).LT.DTFCS)GO TO 3200
      TPAST=T
      TFT=T+TFO
      CALL SDFDES
C***INTERFACE PARAMETERS TO BE TRANSFERRED TO HYTRAN

C***ELEVON COMMANDS
      XD(LOBE)=FC(172)*DTVEO+VBO
      XD(LIBE)=FC(172)*DTVEI+VBI
      XD(ROBE)=FC(171)*DTVEO+VBO
      XD(RIBE)=FC(171)*DTVEI+VBI
C***ELEVON HINGE MOMENT INTERCEPTS
      XD(LOBE+3)=FC(132)-FC(105)*RITDEO*XD(LOBE+37)
      XD(LIBE+3)=FC(130)-FC(103)*RITDEI*XD(LIBE+37)
      XD(ROBE+3)=FC(133)-FC(106)*RITDEO*XD(ROBE+37)
      XD(RIBE+3)=FC(131)-FC(104)*RITDEI*XD(RIBE+37)
C***ELEVON H.M. SLOPES
      XD(LOBE+4)=FC(105)*RITDEO
      XD(LIBE+4)=FC(103)*RITDEI
      XD(ROBE+4)=FC(106)*RITDEO
      XD(RIBE+4)=FC(104)*RITDEI
C***RUDDER COMMAND
      XD(RSB)=FC(174)*DTVR
C***SB COMMAND
      XD(RSB+1)=FC(173)*DTVSB
C***RUDDER/SB H.M. INTERCEPTS (LEFT),(RIGHT)
      XD(RSB+3)=FC(126)-XD(RSB+7)*FC(107)
      XD(RSB+5)=FC(127)-XD(RSB+8)*FC(108)
```

6.99.7 (Continued)

```
C***RUDDER/SB H.M. SLOPES (LEFT),(RIGHT)
      XD(RSB+4)=FC(107)
      XD(RSB+6)=FC(108)
C***BODYFLAP RATE COMMAND
      XD(BF)=FC(169)
C***BODYFLAP H.M. INTERCEPT
      XD(BF+3)=FC(134)-XD(BF+1)*FC(109)
C***BODYFLAP H.M. SLOPE
      XD(BF+4)=FC(109)
C***ELEV. CMD. L,R
      DT(1)=FC(172)
      DT(2)=FC(171)
C***RUD. CMD.
      DT(7)=FC(174)
C***SB CMD.
      DT(8)=FC(173)
C***BF RATE CMD.
      DT(11)=FC(169)
      FC(111)=XD(LOBE+3)
      FC(112)=XD(LIBE+3)
      FC(113)=XD(ROBE+3)
      FC(114)=XD(RIBE+3)
      FC(115)=XD(RSB+3)
      FC(116)=XD(RSB+5)
      FC(117)=XD(BF+3)
      FC(118)=XD(LOBE+68)
      FC(119)=XD(LIBE+68)
      CALL FDUMP(FC,1,200)
 3200 CONTINUE
C***ELEV. POS. LO,LI,RO,RI
      DT(3)=FC(148)
      DT(4)=FC(146)
      DT(5)=FC(149)
      DT(6)=FC(147)
      DT(13)=FC(71)*RTD
      DT(14)=FC(72)*RTD
      DT(15)=FC(79)*RTD
      DT(16)=FC(80)*RTD
      DT(17)=FC(81)*RTD
      DT(9)=FC(136)
      DT(10)=FC(137)
      DT(12)=FC(143)
      DT(18)=FC(145)
      DT(19)=FC(144)
      DT(20)=FC(126)
      DT(21)=FC(127)
      DT(22)=FC(134)
      DT(23)=FC(132)
      DT(24)=FC(130)
      DT(25)=FC(133)
```

6.99.7 (Continued)

```
      DT(26)=FC(131)
C
C
      RETURN
      END
```

## 6.101 SUBROUTINE ACT101

This subroutine models a simple servo actuator with a mechanical input to the servo valve,which operates open loop, without feedback.

A time history of valve position is inputed and a first order or straight line interpolation is used between the input points.

The valve is assumed to be a linear square port configuration, with zero lap. The width of each port slot is inputed independently, to allow the valve areas to be matched to the actuator piston areas.

The initial actuator position is inputed, together with the external loads at the fully retracted and extended stroke positions. The load stroke curve is assumed to be linear between these positions.

The steady state balancing system uses the load at the initial position to determine the pressure drop across the piston. The effects of atmospheric pressure is incorporated into the load.

### 6.101.1 Math Model

The math model can be divided into two sections, the flow calculations and the integral calculations.

The flow calculations are based on a combination of the line equations valve orifice equations, and the volumeric impedance.

The sign convention used is, flows into the actuator cavities (Q1), and (Q2) are +ve, which is the same as the line convention. The flows are calculated using the general formulae

```
FLOW = (SQRT(FN2**2+4*FN1*ABS (C -PT))-FN2)/2.0

where FN1 = (XV*VK)**2
      FN2 = FN1*Z
       XV = Valve Position
       VK = W*SQRT(2/RHO(IT))*.65*Slot width
        Z = Line Characteristic Impedance + Volumetric Impedance
        C = Line Characteristic Pressure
       PT = Chamber volumetric pressure
       IT = Fluid Temperature/Pressure Indication
```

where the volumetric impedance ZV1 = DT(KBULK)/(D(VOL1)+D(AREA1)*DT(PX)) and the chamber volumetric pressure.

PT = DT(PP1)-DT(VEL)*D(AREA1)*ZV1

The actuator velocity is calculated using the equivalent network given in Figure 6.101-2.

The two flows DT(Q1) and DT(Q2) are calculated using the value orifice equations and predicted actuator pressures DT(PP1P) and DT(PP2P).

The network is solved for piston velocity DT(VEL). The damping term includes an inertia term which accounts for the load required to change the velocity during the time step DELT.

A load due the old or previous velocity is included with the actuator load term.

The network solution takes into account the volumetric effect of the two actuator cavities. The assumption is that a portion of the flow is lost too or obtained from these volumes due to changes in pressure within the cavities.

The basis network equations are: -

VELO=DT(VEL) = (OLD VELOCITY)

DT(VEL)*D(AREA1) = DT(Q1) + (DT(PP1)-P1)*GV1 (1)

DT(VEL)*D(AREA2) = -DT(Q2)-(DT(PP2)-P2)*GV2 (2)

DT(VEL)*D(KDAMP) = P1*D(AREA1)-P2*D(AREA2) - DP+VELO*DT(INERT) (3)

where

DP = External Load

= DT(LOADZ)+DT(PX)*DT(LOADS)

+ DT(LOADEX)

The atmospheric load is kept separate to allow time varying computations of values for DT(LOADZ) AND DT(LOADS) to be added later.

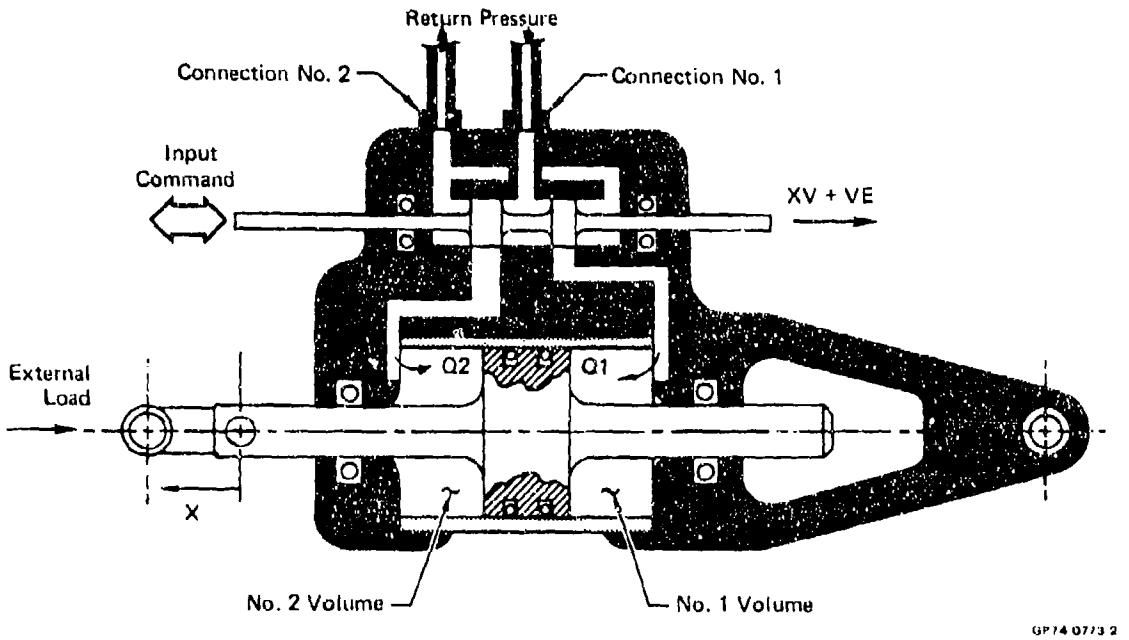



**FIGURE 6.101-1**
**TYPE NO. 101 VALVE CONTROLLED ACTUATOR**

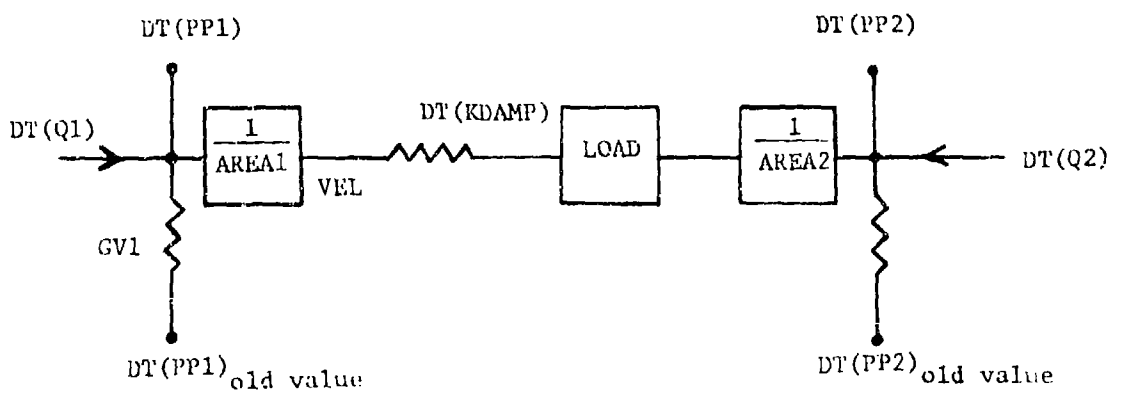



FIGURE 6.101-2

SCHEMATIC DIAGRAM FOR TYPE NO. 101
VALVE CONTROLLED ACTUATOR

Equations (1), (2) and (3) are combined to eliminate P1 and P2, giving the solution in the listing.

6.101.2 Assumptions

It was intended that this particular subroutine should be kept as simple as possible, hence a number of relatively important actuator characteristics have not been included in the model.

Those ommited are cross piston leakage, static and dynamic seal friction, volumetric expansion of the actuator cavity, external load spring mass characteristics, valve leakage and neutral gain characteristics, and cavitation effects, within the actuator cavity.

This model fills the need for a simple actuator model to act as a system load where the detailed dynamics of the actuator are of no particular concern.

6.101.3 Computation

Section 1000

The external load input data is modified by calculating the load displacement curve, and adding the force due to atmospheric pressure.

Finally, the valve impedances, and the location of the node, are calculated for the steady state valve position, for use in the steady state section.

The cavity pressures and piston velocity were initialized in the steady state section.

Section 1500

This section is called from LEGCAL via COMPE for each connection number for each iteration. (INEL, 7) = 5 requires calls to be made for each iteration because the overboard flow and pressure drop across the piston head vary with the flow into the actuator and the pressure in the piston cavity.

One of the cavities is required to be a system node. Which cavity it is depends on the valve position at time zero.

If NODE = 1 it is in #1 cavity, if NODE = 2 it is in #2 cavity.

The steady state section is complicated by the need to determine if the actuator is at its stroke limits, and if the flow guess is taking it toward or away from the limit.

When it is at its limits, and is being driven into the limit, a high impedance is added into the leg (ZQ = 40000.), and the overboard flow is set to zero. (Overboard flow is a displacement flow due to unequal areas).

The steady state calculation set up requires that connection #1 must be the last or only element in the upstream leg, and connection #2 is the first element in the downstream leg.

The upstream leg flow is used to calculate the overboard flow and piston velocity. If the valve is closed the overboard flow is set to zero.

For the upstream leg the valve impedance DT(PP1P) is added into PQLEG(INEL,8). For the downstream leg, the valve impedance PT(PP2P) is added into PQLEG(INEL,8), ZQ is added into PQLEG(INEL,6) and the constant pressure drop DELTP across the piston is subtracted from PQLEG(INEL,5) or added if it is a pressure rise.

Section 3000

This section calculates the transient response of the actuator using an integration step size of DELT.

INTERP is called to obtain an interpolated value of valve position XV. With this value of XV, the flows into the actuator chambers are calculated.

If XV is zero, the flows are set to zero. For XV >0, Q1 is the flow from connection #1 to chamber #1, and Q2 the flow from chamber '2 to connection #2. For XV<0 the flows are reversed.

A common formulae is used to calculate FLOW with a computer go to, to branch back, with ICALC acting as the branch indicator.

With Q1 and Q2 calculated the next section calculates the value of DT(PP1) and DT(PP2) and the velocity of the piston.

The piston velocity is calculated by summing the forces acting on it and dividing by the mass plus the damping coefficients.

A check is then made to see if the piston is at a stroke limit. If it is, the velocity is set to zero if it is in the limit direction.

The pressures are calculated by taking the sum of the flows into the cavity, including the piston velocity and multiplying by DT(KBULK) cavity volume.

6.101.4 Approximations - To be added later.

6.101.5 Limitations

The straight line flow characteristics of the valve and the straight line load characteristics limit the applicability of this subroutine to a rather rudimentary type of actuator. It can however be used to generate waterhammer transients due to rapidly closing valve motion, with the advantage of a good simulation of the actuator cavity pressure response.

6.101.6 Variable Names

| VARIABLE | DESCRIPTION | DIMENSIONS |
|---|---|---|
| D(AREA1) | #1 piston area | IN**2 |
| D(AREA2) | #2 piston area | IN**2 |
| C(L1) | Connection #1 line characteristic | PSI |
| C(L2) | Connection #2 line characteristic | PSI |
| D(DAMP) | Piston damping factor | LBS/SEC/IN |
| DT(DELTP) | Pressure drop across piston | PSI |
| FLOW | Temporary value of line flow | CIS |
| FN1 | Temporary variable | -- |
| FN2 | Temporary variable | -- |
| DT(FORCE) | Load at maximum stroke | LBS |
| ICALC | Calculation counter | -- |
| GV1 | Conductance of Vol 1 | CIS/PSI |
| GV2 | Conductance of Vol 2 | CIS/PSI |
| DT(INERT) | Load Inertia | LB SEC/IN |
| IN2 | Interpolation indicator returned from INTERP | -- |
| D(INPOS) | Initial actuator position | IN |
| DT(KBULK) | Oil bulk modulus times AT | PSI*SEC |

6.101.6 Variable Names (cont'd)

| VARIABLE | DESCRIPTION | DIMENSIONS |
|---|---|---|
| DT(KDAMP) | Damping factor | LB/SEC/IN |
| L(IY) | Address of first value in valve position table | -- |
| DT(LOADS) | Load slope | LB/IN |
| DT(LOADZ) | Load at zero stroke | LBS |
| DT(LOADEX) | External load | LBS |
| L1 | Dummy variable | -- |
| L2 | Dummy variable | -- |
| N | Integer counter | -- |
| D(MASS) | Load mass | LB $SEC^2$/IN |
| D(MINST) | Minimum actuator stroke | IN |
| D(MAXST) | Maximum actuator stroke | IN |
| D(MINL) | Load - actuator fully retracted | LBS |
| D(MAXL) | Load - actuator fully extended | LBS |
| DT(NCAV) | Flow for cavitation volume | CIS |
| L(NODE) | Node location indicator | -- |
| L(NTAB) | Number of points in time table | -- |
| DT(PFORCE) | External actuator load | LBS |
| DT(PP1) | Pressure in #1 cavity) | PSI |
| DT(PP2) | ssure in #2 cavity | PSI |
| DT(PP1P) | Predicted pressure in #1 cavity | PSI |
| DT(PP2P) | Predicted pressure in #2 cavity | PSI |
| DT(PX) | Predicted piston position | IN |
| QA | Leg flow in steady state section | CIS |
| DT(Q1) | Flow into cavity #1 | CIS |
| DT(Q2) | Flow out of cavity #2 | CIS |

6.101.6 Variable Names (cont'd)

| VARIABLE | DESCRIPTION | DIMENSIONS |
|---|---|---|
| QS | Flow sign | -- |
| D(SLOTW1) | Sloth width Vol #1 to Con #1 | IN |
| D(SLOTW2) | Sloth width Vol #1 to Con #2 | IN |
| D(SLOTW3) | Sloth width Vol #2 to Con #1 | IN |
| D(SLOTHW4) | Sloth width Vol #2 to Con #2 | IN |
| DT(VEL) | Actuator Velocity | IN/SEC |
| VELO | Previous actuator velocity | IN/SEC |
| VK | Temporary value of SLOTW | IN |
| D(VOL1) | Minimum volume of cavity #1 | IN**3 |
| D(VOL2) | Maximum volume of cavity #2 | IN**3 |
| DT(X) | Piston position | IN |
| XV | Valve position | IN |
| ZQ | Cross piston leakage impedance | PSI/CIS |
| ZT | Temporary value of line impedance | -- |
| ZV1 | CON #1 valve impedance | PSI/CIS**2 |
| ZV2 | CON #2 valve impedance | PSI/CIS**2 |

## 6.101.7 Subroutine Listing

```
      SUBROUTINE ACT101 (D,DT,DD,L)
C ***REVISED JUNE  1976 ****
      DOUBLE PRECISION DD
      DIMENSION D(32),DT(18),DD(1),L(12)
      COMMON NTELPL,NTOLPL,IPT,IPOINT,NPTS,INEL,KNEL,NTOPL,NLPLT(61,3),
     1 PQLEG(90,12),LCS(90,10),ILEG(1400),PN(90),QN(90)
      COMMON/SUB/PARM(150,9),PM(1500),QM(1500),P(300),Q(300),C(300)
     1,Z(300),RHO(20),S2ORHO(20),VISC(20),BULK(20),TEMP(20),PVAP(20)
     2,ATPRES,T,DELT,TFINAL,PLTDEL,PI,TITLE(20),LEGN,ICON
     3,KTEMP(99),LSTART(150),NLPT(150),LTYPE(99),NC(99),INX,INZ
     4,INV,ISTEP,NLINE,NEL,IND,IENTR,MNLINE,MNEL,MNLEG,MNNODE,MNPLOT
     5,MNLPTS,MDS
      INTEGER AREA1,AREA2,VOL1,VOL2,SLOTW1,SLOTW2
     1,SLOTW3,SLOTW4,DAMP,X,PFORCE,FORCE,PP1P,PP2P
     2,PP1,PP2,Q1,Q2,VEL,PX
C      D ARRAY VARIABLES
      DATA AREA1/1/,AREA2/2/,VOL1/3/,VOL2/4/,MINST/5/,MAXST/6/,
     1 DAMP/7/,MASS/8/,SLOTW1/9/,SLOTW2/10/,SLOTW3/11/,SLOTW4/12/,
     2 MINL/13/,MAXL/14/,INPOS/15/
C      L ARRAY VARIABLES
      DATA NTAB/5/,IY/6/,NODE/7/
C      DT ARRAY VARIABLES
      DATA X/1/,VEL/2/,LOADZ/3/,LOADS/4/,PP1/5/,PP2/6/,
     1 Q1/7/,Q2/8/,KBULK/9/,NCAV/10/,PP1P/11/,PP2P/12/,
     2 PFORCE/13/,FORCE/14/,KDAMP/15/,PX/16/,INERT/17/,LOADEX/18/
C
      IF(IENTR) 1000,2000,3000
C *** 1000 SECTION
 1000 CONTINUE
      IF(INEL.NE.0)GO TO 1500
C      ACTUATOR PARAMETER INPUT
      D(SLOTW1)=D(SLOTW1)*0.65*S2ORHO(KTEMP(IND))
      D(SLOTW2)=D(SLOTW2)*0.65*S2ORHO(KTEMP(IND))
      D(SLOTW3)=D(SLOTW3)*0.65*S2ORHO(KTEMP(IND))
      D(SLOTW4)=D(SLOTW4)*0.65*S2ORHO(KTEMP(IND))
      DT(LOADS)=(D(MAXL)-D(MINL))/(D(MAXST)-D(MINST))
      DT(LOADZ)=D(MAXL)-DT(LOADS)*D(MAXST)
      DT(FORCE)=DT(LOADZ)+DT(LOADS)*D(INPOS)
      DT(LOADEX)=ATPRES*(D(AREA1)-D(AREA2))
      DT(INERT)=D(MASS)/DELT
      DT(KDAMP)=D(DAMP)+DT(INERT)
      DT(X)=D(INPOS)
      DT(KBULK)=BULK(KTEMP(IND))*DELT
      DELTO2=DELT/2.0
      L(NTAB)=L(12)
      L(IY)=(L(NTAB)+7)/8
      L(IY)=17+L(IY)*8
      L(NODE)=1
      XV=D(L(IY))
      DT(NCAV)=1.0
```

6.101.7 (Continued)

```
      DT(PFORCE)=DT(FORCE)/D(AREA2)
      IF(XV) 60,70,80
   60 DT(PP1P)=1/(D(SLOTW2)*XV)**2
      DT(PP2P)=1/(D(SLOTW3)*XV)**2
      L(NODE)=2
      DT(PFORCE)=DT(FORCE)/D(AREA1)
      GO TO 90
   70 DT(PP1P)=400000.
      DT(PP2P)=400000.
      DT(NCAV)=0.0
      GO TO 90
   80 DT(PP1P)=1/(D(SLOTW1)*XV)**2
      DT(PP2P)=1/(D(SLOTW4)*XV)**2
   90 RETURN
C
C *** 1500 SECTION
C     THE STEADY STATE SECTION
 1500 CONTINUE
      QA=PQLEG(INEL,1)
      QS=PQLEG(INEL,2)
      LCS(INEL,7)=5
      IF(KNEL.EQ.2) GO TO 1750
      IF(KNEL.NE.1) GO TO 1900
      IF(L(NODE).EQ.2) QS=-QS
      ZQ=0.0
      N=LCS(INEL,3)
      IF(DT(X).GT.0.0) GO TO 1600
      IF(QS.GT.0.0) GO TO 1650
 1550 QN(N)=0.0
      ZQ=40000.
      IF(L(NODE).EQ.1) GO TO 1700
      QS=-QS
      GO TO 1850
 1600 IF(DT(X).LT.D(MAXST)) GO TO 1550
      IF(QS.GT.0.0) GO TO 1550
 1650 IF(L(NODE).EQ.2) GO TO 1800
      QN(N)=-DT(NCAV)*QA*QS*(D(AREA1)-D(AREA2))/D(AREA1)
 1700 DT(PFORCE)=(DT(FORCE)-PN(N)*(D(AREA1)-D(AREA2)))/D(AREA2)
      DT(PP1)=PN(N)
      DT(PP2)=PN(N)-DT(PFORCE)-QA*QS*ZQ
      DT(VEL)=QA*QS/D(AREA1)
      PQLEG(INEL,8)=PQLEG(INEL,8)+DT(PP1P)
      RETURN
 1750 PQLEG(INEL,8)=PQLEG(INEL,8)+DT(PP2P)
      PQLEG(INEL,6)=PQLEG(INEL,6)+ZQ
      PQLEG(INEL,5)=PQLEG(INEL,5)-DT(PFORCE)
      PQLEG(INEL,11)=PQLEG(INEL,11)-DT(PFORCE)-QA*QS*(ZQ+QA*DT(PP2P))
      RETURN
 1800 QS=-QS
      QN(N)=-DT(NCAV)*QA*QS*(D(AREA1)-D(AREA2))/D(AREA2)
```

6.101.7 (Continued)

```
 1350 DT(PFORCE)=-(PN(N)*(D(AREA2)-D(AREA1))+DT(FORCE))/D(AREA1)
      DT(PP2)=PN(N)
      DT(PP1)=PN(N)-DT(PFORCE)-QA*QS*ZQ
      DT(VEL)=-QA*QS/D(AREA2)
      RETURN
 1900 WRITE(6,1950) IND,KNEL,INEL
 1950 FORMAT(5X,7HCOMP NO,I3,20H, HAS INVALID CON NO  ,I3,
     1 11H, IN LEG NO ,I4)
C *** 2000 SECTION
 2000 CONTINUE
      DT(NCAV)=0.0
      DT(PP1P)=DT(PP1)
      DT(PP2P)=DT(PP2)
      DT(PX)=DT(X)-DT(VEL)*DELT
      DT(FORCE)=0.0
      XV=D(L(IY))
      DT(Q2)=Q(L(2))
      DT(Q1)=Q(L(1))
      IF(XV.GT.0.0) GO TO 2020
      DT(Q2)=Q(L(1))
      DT(Q1)=Q(L(2))
 2020 CONTINUE
      RETURN
C
C *** 3000 SECTION
 3000 CONTINUE
C
      L1=L(1)
      L2=L(2)
      GV1=(D(VOL1)+DT(PX)*D(AREA1))/(DT(KBULK)*DELT)
      ZV1=1.0/GV1
      DT(PP1P)=DT(PP1)-DT(VEL)*D(AREA1)*ZV1
      GV2=(D(VOL2)-DT(PX)*D(AREA2))/(DT(KBULK)*DELT)
      ZV2=1.0/GV2
      DT(PP2P)=DT(PP2)+DT(VEL)*D(AREA2)*ZV2
C
      CALL INTERP (T,D(17),D(L(IY)),10,L(NTAB),XV,IN2)
C
C     CALCULATE LINE FLOWS AND PRESSURES
      IF (XV) 140,170,130
C     XV LESS THAN 0
  140 VK=D(SLOTW3)
      ZT=Z(L1)+ZV2
      DP=C(L1)-DT(PP2P)
      ICALC=1
      GO TO 210
  150 DT(Q2)=SIGN(FLOW,DP)
      Q(L1)=DT(Q2)
      VK=D(SLOTW2)
      ZT=Z(L2)+ZV1
```

6.101.7 (Continued)

```
      DP=C(L2)-DT(PP1P)
      ICALC=2
      GO TO 210
  160 DT(Q1)=SIGN(FLOW,DP)
      Q(L2)=DT(Q1)
      GO TO 220
C
C     XV = 0
  170 DT(Q1)=0.
      DT(Q2)=0.
      Q(L2)=0.0
      Q(L1)=0.0
      ICALC=5
      GO TO 220
C
C     XV GREATER THAN 0
  180 VK=D(SLOTW1)
      ZT=Z(L1)+ZV1
      DP=C(L1)-DT(PP1P)
      ICALC=3
      GO TO 210
  190 DT(Q1)=SIGN(FLOW,DP)
      Q(L1)=DT(Q1)
      VK=D(SLOTW4)
      ZT=Z(L2)+ZV2
      DP=C(L2)-DT(PP2P)
      ICALC=4
      GO TO 210
  200 DT(Q2)=SIGN(FLOW,DP)
      Q(L2)=DT(Q2)
      GO TO 220
C
C     CALCULATE ABSOLUTE VALUE OF FLOWS
  210 FN1=XV*XV*VK*VK
      FN2=FN1*ZT
      FLOW=(SQRT(FN2**2+4.0*FN1*ABS(DP))-FN2)/2.0
      GO TO (150,160,190,200),ICALC
C
  220 CONTINUE
      P(L1)=C(L1)-Q(L1)*Z(L1)
      P(L2)=C(L2)-Q(L2)*Z(L2)
      VAP=PVAP(KTEMP(IND))
      IF(P(L2).GT.VAP.AND.DT(NCAV).LE.0.0) GO TO 270
      FLOW=0.0
      GO TO (230,230,240,240,250),ICALC
  230 FN2=FN1*ZV1
      DP=VAP-DT(PP1P)
      FLOW=(SQRT(FN2**2+4.0*FN1*ABS(DP))-FN2)/2.0
      FLOW=SIGN(FLOW,DP)
      DT(Q1)=FLOW
```

6.101.7 (Continued)

```
      GO TO 250
  240 FN2=FN1*ZV2
      DP=VAP-DT(PP2P)
      FLOW=(SQRT(FN2**2+4.0*FN1*ABS(DP))-FN2)/2.0
      FLOW=SIGN(FLOW,DP)
      DT(Q2)=FLOW
  250 Q(L2)=(C(L2)-VAP)/Z(L2)
      P(L2)=VAP
      DT(NCAV)=DT(NCAV)+FLOW-Q(L2)
  270 CONTINUE
C
C     CALCULATE ACTUATOR VELOCITY
      G1=GV1/D(AREA1)
      G2=GV2/D(AREA2)
      ZN=DT(KDAMP)+D(AREA1)/G1+D(AREA2)/G2
      DP=DT(LOADEX)+DT(LOADZ)+DT(PX)*DT(LOADS)
      DELTP=DT(Q1)/G1+DT(PP1)*D(AREA1)-DT(Q2)/G2-DT(PP2)*D(AREA2)
      VELO=DT(VEL)
      DT(VEL)=(DELTP-DP+VELO*DT(INERT))/ZN
      DT(PX)=DT(X)
      DT(X)=DT(X)+DELTO2*(DT(VEL)+VELO)
      CALL XLIMIT(DT(X),DT(VEL),DP,D(MINST),D(MAXST))
      DT(PX)=DT(X)*2.0-DT(PX)
      IF(DP.NE.0.0) DT(PX)=DT(X)
      IF(DP.EQ.0.0) VELO=DT(VEL)
      DT(PP1)=DT(PP1)+(DT(Q1)-VELO*D(AREA1))/GV1
      DT(PP2)=DT(PP2)+(DT(Q2)+VELO*D(AREA2))/GV2
      RETURN
      END
```

## 6.102 SUBROUTINE ACT102

ACT102 simulates a basic utility actuator. The current subroutine allows the input of piston rod loads at zero and maximum stroke. Straight line interpolation is used between these two loads.

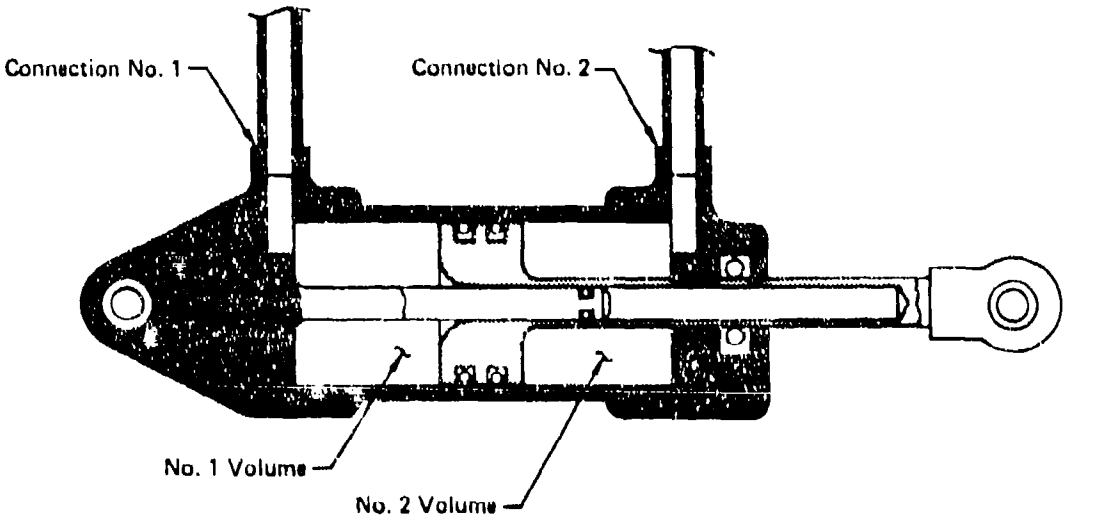



**FIGURE 6.102-1**
**TYPE NO. 102 UTILITY ACTUATOR**

6.102.1 Math Model

The ACT102 subroutine simulates a simple utility actuator. Figure 6.102.2 shows the various forces acting on the actuator.

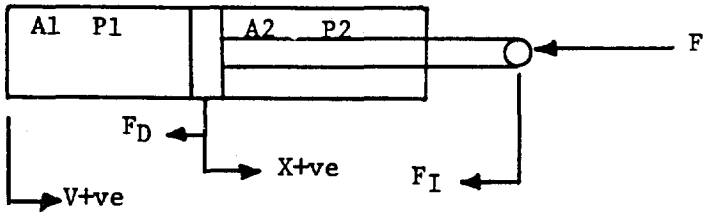



Figure 6.102.2

Summing the forces on the actuator yield

$$F = P1*A1-P2*A2-F_D-F_I \qquad 6.102.1$$

where F is the load on the actuator, lb,
P1 is the head side pressure, psi,
A1 is the head side area, $in^2$,
P2 is the rod side pressure, psi,
A2 is the rod side area, $in^2$,
$F_D$ is the damping force, lb,
$F_I$ is the inertial force, lb.

The damping force, $F_D$, is

$$F_D = V*DAMP$$

where, V is the velocity of the piston, in/sec
DAMP is the piston damping coefficient, lb sec/in.

The inertia force, $F_I$, is

$$F_I = (V-Vo)*M/T$$

where Vo is the previously calculated velocity, in/sec,
M is the mass of the piston and load, lb-in/$sec^2$
T is the calculation time step.

The unknown quantities V, P1 and P2 must be determined. The value of P1 and P2 may be determined from figure 6.102-3.

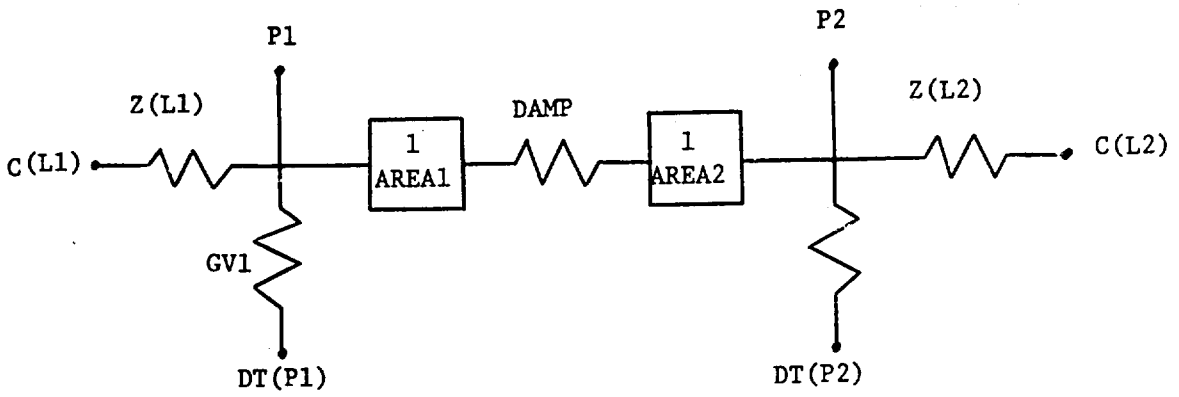



Figure 6.102-3
ACT102 Schematic

The past values of P1 and P2, DT(P1) and DT(P2) respectively, are shown at the ends of pseudo passages along with their conductances, GV1 and GV2.

The conductances GV1 is calculated using the equation

$$GV1 = (VOL1+X*A1)/BULK* T$$

where, VOL1 is the head side volume, $in^3$,

X is the displacement of the piston, in,

BULK is the bulk modulus of the fluid, psi.

Similarly, GV2 is calculated using

$$GV2 = (VOL2-X*A2)/BULK* T$$

Summing the flows about P1 and P2 yields,

$$(C(L1)-P1)/Z(L1)+(DT(P1)-P1)*GV1 = V*A1$$

$$(PL-C(L2))/Z(L2)+(P2-DT(P2)*GV2 = V*A2$$

where, DT(P1) is the previously calculated head side pressure, psi,

DT(P2) is the previously calculated rod side pressure, psi

Solving the above equations for P1 and P2,

$$P1 = \left[\frac{C(L1)}{Z(L1)} + DT(P1)*GV1 - V*A1\right] / \left[\frac{1}{Z(L1)} + GV1\right]$$

$$P2 = \left[\frac{C(L2)}{Z(L2)} + DT(P2)*GV2 + U*A2\right] / \left[\frac{1}{Z(L2)} + GV2\right]$$

Let,

$$C1 = C(L1)/Z(L1) + DT(P1)*GV1$$

$$C2 = C(L2)/Z(L2) + DT(P2)*GV2$$

$$G1 = 1/Z(L1) + GV1$$

$$G2 = 1/Z(L2) + GV2$$

Substituting C1, C2, G1 and G2 into the equations for P1 and P2 yields,

$$P1 = (C1 - V*A1)/G1 \qquad 6.102.2$$

$$P2 = (C2 + V*A2)/G2$$

Rewriting 6.102.1 using the equations for P1, P2, $F_D$ and $F_I$ gives,

$$F = ((C1-V*A1)/G1)*A1-((C2*V*A2)/G2)*A2-V*DAMP-(V-Vo)*M/\Delta T$$

Solving for V gives

$$V = \left[\frac{C1*A1}{G1} - \frac{C2*A2}{G2} - F+Vo*M/\Delta T\right] \Bigg/ \left[\frac{A1^2}{G1} + \frac{A2^2}{G2} + (DAMP+M/\Delta T)\right]$$

With the velocity of the piston, V, known the pressures in the head and rod side of the actuator may be determined using,

$$P1 = (C1 - V*A1)/G1$$

$$P2 = (C2 + V*A2)/G2$$

The displacement of the piston is given by

$$X = X_O + (V-Vo)\Delta T$$

where $X_O$ is the previously calculated displacement, in.

The pressures and flows at the connections to the actuator may then be calculated as follows,

P(L1) = P1

P(L2) = P2

Q(L1) = (C(L1) - P(L1))/Z(L1)

Q(L2) = (C(L2) - P(L2))/Z(L2)

where, P(L1) and P(L2) are the pressures at the connections, psi

Q(L1) and Q(L2) are the flows at the connections, cis.

6.102.2 Assumptions

Friction and stiction are assumed to be zero

6.102.3 Computational Method

1000 Section

The slope of the actuator load stroke curve is calculated

DT(LOADS) = (D(MAXL)-D(MINL))/(D(MAXST)-D(MINST))

The load at zero stroke and the net external load are then calculated.

DT(LOADZ) = D(MAXL) - DT(LOADS)*D(MAXST)

DT(LOADEX) = DT(LOADS)*D(INPOS)+DT(LOADZ)+(D(AREA1)

-D(AREA2))*ATPRES

A sign convention is established such that flow into the volume 1 chamber and the resulting piston velocity are positive.

Various variables are initialized to zero.

1500 Section

The entry first determines whether connection no. 1 is attached to an upstream or downstream line. This establishes the actuator steady state mode of operation. If entry is made using connection no. 2, leg pressure gain (or loss) and leg laminar constant are updated. Pressure at connection no. 2 is also calculated and stored. If entry is made using connection no. 1, tests are performed to verify that the piston is free to move as prescribed by the flow guess.

If the piston is on a stop and the flow guess is such that motion would be into the stop, the node overboard flow is set to zero and impedance is set to a very large number (40000).

If the piston is free to move, the overboard flow, piston velocity, P across piston and pressure at connection no. 2 are calculated.

2000 Section

The variables used in the 1500 section are changed to the values needed in the 3000 section.

3000 Section

A series of temporary variables GV1, G1, C1, C11 and DP etc. are calculated using the predicted actuator position DT(PX). The variables are used in the solution of the network shown in Figure 6.102-3.

The velocity of the piston is calculated using

PT(VEL) = (C11-C22-DP+VELO*DT(INERT))/ZN

The DT(INERT) term is used to incorporate the effect of the load mass in much the same way as the fluid flow is included in the pressure to give a characteristic pressure. The inertia term is equal to the force required to decelerate the piston from the old velocity to zero in the time interval DELT, and hence contributes to the net force on the piston.

If the velocity is constant then the inertia term included in DT(NDAMP) balances out the DT(INERT) term and produces no net effect.

The position of the piston is the calculated from

DT(X) = DT(X)+(VELO+DT(VEL))*DELTO2

The piston position is checked to determine whether or not it exceeds the maximum or minimum actuator stroke specified in the input data.

The pressures within volumes 1 and 2 are then calculated using,

DT(P1) = (C1-DT(VEL)*D(AREA1))/G1

DT(P2)= (C2+DT(VEL)*D(AREA2))/G2

The flows and pressures into and out of the actuator is then calculated by

D(L1) = (C(L1) -DT(P1))/Z(L1)

P(L1) = DT(P1)

Q(L2) = (C(L2) - DT(P2))/Z(L2)

P(L2) = DT(P2)

6.102.4 Approximations

None

6.102.5 Limitations

With no friction or stiction the model response to small pressure changes or load changes will not be accurate.

The model is inaccurate if the actuator volumes are allowed to cavitate.

6.102.6 Variable Names

| Variable | Description | Units |
|---|---|---|
| D(AREA1) | #1 Piston area (Extend) | $in^2$ |
| D(AREA2) | #2 Piston area (Retract) | $in^2$ |
| C1 | Temporary variable | -- |
| C11 | Temporary variable | -- |
| C2 | Temporary variable | -- |
| C22 | Temporary variable | -- |
| D(DAMP) | Seal friction | $\frac{lb\text{-}sec}{in}$ |
| DELTO2 | Integration constant | sec |
| DT(DELTP) | Pressure drop across piston | psi |
| DP | External load | lb |
| GV1 | Conductance of Volume 1 | cis/psi |
| GV2 | Conductance of Volume 2 | cis/psi |
| G1 | Temporary variable | -- |
| G2 | Temporary variable | -- |
| DT(INERT) | Load inertia | lb sec/in |
| D(INPOS) | Initial actuator position | in |
| DT(KBULK) | Oil bulk modulus times T | psi*sec |
| DT(LOADEX) | External load | lbs |
| DT(LOADS) | Load slope | lb/in |
| DT(LOADZ) | Load at zero stroke | lbs |
| LS | Connection sign | -- |
| L1 | Dummy variable | -- |
| L2 | Dummy variable | -- |

| Variable | Description | Units |
|---|---|---|
| D(MASS) | Load mass | lb $sec^2$/in |
| D(MAXL) | Load - actuator fully extended | lbs |
| D(MAXST) | Maximum actuator stroke | in |
| D(MINL) | Load-actuator fully retracted | lbs |
| D(MINST) | Minimum actuator stroke | in |
| N | Integer counter | |
| DT(NDAMP) | Load damping factor | lb sec/in |
| DT(PX) | Predicted piston position | in |
| DT(P1) | Pressure in volume 1 | psi |
| DT(P2) | Pressure in volume 2 | psi |
| TLOAD | Temporary variable | -- |
| DT(VEL) | Actuator velocity | in/sec |
| VELO | Previous actuator velocity | in/sec |
| D(VOL1) | Volume 1 at zero stroke | $in^3$ |
| D(VOL2) | Volume 2 at zero stroke | $in^3$ |
| DT(X) | Piston position | in |
| ZN | Temporary variable | - |

## 6.102.7 Subroutine Listing

```
      SUBROUTINE ACT102 (D,DT,DD,L)
C ****REVISED JUNE 1976 ****
      DOUBLE PRECISION DD
      DIMENSION D(11),DT(11),DD(1),L(4)
      COMMON NTELPL,NTOLPL,IPT,IPOINT,NPTS,INEL,KNEL,NTOPL,NLPLT(61,3),
     1 PQLEG(90,12),LCS(90,10),ILEG(1400),PN(90),QN(90)
      COMMON/SUB/PARM(150,9),PM(1500),QM(1500),P(300),Q(300),C(300)
     1,Z(300),RHO(20),S2ORHO(20),VISC(20),BULK(20),TEMP(20),PVAP(20)
     2,ATPRES,T,DELT,TFINAL,PLTDEL,PI,TITLE(20),LEGN,ICON
     3,KTEMP(99),LSTART(150),NLPT(150),LTYPE(99),NC(99),INX,INZ
     4,INV,ISTEP,NLINE,NEL,IND,IENTR,MNLINE,MNEL,MNLEG,MNNODE,MNPLOT
     5,MNLPTS,MDS
      INTEGER AREA1,AREA2,VOL1,VOL2,DAMP,X,VEL,P1,P2,ZQ,DELTP,PX
C     D ARRAY VARIABLES
      DATA AREA1/1/,AREA2/2/,VOL1/3/,VOL2/4/,MINST/5/,MAXST/6/,
     1 DAMP/7/,MASS/8/,MINL/9/,MAXL/10/,INPOS/11/
C     DT ARRAY VARIABLES
      DATA X/1/,VEL/2/,LOADZ/3/,LOADS/4/,LOADEX/5/,P1/6/,P2/7/,
     1 PX/8/,ZQ/9/,NDAMP/10/,INERT/11/,DELTP/12/,KBULK/13/
      IF(IENTR) 1000,2000,3000
 1000 CONTINUE
      IF (INEL.NE.0) GO TO 1500
      DO 1010 I=1,20
 1010 DT(I)=0.0
      L(3)=1
      IF(L(1)/2.NE.(L(1)+1)/2) L(3)=-1
      L(4)=-1
      IF(L(2)/2.NE.(L(2)+1)/2) L(4)=1
      DT(ZQ)=0.0
      DT(DELTP)=0.0
      DT(X)=D(INPOS)
      DT(PX)=DT(X)
      DT(LOADS)=(D(MAXL)-D(MINL))/(D(MAXST)-D(MINST))
      DT(LOADZ)=D(MAXL)-DT(LOADS)*D(MAXST)
      DT(LOADEX)=DT(LOADS)*D(INPOS)+DT(LOADZ)+(D(AREA1)-D(AREA2))*ATPRES
      DT(KBULK)=BULK(KTEMP(IND))*DELT
      DELTO2=DELT/2.0
      RETURN
C
C****STEADY STATE SECTION****
C
 1500 CONTINUE
      QA=PQLEG(INEL,1)
      LS=L(2+KNEL)
      QS=PQLEG(INEL,2)*LS
      N=LCS(INEL,3)
      LCS(INEL,7)=5
      IF(LS.GT.0) GO TO 1510
      IF(INX.NE.1.AND.KNEL.EQ.1) GO TO 1900
      N=LCS(INEL,2)
```

6.102.7 (Continued)

```
 1510 IF(KNEL.EQ.2) GO TO 1850
      IF(D(INPOS).GT.D(MINST)) GO TO 1600
      IF(DT(LOADEX).GE.0.0) GO TO 1550
      IF(QS.GT.0.0) GO TO 1650
 1550 QN(N)=0.0
      DT(ZQ)=40000.
      TLOAD=0.0
      DT(VEL)=0.0
      GO TO 1700
 1600 IF(D(INPOS).LT.D(MAXST)) GO TO 1650
      IF(DT(LOADEX).LE.0.0) GO TO 1550
      IF(QS.GT.0.0) GO TO 1550
 1650 DT(ZQ)=0.0
      TLOAD=DT(LOADEX)
      QN(N)=(-1.)*QA*QS*(D(AREA1)-D(AREA2))/D(AREA1)
      DT(VEL)=QA*QS/D(AREA1)
 1700 DT(DELTP)=(PN(N)*(D(AREA1)-D(AREA2))-
     1TLOAD-DT(VEL)*D(DAMP))/D(AREA2)
 1750 DT(P1)=PN(N)
      DT(P2)=PN(N)+DT(DELTP)-QA*QS*DT(ZQ)
 1800 RETURN
 1850 IF(INX.EQ.1.AND.LS.EQ.-1) GO TO 1900
      PQLEG(INEL,6)=PQLEG(INEL,6)+DT(ZQ)
      PQLEG(INEL,5)=PQLEG(INEL,5)+DT(DELTP)*LS
      PQLEG(INEL,11)=PQLEG(INEL,11)+DT(DELTP)*LS-DT(ZQ)*QS*QA
      RETURN
 1900 WRITE(6,1950) IND,KNEL,INEL
 1950 FORMAT(5X,7HCOMP NO,I3,20H, HAD INVALID CON NO      ,I3,
     1 11H, IN LEG NO      ,I4)
      WRITE(6,999)
  999 FORMAT(10X,33HPROGRAM STOP IN SUBROUTINE ACTI02)
      STOP
 2000 CONTINUE
      DT(INERT)=D(MASS)/DELT
      DT(NDAMP)=D(DAMP)+DT(INERT)
      DT(LOADEX)=(D(AREA1)-D(AREA2))*ATPRES
      DT(ZQ)=0.0
      RETURN
 3000 CONTINUE
      L1=L(1)
      L2=L(2)
      GV1=(D(VOL1)+DT(PX)*D(AREA1))/DT(KBULK)
      GV2=(D(VOL2)-DT(PX)*D(AREA2))/DT(KBULK)
      G1 =1.0/Z(L1)+GV1
      G2 =1.0/Z(L2)+GV2
      C1=C(L1)/Z(L1)+DT(P1)*GV1
      C11=C1*D(AREA1)/G1
      C2=C(L2)/Z(L2)+DT(P2)*GV2
      C22=C2*D(AREA2)/G2
      DP=DT(LOADEX)+DT(LOADZ)+DT(LOADS)*DT(PX)
      DT(ZQ)=DP
      VELO=DT(VEL)
```

6.102.7 (Continued)

```
ZN=DT(NDAMP)+D(AREA1)**2/G1+D(AREA2)**2/G2
DT(VEL)=(C11-C22-DP+VELO*DT(INERT))/ZN
DT(PX)=DT(X)
DT(X)=DT(X)+(VELO+DT(VEL))*DELTO2
CALL XLIMIT(DT(X),DT(VEL),DP,D(MINST),D(MAXST))
DT(PX)=DT(X)*2.0-DT(PX)
IF(DP.NE.0.0) DT(PX)=DT(X)
DT(P1)=(C1-DT(VEL)*D(AREA1))/G1
DT(P2)=(C2+DT(VEL)*D(AREA2))/G2
Q(L1)=(C(L1)-DT(P1))/Z(L1)
P(L1)=DT(P1)
Q(L2)=(C(L2)-DT(P2))/Z(L2)
P(L2)=DT(P2)
RETURN
END
```

## 6.103 SUBROUTINE ACT103

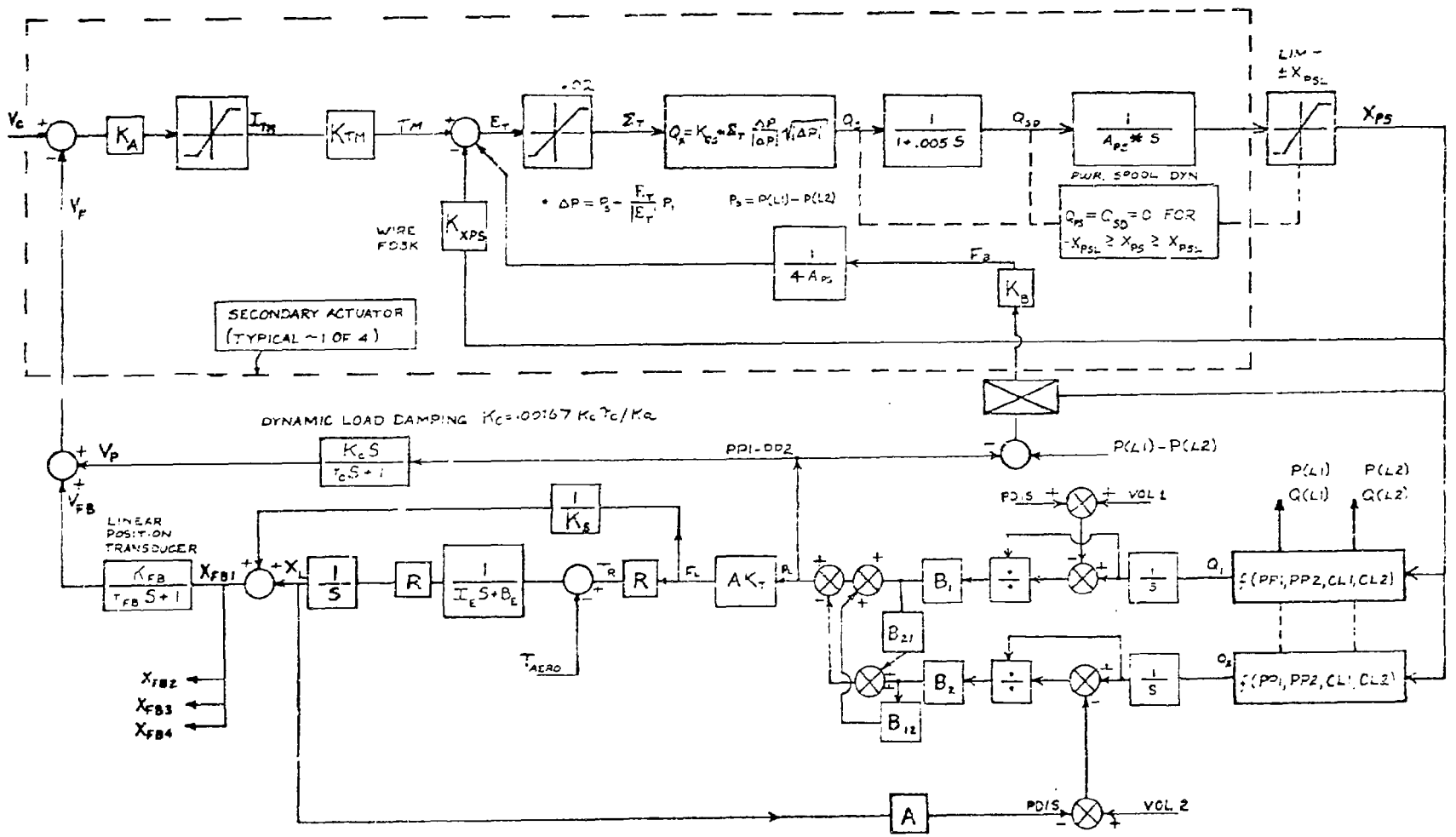
Subroutine ACT103 models the shuttle elevon actuators, the layout of which is shown in Figure 6.103-1. The elevon actuators operate from three pairs of hydraulic supply and return lines. The actuator supply and return is selected from the three hydraulic systems by a switching valve module as shown in Figure 6.103-2. For the purposes of modeling the switching valve module is considered to be an independent model connected to the actuator via pseudo lines. The input command and hinge moments are supplied by the SDF Program which updates the values at each sample time interval of the guidance system which is .04 seconds.

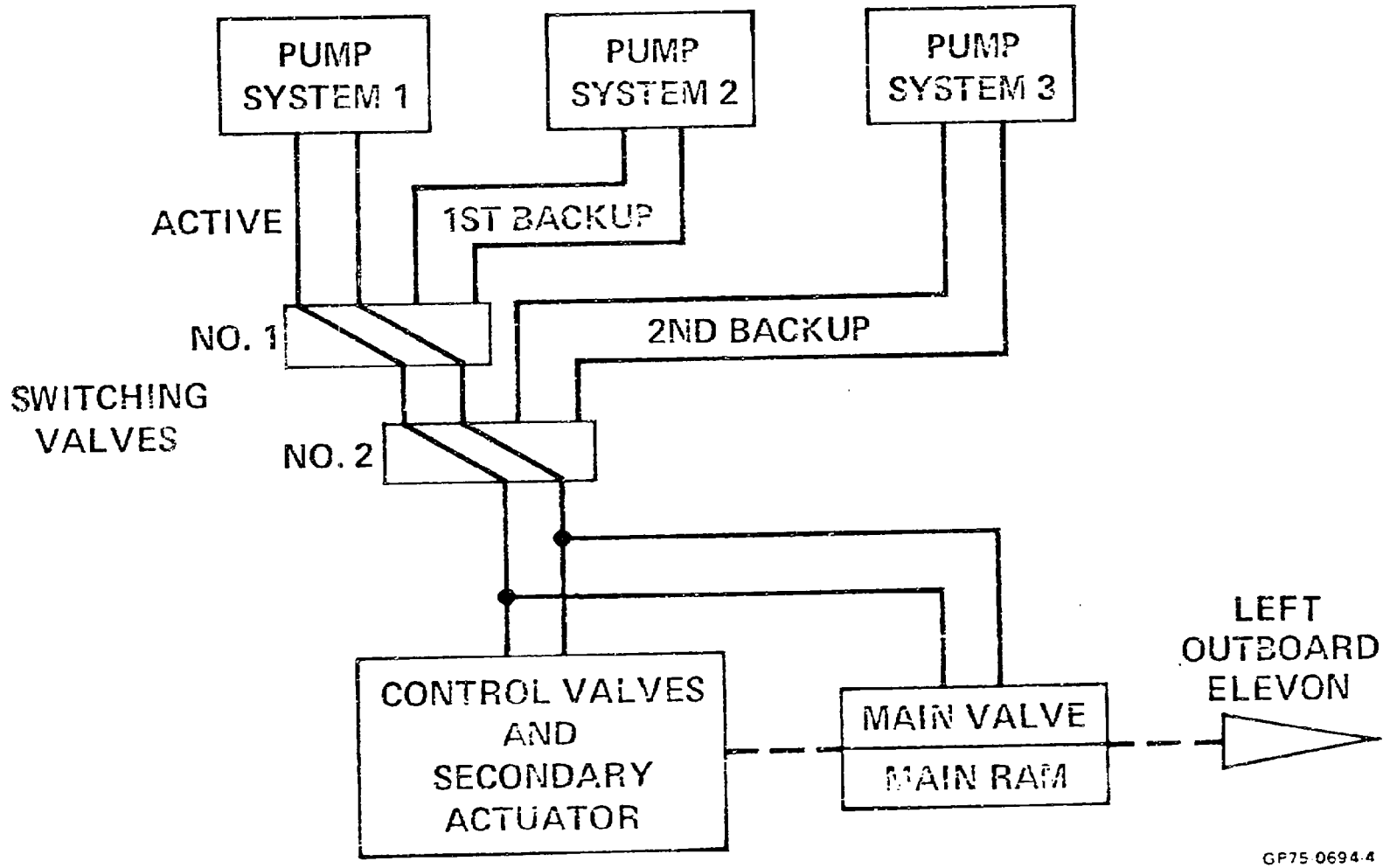
### 6.103.1 Math Model

This elevon math model was derived from information and a model provided by RI. The model shown in Figure 6.103-1 takes the difference between the position command signal VC and the position feedback signal VFB, to generate commands to the servo valve torque motor. The net applied torque deflects the servo valve causing differential flow to the secondary actuator piston. The subsequent displacement of the secondary actuator which is attached to the servo valve by a wire feedback, continues until the valve is returned to the equilibrium flow position. Changes in the position of the secondary actuator are also transmitted to the power spool through the summing linkage. The power spool displacements, XPS, generate differential flow the main ram causing elevon deflection. The elevon deflection is sensed as VFB and compared with the elevon position command $V_c$ to close the loop. Demand flow on the hydraulic system is generated by the sum of the flow to the main ram with the flow to four secondary actuators plus first stage leakage.

# ELEVON ACTUATION CONTROL SYSTEM MODEL

REVISION A




PUMP SYSTEM 1
PUMP SYSTEM 2
PUMP SYSTEM 3
ACTIVE
1ST BACKUP
NO. 1
2ND BACKUP
SWITCHING VALVES
NO. 2
CONTROL VALVES AND SECONDARY ACTUATOR
MAIN VALVE
MAIN RAM
LEFT OUTBOARD ELEVON
GP75-0694-4

6.103.2 Assumptions

The model is limited to simulating one channel of the secondary actuator and it is assumed that the other channels if operative will give identical outputs, in terms of available force at the main control valve.

The model of the differential pressure transducer does not have static and dynamic friction. To save computing time a common hinge moment arm has been used for translating force to hinge moment and elevon velocity to actuator velocity.

The actuator velocity conversion is not quite correct since the actuator position is the sum of the elevon deflection and structural deflection.

6.103.3 Computation Method

The method used for computation is based on the previous program developed by J. Fivel and J. Callihan.

This program uses Tustin's method of integration and is arranged such that each differential is evaluated using the latest value of the previous integral, in a series calculation, rather than the more usual parallel integration methods, where the integrals are effectively evaluated simultaneously using previous values to evaluate the derivatives.

The use of the series integration used together with predicted values for selected variables works better than other methods under certain conditions. The computation follows the usual component subroutine layout shown in Figure 6.1-1 on page 6.1-2.

1000 Section.

This section is used to initialize variables used by the program. For the most part, the integration variables are zeroed and the Tustin subroutine is called to initialize the integration constants.

1500 Section

This section is used to return values for the steady state calculation so that the system state conditions can be calculated. It is assumed that the secondary and surface actuators are at zero velocity, hence the only steady state flow is that due to the first stage servo valve leakage. The first stage impedance 1.0/(D(KQLOSS)*D(KCHAN) is added to the $Q^2$ value in PQLEG (INEL,8), and the pressure drop DELP is subtracted from the inlet pressure PQLEG(INEL,11).

2000 Section

This section is used to initialize the variables to their steady state values. The only variables involved for the subroutine are the surface actuator pressures which are set at the mean of the inlet and return pressures. The effect of the internal load is to vary the pressures either side of this mean. The actuator position is initialized to the trim position calculated by the SDF program. Most of the other variables are zeroed since to initialize the elevon variables at any condition other than zero velocity would be very difficult.

3000 Section

The transient section of the program is coded to follow the flow path of the model diagram, starting with the input position command DT(VC). DT(VC), is varied by the SDF program with the value being updated at 40 millisecond intervals. The SDF program uses the latest value of actuator position feedback voltage DT(VFB) in its calculations as an indication of the elevon position. SDF also calculates the surface hinge moment at this position and using that at the previous time step uses a linear interpolation to obtain a hinge moment at zero actuator position and the slope of hinge moment versus position.

ACT103 uses the zero and slope values, to interpolate values of TAERO until those values are updated by the next SDF computation. The interface between this subroutine ACT103 and the SDF program is via the following variables DT(1) thru DT(4).

| Variable | Description | Dimensions |
|---|---|---|
| DT(VC) (Calculated by SDF) | Input position command | Volts |
| DT(VFB) (Calculated by ACT103) | Elevon actuator position | Volts |
| DT(KAERO) (Calculated by SDF) | Elevon hinge moment at zero stroke | In lbs |
| DT(KAEROP) (Calculated by SDF) | Elevon hinge moment slope | In lbs/in |

The input command DT(VC) is feed via a first order lag to the junction, where it is summed with a predicted feedback value.

The error is passed through a limiter to the servo valve which converts it to torque. The torque is summed with the spring feedback from the secondary actuator and the pressure feedback across the secondary actuator piston.

The flow calculation uses the supply and pressures from the previous time step to calculate a new flow. The flow is passed through a simple lag to denote the servo valve time delay, and is then integrated to obtain a new secondary actuator (main control valve) position.

The actuator position is tested against its limits, if it is at a limit then the secondary actuator flow is set to zero.

The flow through the main control valve is calculated using the predicted actuator cavity pressure and the valve orifice equations which are solved in conjunction with the line characteristic equation.

The line characteristic pressure is reduced by an amount equal to the

secondary actuator flow plus the leakage flow times the characteristic impedance. This method avoids the need for a simultaneous solution which would have required a more expensive iterative solution.

The computed flows, which are positive when flowing into the actuator cavities, are then integrated. The integrated flows are summed with the actuator displacement flow and the difference is used to compute a new actuator cavity pressure.

The cavity pressure computation is complicated by the actuator deflection due to differential pressure acting on the structural stiffness.

If solved sequentially, an arithmetic loop is created which creates computational instability. This was avoided by reformulating the loop to give a direct solution for actuator cavity pressures, with the structural stiffness being integrated into the calculation.

The remaining part of the computation follows the block diagram.

Switching Valve Module

The switching valve module is the same as that used in the TVC subroutine ACT105 and described in the TVC model description. This model will need to be revised when the valve design details are finalized.

The revisions will be in the area trigger valve operation which is different from the technique used in the TVC design.

6.103.4 Approximations

The servo valve first order lag of 0.005 seconds is an approximation.

6.103.5 Limitations

The Elevon module can be used to simulate the overall response of the actuator and surface to commands from the SDF program. Because of the simplification in the area of the secondary actuator the small signal response is better than can be expected in practice. The simulation of the switching valve is not ideal, because of the very fast response of these valves. To improve the model would require a much smaller time step. Should these limitations become a problem, a smaller time step can be used with some minor program changes and of course an increase in cost.

6.103.6 Variable Names

| Variable | Description | Outboard Valve* | Inboard Valve | Dimensions |
|---|---|---|---|---|
| APS | Working Area of Power Spool | | 0.193 | $in^2$ |
| D(AR) | Effective Ram Area | 18.02 | 21.80 | $in^2$ |
| D(BE) | Effective Elevon Damping Coefficient | 15000. | 45000. | in-lb-sec |
| EPST | Net Error Torque | | | in-lb |
| EYE | Torque Motor Input Current | | | ma |
| EYEL | Servo Amplifier Saturation Limit | | 8.0 | ma |
| FL | Ram Force to Load | | | lb |
| D(IE) | Elevon Moment of Inertia About Hinge Line | 2663. | 9743 | $in-lb-sec^2$ |
| L(ISYS) | Active Hydraulic Connections | | | - |
| KA | Servo Amplifier Gain | | 15. | ma/volt |
| DT(KAERO) | Hinge Moment at Zero Stroke | | | in-lb |
| DT(KAEROP) | Hinge Moment Slope | | | in-lb/in |
| D(KB) | Bernoulli Force Coefficient | 0.319 | 0.755 | in |
| D(KCHAN) | Number of Active Channels | | 4 | - |
| D(KFB) | Linear Position Transducer Gain | 1.173 | 0.683 | volts/in |
| KQLOSS | First Stage Leakage Coefficient | | 0.0237 | CIS/ PSI |
| D(KQPS) | Power Spool Flow Gain | 51.8 | 124.7 | $in^3/(sec\ lb)$ |
| KQS | Secondary Valve Flow Gain | | 0.387 | $in^3/(sec\ lb)$ |
| D(KS) | Structural Stiffness | 154000. | 298000. | lb/in |
| KTM | Torque Motor Gain Constant | | 0.045 | in-lb/ma |
| KXPS | Wire Feedback, Power Spool to Torque Motors | | 6.22 | in-lb/in |
| D(MA0) | Initial Position of Elevon Actuator | 8.793 | 15.09 | in |
| D(MA1) | 1st Moment Arm Constant | -4.063E-2 | -2.90E-2 | in/in |

| | | | | |
|---|---|---|---|---|
| D(MA2) | 2nd Moment Arm Constant | -5.51E-2 | -3.22e-3 | $in/in^2$ |
| D(MA3) | 3rd Moment Arm Constant | 1.40E-3 | 6.40E-4 | $in/in^3$ |
| D(MA4) | 4th Moment Arm Constant | 6.40E-4 | -5.88E-5 | $in/in^4$ |
| DT(PL) | Load Pressure Across Ram Piston | | | PSI |
| DT(P1) | Secondary Actuator Pressure | | | PSI |
| DT(QS) | Secondary Actuator Flow to Mod Piston | | | CIS |
| DT(RFXL) | Effective Moment Arm | | | in |
| DT(TAERO) | Hinge Moment at Zero Stroke | | | in-lb |
| TAUC | Dynamic Load Damping Time Constant | | 0.1 | sec |
| TAUFB | Linear Position Transducer Demod. Time Constant | | 0.004 | sec |
| VAP | Vapor Pressure | | | PSI |
| DT(VC) | Position Command Signal | | | volts |
| DT(VFB) | Ram Position Feedback Voltage | | | volts |
| VLVOL | Power Valve Overlap | | 0.0006 | in |
| D(VOL1) | #1 Cavity Volume at Mid-Stroke | 82.6 | 169.0 | $in^3$ |
| D(VOL2) | #2 Cavity Volume at Mid-Stroke | 82.6 | 169.0 | $in^3$ |
| DT(XFB) | Ram Pistion Position | | | in |
| DT(XPS) | Power Valve Displacement | | | in |

* Same as inboard unless otherwise noted.

6.103.7 Subroutine Listing

```
      SUBROUTINE ACT103 (D,DT,DD,L)
C
C     ****** REVISED JUNE 1976 ******
C     SHUTTLE ELEVON ACTUATOR MODEL WITH SWITCHING VALVE
C
      DOUBLE PRECISION DD
      COMMON NTELPL,NTOLPL,IPT,IPOINT,NPTS,INEL,KNEL,NTOPL,NLPLT(61,3),
     1 PQLEG(90,12),LCS(90,10),ILEG(1400),PN(90),QN(90)
      COMMON/SUB/PARM(150,9),PM(1500),QM(1500),P(300),Q(300),C(300)
     1,Z(300),RHO(20),S2ORHO(20),VISC(20),BULK(20),TEMP(20),PVAP(20)
     2,ATPRES,T,DELT,TFINAL,PLTDEL,PI,TITLE(20),LEGN,ICON
     3,KTEMP(99),LSTART(150),NLPT(150),LTYPE(99),NC(99),INX,INZ
     4,INV,ISTEP,NLINE,NEL,IND,IENTR,MNLINE,MNEL,MNLEG,MNNODE,MNPLOT
     5,MNLPTS,MDS
      REAL KA,KC,KTM,KQS,KXPS,KQLOSS
      INTEGER AR,BE,VC,VFB,VFBP,P1,XPS,
     2 TXPS,PS,XYQS,QS,XQI,XQ,
     3 XYXPSD,XPSD,XPSI,Q1,Q2,PP1,PP2,XYTE,TE,DELEDI,
     4 XYDLED,VXL,RFXL,XL,XSD,XYPL,PL,VPLI,XYXFB,XFB,VFBI,
     5 VOL1,VOL2,VXLP,XFBP,P1P
     6 ,FIBETA,XLP,XIQ1,XIQ2,VCS,VCN,VCL
     7 ,CRA,CSA,CRV,CSV
     8 ,XP,XPP,VELP,XS,XSP,VELS,COEFVP,COEFVS
      DIMENSION D(16),DT(100),DD(1),L(10)
C
C *** D ARRAY VARIABLES
C
      DATA KCHAN/1/,KB/2/,KQPS/3/,AR/4/,KS/5/,IE/6/,BE/7/,KFB/8/,
     2 VOL1/9/,VOL2/10/,MA0/12/,MA1/13/,MA2/14/,MA3/15/,MA4/16/
C
C *** DT ARRAY VARIBLES
C
      DATA VC/1/,VFB/2/,VFBP/3/,KAERO/4/,KAEROP/5/,
     1 P1/6/,XPS/7/,TXPS/8/,PS/9/,XYQS/10/,QS/11/,
     2 XQI/12/,XQ/13/,NQLOSS/14/,NQS/15/,NQPS/16/,NCAV/17/,
     3 IBETA/18/,XYXPSD/19/,XPSD/20/,XPSI/21/,Q1/22/,
     4 Q2/23/,PP1/24/,PP2/25/,XYTE/26/,TE/27/,DELEDI/28/,
     5 XYDLED/29/,VXL/30/,RFXL/31/,XL/32/,XSD/33/,XYPL/34/,PL/35/,
     6 VPLI/36/,XYXFB/37/,XFB/38/,KINT/39/,KFLOW/42/,VXLP/45/,XFBP/46/,
     1 P1P/47/,VFBI/48/,DPFB/49/,DPFBP/50/,FIBETA/51/,
     2 XLP/52/,XIQ1/53/,XIQ2/54/,KCOM/55/,VCS/58/,VCN/59/,VCL/60/,
     3 XP/61/,XPP/62/,VELP/63/,XS/64/,XSP/65/,VELS/66/,
     4 COEFVP/67/,COEFVS/68/,LOSS/69/,LEAK/70/,LSWACT/71/,
     5 NSUPRP/72/,NRETRP/73/,ISYS/8/
     6 ,CSA/74/,CSV/75/,CRA/76/,CRV/77/,TP/78/,KDAMP/79/,KTRANS/82/
C
C *** INITIALIZE CONSTANTS COMMON TO ALL ACTUATORS ***
      DATA VLVOL/.0006/,
     1 DORFCP/.5/,DORFCS/.5/,OLP/.005/,OLS/.005/,ZSPP/15./,SSPP/50./,
     2 A1PS1/.338/,A2PS2/.274/,A3PB/.263/,A4PR2/.327/,
```

6.103.7 (Continued)

```
     3 A1SS1/.338/,A2SS2/.230/,A3SB/.263/,A4SR2/.382/,
     4 KA/15./,KTM/.045/,EYEL/8.0/,KQS/.387/,KXPS/6.22/,
     5 APS/.193/,XPSL/.065/,KQLOSS/.0237/,ZS/6.5/,ZR/6.5/,
     6 KC/.000019/,TAUC/.1/,TAUFB/.004/
      IF (IENTR) 1000,2000,3000
 1000 IF (INEL.NE.0) GO TO 1500
C
C     INITIALIZATION
C
      DO 1001 I=1,100
 1001 DT(I)=0.0
C     CORRECT INPUT DATA FOR FLUID TEMPERATURE
      I=KTEMP(IND)
      DT(NQS)=S2ORHO(I)/SQRT(2./7.82E-5)
      DT(NQPS)=D(KQPS)*DT(NQS)*SQRT(2.)
      DT(NQLOSS)=KQLOSS*DT(NQS)
      DT(NQS)=KQS*DT(NQS)
      DT(IBETA)=BULK(I)
      DT(FIBETA)=DT(IBETA)*D(AR)*D(AR)/D(KS)
      CALL TUSTIN(1,1.,DT(KINT),DELT)
      CALL TUSTIN(2,.005,DT(KFLOW),DELT)
      CALL TUSTIN(2,.0159,DT(KCOM),DELT)
      CALL TUSTIN(3,TAUC,DT(KDAMP),DELT)
      CALL TUSTIN(2,TAUFB,DT(KTRANS),DELT)
C
C     SET UP SWITCHING VALVE CONSTANTS
      DINP=.75/(12*32.2*DELT)
      DELTO2=DELT/2.0
      BBVP=(DORFCP/A3PB)**2
      BVP=BBVP*(.1+DINP)/2.0
      SQBVP=BVP**2
      BBVS=(DORFCS/A3SB)**2
      BVS=BBVS*(.1+DINP)/2.0
      SQBVS=BVS**2
      DT(COEFVP)=2.0*.65*S2ORHO(I)
      DT(COEFVS)=2.0*.65*S2ORHO(I)
      DT(LEAK)=1000000.
C
      L(ISYS)=1
      IF(INV) 1220,1250,1200
 1200 IF(L(9).EQ.INV) GO TO 1250
      IF(L(10).EQ.INV) GO TO 1230
      L(ISYS)=3
      GO TO 1250
 1220 IF(L(9).NE.-INV) GO TO 1250
 1230 L(ISYS)=2
 1250 CONTINUE
      DT(LOSS)=RHO(1)*6.06
      DT(LSWACT)=1.0/(DT(COEFVS)*.2)**2+DT(LOSS)
      IF(L(ISYS).GT.2) GO TO 1260
```

6.103.7 (Continued)

```
      DT(LSWACT)=DT(LSWACT)+1.0/(DT(COEFVP)*.2)**2+DT(LOSS)
 1260 CONTINUE
      RETURN
C
C             --------- STEADY STATE SECTION
C
 1500 CONTINUE
      QLOSS = (1.0/(DT(NQLOSS)*D(KCHAN)))**2
      COES=DT(LEAK)
      IF((KNEL+1)/2.NE.L(ISYS)) GO TO 1600
      DT(QS)=PQLEG(INEL,1)*PQLEG(INEL,2)
      COES=0.0
 1600 QSA=PQLEG(INEL,1)
      PDELTA=PQLEG(INEL,2)*QSA*(QSA*DT(LSWACT)+COES)
      DELP=PQLEG(INEL,2)*QSA**2*QLOSS
      PSA=PQLEG(INEL,11)-PDELTA
      PRA=PSA-DELP
      PQLEG(INEL,11)=PRA-PDELTA
      PQLEG(INEL,6)=PQLEG(INEL,6)+COES*2.0
      PQLEG(INEL,8)=PQLEG(INEL,8)+2.0*DT(LSWACT)+QLOSS
      IF(COES.NE.0.0) GO TO 1700
      DT(PP1)=PSA
      DT(PP2)=PRA
 1700 RETURN
C
C        --------- VARIABLE INITIALIZATION
C
 2000 CONTINUE
      DT(NSUPRP)=DT(PP1)
      DT(NRETRP)=DT(PP2)
      DT(CSA)=DT(PP1)+DT(QS)*ZS
      DT(CSV)=DT(PP1)-DT(QS)*ZS
      DT(CRA)=DT(PP2)-DT(QS)*ZR
      DT(CRV)=DT(PP2)+DT(QS)*ZR
      DT(VCL)=DT(VC)
      DT(VCS)=DT(VC)
      DT(VFB)=DT(VC)
      DT(XFB)=DT(VFB)/D(KFB)
      DT(XYXFB)=DT(XFB)
      DT(VFBI)=DT(XFB)
      DT(XYXFB)=DT(XFB)
      DT(XFBP)=DT(XFB)
      TAERO = DT(KAERO)+DT(KAEROP)*DT(XFB)
      DT(RFXL) = D(MA0)+D(MA1)*DT(XFB)+D(MA2)*DT(XFB)**2
     1  +DT(XFB)**3*(D(MA3)+DT(XFB)*D(MA4))
      DT(XL) = DT(XFB)-(TAERO/DT(RFXL))/D(KS)
      DT(XLP)=DT(XL)
      F1 = TAERO/DT(RFXL)/D(AR)
      DT(PS) = DT(PP1)-DT(PP2)
      DELP = (DT(PP1)+DT(PP2))/2
```

6.103.7 (Continued)

```
      DT(PP1) = DELP+F1/2
      DT(PP2) = DELP-F1/2
      DT(PL) = F1
      DT(XYPL)=F1
      DT(XIQ1)=(D(VOL1)+DT(XFB)*D(AR))/(1.-DT(PP1)/DT(IBETA))
      DT(XIQ2)=(D(VOL2)-DT(XFB)*D(AR))/(1.-DT(PP2)/DT(IBETA))
      DT(VCL)=DT(VC)
      DT(VCS)=DT(VC)
      DT(XP)=0.2
      DT(XS)=0.2
      IF(L(ISYS)-2)2040,2020,2010
 2010 DT(XS)=-.2
      GO TO 2040
 2020 DT(XP)=-.2
 2040 DT(XPP)=DT(XP)
      DT(XSP)=DT(XS)
      DT(QS)=0.0
      RETURN
 3000 CONTINUE
C
C              **** TRANSIENT CALCULATIONS ****
C
C            **** INTIALIZE VARIABLES ****
      RFXLS=DT(RFXL)
      DT(RFXL) = D(MA0)+D(MA1)*DT(XL)+D(MA2)*DT(XL)**2
     1  +DT(XFB)**3*(D(MA3)+DT(XFB)*D(MA4))
      RFXLP=2.*DT(RFXL)-RFXLS
      TAERO=DT(KAERO)+DT(KAEROP)*DT(XFBP)
C
C     SECONDARY ACTUATOR
C
C              ----- SERVO AMPLIFIER
C
      DT(VCN)=DT(VC)
      DT(VCL)=DYNAM(DT(VCS),DT(VCL),DT(KCOM))
      DT(VCS)=DT(VCN)
      EYE=KA*(DT(VCL)-DT(VFBP))
      IF(ABS(EYE).GT.EYEL) EYE=SIGN(EYEL,EYE)
C
C              ----- TORQUE MOTOR
C
      TXPSS=DT(TXPS)
      DT(TXPS) = DT(XPS)*KXPS
      TXPSP=2.*DT(TXPS)-TXPSS
      TM = EYE*KTM
      EPST=TM-TXPSP
      IF(ABS(EPST).GT.0.02) EPST=SIGN(0.02,EPST)
      DELP=DT(PS)-SIGN(1.0,EPST)*DT(P1P)
      IF(DELP.LT.0.0) DELP=0.0
      DT(QS) = EPST*DT(NQS)*SQRABS(DELP)
```

6.103.7 (Continued)

```
C
C            ------ MAIN CONTOL VALVE POSITION
C
      DT(XQI) = DYNAM(DT(XYQS),DT(XQI),DT(KFLOW))
      DT(XPSD) = DT(XQI)/APS
      DT(XPS) = DYNAM(DT(XYXPSD),DT(XPS),DT(KINT))
      CALL XLIMIT(DT(XPS),DT(XPSD),TM,-XPSL,XPSL)
      IF(TM.EQ.0.0) GO TO 60
      DT(QS)=0.0
      DT(XQI)=0.0
   60 CONTINUE
      DT(XYXPSD)=DT(XPSD)
      DT(XYQS)=DT(QS)
      QLOSS = (ABS(DT(XQI))+DT(NQLOSS)*SQRABS(DT(PS)))*D(KCHAN)
      Q1S=DT(Q1)
      Q2S=DT(Q2)
      NXV=DT(XPS)/VLVOL
      IF(NXV) 130,170,180
C
C     XPS LESS THAN 0
C
  130 XPSS=((DT(XPS)+VLVOL)*DT(NQPS))**2
      SGN=DT(CRA)-DT(PP1)+QLOSS*ZR
      FN2=XPSS*ZR
      ICALC=1
C
C     CALCULATE MAIN CONTROL VALVE FLOWS
C
  140 FLOW=(SQRT(FN2**2+4.0*XPSS*ABS(SGN))-FN2)/2.0
      ZCALC = ICALC
      GO TO (150,160,190,200),ICALC
C
  150 DT(Q1)=SIGN(FLOW,SGN)
      QRA = DT(Q1)-QLOSS
      FN2=XPSS*ZS
      SGN=DT(CSA)-DT(PP2)-QLOSS*ZS
      ICALC=2
      GO TO 140
  160 DT(Q2)=SIGN(FLOW,SGN)
      QSA = DT(Q2)+QLOSS
      GO TO 220
C
C     XPS = 0
C
  170 DT(Q1)=0.
      FLOW=0.0
      QSA = QLOSS
      DT(Q2)=0.
      QRA = -QLOSS
      ICALC=5
```

6.103.7 (Continued)

```
      GO TO 220
C
C     XPS GREATER THAN 0
C
  180 XPSS=((DT(XPS)-VLVOL)*DT(NQPS))**2
      FN2=ZS*XPSS
      SGN=DT(CSA)-DT(PP1)-QLOSS*ZS
      ICALC=3
      GO TO 140
  190 DT(Q1)=SIGN(FLOW,SGN)
      QSA = DT(Q1)+QLOSS
      FN2=ZR*XPSS
      SGN=DT(CRA)-DT(PP2)+QLOSS*ZR
      ICALC=4
      GO TO 140
  200 DT(Q2)=SIGN(FLOW,SGN)
      QRA = DT(Q2)-QLOSS
C
  220 PSA = DT(CSA)-ZS*QSA
      PRA =DT(CRA)-ZR*QRA
      VAP=PVAP(KTEMP(IND))
      IF(PRA.GT.VAP.AND.DT(NCAV).LE.0.0) GO TO 270
      GO TO (230,230,240,240,250),ICALC
  230 FLOW=SQRABS(XPSS*(VAP-DT(PP1)))
      DT(Q1)=FLOW
      GO TO 250
  240 FLOW=SQRABS(XPSS*(VAP-DT(PP2)))
      DT(Q2)=FLOW
  250 QRA=(DT(CRA)-VAP)/ZR
      PRA=VAP
      DT(NCAV)=DT(NCAV)+FLOW-QLOSS-QRA
  270 CONTINUE
      DT(PS) = PSA-PRA
C
C              ----- ACTUATOR CHAMBER PRESSURES
      PFLOW=DT(VXLP)*D(AR)
      PDIS=DT(XLP)*D(AR)
      DT(XIQ1)=DT(XIQ1)+.5*DELT*(DT(Q1)+Q1S)
      DT(XIQ2)=DT(XIQ2)+.5*DELT*(DT(Q2)+Q2S)
      FQ1=DT(FIBETA)/DT(XIQ1)
      FQ2=DT(FIBETA)/DT(XIQ2)
      BETA1=DT(IBETA)*(1.+FQ2)/(1.+FQ1+FQ2)
      BETA2=DT(IBETA)*(1.+FQ1)/(1.+FQ1+FQ2)
      PP11=BETA1*(1.-(PDIS+D(VOL1))/DT(XIQ1))
      PP22=BETA2*(1.+(PDIS-D(VOL2))/DT(XIQ2))
      PP21=PP11*FQ2/(1.+FQ2)
      PP12=PP22*FQ1/(1.+FQ1)
      DT(PP1)=PP11+PP12
      DT(PP2)=PP22+PP21
      IF(DT(PP1).LT.VAP) DT(PP1)=VAP
```

6.103.7 (Continued)

```
      IF(DT(PP2).LT.VAP) DT(PP2)=VAP
      DT(PL) = DT(PP1)-DT(PP2)
C
C              -----  LOAD DYNAMICS
C
       FL=DT(PL)*D(AR)
      TR = FL*RFXLP
      DT(XSD)=FL/D(KS)
      FDRIVE=TR-TAERO
      CALL CFRIC2(FDRIVE,FF,DT(VXLP),DT(VXL),DT(TE),DT(XYTE),
     1  DT(DELEDI),DELS,201000.,1.263,1.)
      DT(TE)=FDRIVE+FF-DT(VXLP)*D(BE)/RFXLP
      DT(DELEDI) = DYNAM(DT(XYTE),DT(DELEDI),DT(KINT))
      VXLS=DT(VXL)
      DT(VXL) = DT(DELEDI)*RFXLP/D(IE)
      DT(VXLP)=2.*DT(VXL)-VXLS
      DT(XYTE) = DT(TE)
      XLS=DT(XL)
      DT(XL) = DYNAM(DT(XYDLED),DT(XL),DT(KINT))
      DT(XLP)=2.*DT(XL)-XLS
      DT(XYDLED) = DT(VXL)
      XFBS=DT(XFB)
      DT(XFB) = DT(XL)+DT(XSD)
      DT(XFBP)=2.*DT(XFB)-XFBS
      P1S=DT(P1)
      DT(P1)=DT(XPS)*D(KB)*(DT(PS)-SIGN(DT(PL),DT(XPS)))/APS/D(KCHAN)
      DT(P1P)=2.*DT(P1)-P1S
C
C     POSITION AND PRESSURE FEEDBACK
C
      DT(VPLI) = DYNAM(DT(XYPL),DT(VPLI),DT(KDAMP))
      DT(XYPL) = DT(PL)
      DT(VFBI) = DYNAM(DT(XYXFB),DT(VFBI),DT(KTRANS))
      DT(XYXFB) = DT(XFB)
      VFBS=DT(VFB)
      DT(VFB)=DT(VFBI)*D(KFB)+DT(VPLI)*KC
      DT(VFBP)=2.0*DT(VFB)-VFBS
C
C     SWITCHING VALVE CALCULATIONS
C
      DO 3010 I=1,6
      IL=L(I)
      Q(IL)=0.0
 3010 P(IL)=C(IL)
      QSV=0.0
      QRV=0.0
      PSV=DT(CSV)
      PRV=DT(CRV)
      NS=1
      PXP=DT(XP)*2.0-DT(XPP)
```

6.103.7 (Continued)

```
      PXS=DT(XS)*2.0-DT(XSP)
      NDISVP=PXP/OLP
      NDISVS=PXS/OLS
      IF(NDISVS) 3020,3040,3029
 3020 COE=((PXS+OLS)*DT(COEFVS))**2
      COE=COE/(COE*DT(LOSS)+1.0)
      NS=3
      GO TO 3036
 3029 IF(NDISVP) 3030,3045,3035
 3030 NS=2
 3035 COEP=1.0/((PXP-SIGN(OLP,PXP))*DT(COEFVP))**2+DT(LOSS)
      COES=1.0/((PXS-OLS)*DT(COEFVS))**2+DT(LOSS)
      COE=1.0/(COEP+COES)
 3036 LS=NS*2
      LR=L(LS)
      LS=L(LS-1)
C     SUPPLY CALCULATION
      ZA=(Z(LS)+ZS)*COE/2.0
      QSV= -ZA+SQRT(ZA**2+ABS(C(LS)-DT(CSV))*COE)
      QSV= SIGN(QSV,C(LS)-DT(CSV))
      PSV=DT(CSV)+ZS*QSV
      Q(LS)=QSV
      P(LS)=C(LS)-QSV*Z(LS)
      DT(NSUPRP)=C(LS)-QRV*(Z(LS)-ABS(QRV)*COEP)
C     RETURN CALCULATION
      ZA=(Z(LR)+ZR)*COE/2.0
      QRV= -ZA+SQRT(ZA**2+ABS(C(LR)-DT(CRV))*COE)
      QRV= SIGN(QRV,C(LR)-DT(CRV))
      PRV=DT(CRV)+ZR*QRV
      Q(LR)=QRV
      P(LR)=C(LR)-QRV*Z(LR)
      DT(NRETRP)=C(LR)-QRV*(Z(LR)-ABS(QRV)*COEP)
 3040 N=1
      IF(NDISVP) 3043,3100,3044
 3043 N=N+2
 3044 DT(NSUPRP)=P(L(N))
      DT(NRETRP)=P(L(N+1))
      GO TO 3100
 3045 DT(NSUPRP)=PSV
      DT(NRETRP)=PRV
 3100 CONTINUE
      VELPO=DT(VELP)
      PBST=P(L(1))
      IF(PXP.LT.0.1) PBST=P(L(2))
      PSPL=(ZSPP-SSPP*PXP+P(L(3))*A2PS2+P(L(4))*A4PR2-P(L(1))*A1PS1)
      PSPL=(PSPL-VELPO*DINP)/A3PB
      DT(VELP)=-BVP+SQRT(SQBVP+3BVP*ABS(PBST-PSPL))
      DT(VELP)=SIGN(DT(VELP),PBST-PSPL)
      DT(XPP)=DT(XP)
      DT(XP)=DT(XP)+(VELPO+DT(VELP))*DELTO2
```

6.103.7 (Continued)

```
      CALL XLIMIT(DT(XP),DT(VELP),PSPL,-.2,.2)
      IF(PSPL.NE.0.0) DT(XPP)=DT(XP)
C
      VELSO=DT(VELS)
      PBST=DT(NSUPRP)
      IF(PXS.LT.0.1) PBST=DT(NRETRP)
      PSPL=(ZSPP-SSPP*PXS+P(L(5))*A2SS2+P(L(6))*A4SR2
     1 -DT(NSUPRP)*A1SS1)
      PSPL=(PSPL-VELSO*DINP)/A3SB
      DT(VELS)=-BVS+SQRT(SQBVS+3BVS*ABS(PBST-PSPL))
      DT(VELS)=SIGN(DT(VELS),PBST-PSPL)
      DT(XSP)=DT(XS)
      DT(XS)=DT(XS)+(VELSO+DT(VELS))*DELTO2
      CALL XLIMIT(DT(XS),DT(VELS),PSPL,-.2,.2)
      IF(PSPL.NE.0.0) DT(XSP)=DT(XS)
      DT(CSA)=PSV+ZS*QSV
      DT(CSV)=PSA-ZS*QSA
      DT(CRA)=PRV+ZR*QRV
      DT(CRV)=PRA-ZR*QRA
      DT(89)=TAERO
      DT(90)=TR
      RETURN
      END
```

## 6.104 SUBROUTINE ACT104

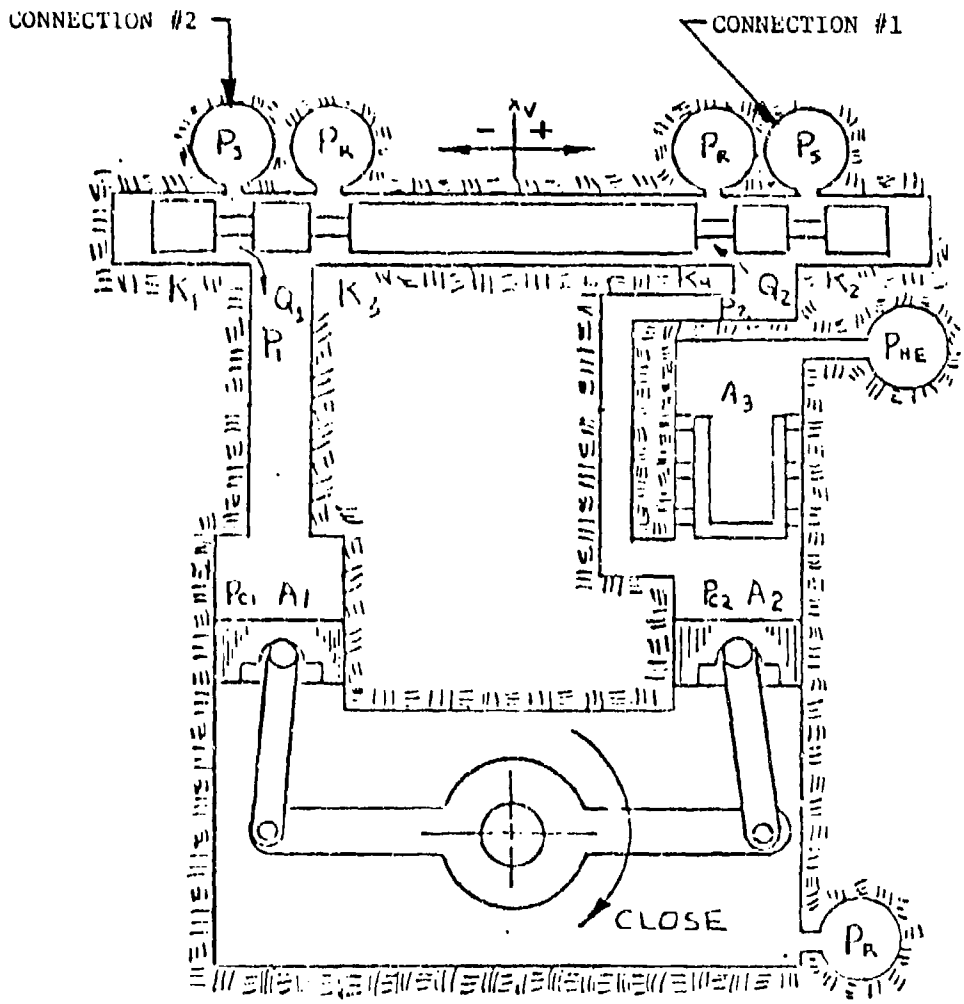



FIGURE 6.104-1

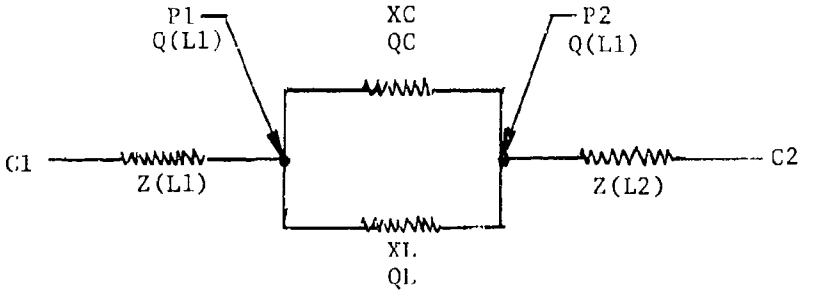
TYPE NO. 104 SHUTTLE ENGINE CONTROL ACTUATOR

The No. 104 actuator is a model of a push-push servoactuator. External loads as well as friction are not included in the model. Data inputs required are position transducer gain, servovalve gain constant, average effective moment arm and piston area.

6.104.1 Math Model

This EC actuator math model was derived from "Models for Use in Pressure Transient Simulation Report" (388-202-75-080), written by A. C. Morseof Rockwell International Space Division. The model shown in Figure 6.104-2 takes the difference between the position command signal KCOM and the position feedback signal KPH, to generate commands to the servo valve torque motor. The net applied torque deflects the servo valve causing differential flow to the secondary actuator piston. The subsequent displacement of the secondary actuator which is attached to the servo valve by a wire feedback, continues until the valve is returned to the equilibrium flow position. Changes in the position of the secondary actuator are also transmitted to the power spool through the summing linkage. The power spool displacements XV generate differential flow to the actuator piston. The actuator position is sensed as VF and compared with the position command VCOM to close the loop.

For purposes of interfacing with HYTRAN program the servoactuator schematic can be considered as follows:




P1 = C1 - Q(L1)*Z(L1)

P2 = C2 + Q(L1)*Z(L2)

QC = XC*SQRT(P1-P2)

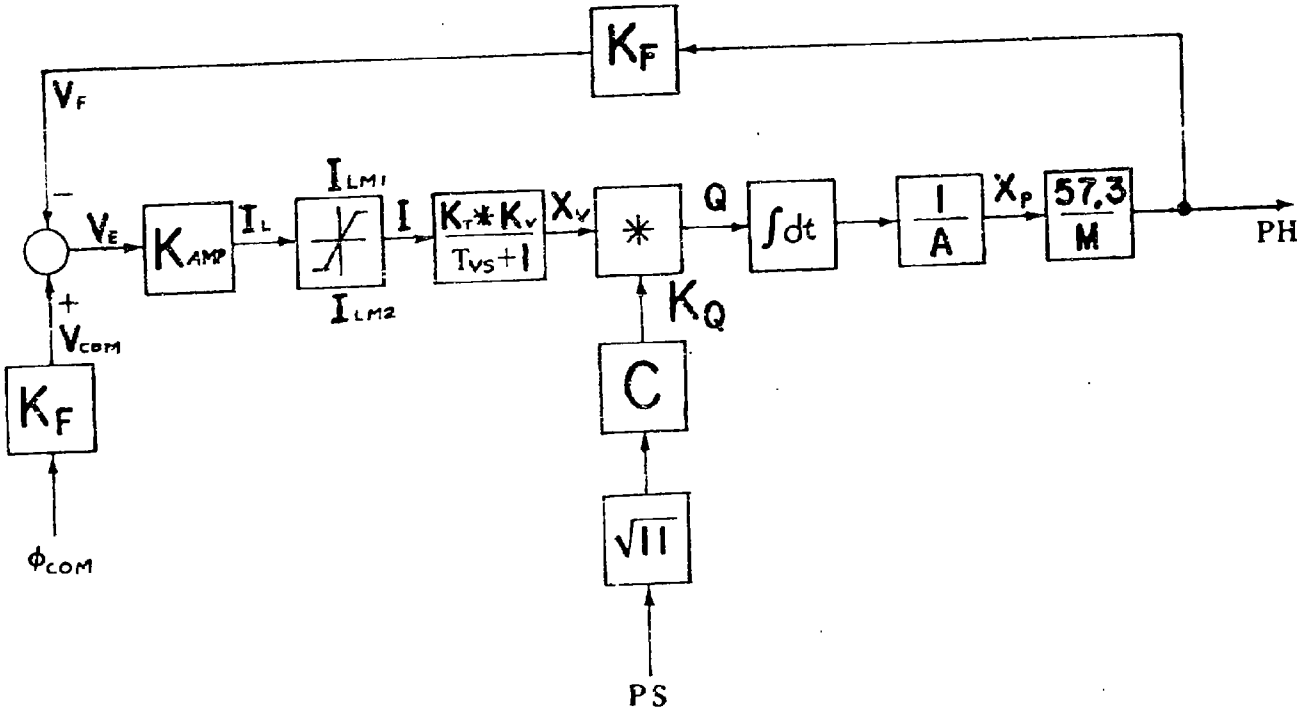



FIGURE 6.104-2
BLOCK DIAGRAM OF SSME - ENGINE CONTROL ACTUATOR

$QL = XL*SQRT(P1-P2)$

$Q(L1) = QC + QL$

$Q(L1) = (XC + XL)*SQRT(P1-P2)$

$$\frac{Q(L1)^2}{(XC+XL)^2} = (C1-C2)-Q(L1)*(Z1+Z2)$$

Solving for Q(L1) by putting in quadratic form

$Q(L1) = (SQRT(ZT**2+4.0*ABS(CDIFF)/XCL)-ZT)*XCL/2.0$

where:

$ZT = Z(L1)+Z(L2)$

$CDIFF = C1 - C2$

$XCL = (XC+XL)^2$

6.104.2 Assumptions

The model does not include static and dynamic friction. To save computing time a common hinge moment arm has been used for translating force to hinge moment and actuator velocity to output rotational velocity.

6.104.3 Computation Method

This program uses Tustin's method of integration and is arranged such that each differential is evaluated using the latest value of the previous integral, in a series calculation, rather than the more usual parallel integration methods, where the integrals are effectively evaluated simultaneously using previous values to evaluate the derivatives.

The use of the series integration is difficult to justify except by stating that it seems to work better than the other method under certain conditions.

1000 Section

This section adjusts RHØ dependent variables for the input temperature.

1500 Section

This section is used to return values for the steady state calculation so that the system state conditions can be calculated. It is assumed that the actuator is at zero velocity, hence the only steady state flow is that due to the first stage servo valve leakage. The first stage impedance 1.0/(DT(KQL)**2 is added to the $Q^2$ value in PQLEG (INEL,8), and the pressure drop DT(KQ)*XCL is subtracted from the inlet pressure PQLEG(INEL,11).

2000 Section

This section is used to initialize the variables to their steady state values. The variables are zeroed since to initialize the variables at any condition other than zero velocity would be very difficult.

3000 Section

The transient section of the program is coded to follow the flow path of the model diagram, starting with the input position command DT(KCOM). DT(KCOM), is varied by the guidance and control subroutine CAD98 or its equivalent in the same way as it is in the actual vehicle, with the value being updated at 20 millisecond intervals. The CAD98 subroutine uses the latest value of actuator position feedback voltage DT(KPH) in its calculations as an indication of the actuator position.

The interface between this subroutine ACD_14 and the guidance and control subroutine CAD98 is via the following variables DT(1) and DT(2).

| Variable | Description | Dimensions |
|---|---|---|
| DT(KCOM) (Calculated by CAD98) | Input position command | Volts |
| DT(KPH) (Calculated by ACT104) | Actuator position | Volts |

The line pressures and flows are returned via the P and Q arrays using the address L(1) and L(2) for the supply and return connections respectively.

6.104.4 APPROXIMATIONS

None.

6.104.5 LIMITATION

The assumption of "no load" on the actuator may give the actuator a higher response than would actually occur under a high load condition.

6.104.6 Variable Names

| Variable | Description | Dimensions |
|---|---|---|
| AMPKT | Constant (KAMP*KT*KV) | -- |
| CDIFF | C(L1)-C(L2) | PSI |
| DELTO2 | DELT/2 | SEC |
| D(KAMPI) | Constant (57.3/(Effective Moment Arm* Piston Area)) | $DEG/IN^3$ |
| D(KC) | Servovalve Gain Constant for 100°F | -- |
| DT(KCOM) | Position Command | VOLTS |
| DT(KCT) | Servovalve Gain Constant for Run Temperature | -- |
| D(KF) | Position Transducer Gain | V/DEG. |
| DT(KPH) | Actuator Position Feedback Voltage | VOLTS |
| DT(KQ) | Actuator Flow | CIS |
| DT(KQI) | Actuator Displacement | $IN^3$ |
| DT(KQL) | Old Value of Actuator Flow | CIS |
| DT(KXLK) | Leakage Leg Constant for Run Temperature | -- |
| L1 | Dummy Variable for L(1) | -- |
| L2 | Dummy Variable for L(2) | -- |
| VE | Difference Between Command and Valve Position | VOLTS |
| VILM | Constant (ILM/KAMP) | -- |
| XC | Flow Constant for Actuator Leg | -- |
| XCL | Constant (Sum of Flow and Leakage Leg Constants Squared) | -- |
| XLK | Leakage Leg Constant for 100°F | -- |

| | | |
|---|---|---|
| ZT | Sum of L1 and L2 Characteristic Impedance | PSI |
| DT(KLAG) | Integration Constant | -- |
| D(LAG) | First Order Lag Time Constant | sec |

6.104.7 Subroutine Listing

```
      SUBROUTINE ACT104 (D,DT,DD,L)
C *** REVISED NOV 3,1975 ***
C *** SHUTTLE SSME ENCINE CONTROL ACTUATOR MODEL
      DOUBLE PRECISION DD
      DIMENSION D(1),DT(1),DD(1),L(1)
      COMMON NTELPL,NTOLPL,IPT,IPOINT,NPTS,INEL,KNEL,
     1 NTOPL,MLPLT(61,3),PQLEG(90,12),LCS(90,10),ILEG(1400),
     2 PN(90),QN(90)
      COMMON/SUB/PARM(150,9),PM(1500),QM(1500),P(300),Q(300),C(300)
     1,Z(300),RHO(20),S2ORHO(20),VISC(20),BULK(20),TEMP(20),PVAP(20)
     2,ATPRES,T,DELT,TFINAL,PLTDEL,PI,TITLE(20),LEGN,ICON
     3,KTEMP(99),LSTART(150),NLPT(150),LTYPE(99),NC(99),INX,INZ
     4,INV,ISTEP,NLINE,NEL,IND,IENTR,MNLINE,MNEL,MNLEG,MNNODE,MNPLOT
     5,MNLPTS,MDS
C *** D ARRAY VARIABLES
      DATA KF/1/,KC/2/,KAMPI/3/,LAG/4/
C *** DT ARRAY VARIABLES
      DATA KCOM/1/,KPH/2/,KQL/3/,KQ/4/,KQI/5/,KXLK/6/,KCT/7/
     1 ,KLAG/8/,KCOMS/11/,KCOMN/12/,KCOML/13/
      DATA AMPKT/.08688/,VILM/.184162/,XLK/.02108/
C     AMPKT=KAMP*KT*KV,VILM=ILM/KAMP
C     XLK=1.155*(1/SQRT(3000)),  KAMPI=57.3/(A*M)
      IF(IENTR) 1000,2000,3000
 1000 CONTINUE
      IF (INEL.NE.0) GO TO 1500
      XCL=S2ORHO(KTEMP(IND))/SQRT(2./7.82E-5)
      DT(KXLK)=XLK*XCL
      DT(KCT)=D(KC)*XCL
      DT(KCOM)=0.0
      CALL TUSTIN(2,D(LAG),DT(KLAG),DELT)
      RETURN
 1500 XCL=1.0/DT(KXLK)**2
      PQLEG(INEL,8)=PQLEG(INEL,8)+XCL
      DT(KQ)=PQLEG(INEL,1)**2*PQLEG(INEL,2)
      PQLEG(INEL,11)=PQLEG(INEL,11)-DT(KQ)*XCL
      RETURN
 2000 CONTINUE
      DT(KQ)=0.0
      DT(KPH)=DT(KCOM)
      DT(KQI)=DT(KPH)/D(KAMPI)
      DELTO2=DELT/2.0
      DT(KCOML)=DT(KCOM)
      DT(KCOMS)=DT(KCOM)
      RETURN
 3000 CONTINUE
      L1=L(1)
      L2=L(2)
      ZT=Z(L1)+Z(L2)
      DT(KCOMN)=DT(KCOM)
      DT(KCOML)=DYNAM(DT(KCOMS),DT(KCOML),DT(KLAG))
```

6.104-7 (Continued)

```
      DT(KCOMS)=DT(KCOMN)
      VE=D(KF)*(DT(KCOML)-DT(KPH))
      CDIFF=C(L1)-C(L2)
      IF(ABS(VE).GT.VILM) VE=SIGN(VILM,VE)
      XC=VE*AMPKT*DT(KCT)
      XCL=(DT(KXLK)+ABS(XC))**2
      Q(L1)=(SQRT(ZT**2+4.0*ABS(CDIFF)/XCL)-ZT)*XCL/2.0
      Q(L1)=SIGN(Q(L1),CDIFF)
      P(L1)=C(L1)-Q(L1)*Z(L1)
      Q(L2)=-Q(L1)
      P(L2)=C(L2)-Q(L2)*Z(L2)
      DT(KQL)=DT(KQ)
      DT(KQ)=XC*SQRABS(P(L1)-P(L2))
      DT(KQI)=DT(KQI)+(DT(KQL)+DT(KQ))*DELTO2
      DT(KPH)=DT(KQI)*D(KAMPI)
      RETURN
      END
```

## 6.105 SUBROUTINE ACT105

Subroutine ACT105 models the shuttle thrust vector control (TVC) actuators, the layout of which is shown in Figure 6.105-1. The TVC actuators operate from three pairs of hydraulic supply and return lines. The supply and return used is selected by a switching valve module as shown in Figure 6.105-2. For the purpose of modeling the switching valve module is considered to be separated from the actuator by a pair of pseudo lines. The input command and hinge moments are supplied by a Guidance and Control subroutine which updates the values at each sample time interval of the guidance system which is .04 seconds.

### 6.105.1 Math Model

This TVC math model was derived from that proposed by Rockwell International. The model shown in Figure 6.105-1 uses the position command signal VC to generate commands to the servo valve torque motor. The applied torque deflects the servo valve causing differential flow to the secondary actuator piston. The subsequent displacement of the secondary actuator, which is attached to the servo valve by a wire feedback, and to the main ram by mechanical linkage, continues until the valve is returned to the equilibrium flow position. Changes in the position of the secondary actuator are transmitted to the power spool through the summing linkage. The power spool displacement, XPS, generates differential flow to the main ram causing actuator deflection. The deflection is sensed as XFB and is feed back to the servo valve as a torque to close the loop. Demand flow on the hydraulic system is generated by the sum of the flow to the main ram with the flow to four secondary actuators plus first stage leakage.

# SSME TVC ACTUATION CONTROL SYSTEM MODEL

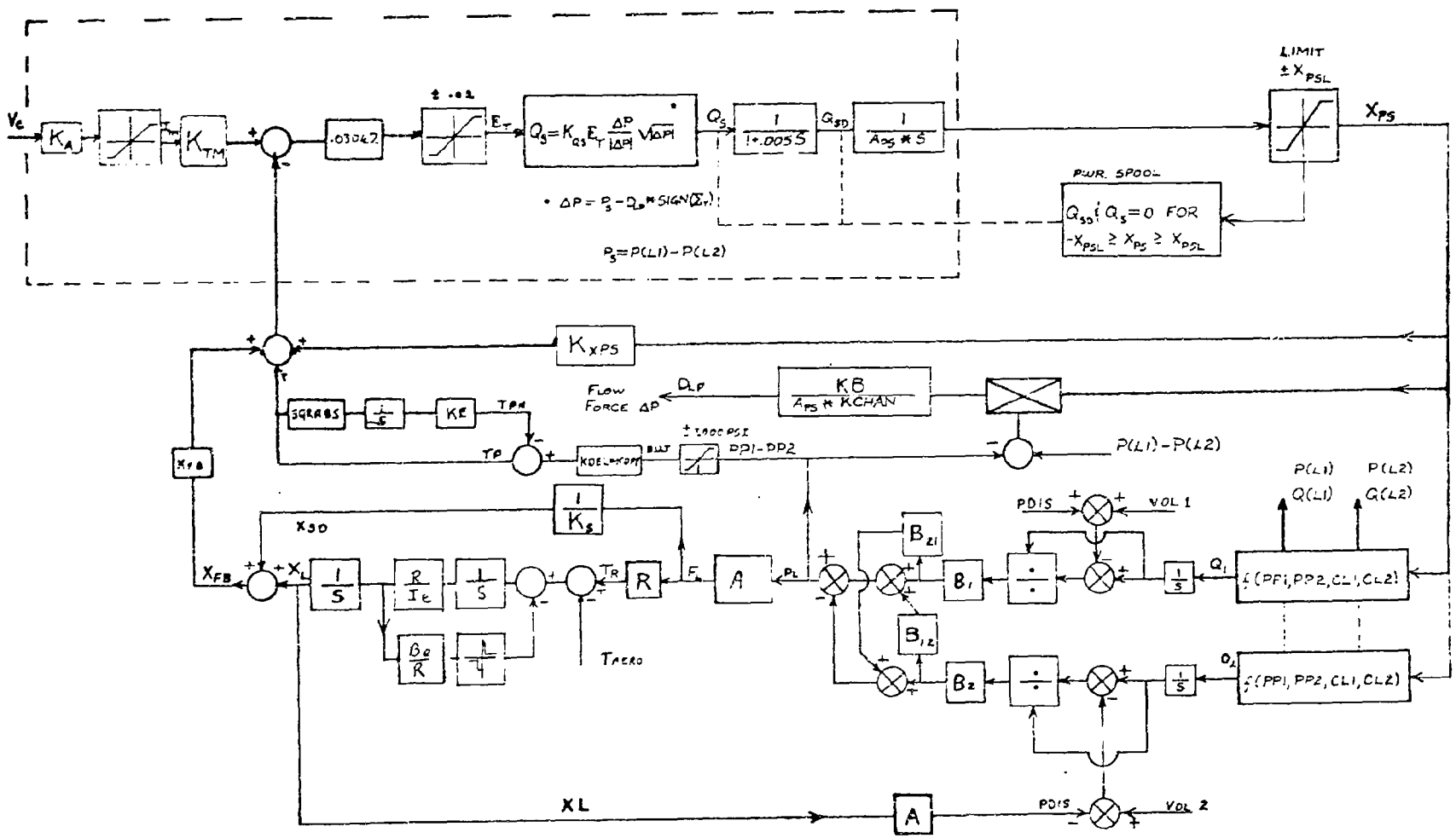



FIGURE 6.105-1

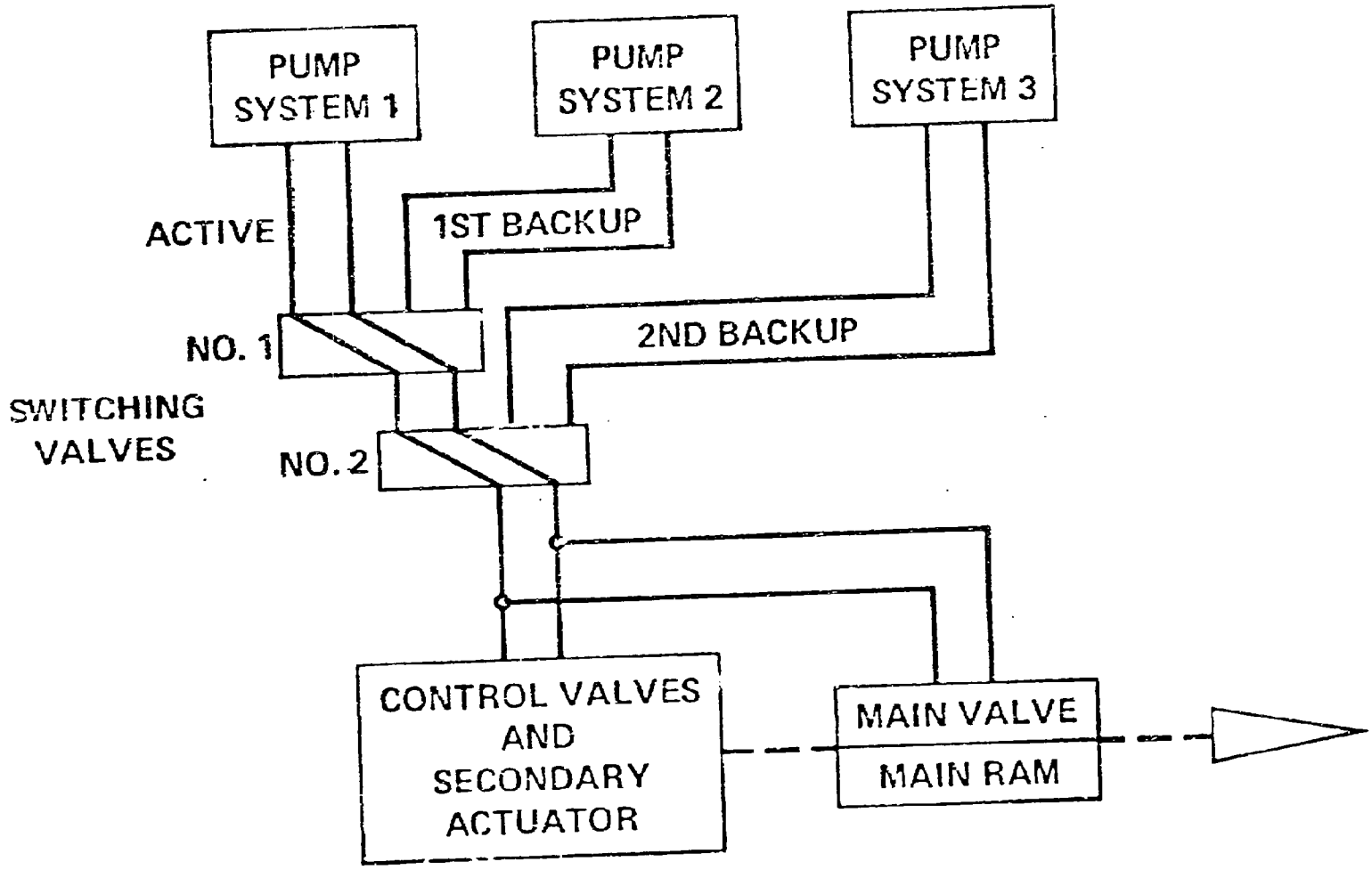



FIGURE 6.105-2

6.103.2 Assumptions

The model is limited to simulating one channel of the secondary actuator and it is assumed that the other channels if operative will give identical outputs, in terms of available force at the main control valve.

The model of the surface actuator includes static and dynamic friction. To save computing time a common hinge moment arm has been used for translating force to hinge moment and TVC velocity to actuator velocity.

The actuator velocity conversion is not quite correct since the actuator position is the sum of the TVC deflection and structural deflection.

6.103.3 Computation Method

The method used for computation is based on the previous program developed for the elevon actuator.

This program uses Tustin's method of integration and is arranged such that each differential is evaluated using the latest value of the previous integral, in a series calculation, rather than the more usual parallel integration methods, where the integrals are effectively evaluated simultaneously using previous values to evaluate the derivatives.

Predicted values are used to improve the calculation where the use of a previous value would incur a significant error. The computation follows the usual component subroutine layout shown in Figure 6.1-1 on page 6.1-2.

1000 Section.

This section is used to initialize variables used by the program. For the most part, the integration variables are zeroed and the Tustin

subroutine is called to initialize the integration constants.

1500 Section

This section is used to return values for the steady state calculation so that the system state conditions can be calculated. It is assumed that the secondary and power actuators are at zero velocity, hence the only steady state flow is that due to the first stage servo valve leakage. The first stage impedance 1.0/(D(KQLOSS)*D(KCHAN))**2 is added to the $Q^2$ value in PQLEG (INEL,8), and the pressure drop DELP is subtracted from the inlet pressure PQLEG(INEL,11).

2000 Section

This section is used to initialize the variables to their steady state values. The only variables involved for the subroutine are the actuator pressures which are set at the mean of the inlet and return pressures. The effect of the internal load is to vary the pressures either side of this mean. Most of the other variables are zeroed since to initialize the TVC actuator variables at any condition other than zero velocity would be very difficult.

3000 Section

The transient section of the program is coded to follow the flow path of the model diagram, starting with the input position command DT(VC). DT(VC), is varied by the guidance and control subroutine or its equivalent in the same way as it is in the actual vehicle, with the value being updated at 40 millisecond intervals. The six degree of freedom (SDF) subroutine uses the latest value of actuator position feedback DT(XFB) in its calculations as an indication of the actuator position. The SDF program also calculates a hinge moment at zero actuator position and the slope of hinge moment versus position.

ACT105 uses the zero and slope values, ' interpolate values of TAERO until those values are updated by the next guidance and control computation. The interface between this subroutine ACT105 and the guidance and control subroutine is via the following variables DT(1) thru DT(5).

| Variable | Description | Dimensions |
|---|---|---|
| DT(VC) (Calculated by SDF) | Input position command | Volts |
| DT(XFB) (Calculated by ACT105) | TVC actuator position | In |
| DT(KAERO) (Calculated by SDF) | TVC hinge moment at zero stroke | In lbs |
| DT(KAEROP) (Calculated by SDF) | TVC hinge moment slope | In lbs/in |

The line pressures and flows are returned via the P and Q arrays using the address L(1) - L(6) for the supply and return connections.

The input command DT(VC) is feed via a first order lag to the junction, where it is summed with a predicted feedback value.

The error is passed through a limiter to the servo valve which converts it to torque. The torque is summed with the spring feedback from the secondary actuator and the pressure feedback across the secondary actuator piston.

The flow calculation uses the supply and pressures from the previous time step to calculate a new flow. The flow is passed through a simple lag to denote the servo valve time delay, and is then integrated to obtain a new secondary actuator (main control valve) position.

The actuator position is tested against its limits, if it is at a limit then the secondary actuator flow is set to zero.

The flow through the main control valve is calculated using the predicted actuator cavity pressure and the valve orifice equations which are

solved in conjunction with the line characteristic equation.

The line characteristic pressure is reduced by an amount equal to the secondary actuator flow plus the leakage flow times the characteristic impedance. This method avoids the need for a simultaneous solution which would have required a more expensive iterative solution.

The computed flows, which are positive when flowing into the actuator cavities, are then integrated. The integrated flows are summed with the actuator displacement flow and the difference is used to compute a new actuator cavity pressure.

The cavity pressure computation is complicated by the actuator deflection due to differential pressure acting on the structural stiffness.

If solved sequentially, an arithmetic loop is created which creates computational instability. This was avoided by reformulating the loop to give a direct solution for actuator cavity pressures, with the structural stiffness being integrated into the calculation.

The remaining part of the computation follows the block diagram.

SWITCHING VALVE SECTION

The final section of the TVC subroutine describes the operation of the switching valve or selector valve as it is sometimes called. The valve model is set up as an independent model to that of the TVC and is connected to the actuator via pseudo lines, which provides the numerical isolation required.

Because of the TVC package dimensions, these pseudo lines do have some factual basis in that there is a significant flow path length between the connections and the main control valve. The valve pressure loss in each flow path is assumed to be contributed by two separate orifice coefficients, due to the inlet holes through the sleeve, and due to the valve metering orifices.

There are four .375" dia inlet holes. The metering orifice has four slots, 0.5" wide 0.2" long. This configuration is repeated for each flow path, in the primary and secondary sections of the valve.

The pressure drop for each orifice will be

$= (Q/(AREA*CD))^2*RHO/2$

$= 8*10^{-5}/.65^2/2 * Q^2/AREA^2$

for Q = 150 CIS

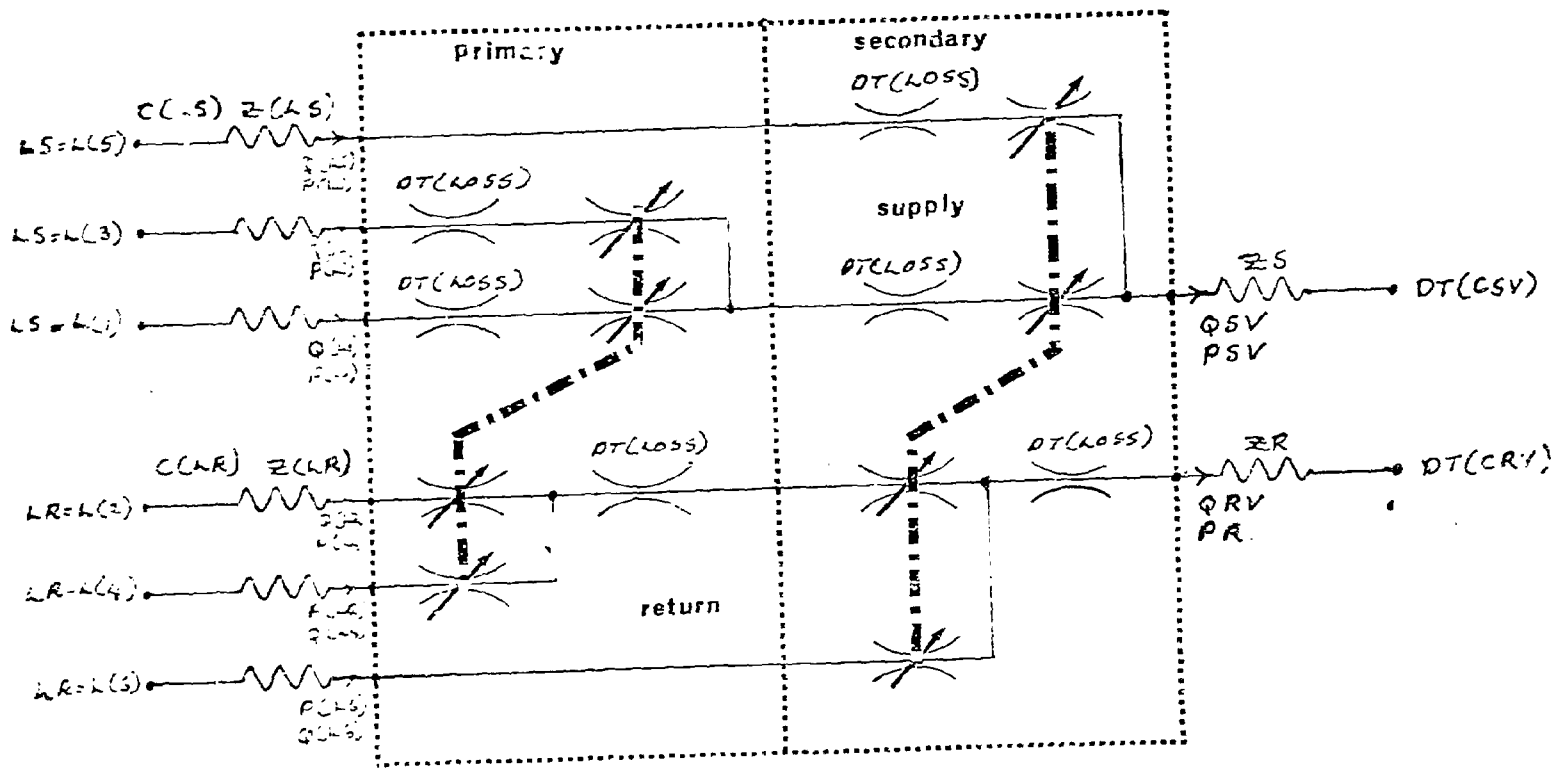
$= 2.13/AREA^2$

= 13.31 psi for metering orifice

= 10.9 psi for the four holes

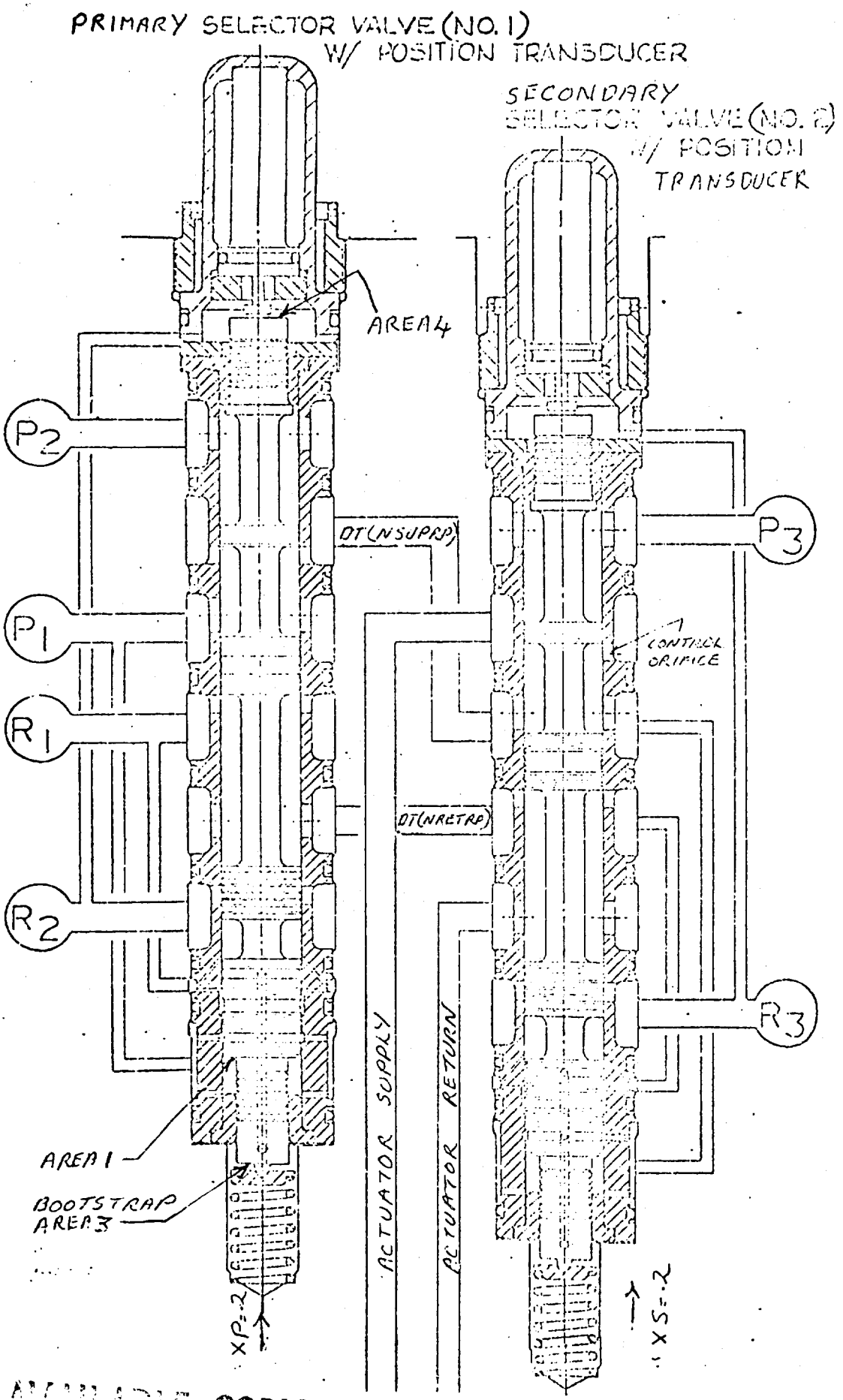
= 26.22 for each valve flow path

= 96.99 psi for supply and return for the primary system




TVC SWITCHING VALVE MODULE

FIGURE 6.105-3




SWITCHING VALVE LAYOUT
FIGURE 6.105-4

The pressure drop across the valve holes

$\Delta P = Q^2 * DT(LOSS)$

where the loss coefficient

$DT(LOSS) = RHO/(2*.65^2*AREA^2)$

$= .000485 @ 130°F$

or $= RHO * 6.06$

This combination of orifices is only significant when the valves are fully open, the predominant loss occurs when the valve displacement is less than 0.1" during the switching cycle.

It is under switching conditions that the actual stability of the model becomes significant. When the start up conditions are normal, switching occurs from #1 to #2, and then #2 to #3. Under these conditions the bootstrap arrangement will give a positive or stable switching characteristic. The real stability problem occurs when the system is switching from #3 to #2 or #2 to #1 when the pump is being repressurized. Fortunately this particular situation will occur at low or zero flows. The high water hammer pressures, associated with valve closure when there is a high flow demand, will probably not occur during reverse switching.

Math Model

Valve Positions - The valve positions are predicted based on the positions at the last two time steps.

PXP = DT(XP)*2-DT(XPP)

Predicted values are used to improve the computation accuracy during switching, no effort is made to correct the computations when the final positions are computed. The choice of a predicted position computation was based on the need to use a valve position nearer to the

true value, but still avoiding the predictor - corrector type of computation, which leads to stability problems.

The computation would be improved by the use of a very small time step, but it is doubtful if there would be any significant improvement in the overall system accuracy.

The initial position of the valve is determined from the system indicator INV which indicates the conditions of the three systems. This is translated into the local system indicated (ISYS).

where

L(ISYS) = 1 primary system active

L(ISYS) = 2 first back-up active primary failed

L(ISYS) = 3 second back-up active primary and first back-up failed

The valve positions at the L(ISYS) = 1 condition are both +0.2 inches. The positions for L(ISYS) = 3 are -0.2 inches.

There are two valves in series for the No. 1 and No. 2 systems, and just one for the No. 3 system. The two valves are normally called No. 1 and No. 2 selector valves, but to avoid confusion with the line connections, we have used primary and secondary as the valve names. The two valves are almost identical except for the end areas, which are varied to provide some difference in the switching pressures. The basic philosophy is to make the primary valve move and switch before the secondary valve responds.

The achievement of this will depend on the transient pressures that occur during switching.

The reference pressures DT(NSUPRP) and DT(NRETRP) are the supply and return pressures at the outlet side of the primary valve. These reference pressures control the switching of the secondary valves.

Springs are used to position the valves on start up.

At the time of writing, there is no data on the damping effects of the small diameter holes connecting the return or pressure ports and the bootstrap area end of the valve. Scaling the drawings available show the hole to be approximately .1" diameter and about 1.2" long.

The max valve velocity is expected to be about 80 in/sec which would give a flow of approximately 21 CIS. Using the diameter and length we can generate an equivalent orifice coefficient for the flow restriction. A plain orifice of .1" dia gives a coeff of 1.25 PSI/$(CIS)^2$ plus a smaller contribution from the pole length gives a total of 1.4 PSI/$CIS^2$ At the 21 CIS flow this will give a pressure drop of 617 PSI which will have a significant effect on the bootstrap pressure.

In going from the initial position of PXP=.2" to the switches position of PXP= -.2", oil is pumped out of the bootstrap cavity, initially into the P1 supply and later when PXP <.1", it is pumped into the R1 return. The bootstrap reference pressure PBST is switched when PXP <.1 from PBST = P(L(1)) to PBST = P(L(2)).

The spring pre-load is 5 lbs @ XS = .2 and the spring rate is 50 lbs/in.

This makes the spring load

= ZSPP-DDISVP*SSPP

where ZSPP = 5 + .2*50

= 15 lbs.

and SSPP = 50 lbs/in

The predicted valve displacement PDISVP is used for the calculation. In calculating the valve velocity, the inertia and damping forces are considered. The inertial forces are normally small, but very high accelerations are required to achieve switching in 10 Milliseconds. With a stroke of .4" and assuming constant acceleration

$$ACCN = .4*2*10^4 = 8000 \text{ in/sec}^2$$

$$FORCE = .002x8000 = 16 \text{ lbs.}$$

6.105.5 **Approximations**

The servo valve first order lag of 0.005 seconds is an approximation.

6.105.5 Limitations

The TVC module can be used to simulate the overall response of the actuator and surface to commands vrom the SDF program. Because of the simplification in the area of the secondary actuator the small signal response is better than can be expected in practice. The simulation of the switching valve is not ideal, because of the very fast response of these valves. To improve the model would require a much smaller time step. Should these limitations become a problem, a smaller time step can be used with some minor program changes and of course an increase in cost.

6.105.6 Variable Names

| Variable | Description | E3P* | Dimensions |
|---|---|---|---|
| APS | Working Area of Power Spool | | $in^2$ |
| D(AR) | Effective Ram Actuator Area | 20.03 | $in^2$ |
| D(BE) | Effective TVC Damping Coefficient | 1000. | $\frac{16\text{-sec}}{in}$ |
| EPS7 | Net Error Torque | | in-lb |
| EYE | Input Current | | ma |
| EYEL | Servo Amplifier Saturation Limit | 50.0 | ma |

| Variable | Description | E3P* | Dimensions |
|---|---|---|---|
| FDRIVE | Net Ram Force | | lb |
| FF | Friction Force on Ram | | lb |
| FL | Ram Force Applied to Load | | lb |
| D(IC) | Engine Moment of Inertia About Hinge Line | 48400 | $in\text{-}lb\text{-}sec^2$ |
| KA | Servo Amplifier Gain | 11. | ma/v |
| KB | Bernoulli Force Coefficie | .702 | in |
| D(KCHAN) | Number of Operative Channels | | -- |
| KDEL | Dynamic Pressure Feedback Gain | .2 | -- |
| KDPF | Dynamic Pressure Feedback Gain | .00062 | in lb/PSI |
| KE | Dynamic Pressure Feedback Integrator Gain | 59.4 | -- |
| D(KFB) | Actuator Feedback Gain | .367 | $\frac{in\text{-}lb}{in}$ |
| KQLOSS | First Stage Leakage Coefficient | .0237 | CIS/PSI |
| D(KQPS) | Power Spool Flow Gain | 121.0 | CIS/lb |
| KQS | Secondary Valve Flow Gain | 3.743 | $in^3/(SEC\ lb^3)$ |
| D(KS) | Structural Stiffness | 226000. | lb/in |
| KTM | Torque Motor Gain | .04 | $\frac{in\text{-}lb}{ma}$ |
| D(KXPS) | Power Spool Feedback Gain | 5.82 | $\frac{in\text{-}lb}{in}$ |
| D(MA0) | First Moment Arm Constant | 27.79 | in |
| D(MA1) | Second Moment Arm Constant | -.3499 | in/in |
| D(MA2) | Third Moment Arm Constant | -.00787 | $in/in^2$ |
| VLVOL | Power Spool Overlap | .0006 | in |
| D(VOL1) | #1 Cavity Volume at Mid-stroke | 122.0 | $in^3$ |
| D(VOL2) | #2 Cavity Volume at Mid-stroke | 115.0 | $in^3$ |

| Variable | Description | E3P* | Dimensions |
|---|---|---|---|
| XPSL | Power Spool Displacement Limit | .05 | in |

* Engine No. 3 Pitch TVC.

| | | |
|---|---|---|
| A1PS1 | Primary valve AREA1 | $IN^2$ |
| A1SS1 | Secondary valve AREA1 | |
| A2PS2 | Primary valve AREA2 | " |
| A2SS2 | Secondary valve AREA2 | |
| A3PB | Primary valve AREA3 | " |
| A3SB | Secondary valve AREA3 | |
| A4PR2 | Primary valve AREA4 | " |
| A4SR2 | Secondary valve AREA4 | |
| BBVP | Damping coeff. | - |
| BVP | Damping & inertial coeff. | |
| COE | Combined orifice coef. | |
| DT(COEFVP) | Metering orifice coef. | $CIS/(PSI)^{1/2}/IN$ |
| DT(COEFVS) | Metering orifice coef. | " |
| COEP | Primary combined orifice coef. | $PSI/(CIS)^2$ |
| COES | Secondary combined orifice coef. | " |
| DELT02 | DELT 2.0 | |
| DINP | Spool inertia coef | |

| | | |
|---|---|---|
| DORFCP | Spool damping orifice coef | |
| DORFCS | Spool damping orifice coef | |
| I | Do loop counter | |
| IA | Actuator line address | |
| IL | Line address | |
| IN | Connection number | |
| IPASS | Calculation counter | |
| L(ISYS) | Active system indicator | |
| DT(LEAK) | Leakage coeff when valves are closed | |
| DT(LOSS) | Orifice coef of inlet holes | $PSI/CIS^2$ |
| DT(LSWACT) | Combined coef oc metering orifice and hose coef | $PSI/CIS^2$ |
| N | Connection counter | |
| NDISVP | Integer position indicator | |
| NDISVS | Integer position indicator | |
| DT(NRETRP) | Return reference pressure | |
| NS | System indicator | |
| DT(NSUPRP) | Supply reference pressure | |
| OLP | Primary valve overlap | IN |
| OLS | Secondary valve overlap | IN |
| PBST | Bootstrap reference pressure | PSI |
| PDELTA | Steady state pressure drop | PSI |
| PXP | Predicted value of XP | IN |
| PXS | Predicted value of XS | IN |
| PSPL | Pressure exerted by spool forces on bootstrap volume | PSI |
| QS | Flow through valve | |

| | | |
|---|---|---|
| SQBVP | $BUP^2$ | |
| SSPP | Spring slope or rate | LBS/IN |
| DT(VELP) | Primary spool velocity | IN/SEC |
| VELPO | Previous value of DT(VELP) | IN/SEC |
| DT(VELS) | Secondary spool velocity | IN/SEC |
| VELSO | Previous value DT(VELS) | IN/SEC |
| DT(XP) | Primary valve position | IN |
| DT(XPP) | Previous value of XP | IN |
| DT(XS) | Secondary value position | IN |
| DT(XSP) | Previous value of XS | IN |
| ZA | Temporary impedance | PSI/CIS |
| ZSPP | Spring load at zero position | LBS |

6.105.7 Subroutine Listing

```
      SUBROUTINE ACT105 (D,DT,DD,L)
C
C       *****REVISED NOVEMBER 1975 ****
C       SHUTTLE SSME TVC ACTUATOR MODEL
C
      DOUBLE PRECISION DD
      COMMON NTELPL,NTOLPL,IPT,IPOINT,NPTS,INEL,KNEL,NTOPL,NLPLT(61,3),
     1 PQLEG(90,12),LCS(90,10),ILEG(1400),PN(90),QN(90)
      COMMON/SUB/PARM(150,9),PM(1500),QM(1500),P(300),Q(300),C(300)
     1,Z(300),RHO(20),S2ORHO(20),VISC(20),BULK(20),TEMP(20),PVAP(20)
     2,ATPRES,T,DELT,TFINAL,PLTDEL,PI,TITLE(20),LEGN,ICON
     3,KTEMP(99),LSTART(150),NLPT(150),LTYPE(99),NC(99),INX,INZ
     4,INV,ISTEP,NLINE,NLL,IND,IENTR,MNLINE,MNEL,MNLEG,MNNODE,MNPLOT
     5,MNLPTS,MDS
      REAL KA,KTM,KQS,KDEL,KDPF,KE,KQLOSS
      INTEGER AR,BE,VC,P1,XPS,
     2 TXPS,PS,XYQS,QS,XQI,
     3 XYXPSD,XPSD,XPSI,Q1,Q2,PP1,PP2,XYTE,TE,DELEDI,
     4 XYDLED,VXL,RFXL,XL,XSD,PL,VPI,XFB,
     5 VOL1,VOL2,VXLP,XFBP,P1P
     6 ,TP,TPN,DPFB,DPFBP,FIBETA,XLP,XIQ1,XIQ2,VCS,VCN,VCL
     7 ,CRA,CSA,CRV,CSV
     8 ,XP,XPP,VELP,XS,XSP,VELS,COEFVP,COEFVS
      DIMENSION D(16),DT(100),DD(1),L(10)
C
C *** D ARRAY VARIABLES
C
      DATA KCHAN/1/,KB/2/,KQPS/3/,AR/4/,KS/5/,IE/6/,BE/7/,KFB/8/,
     2 VOL1/9/,VOL2/10/,KXPS/11/,MA0/12/,MA1/13/,MA2/14/
C
C *** DT ARRAY VARIBLES
C
      DATA VC/1/,KAERO/4/,KAEROP/5/,
     1 P1/6/,XPS/7/,TXPS/8/,PS/9/,XYQS/ 0/,QS/11/,
     2 XQI/12/,NQLOSS/14/,NQS/15/,NQPS /16/,NCAV/17/,
     3 IBETA/18/,XYXPSD/19/,XPSD/20/,XPSI/21/,Q1/22/,
     4 Q2/23/,PP1/24/,PP2/25/,XYTE/26/,TE/27/,DELEDI/28/,
     5 XYDLED/29/,VXL/30/,RFXL/31/,XL/32/,XSD/33/,
     6 PL/35/,VPI/36/,XFB/38/,KINT/39/,KFLOW/42/,VXLP/45/,XFBP/46/,
     1 P1P/47/,TPN/48/,DPFB/49/,DPFBP/50/,FIBETA/51/,
     2 XLP/52/,XIQ1/53/,XIQ2/54/,KCOM/55/,VCS/58/,VCN/59/,VCL/60/,
     3 XP/61/,XPP/62/,VELP/63/,XS/64/,XSP/65/,VELS/66/,
     4 COEFVP/67/,COEFVS/68/,LOSS/69/,LEAK/70/,LSWACT/71/,
     5 NSUPRP/72/,NRETRP/73/,ISYS/8/
     6 ,CSA/74/,CSV/75/,CRA/76/,CRV/77/,TP/78/
C
C *** INITIALIZE CONSTANTS COMMON TO ALL ACTUATORS ***
      DATA VLVOL/.0006/,KDEL/.2/,KDPF/.00062/,KE/59.4/,
     1 DORFCP/.5/,DORFCS/.5/,OLP/.005/,OLS/.005/,ZSPP/15./,SSPP/50./,
     2 A1PS1/.338/,A2PS2/.274/,A3PB/.263/,A4PR2/.327/,
```

6.105.7 (Continued

```
     3 A1SS1/.338/,A2SS2/.230/,A3SB/.263/,A4SR2/.382/,
     4 KA/11./,KTM/.04/,EYEL/50./,KQS/3.743/,
     5 APS/.393/,XPSL/.05/,KQLOSS/.0237/,ZS/6.5/,ZR/6.5/
      IF (IENTR) 1000,2000,3000
 1000 IF (INEL.NE.0) GO TO 1500
C
C     INITIALIZATION
C
      DO 1001 I=1,100
 1001 DT(I)=0.0
C     CORRECT INPUT DATA FOR FLUID TEMPERATURE
      I=KTEMP(IND)
      DT(NQS)=S2ORHO(I)/SQRT(2./7.82E-5)
      DT(NQPS)=D(KQPS)*DT(NQS)*SQRT(2.)
      DT(NQLOSS)=KQLOSS*DT(NQS)
      DT(NQS)=KQS*DT(NQS)
      DT(IBETA)=BULK(I)
      DT(FIBETA)=DT(IBETA)*D(AR)*D(AR)/D(KS)
      CALL TUSTIN(1,1.,DT(KINT),DELT)
      CALL TUSTIN(2,.005,DT(KFLOW),DELT)
      CALL TUSTIN(2,.01872,DT(KCOM),DELT)
C
C     SET UP SWITCHING VALVE CONSTANTS
      DINP=.75/(12*32.2*DELT)
      DELTO2=DELT/2.0
      BBVP=(DORFCP/A3PB)**2
      BVP=BBVP*(.1+DINP)/2.0
      SQBVP=BVP**2
      BBVS=(DORFCS/A3SB)**2
      BVS=BBVS*(.1+DINP)/2.0
      SQBVS=BVS**2
      DT(COEFVP)=2.0*.65*S2ORHO(1)
      DT(COEFVS)=2.0*.65*S2ORHO(I)
      DT(LEAK)=1000000.
C
      L(ISYS)=1
      IF(INV) 1220,1250,1200
 1200 IF(L(9).EQ.INV) GO TO 1250
      IF(L(10).EQ.INV) GO TO 1230
      L(ISYS)=3
      GO TO 1250
 1220 IF(L(9).NE.-INV) GO TO 1250
 1230 L(ISYS)=2
 1250 CONTINUE
      DT(LOSS)=RHO(1)*6.06
      DT(LSWACT)=1.0/(DT(COEFVS)*.2)**2+DT(LOSS)
      IF(L(ISYS).GT.2) GO TO 1260
      DT(LSWACT)=DT(LSWACT)+1.0/(DT(COEFVP)*.2)**2+DT(LOSS)
 1260 CONTINUE
      RETURN
```

6.105.7 (Continued)

```
C
C              --------- STEADY STATE SECTION
C
 1500 CONTINUE
      QLOSS = (1.0/(DT(NQLOSS)*D(KCHAN)))**2
      COES=DT(LEAK)
      IF((KNEL+1)/2.NE.L(ISYS)) GO TO 1600
      DT(QS)=PQLEG(INEL,1)*PQLEG(INEL,2)
      COES=0.0
 1600 QSA=PQLEG(INEL,1)
      PDELTA=PQLEG(INEL,2)*QSA*(QSA*DT(LSWACT)+COES)
      DELP=PQLEG(INEL,2)*QSA**2*QLOSS
      PSA=PQLEG(INEL,11)-PDELTA
      PRA=PSA-DELP
      PQLEG(INEL,11)=PRA-PDELTA
      PQLEG(INEL,6)=PQLEG(INEL,6)+COES*2.0
      PQLEG(INEL,8)=PQLEG(INEL,8)+2.0*DT(LSWACT)+QLOSS
      IF(COES.NE.0.0) GO TO 1700
      DT(PP1)=PSA
      DT(PP2)=PRA
 1700 RETURN
C
C         --------- VARIABLE INITIALIZATION
C
 2000 CONTINUE
      DT(NSUPRP)=DT(PP1)
      DT(NRETRP)=DT(PP2)
      DT(CSA)=DT(PP1)+DT(QS)*ZS
      DT(CSV)=DT(PP1)-DT(QS)*ZS
      DT(CRA)=DT(PP2)-DT(QS)*ZR
      DT(CRV)=DT(PP2)+DT(QS)*ZR
      DT(XFB)=DT(VC)*KA*KTM/D(KFB)
      DT(XFBP)=DT(XFB)
      TAERO = DT(KAERO)+DT(KAEROP)*DT(XFB)
      DT(RFXL) = D(MA0)+D(MA1)*DT(XFB)+D(MA2)*DT(XFB)**2
      DT(XL) = DT(XFB)-(TAERO/DT(RFXL))/D(KS)
      DT(XLP)=DT(XL)
      F1 = TAERO/DT(RFXL)/D(AR)
      DT(PS) = DT(PP1)-DT(PP2)
      DELP = (DT(PP1)+DT(PP2))/2
      DT(PP1) = DELP+F1/2
      DT(PP2) = DELP-F1/2
      DT(PL) = F1
      DT(XIQ1)=(D(VOL1)+DT(XFB)*D(AR))/(1.-DT(PP1)/DT(IBETA))
      DT(XIQ2)=(D(VOL2)-DT(XFB)*D(AR))/(1.-DT(PP2)/DT(IBETA))
      DT(VCL)=DT(VC)
      DT(VCS)=DT(VC)
      DT(DPFB)=DT(PL)*KDEL*KDPF/KE
      DT(XP)=0.2
      DT(XS)=0.2
```

6.105.7 (Continued)

```
      IF(L(ISYS)-2) 2040,2020,2010
 2010 DT(XS)=-.2
      GO TO 2040
 2020 DT(XP)=-.2
 2040 DT(XPP)=DT(XP)
      DT(XSP)=DT(XS)
      DT(QS)=0.0
      RETURN
 3000 CONTINUE
C
C               **** TRANSIENT CALCULATIONS ****
C
C             **** INTIALIZE VARIABLES ****
      RFXLS=DT(RFXL)
      DT(RFXL) = D(MA0)+D(MA1)*DT(XL)+D(MA2)*DT(XL)**2
      RFXLP=2.*DT(RFXL)-RFXLS
      TAERO=DT(KAERO)+DT(KAEROP)*DT(XFBP)
C
C     SECONDARY ACTUATOR
C
C               ----- SERVO AMPLIFIER
C
      DT(VCN)=DT(VC)
      DT(VCL)=DYNAM(DT(VCS),DT(VCL),DT(KCOM))
      DT(VCS)=DT(VCN)
      EYE=DT(VCL)*KA
      IF(ABS(EYE).GT.EYEL) EYE=SIGN(EYEL,EYE)
C
C               ----- TORQUE MOTOR
C
      TXPSS=DT(TXPS)
      DT(TXPS) = DT(XPS)*KXPS
      TXPSP=2.*DT(TXPS)-TXPSS
      TM = EYE*KTM
      EPST=TM-DT(XFBP)*D(KFB)-TXPSP-DT(TP)
      EPST = .03042*EPST
      IF(ABS(EPST).GT.0.02) EPST=SIGN(0.02,EPST)
      DELP=DT(PS)-SIGN(1.0,EPST)*DT(PLP)
      IF(DELP.LT.0.0) DELP=0.0
      DT(QS) = EPST*DT(NQS)*SQRABS(DELP)
C
C           ------ MAIN CONTOL VALVE POSITION
C
      DT(XQI) = DYNAM(DT(XYQS),DT(XQI),DT(KFLOW))
      DT(XPSD) = DT(XQI)/APS
      DT(XPS) = DYNAM(DT(XYXPSD),DT(XPS),DT(KINT))
      CALL XLIMIT(DT(XPS),DT(XPSD),TM,-XPSL,XPSL)
      IF(TM.EQ.0.0) GO TO 60
      DT(QS)=0.0
      DT(XQI)=0.0
```

6.105.7 (Continued)

```
   60 CONTINUE
      DT(XYXPSD)=DT(XPSD)
      DT(XYQS)=DT(QS)
      QLOSS = (ABS(DT(XQI))+DT(NQLOSS)*SQRABS(DT(PS)))*D(KCHAN)
      Q1S=DT(Q1)
      Q2S=DT(Q2)
      NXV=DT(XPS)/VLVOL
      IF(NXV) 130,170,180
C
C     XPS LESS THAN 0
C
  130 XPSS=((DT(XPS)+VLVOL)*DT(NQPS))**2
      SGN=DT(CRA)-DT(PP1)+QLOSS*ZR
      FN2=XPSS*ZR
      ICALC=1
C
C     CALCULATE MAIN CONTROL VALVE FLOWS
C
  140 FLOW=(SQRT(FN2**2+4.0*XPSS*ABS(SGN))-FN2)/2.0
      ZCALC = ICALC
      GO TO (150,160,190,200),ICALC
C
  150 DT(Q1)=SIGN(FLOW,SGN)
      QRA = DT(Q1)-QLOSS
      FN2=XPSS*ZS
      SGN=DT(CSA)-DT(PP2)-QLOSS*ZS
      ICALC=2
      GO TO 140
  160 DT(Q2)=SIGN(FLOW,SGN)
      QSA = DT(Q2)+QLOSS
      GO TO 220
C
C     XPS = 0
C
  170 DT(Q1)=0.
      FLOW=0.0
      QSA = QLOSS
      DT(Q2)=0.
      QRA = -QLOSS
      ICALC=5
      GO TO 220
C
C     XPS GREATER THAN 0
C
  180 XPSS=((DT(XPS)-VLVOL)*DT(NQPS))**2
      FN2=ZS*XPSS
      SGN=DT(CSA)-DT(PP1)-QLOSS*ZS
      ICALC=3
      GO TO 140
  190 DT(Q1)=SIGN(FLOW,SGN)
```

6.105.7 (Continued)

```
      QSA = DT(Q1)+QLOSS
      FN2=ZR*XPSS
      SGN=DT(CRA)-DT(PP2)+QLOSS*ZR
      ICALC=4
      GO TO 140
  200 DT(Q2)=SIGN(FLOW,SGN)
      QRA = DT(Q2)-QLOSS
C
  220 PSA = DT(CSA)-ZS*QSA
      PRA =DT(CRA)-ZR*QRA
      VAP=PVAP(KTEMP(IND))
      IF(PRA.GT.VAP.AND.DT(NCAV).LE.0.0) GO TO 270
      GO TO (230,230,240,240,250),ICALC
  230 FLOW=SQRABS(XPSS*(VAP-DT(PP1)))
      DT(Q1)=FLOW
      GO TO 250
  240 FLOW=SQRABS(XPSS*(VAP-DT(PP2)))
      DT(Q2)=FLOW
  250 QRA=(DT(CRA)-VAP)/ZR
      PRA=VAP
      DT(NCAV)=DT(NCAV)+FLOW-QLOSS-QRA
  270 CONTINUE
      DT(PS) = PSA-PRA
C
C                ----- ACTUATOR CHAMBER PRESSURES
      PFLOW=DT(VXLP)*D(AR)
      PDIS=DT(XLP)*D(AR)
      DT(XIQ1)=DT(XIQ1)+.5*DELT*(DT(Q1)+Q1S)
      DT(XIQ2)=DT(XIQ2)+.5*DELT*(DT(Q2)+Q2S)
      FQ1=DT(FIBETA)/DT(XIQ1)
      FQ2=DT(FIBETA)/DT(XIQ2)
      BETA1=DT(IBETA)*(1.+FQ2)/(1.+FQ1+FQ2)
      BETA2=DT(IBETA)*(1.+FQ1)/(1.+FQ1+FQ2)
      PP11=BETA1*(1.-(PDIS+D(VOL1))/DT(XIQ1))
      PP22=BETA2*(1.+(PDIS-D(VOL2))/DT(XIQ2))
      PP21=PP11*FQ2/(1.+FQ2)
      PP12=PP22*FQ1/(1.+FQ1)
      DT(PP1)=PP11+PP12
      DT(PP2)=PP22+PP21
      IF(DT(PP1).LT.VAP) DT(PP1)=VAP
      IF(DT(PP2).LT.VAP) DT(PP2)=VAP
      PLS=DT(PL)
      DT(PL) = DT(PP1)-DT(PP2)
      DLLT = DT(PL)
      IF(ABS(DLLT).GT.2000.)DLLT=SIGN(2000.,DLLT)
      DT(TP) = DLLT*KDEL*KDPF-KE*DT(DPFBP)
      TPNS =DT(TPN)
      DT(TPN) = SQRABS(DT(TP))
      DPFBS=DT(DPFB)
      DT(DPFB) =DT(DPFB) + .5*DELT*(DT(TPN)+TPNS)
```

6.105.7 (Continued)

```
      DT(DPFBP) = 2.*DT(DPFB)-DPFBS
C
C                -----  LOAD DYNAMICS
C
       FL=DT(PL)*D(AR)
      TR = FL*RFXLP
      DT(XSD)=FL/D(KS)
      FDRIVE=TR-TAERO
      CALL CFRIC2(FDRIVE,FF,DT(VXLP),DT(VXL),DT(TE),DT(XYTE),
     1  DT(DELEDI),DELS,201000.,1.263,1.)
      DT(TE)=FDRIVE+FF-DT(VXLP)*D(BE)/RFXLP
      DELS=DT(DELEDI)
      DT(DELEDI) = DYNAM(DT(XYTE),DT(DELEDI),DT(KINT))
      VXLS=DT(VXL)
      DT(VXL) = DT(DELEDI)*RFXLP/D(IE)
      DT(VXLP)=2.*DT(VXL)-VXLS
      DT(XYTE) = DT(TE)
      XLS=DT(XL)
      DT(XL) = DYNAM(DT(XYDLED),DT(XL),DT(KINT))
      DT(XLP)=2.*DT(XL)-XLS
      DT(XYDLED) = DT(VXL)
      XFBS=DT(XFB)
      DT(XFB) = DT(XL)+DT(XSD)
      DT(XFBP)=2.*DT(XFB)-XFBS
      DELP=DT(PP1)-DT(PP2)
      P1S=DT(P1)
      DT(P1)=DT(XPS)*D(KB)*(DT(PS)-SIGN(DELP,DT(XPS)))/APS/D(KCHAN)
      DT(P1P)=2.*DT(P1)-P1S
C
C     SWITCHING VALVE CALCULATIONS
C
      DO 3010 I=1,6
      IL=L(I)
      Q(IL)=0.0
 3010 P(IL)=C(IL)
      QSV=0.0
      QRV=0.0
      PSV=DT(CSV)
      PRV=DT(CRV)
      NS=1
      PXP=DT(XP)*2.0-DT(XPP)
      PXS=DT(XS)*2.0-DT(XSP)
      NDISVP=PXP/OLP
      NDISVS=PXS/OLS
      IF(NDISVS) 3020,3040,3029
 3020 COE=((PXS+OLS)*DT(COEFVS))**2
      COE=COE/(COE*DT(LOSS)+1.0)
      NS=3
      GO TO 3036
 3029 IF(NDISVP) 3030,3045,3035
```

6.105.7 (Continued)

```
 3030 NS=2
 3035 COEP=1.0/((PXP-SIGN(OLP,PXP))*DT(COEFVP))**2+DT(LOSS)
      COES=1.0/((PXS-OLS)*DT(COEFVS))**2+DT(LOSS)
      COE=1.0/(COEP+COES)
 3036 LS=NS*2
      LR=L(LS)
      LS=L(LS-1)
C     SUPPLY CALCULATION
      ZA=(Z(LS)+ZS)*COE/2.0
      QSV= -ZA+SQRT(ZA**2+ABS(C(LS)-DT(CSV))*COE)
      QSV= SIGN(QSV,C(LS)-DT(CSV))
      PSV=DT(CSV)+ZS*QSV
      Q(LS)=QSV
      P(LS)=C(LS)-QSV*Z(LS)
      DT(NSUPRP)=C(LS)-QRV*(Z(LS)-ABS(QRV)*COEP)
C     RETURN CALCULATION
      ZA=(Z(LR)+ZR)*COE/2.0
      QRV= -ZA+SQRT(ZA**2+ABS(C(LR)-DT(CRV))*COE)
      QRV= SIGN(QRV,C(LR)-DT(CRV))
      PRV=DT(CRV)+ZR*QRV
      Q(LR)=QRV
      P(LR)=C(LR)-QRV*Z(LR)
      DT(NRETRP)=C(LR)-QRV*(Z(LR)-ABS(QRV)*COEP)
 3040 N=1
      IF(NDISVP) 3043,3100,3044
 3043 N=N+2
 3044 DT(NSUPRP)=P(L(N))
      DT(NRETRP)=P(L(N+1))
      GO TO 3100
 3045 DT(NSUPRP)=PSV
      DT(NRETRP)=PRV
 3100 CONTINUE
      VELPO=DT(VELP)
      PBST=P(L(1))
      IF(PXP.LT.0.1) PBST=P(L(2))
      PSPL=(ZSPP-SSPP*PXP+P(L(3))*A2PS2+P(L(4))*A4PR2-P(L(1))*A1PS1)
      PSPL=(PSPL-VELPO*DINP)/A3PB
      DT(VELP)=-BVP+SQRT(SQBVP+BBVP*ABS(PBST-PSPL))
      DT(VELP)=SIGN(DT(VELP),PBST-PSPL)
      DT(XPP)=DT(XP)
      DT(XP)=DT(XP)+(VELPO+DT(VELP))*DELTO2
      CALL XLIMIT(DT(XP),DT(VELP),PSPL,-.2,.2)
      IF(PSPL.NE.0.0) DT(XPP)=DT(XP)
C
      VELSO=DT(VELS)
      PBST=DT(NSUPRP)
      IF(PXS.LT.0.1) PBST=DT(NRETRP)
      PSPL=(ZSPP-SSPP*PXS+P(L(5))*A2SS2+P(L(6))*A4SR2
     1 -DT(NSUPRP)*A1SS1)
      PSPL=(PSPL-VELSO*DINP)/A3SB
```

6.105.7 (Continued)

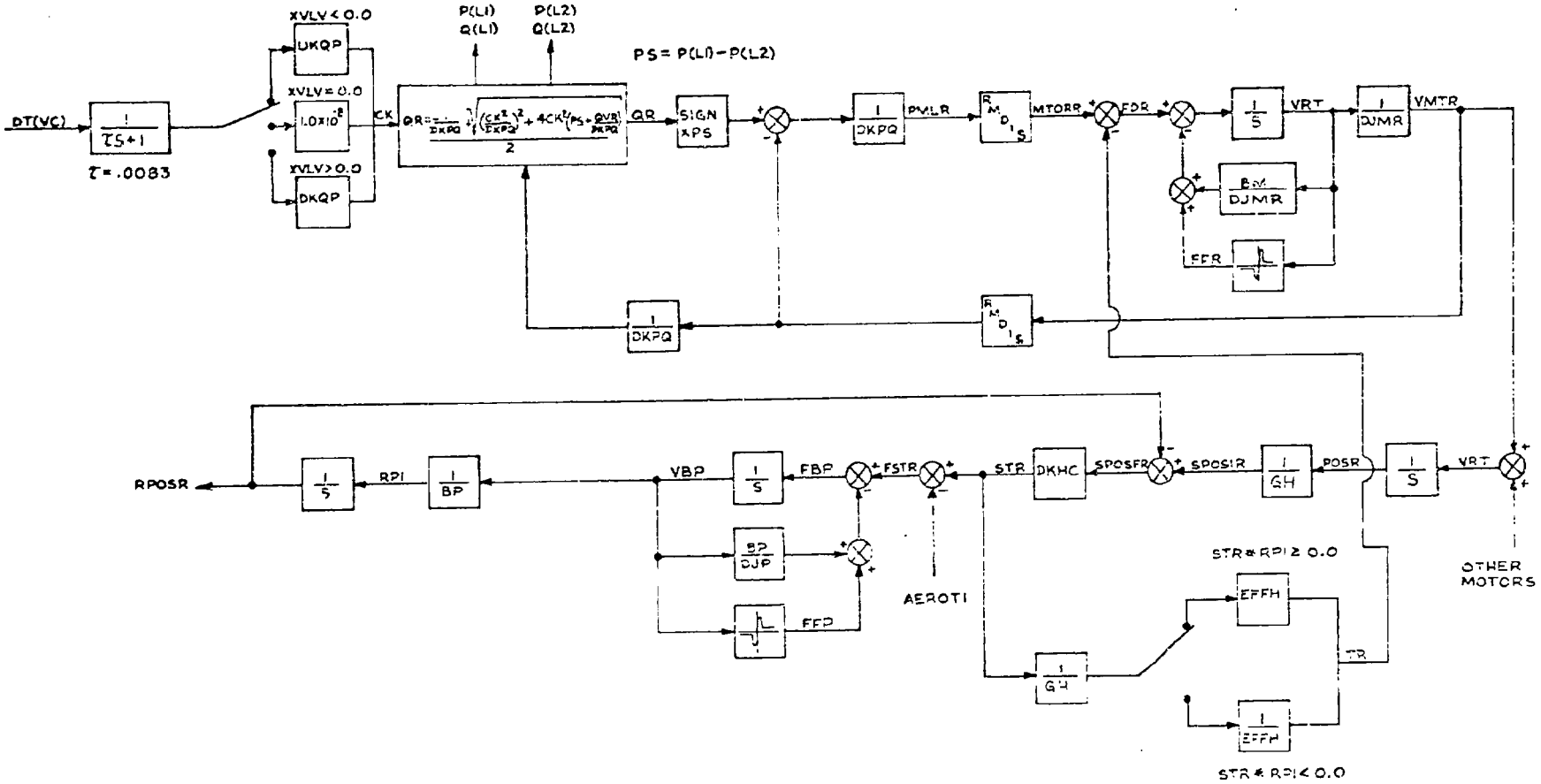
```
      DT(VELS)=-BVS+SQRT(SQBVS+BBVS*ABS(PBST-PSPL))
      DT(VELS)=SIGN(DT(VELS),PBST-PSPL)
      DT(XSP)=DT(XS)
      DT(XS)=DT(XS)+(VELSO+DT(VELS))*DELTO2
      CALL XLIMIT(DT(XS),DT(VELS),PSPL,-.2,.2)
      IF(PSPL.NE.0.0) DT(XSP)=DT(XS)
      DT(CSA)=PSV+ZS*QSV
      DT(CSV)=PSA-ZS*QSA
      DT(CRA)=PRV+ZR*QRV
      DT(CRV)=PRA-ZR*QRA
      RETURN
      END
```

## 6.106 SUBROUTINE ACT106

Subroutine ACT106 is a model of the spaceshuttle body flap actuation subsystem, a schematic of which is shown in Figure 6.106-1. The subsystem basically consists of three hydraulic motors, a valve, a mechanical drive unit, and rotary surface actuators. The component has six hydraulic connections, two for each hydraulic system, attached to it. Each system powers a motor. The output of the motors is summed in the mechanical drive which in turn drives the rotary surface actuators to position the body flap.

A single valve controls the flow to all three motors. The guidance and control subroutine provides the input commands and hinge moments at each sample time interval of the guidance system, which is .04 seconds. The valve may be commanded to open in the extend direction, open in the retract direction or close.

The body flap subsystem is essentially an open loop system with no feedback between the body flap and the valve. The position of the body flap is supplied to the command and control subroutine which commands the position of the valve.




BODY FLAP ACT106

FIGURE 6.106-1

6.106.1 MATH MODEL

The body flap model, Figure 6.106-1, receives input commands from the guidance and control program. The command signal, DT(VC), commands the valve to open in either the extend or retract direction, or to close.

The pressures and flows at the component connections, Q(L1), Q(L2), P(L1) and P(L2) are then calculated. The pressure drop due to the load on the motor is,

$$PMLR = (QR-VMTR*RMDIS)/_{DKPQ} \qquad 6.106\text{-}1$$

where

PMLR is the pressure drop due to motor load,

QR is the flow through the valve,

VMTR is the velocity of the motor,

RMDIS is the displacement of the motor,

DKPQ is the motor leakage coefficient.

The total pressure drop through the valve is,

$$PS-PMR = QR^2/CK^2 \qquad 6.106\text{-}2$$

where

PS is the pressure drop between connections,

CK is the valve coefficient.

Combining Equations 6.106-1 and 6.106-2 and simplifying,

$$QR^2/CK^2 + QR/DKPQ = PS + (VMTR*RMDIS)/DKPQ$$

With QR known the loads on the body flap and the position of the body flap are determined using the model in Figure 6.106-2.

6.106.2 Assumptions

The model of the mechanical drive was derived from the rudder/speedbrake model. Values of panel inertia, stiffness and damping are approximations based upon similar rudder/speedbrake value.

6.106.3 Computational Method

This subroutine uses Tustins method of integration and is arranged such that each differential is evaluated using the latest value of the previous integral in a series calculation, rather than the more usual parallel integration methods where the integrals are effectively evaluated simultaneously using previous values to evaluate the derivatives.

Predicted values are used to improve the calculation, where the use of a previous value would incur a significant error. The computation follows the usual component model layout shown in Figure 6.1-1 on Page 6.1-2.

1000 SECTION

This section is used to initialize the integration constants used by the subroutine. Subroutine Tustin is called to initialize each of the constants. Subroutine variables are zeroed.

1500 SECTION

This section is used to return values for the steady state calculation. In the steady state calculation the motors are assumedto be stopped and the valves are closed. The steady state flow for each system is assumed to be a leakage flow through the valve.

2000 SECTION

This section is used to initialize the subroutine variables to their steady state values. If the body flap has an initial position other than zero the variables used to calculate the body flap position are evaluated.

Constants used by the 3000 section are evaluated.

3000 SECTION

The 3000 section models the transient response of the body flap and follows the model in Figure 6.106-2. It is divided into four parts. Part one models the valve, part two models the pressure and flow demands of the

hydraulic motors, part three models the load dynamics of the rotary actuators and aerodynamic panel and part four models the gear efficiency.

In part one a command is received from the guidance and control program. This command may be +1, -1 or 0. The command is delayed .05 seconds before being used by the subroutine. If within the .05 second delay the command is changed the new command is used at the end of the delay. The command, A(COM), has a lag with a time constant of .0083 seconds. The lagged command, A(XPSL), is multiplied by the power valve flow coefficient. If A(XPSL) is positive the flap up flow coefficient, UKQP, is used. If A(XPSL) is negative the flap down flow coefficient DKQP, is used. If A(XPSL) is zero the flow coefficient is assumed to have a leakage value of .00000001. The resulting power valve flow coefficient, CK, is used to calculate the flow through the body flap.

Part two of the 3000 section uses the power valve flow coefficient, CK, calculated in part one to calculate the pressure and flow demands of the body flap motors. The calculations in part two are performed three times, once for each hydraulic system connected to the body flap. The flow through the valve, QR, is calculated. This flow is equal to the flow through the body flap. With the flow known the pressures at the connections are calculated. The flow, QR, is then used to calculate the velocity of the body flap motor, A(VMTR+MA), (MA may equal 0, 1 or 2 depending on which system is being solved).

Part three sums the individual body flap motor velocities. The total velocity is integrated to give the total angular displacement of the power drive unit, A(POSR). The angular displacement is then divided by the total mechanism gear ratio to give the total angular displacement of the drive mechanism. The torque caused by the load on the panel is calculated

next. The torque is integrated twice to yield the position of the body flap aerodynamic panel under load.

Part four models the gear efficiency of the body flap drive mechanism.

6.106.4 APPROXIMATION

The values for the panel friction and damping were estimated since no data was available. The inertia of the panel is based upon the rudder/speedbrake panel inertia.

6.106.5 LIMITATIONS

Little data was available for the body flap, therefore, when necessary, rudder/speedbrake data was used. This, obviously, will produce some error in the body flap performance. However, the flow requirements and surface rates do approximate those specified for the body flap.

6.106.6 Variable Names

| Variable | Description | Dimensions |
|---|---|---|
| AEROTI | Body flap aerodynamic load | in-# |
| BM | Motor visions friction coefficient | .015 in-lb-sec |
| BP | Viscous damping of panel | 40000. in-lb-sec |
| BRK | Motor brake torque | in-lb |
| A(COM) | Time delayed input command | deg |
| CK | Valve flow characteristic | cis/ psi |
| DJMR | Motor moment of inertia | .00636 in-lb-sec$^2$ |
| DJP | Inertia of body flap panel | 2579. in-lb-sec$^2$ |
| DKHC | Rotary actuator spring constant | 16.5E6 in-lb/rad |
| DKPQ | Motor leakage coefficient | .001 cis/psi |
| DKQP | Body flap down valve coefficient | cis/ psi |
| DLAG | Power valve and linkage response time | .0083 sec |
| DMLK | Valve leakage impedance | 1.0E6 psi/cis |
| EFFH | Gear efficiency | 96 percent |
| A(FAR+MA) | Net motor torque | in-lb |
| A(FBP) | Net torque on panel | in-lb |
| FDR | Gross developed motor torque | in-lb |
| FFP | Friction force on body flap panel | in-lb |
| FFR | Friction force on motor | in-lb |
| FRIC | Motor columb friction | 9.0 in-b |
| FRICP | Panel columb friction | 9.0 in-lb |
| A(FSTR) | Body flap panel load | in-lb |
| GH | Power train gear ratio | 56640 |
| A(KINT) | Integration constant | - |
| A(LAG) | Integration constant | - |

| | | |
|---|---|---|
| L1 | Supply connection number | - |
| L2 | Return connection number | - |
| MA | System indicator | - |
| A(PMLR+MA) | Body flap motor load pressure | psi |
| A(POSR) | Total drive unit rotation | rad |
| A(PSC+MA) | System pressure loss across body flap | psi |
| QR | Flow through body flap | cis |
| RMDIS | Motor volumetric displacement | .01432 $in^3$/rod |
| A(RPOSR) | Body flap position | rad |
| SPOSFR | Body flap deflection | rad |
| SPOSIR | Body flap hinge rotation | rad |
| SR | Ratio of motor static to columb friction | 2.0 - |
| SRP | Ratio of panel static to columb friction | 2.0 - |
| STR | Torque due to body flap deflection | in-lb |
| SXPS | Sign of valve opening | - |
| DT(TAERO) | Aerodynamic load from SDF | in-lb |
| TR | Gear box reaction torque | in-lb |
| TSW | Valve delay counter | sec |
| UKQP | Body flap up valve coefficient | cis/ psi |
| DT(VC) | Body flap position command | - |
| VLVOL | Power valve overlap | .043 in |
| VM | Motor fluid volume | 2.77 $in^3$ |
| A(UMTR+MA) | Body flap motor velocity | rad/sec |
| A(VRT) | Sum of motor velocities | rad/sec |
| A(XPS) | Valve position | |

6.106.7 Subroutine Listing

```
      SUBROUTINE ACT106 (D,DT,DD,L)
C     SHUTTLE BODY FLAP MODEL
C     REVISED AUGUST 28, 1976
      COMMON NTELPL,NTOLPL,IPT,IPOINT,NPTS,INEL,KNEL,NTOPL,NLPLT(61,3),
     1 PQLEG(90,12),LCS(90,10),ILEG(1400),PN(90),QN(90)
      COMMON/SUB/FARM(150,9),PM(1500),QM(1500),P(300),Q(300),C(300)
     1,Z(300),RHO(20),S2ORHO(20),VISC(20),BULK(20),TEMP(20),PVAP(20)
     2,ATPRES,T,DELT,TFINAL,PLTDEL,PI,TITLE(20),LEGN,ICON
     3,KTEMP(99),LSTART(150),NLPT(150),LTYPE(99),NC(99),INX,INZ
     4,INV,ISTEP,NLINE,NEL,IND,IENTR,MNLINE,MNEL,MNLEG,MNNODE,MNPLOT
     5,MNLPTS,MDS
      DIMENSION D(1),DT(13),DD(1),L(1),A(70)
      REAL MTORR
      INTEGER XPS,POSR,RPOSR,RPOSRS,RPOSRP,RPIR,PSC,QRS,PMLR,
     1 VMTR,VTR,TR,VRT,FSTR,RP1,VC,TAERO,VTRP,VTRO,FAR,FARO
     2 ,XPSS,XPSN,XPSL,VBPP,VBP,FBP,FBPO,VBPO,COM
C
C  ------- A ARRAY VARIABLES ---------
C
      DATA XPS/1/,PSC/2/,QRS/5/,PMLR/8/,VMTR/11/,TR/14/,VTR/17/
     1 ,POSR/20/,RPOSR/21/,RPOSRS/22/,RPOSRP/23/,RPIR/24/,FAR/25/,
     2 FARO/28/,VRT/30/,FSTR/32/,RP1/35/,KINT/37/,KSSGN/40/,
     3 KRP1/41/,VTRP/44/,VTRO/47/,LAG/50/,XPSL/53/,XPSS/54/,XPSN/55/
     4 ,VBPP/60/,VBP/61/,FBP/63/,FBPO/65/,VBPO/66/,COM/69/
C
C  --------- DT ARRAY VARIABLES ---------
C
      DATA VC/1/,TAERO/4/
C  ------ INITIALIZATION OF CONSTANTS  -----------
      DATA DKPQ/.001/,VM/2.77/,B/180000./,DJP/2579./,
     1 BP/40000./,DMLK/1.0E6/,DKVC/.002/,DKPI/.000138/,
     2 DKA/15.0/,DKTM/0.045/,DKXPS/6.22/,DAPS/0.193/,
     3 VLVOL/.043/,RMDIS/.01432/,BM/.015/,DJMR/.00636/,
     4 DKPR/.184/,DKHC/16.5E6/,DKB/.52/,
     5 GH/56640./,EFFH/.82/,EFFG/.96/,FRIC/9.0/,
     6 SR/2.0/,BF/1.0/,DH/.006/,DIL/8.0/,DLAG/.0083/
     7 ,FRICP/9.0/,SRP/2.0/,BFP/1.0/
C
      IF (IENTR) 1000,2000,3000
 1000 IF (INEL.NE.0) GO TO 1500
C
C  --------- INITIALIZATION --------
C
      DO 1001 I=1,70
 1001 A(I)=0.0
      DO 1002 I=1,11
 1002 DT(I)=0.0
      TDRHO=S2ORHO(KTEMP(IND))/SQRT(2./7.82E-5)
      CALL TUSTIN (1,DUMMY,A(KINT),DELT)
      CALL TUSTIN (2,DLAG,A(LAG),DELT)
```

6.106.7 (Continued)

```
      CON3=DJP/BP
      CALL TUSTIN(2,CON3,A(KRP1),DELT)
      RETURN
 1500 CONTINUE
      MA=((KNEL+1)/2)-1
      QA=PQLEG(INEL,1)
      QS=PQLEG(INEL,2)
      PQLEG(INEL,6)=PQLEG(INEL,6)+DMLK
      PQLEG(INEL,11)=PQLEG(INEL,11)-QA*QS*DMLK
      A(PSC+MA)=QA*QS*DMLK
      A(QRS+MA)=QA*QS
      RETURN
 2000 CONTINUE
      UKQP=(.65*S2ORHO(KTEMP(IND))*PI/4.0)**2
      DKQP=UKQP*.112**4
      UKQP=UKQP*.063**4
      UKQP=UKQP*DKQP/(UKQP+DKQP)
      DELT2=DELT/2.0
      A(RPOSR)=DT(2)/57.3
      A(RPOSRP)=A(RPOSR)
      A(POSR)=A(RPOSR)+(DT(2)*DT(TAERO+1)+DT(TAERO))/DKHC
      A(POSR)=A(POSR)*GH
      A(COM)=DT(VC)
      RETURN
 3000 CONTINUE
      IF(DT(VC).NE.A(COM))GO TO 3050
      TSW=.050
      GO TO 3070
 3050 TSW=TSW-DELT
      IF(TSW.LE.0.0)A(COM)=DT(VC)
 3070 CONTINUE
      AEROT1=DT(TAERO)+DT(TAERO+1)*A(RPOSR)*57.3
      XPOS=A(RPOSR)*57.3
      CALL XLIMIT(XPOS,A(COM),A(KSSGN),-11.7,22.5)
      A(XPS)=A(COM)
      DT(8)=A(XPS)
      A(XPSN)=A(XPS)
      A(XPSL)=DYNAM(A(XPSS),A(XPSL),A(LAG))
      A(XPSS)=A(XPSN)
      DT(6)=A(XPSL)
      SXPS=SIGN(1.0,A(XPSL))
      NXV=A(XPSL)/VLVOL
      IF(NXV) 3100,3110,3120
C  -------- XPS LESS THAN ZERO --------
C
 3100 XVLV=A(XPSL)+VLVOL
      CK=DKQP*ABS(XVLV)
      GO TO 3130
C  --------- XPS LESS THAN VALVE OVERLAP -------
 3110 XVLV=0.0
```

6.106.7 (Continued)

```
      CK=.00000001
      GO TO 3130
C  --------- XPS GREATER THAN ZERO ---------
 3120 XVLV=A(XPSL)-VLVOL
      CK=DKQP*XVLV
 3130 DT(7)=XVLV
C  ---------- PRESSURE DROP BETWEEN CONNECTIONS ----------
C     THE VARIABLES IN THE A ARRAY WILL BE USED IN SETS OF THREE
C     USING ONE POSITION FOR EACH SYSTEM.  THE POSITION WILL BE
C     DETERMINED BY THE CONSTANT MA.
      B1=CK/DKPQ
C
      DO 3300 I=1,3
      MA=I-1
      L1=L(2*I-1)
      L2=L(2*I)
      PS=A(PSC+MA)
      QVR=A(VMTR+MA)*RMDIS
      C1=(PS+QVR*SXPS/DKPQ)*CK
      QR=(-B1+SQRT(B1**2+4.0*C1))/(2.0)
      QR=SIGN(QR,C1)
      A(PSC+MA)=C(L1)-QR*Z(L1)-C(L2)-QR*Z(L2)
      Q(L1)=QR
      Q(L2)=-QR
      P(L1)=C(L1)-Q(L1)*Z(L1)
      P(L2)=C(L2)-Q(L2)*Z(L2)
      A(PMLR+MA)=(QR*SXPS-A(VMTR+MA)*RMDIS)/DKPQ
      MTORR=A(PMLR+MA)*RMDIS
C
C  ---------- MOTOR FRICTION AND DAMPING EFFECTS ---------
C
      FDR=MTORR-A(TR+MA)
      BRK=0.0
      IF(P(L1).LT.2220.)BRK=(2220.-P(L1))/2220.*73.8
      CALL CFRIC(FDR,FFR,A(VTRP+MA),A(VTR+MA),A(FAR+MA),A(FARO+MA)
     1,FRIC+BRK,SR,BF)
      A(FARO+MA)=A(FAR+MA)
      A(FAR+MA)=FDR-A(VTRP+MA)*(BM/DJMR)+FFR
      A(VTRO+MA)=A(VTR+MA)
      A(VTR+MA)=A(VTRO+MA)+DELT2*(A(FAR+MA)+A(FARO+MA))
      A(VTRP+MA)=2.0*A(VTR+MA)-A(VTRO+MA)
      A(VMTR+MA)=A(VTR+MA)/DJMR
 3300 CONTINUE
C
C  ---------- LOAD DYNAMICS
C
      A(VRT)=A(VRT+1)
      A(VRT+1)=A(VMTR)+A(VMTR+1)+A(VMTR+2)
      A(POSR)=DYNAM(A(VRT),A(POSR),A(KINT))
      SPOSTR=A(POSR)/GH
```

6.106.7 (Continued)

```
      SPOSFR=SPOSIR-A(RPOSRP)
      STR=SPOSFR*DKHC
      A(FSTR)=STR-AEROT1
      CALL CFRIC(A(FSTR),FFP,A(VBPP),A(VBP),A(FBP+1),A(FBPO)
     1,FRICP,SRP,BFP)
      A(FBPO)=A(FBP+1)
      A(FBP+1)=A(FSTR)-A(VBPP)*(BP/DJP)+FFP
      A(VBPO)=A(VBP)
      A(VBP)=DYNAM(A(FBP),A(VBP),A(KINT))
      A(VBPP)=2.0*A(VBP)-A(VBPO)
      A(RP1)=A(RP1+1)
      A(RP1+1)=A(VBP)/BP
      A(RPOSRS)=A(RPOSR)
      A(RPOSR)=DYNAM(A(RP1),A(RPOSR),A(KINT))
      A(RPOSRP)=2.0*A(RPOSR)-A(RPOSRS)
C     WRITE(6,4700)A(VRT),A(VRT+1),A(POSR),A(VBP),STR,AEROT1,A(FSTR),
C    1 A(FBP+1),A(RP1+1),A(RPOSR)
C4700 FORMAT(2X,1H3,2X,10E12.5)
C
C  ------------ GEAR EFFICIENCY CALCULATION -------------
C  ------------ HINGE GEAR EFFICIENCY -------------------
C
      TRP=STR/GH
      DO 3440 I=1,3
      NA=I-1
      IF(STR*A(RP1).GE.0.0) GO TO 3410
      TRPH=TRP/EFFH
      GO TO 3440
 3410 TRPH=TRP*EFFH
 3440 A(TR+NA)=TRPH
      DT(2)=A(RPOSR)*57.3
      DT(9)=A(VRT+1)*60./(2.*3.14159)
      DT(10)=A(POSR)*57.3
      DT(11)=A(RP1+1)
      RETURN
      END
```

## 6.107 SUBROUTINE ACT107

Subroutine ACT107 models the shuttle rudder/speedbrake actuation subsystem, the layout of which is shown in Figure 6.107-1. The subsystem consists of six hydraulic motors, two power valves and two 4-channel servo actuators. Each system powers two motors, one for the rudder and one for the speedbrake. The servo actuators which control the power valves are powered by a single system. All three hydraulic systems power the motors and a switching valve selects one of the three systems to supply the servo actuator. A single power valve controls all three rudder motors and a single power valve controls all three speedbrake motors. The input command and hinge moments are supplied by a guidance and control program which updates the values at each sample time interval of the guidance system which is .04 seconds.

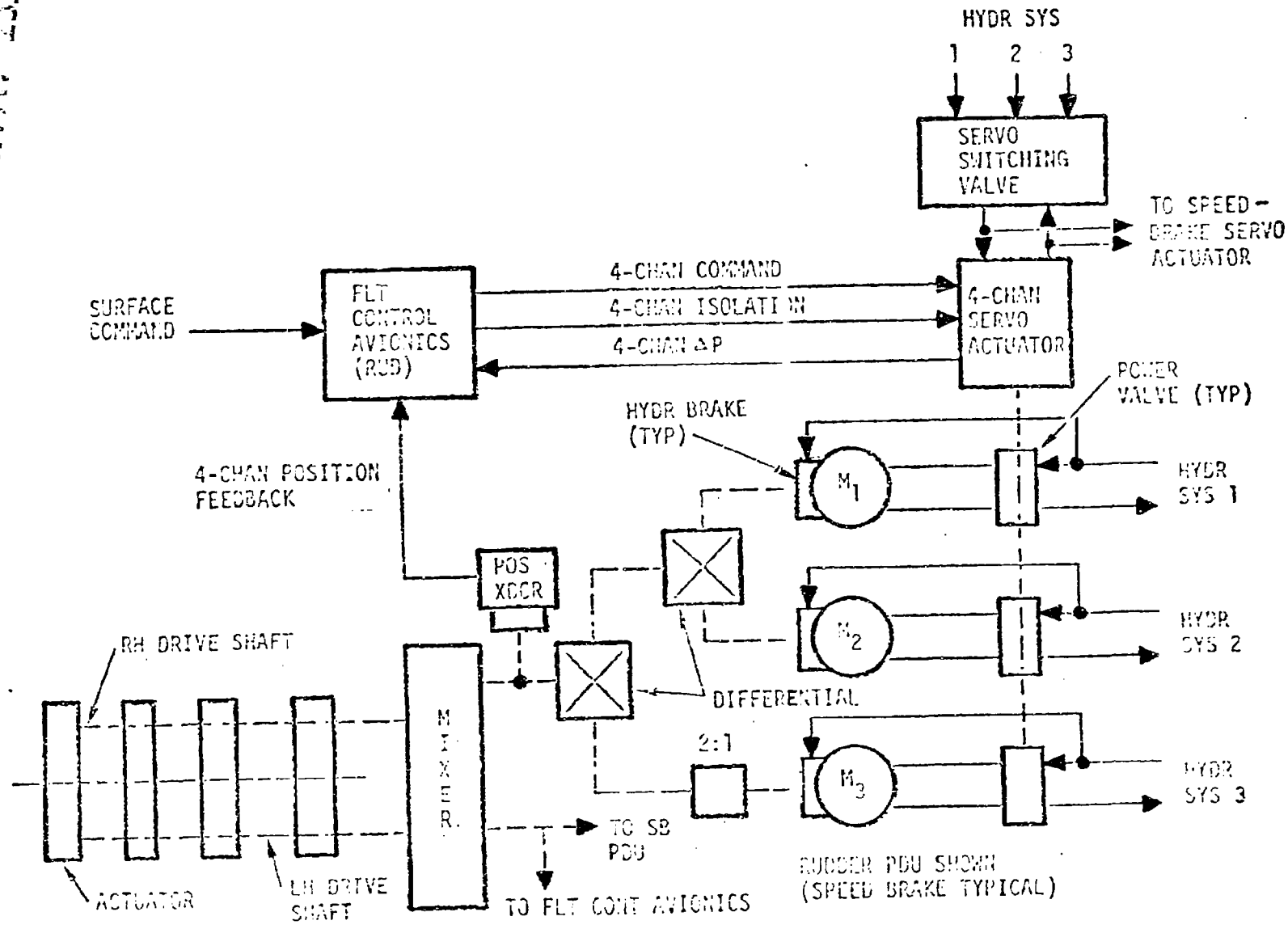
### 6.107.1 Math Model

The ACT107 rudder/speedbrake model is shown in Figure 6.107-2. The figure shows one of the four rudder servovalves, the rudder power valve and the rudder motor. The models of the speedbrake are identical to those of the rudder except for constants which describe the physical characteristics of the rudder and speedbrake. Figure 6.107-2 also shows the rotary actuator and aerodynamic surface model which is common to both the rudder and speedbrake.

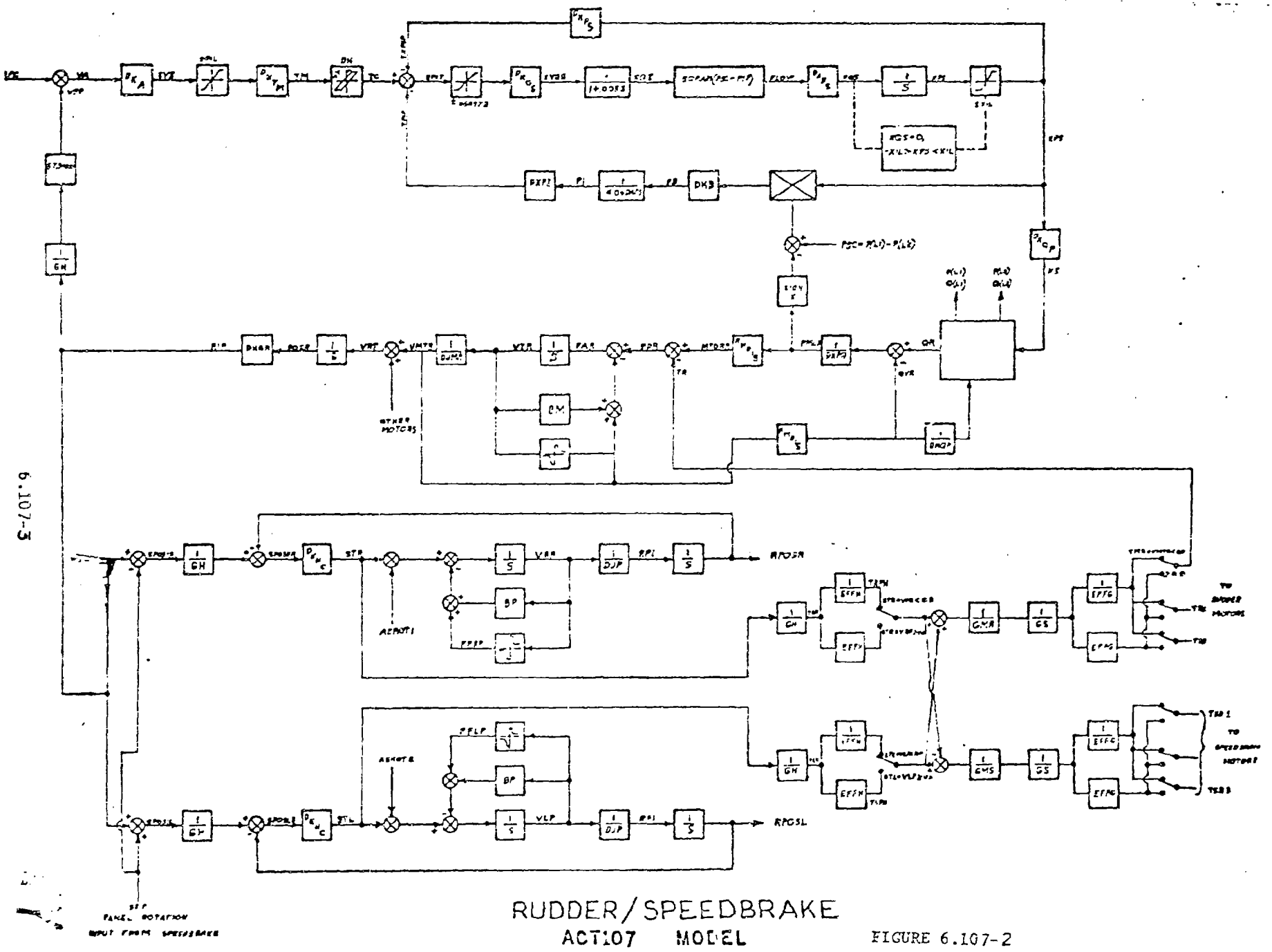
The following portions of the rudder/speedbrake description which specify the rudder apply equally well to the speedbrake.

The rudder receives input commands from the command and guidance program. The input command, VC, and the rudder position feedback, VFP, are summed to yield an error signal which drives the first stage flapper valve of a two stage servovalve.

The ACT107 model uses the error signal to calculate the pressure and flow demands of the rudder/speedbrake. The rudder/speedbrake may be represented




RUDDER/SPEEDBRAKE ACTUATION SUBSYSTEM
FIGURE 6.107-1



RUDDER/SPEEDBRAKE
ACT107 MODEL

FIGURE 6.107-2

by six parallel orifices with a common supply and return, Figure 6.107-3. Two of the orifices represent the servovalves first stage flapper valves. Two of the orifices represent the servovalves second stage valve and the mod piston. The third set of orifices represent the power valves and rudder/speedbrake motors.

The total flow through the system is equal to the sum of the flows through each orifice.

$$Q_T = Q_1 + Q_2 + Q_3 + Q_4 + Q_5 + Q_6 \qquad 6.107.1$$

where, $Q_T$ is the total flow through the rudder/speedbrake

$Q_1$ to $Q_6$ are the flows through each orifice

The flow through each orifice is

$$Q_N = C_N PS^{1/2} \left(1 - \frac{PL_N}{PS}\right)^{1/2} \qquad N = 1,6 \qquad 6.107.2$$

where $Q_N$ is the flow through each orifice,

$C_N$ is the orifice flow coefficient,

PS is the pressure drop across the component connections,

$PL_N$ is the load pressure on each orifice.

Combining 6.107.1 and 6.107.2 $Q_T = PS^{1/2} \Sigma CN \left(1 - \frac{PL_N}{PS}\right)^{1/2}$ 6.107.2a

$C_N$ must be determined for each orifice.

For the two orifices which represent the rudder and speedbrake flapper valve $C_N$ is simply the first stage leakage coefficient, DQLOSS.

The orifice flow coefficient for the rudder and speedbrake second stage valve varies depending upon the error signal to the flapper valve. The error signal produces a torque in the flapper valve torque motor which is summed with the secondary wire feedback torque and the pressure compensated feedback torque to yield a flapper valve error torque.

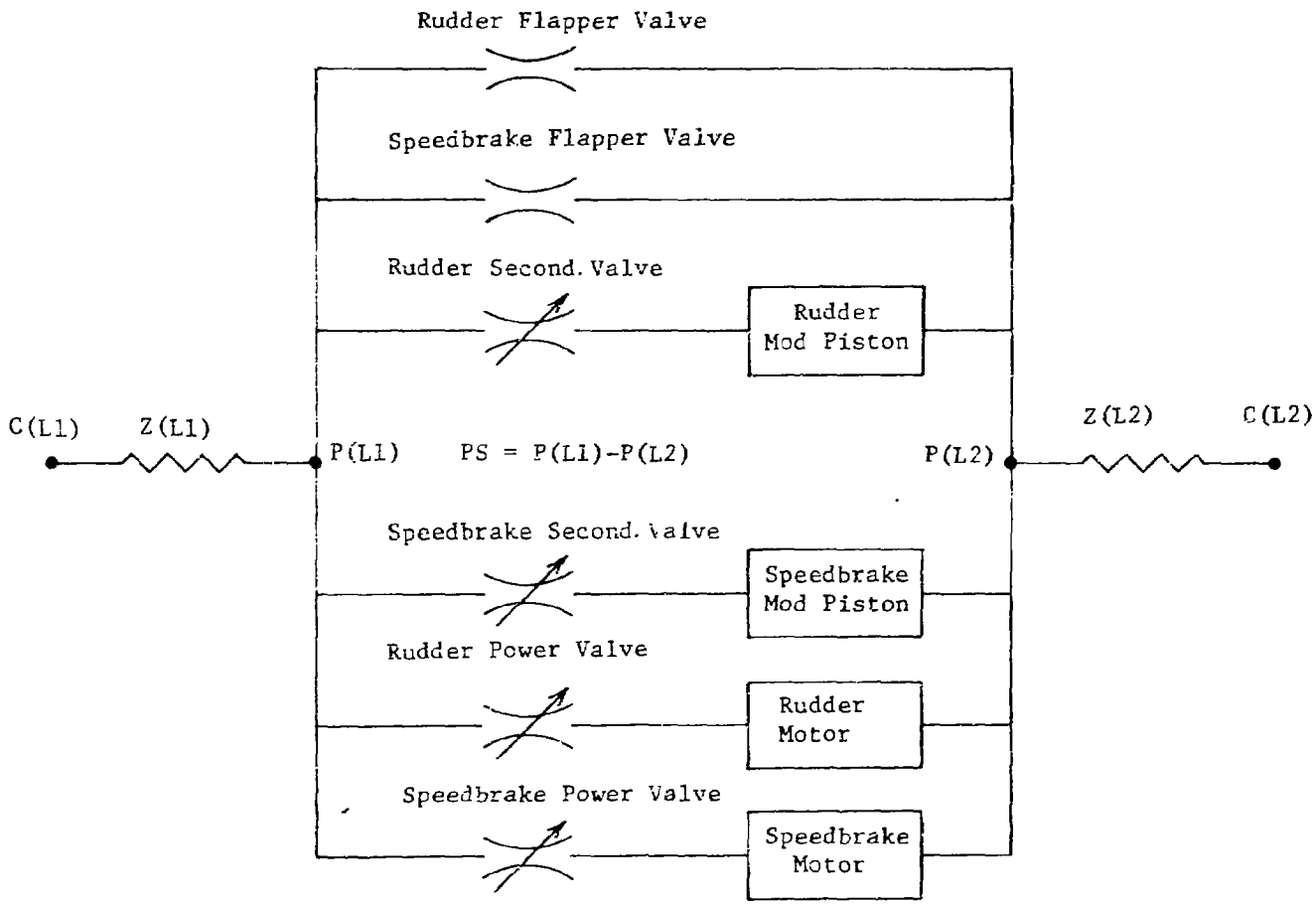



Figure 6.107.3

Rudder/Speedbrake Flow Model

$$EPST = TC - TXPS - TP1$$

where, EPST is the flapper valve error torque,

TC is the torque motor input torque,

TXPS is the wire feedback torque,

TP1 is the pressure compensated feedback torque.

The orifice coefficient is then calculated from

$$K2 = EPST * DKQS$$

where, K2 is the orifice flow coefficient for the rudder,

DKQS is the secondary valve flow gain.

The orifice coefficient for the power valve depends upon the displacement of the power valve. The power valve displacement is equal to the displacement of the mod piston which is,

$$FLOW = XQI \sqrt{PS - P1P}$$

$$XQS = FLOW/DAPS$$

$$XPS = XQS$$

Where, FLOW is the flow to the mod piston,

XQI is the secondary valve flow coefficient,

P1P is the mod piston load pressure,

XQS is the mod piston velocity,

DAPS is the mod piston area,

XPS is the displacement of the mod piston and power valve.

The flow coefficient for the power valve is,

$$K5 = (XPS - VLVOL) * DKQP$$

where, K5 is the power valve flow coefficient,

VLVOL is the valve overlap,

DKQP is the power valve flow gain.

The power valve and motor are modeled as shown in Figure 6.107-4.

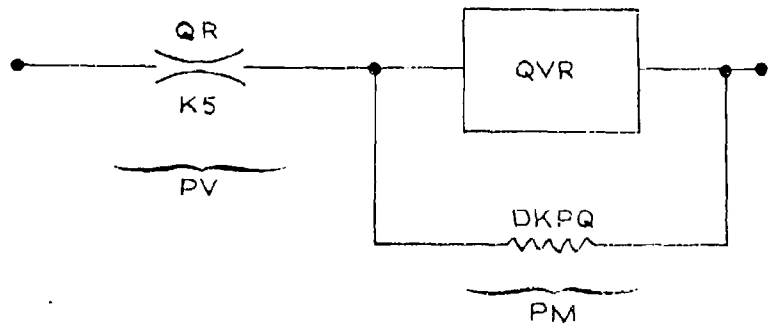



Figure 6.107-4

The pressure drop across the valve and motor, PS, is

$$PS = PV + PM$$

Where, PV is the pressure drop across the valve,

PM is the pressure drop across the motor.

PS in terms of the flows through the orifices is

$$PS = \frac{QR^2}{K5^2} + \frac{QR - QVR}{DKPQ} \qquad 6.107.3$$

where, QR is the flow through the valve,

QVR is the flow displaced by the motor,

DKPQ is the motor leakage coefficient.

With the orifice flow coefficients, CN, calculated for each of the six orifices a total flow coefficient, CT, is calculated

$$C_T = CN(1 - \frac{PL_N}{PS})^{1/2} \qquad N = 1,6 \qquad 6.107.4$$

For the flapper valve orifices and the power valve and motor equivalent orifice, $PL_N$ is equal to zero.

The pressure drop across the component is

$$PS = C1 - C2 - Q_T Z1 - Q_T Z2 \qquad 6.107.5$$

where, C1 and C2 are the characteristic line pressures, Z1 and Z2 are the line impedances. The pressure drop across the component in terms of CT is:

$$PS = \frac{Q_T^2}{C_T^2} \qquad 6.107.6$$

Combining 6.107.5 and 6.107.6 and solving for QT,

$$QT = \frac{-CT^2(Z1 + Z2) + \sqrt{(CT^2(Z1 + Z2))^2 + \frac{4(CT^2(C1 - C2))}{CT^4(Z1 + Z2)^2}}}{2}$$

QT is compared to the QT calculated during the previous time step and if the difference is less than .001 CIS the program proceeds to the next section. If the difference between the old and new QT is greater than .001 CIS and ITER is less than 25, PS is recalculated using equation 6.107.4 and CT and QT are recalculated.

With the pressure drop across the component calculated the flow through the power valve, QR, may be calculated. The pressure drop across the power valve and motor is,

$$PS = \frac{QR^2}{KD^2} + \frac{QR - QVR}{QKPQ}$$

Solving for QR yields,

$$QR = \frac{\frac{-K5^2}{DKPQ} + \sqrt{\left(\frac{K5^2}{DKPQ}\right) + 4\ K5^2\left(\frac{QVR}{DKPQ} + PS\right)}}{2}$$

With QR known, the left and right rudder panel positions may be calculated using the model in Figure 6.107-2.

6.107.2 Assumptions

The model is limited to simulating one channel of the secondary actuator and it is assumed that the other channels are operative and will give identical outputs in terms of available force at the power spool.

6.107.3 Computational Method

This program uses Tustin's method of integration and is arranged such that each differential is evaluated using the latest value of the previous integral in a series calculation, rather than the more usual parallel integration methods, where the integrals are effectively evaluated simultaneously using previous values to evaluate the derivatives.

Predicted values are used to improve the calculation where the use of a previous value would incur a significant error. The computation follows the usual component subroutine layout shown in Figure 6.1-1 on Page 6.1-2.

1000 SECTION

This section is used to initialize the integration constants used by the subroutine. Subroutine Tustin is called to initialize each of the constants.

The system used to supply the servovalve is also selected using the input constant INV which indicates which systems are active or shut down.

1500 SECTION

This section is used to return values for the steady state calculation so that the system state conditions can be calculated. It is assumed that

the secondary actuators and the motors are at zero velocity. For the two hydraulic systems used to power the motors only, the steady state flow consists of leakage flow past the power valve. The leakage impedance is added to the Q value in PQLEG(INEL,6), and the pressure drop A(PSC+MA) is subtracted from the inlet pressure PQLEG(INEL,11). MA may have the value of 0, 1 or 2 depending on whether the pressure for system 1, 2 or 3 is being calculated.

For the system which supplies the two 4-channel servovalves the only steady state flow is that due to the first stage servovalve leakage. It is assumed that the leakage through the second stage control valve and the power valve are small compared to the servovalve leakage and can be ignored. The first stage impedance (1.0/(DQLOSS)*8.0))**2.0 is added to the $Q^2$ value in PQLEG(INEL,8), and the pressure drop (A(PSC+MA)) is subtracted from the inlet pressure PQLEG(INEL,11).

2000 SECTION

This section is used to initialize the variables to their steady state values. If the initial position is other than zero, variables used to calculate the panel positions are evaluated. Other variables are set to zero. In addition some of the constants used in the 3000 section are defined.

3000 SECTION

The 3000 section models the transient response of the rudder/speedbrake. It is divided into four parts. Part one models the servovalve. Part two models the pressure and flow demands of the hydraulic motors. Part three models the load dynamics of the rotary actuators and aerodynamic panels. Part four models the gear efficiency.

Before the servovalve calculations in part one are begun, a test is made to determine which hydraulic system is to supply the servovalve via the switching valve. The aerodynamic loads on the two rudder/speedbrake panels

are then calculated using hinge moments and hinge moment slopes provided by the command and guidance program.

The servovalve calculations in part one are performed twice, once for the rudder commands and once for the speedbrake commands. The command and guidance program supplies a rudder command, A(VC+N), which, when summed with the rudder position feedback, A(VFP+N), commands the servovalve torque motor (N=0 for rudder, N=1 for speedbrake). The command to the servovalve causes a displacement of the mod piston which is connected to the power valve. The resulting power valve displacement, A(XPS+N), is used to calculate a power valve flow coefficient A(K5+N).

Part two of the 3000 section uses the power valve flow coefficient along with the flapper valve leakage coefficient, A(K1+N), and the secondary valve flow coefficient, A(K3+N), also calculated in part one, to calculate the pressure and flow demands of the rudder and speedbrake motors. The calculations in part two are performed three times, once for each hydraulic system connected to the rudder/speedbrake. The calculation technique uses an iterative procedure to solve for the total flow from each system, A(QRS+MA). (MA may be equal to 0, 1 or 2 depending on which system is being solved). With the flow known, the pressures at the connections are calculated. The flow to the motors through the power valve, QR, is calculated next and is used to calculate the velocity of the rudder motor, A(VMTR+MA).

Part three sums the individual rudder motor velocities. The total velocity is integrated to give the total angular displacement of the power drive unit. The angular displacements for the rudder and speedbrake are then summed to give the angular displacement of the rotary actuators. The torque caused by loads on the panels are calculated next. The torques acting on

both panels are integrated twice to yield the position of the rudder/ speedbrake panels.

Part four models the effect of the grear efficiency on the rudder/ speedbrake load reaction torque.

6.107.4 Approximations

The aerodynamic panel friction and damping were unavailable for use in the model, therefore, approximate values were used and should be revised when actual values become available.

6.107.5 Limitations

None.

6.107.6 Variable Names

| Variable | Description | | Dimensions |
|---|---|---|---|
| AEROT1 | Aerodynamic load on R.H. panel | | in-lb |
| AEROT2 | Aerodynamic load on L.H. panel | | in-lb |
| BM | Motor viscous friction coefficient | 0.015 | in-lb-sec |
| BP | Viscous damping, each rudder panel | 1808. | in-lb-sec |
| CER | Combined Rudder Orifice Coefficient | | $CIS/\sqrt{PSI}$ |
| CES | Combined Speedbrake Orifice Coefficient | | $CIS/\sqrt{PSI}$ |
| CK | Total flow characteristic | | $CIS/\sqrt{PSI}$ |
| CT | Flow characteristic array | | $CIS/\sqrt{PSI}$ |
| DAPS | Mod piston area | 0.1930 | $in^2$ |
| DH | Torque motor hystersis | 0.006 | in-lb |
| DlL | Servo amplifier saturation limit | 8.0 | ma |
| DJMR | Motor moment of inertia (rudder) | 0.00636 | $in\text{-}lb\text{-}sec^2$ |
| DJMSB | Motor moment of inertia (speedbrake) | 0.00565 | $in\text{-}lb\text{-}sec^2$ |
| DJP | Inertia of each rudder panel | 1169. | $in\text{-}lb\text{-}sec^2$ |
| DKA | Servo amplifier gain | 15.0 | ma/v |
| DKB | Bernoulli coefficient | 0.52 | in |
| DKHC | Rotary actuator spring constant | 7.0E6 | in-lb/rad |
| A(DKP) | Rudder position gain | 0.184 | v/deg |
| A(DKP+1) | Speedbrake position gain | 0.0937 | v/deg |
| DKPI | Nozzle pressure feedback gain | 0.000138 | $in^3$ |
| DKPQ | Motor leakage coefficient | 0.004 | CIS/PSI |

| | | | |
|---|---|---|---|
| DKQP | Power spool flow gain | 73.24 | $in^3/(sec\text{-}lb^{1/2})$ |
| DKQS | Secondary valve flow gain | 0.3868 | $in^3/(sec\text{-}lb^{3/2})$ |
| DKTM | Torque motor gain | 0.045 | in-lb/ma |
| DKXPS | Wire feedback gain | 6.22 | in-lb/in |
| DMLK | Power valve leakage coefficient | 1.0E6 | PSI/CIS |
| DQLOSS | First stage leakage coefficient | 2.37E-2 | CIS/PSI |
| EFFG | Efficiency of summer and mixer gearing | 96 | percent |
| EFFH | Efficiency of rotary actuators (Hinge gear boxes) | 82 | percent |
| EPST | Net error torque | | in-lb |
| EYE | Torque motor input current | | ma |
| A(FB+N) | Bernoulli force on power spool | | lb |
| FDR | Net rudder motor torque | | in-lb |
| FDSB | Net speedbrake motor torque | | in-lb |
| FFLP | Friction force on left panel | | in-lb |
| FFR | Friction force on rudder motor | | in-lb |
| FFRP | Friction force on right panel | | in-lb |
| FFSB | Friction force on speedbrake motor | | in-lb |
| FI | Dummy variable | | |
| FLOW | Flow to mod piston | | CIS |
| A(FLP) | Net torque on left panel | | in-lb |
| FRIC | Motor Coulomb friction | 9.0 | in-lb |
| FRICP | Panel Coulomb friction | TBD | in-lb |
| A(FRP) | Net torque on right panel | | in-lb |
| A(FSTL) | L.H. panel load | | in-lb |
| A(FSTR) | R.H. panel load | | in-lb |

| | | | |
|---|---|---|---|
| GH | Power hinge gear ratio | 473.921:1 | |
| GMR | Mixer gear train ratio, rudder | 1.47:1 | |
| GMS | Mixer gear train ratio, speedbrake | 4.46:1 | |
| GS | Summer gear train ratio | 7:32:1 | |
| ITER | Iteration counter | | - |
| A(K1+N) | Servovalve leakage characteristic | | |
| A(K3+N) | Secondary actuator load characteristic | | |
| A(K5+N) | Power valve flow characteristic | | |
| L1 | Supply connection number | | - |
| L2 | Return connection number | | - |
| A(PMLR+MA) | Rudder motor load pressure | | PSI |
| A(PMLSB+MA) | Speedbrake motor load pressure | | PSI |
| A(POSR) | Total rudder motor rotation | | Rad |
| A(POSSB) | Total speedbrake motor rotation | | Rad |
| A(PSC+MA) | System pressure loss across rudder/speedbrake | | PSI |
| A(P1+N) | Mod piston pressure | | PSI |
| A(QRS+MA) | System flow through rudder/speedbrake | | CIS |
| A(QRV+MA) | Flow through rudder power valve | | CIS |
| A(QSBV+MA) | Flow through speedbrake power valve | | CIS |
| QVR | Flow through rudder motor | | CIS |
| QVSB | Flow through speedbrake motor | | CIS |
| RMD1S | Motor volumetric displacement (rudder) | 0.08276 | $in^3/rad$ |
| A(RPOSL) | Left panel position | | Rad |
| A(RPOSR) | Right panel position | | Rad |
| SPOSFL | Left panel deflection | | Rad |

| | | | |
|---|---|---|---|
| SBMDIS | Motor volumetric displacement (speedbrake) | 0.08276 | $in^3/rad$ |
| SPOSFR | Left panel deflection | | Rad |
| SR | Ratio of motor static to Coulomb friction | 2.0 | |
| SRP | Ratio of panel static to Coulomb friction | 2.0 | |
| STL | Torque due to left panel deflection | | in-lb |
| STR | Torque due to right panel deflection | | in-lb |
| SWIT | Switching valve switching pressure | | PSI |
| SYS | System supplying servovalve | | - |
| A(TAEROL) | Left panel aerodynamic load | | in-lb |
| A(TAEROR) | Right panel aerodynamic load | | in-lb |
| TC | Torque mortor torque | | in-lb |
| A(TP1P+N) | Mod piston predicted pressure feedback | | in-lb |
| TR | Rudder gearbox reaction torque | | in-lb |
| A(TXPS+N) | Wire feedback torque | | in-lb |
| VA | Net position error | | v |
| DT(VC+N) | Rudder/speedbrake position command | | Deg |
| VLVOL | Power valve overlap | 0.0011 | in. |
| VM | Motor fluid volume | 2.77 | $in^3$ |
| A(VMTR+MA) | Rudder motor velocity | | Rad/sec |
| A(VMTSB+MA) | Speedbrake motor velocity | | Rad/sec |
| A(VRT+1) | Sum of rudder motor velocities | | Rad/sec |
| A(VSBT+1) | Sum of speedbrake motor velocities | | Rad/sec |
| XIL | Power valve displacement limit | 0.065 | in |
| A(XPS+N) | Power valve displacement | | in. |
| XVLV | Effective power valve displacement | | in. |

6.107.7 Subroutine Listing

```
      SUBROUTINE ACT107 (D,DT,DD,L)
C     REVISED JANUARY 28,1976
      COMMON NTELPL,NTOLPL,IPT,IPOINT,NPTS,INEL,KNEL,NTOPL,NLPLT(61,3),
     1 PQLEG(90,12),LCS(90,10),ILEG(1400),PN(90),QN(90)
      COMMON/SUB/PARM(150,9),PM(1500),QM(1500),P(300),Q(300),C(300)
     1,Z(300),RHO(20),S2ORHO(20),VISC(20),BULK(20),TEMP(20),PVAP(20)
     2,ATPRES,T,DELT,TFINAL,PLTDEL,PI,TITLE(20),LEGN,ICON
     3,KTEMP(99),LSTART(150),NLPT(150),LTYPE(99),NC(99),INX,INZ
     4,INV,ISTEP,NLINE,NEL,IND,IENTR,MNLINE,MNEL,MNLEC,MNNODE,MNPLOT
     5,MNLPTS,MDS
      DIMENSION D(1),DT(13),DD(1),L(1),A(170),CT(6),PL(6)
      REAL MTORR,MTORSB
      INTEGER VFP,TXPS,XPS,TP1P,XYQS,XQ1,XQ,XQS,
     1 P1,FB,P1P,POSR,POSSB,RPOSR,RPOSL,VRPP,VLPP,
     2 PSC,QRS,PMLR,PMLSB,VMTR,VMTSB,VTR,VTSB,SYS,TR,TSB,
     3 VRT,VSBT,FSTR,FSTL,RP1,RP2,VC,TAEROR,TAEROL,
     4 VTRP,VTSBP,VTRO,VTSBO,FAR,FARO,FASB,FASBO,DKP,QRV,QSBV
     5 ,FRP,FLP,FRPO,FLPO,VRPO,VLPO,VRP,VLP,RPOSRP,RPOSLP,RPOSRS,RPOSLS
     6 ,COMS,COMN,COML
C
C  ------- A ARRAY VARIABLES ---------
C
      DATA VFP/1/,XPS/5/,TP1P/7/,XQ1/9/,
     1 XYQS/11/,XQ/15/,P1/17/,FB/19/,P1P/21/,PSC/23/,
     2 QRS/26/,PMLR/29/,PMLSB/32/,VMTR/35/,VMTSB/38/,
     3 TR/41/,TSB/44/,VTR/47/,VTSB/50/,POSR/53/,POSSB/54/,
     4 RPOSR/55/,RPOSL/56/,VRP/57/,VLP/58/,QRV/59/,QSBV/62/,XQS/65/,
     5 DKP/69/,FAR/71/,FARO/74/,FASB/77/,FASBO/80/,VRT/83/,VSBT/85/,
     6 FSTR/87/,FSTL/88/,VRPP/89/,VLPP/90/,RP1/91/,RP2/93/,KINT/95/,
     7 KSSGN/98/,KFP1/99/,KPIL/102/,KHYST/105/,K1/115/,K2/116/,K3/117/,
     8K4/118/,K5/119/,K6/120/,VTRP/121/,VTSBP/124/,VTRO/127/,VTSBO/130/
     9 ,KFLOW/133/,TXPS/136/,FRP/138/,FLP/140/,FRPO/142/,FLPO/143/
     A ,VRPO/144/,VLPO/145/,RPOSRP/146/,RPOSLP/147/,RPOSRS/148/,
     B RPOSLS/149/,COMS/150/,COMN/151/,COML/152/,KCOM/156/
C
C  --------- DT ARRAY VARIABLES ---------
C
      DATA VC/1/,TAEROL/4/,TAEROR/6/
C
C ---- RUDDER/SPEEDBRAKE INPUT DATA ------
C
      DATA DKPQ/0.004/,VM/2.77/,B/180000./,DJP/1169./,BP/1808./,
     1 DMLK/1.0E6/,DQLOSS/2.37E-2/,DKPI/0.000138/,DKA/15.0/,
     2 DKTM/0.045/,DKXPS/6.22/,DAPS/0.1930/,VLVOL/.0011/,
     3 RMDIS/0.08276/,SBMDIS/0.08276/,BM/0.015/,DJMR/.00636/,
     4 DJMSB/.00565/,DKHC/7.0E6/,DKB/0.52/,GS/7.32/,GMR/1.47/,
     5 GMS/4.46/,GH/473.921/,EFFH/.82/,EFFG/.96/,FRIC/9.0/,
     6 SR/2.0/,BF/1.0/,FRICP/12.0/,SRP/2.0/,BFP/1.0/,XIL/0.065/,
     7 DH/0.006/,DIL/8.0/
```

6.107.7 (Continued)

```
      IF (IENTR) 1000,2000,3000
 1000 IF (INEL.NE.0) GO TO 1500
C
C  --------- INITIALIZATION --------
C
      DO 1010 I=1,170
 1010 A(I)=0.0
      DO 1020 I=1,30
 1020 DT(I)=0.0
      TDRHO=S2ORHO(KTEMP(IND))/SQRT(2./7.82E-5)
      CALL TUSTIN (1,DUMMY,A(KINT),DELT)
      CON1=DKPQ/(VM/(4*B))
      CALL TUSTIN (2,CON1,A(KPHL),DELT)
      CON3=DJP/BP
      CALL TUSTIN(2,CON3,A(KRP1),DELT)
      CALL TUSTIN(2,.005,A(KFLOW),DELT)
      CALL TUSTIN(2,.0159,A(KCOM),DELT)
      SYS=0
      DO 15 I=1,3
      K=I+6
      SYS=SYS+1
      IF(INV)20,30,10
   10 IF(INV.EQ.L(K)) GO TO 30
   15 CONTINUE
   20 IF(INV.EQ.-L(K))SYS=SYS+1
   30 CONTINUE
      RETURN
 1500 CONTINUE
      MA=((KNEL+1)/2)-1
      QA=PQLEG(INEL,1)
      QS=PQLEG(INEL,2)
      IF(SYS.EQ.(KNEL+1)/2)GO TO 1600
      PQLEG(INEL,6)=PQLEG(INEL,6)+DMLK
      PQLEG(INEL,11)=PQLEG(INEL,11)-QA*QS*DMLK
      A(PSC+MA)=QA*QS*DMLK
      A(QRS+MA)=QA*QS
      RETURN
 1600 CONTINUE
      QLOSS=(1.0/(DQLOSS*8.0))**2.0
      PQLEG(INEL,8)=PQLEG(INEL,8)+QLOSS
      A(PSC+MA)=QLOSS*QS*QA**2.0
      PQLEG(INEL,11)=PQLEG(INEL,11)-A(PSC+MA)
      A(QRS+MA)=QA*QS
      RETURN
 2000 CONTINUE
      SWIT=1300.
      DKQS=0.3868*TDRHO
      DKQP=73.24*TDRHO*SQRT(2.)
      DELT2=DELT/2.0
      A(DKP)=0.184
```

6.107.7 (Continued)

```
      A(DKP+1)=.0937
      DKGR=1/(GS*GMR)
      DKGSB=1/(GS*GMS)
      DICOMR=DT(VC)/(57.3*A(DKP))
      DICOMS=DT(VC+1)/(2.0*57.3*A(DKP+1))
      A(POSR)=(DICOMR*GH)/DKGR
      A(POSSB)=(DICOMS*GH)/DKGSB
      A(VFP)=DT(VC)
      A(VFP+1)=DT(VC+1)
      A(RPOSR)=DICOMR-DICOMS
      A(RPOSL)=DICOMR+DICOMS
      A(RPOSR)=A(RPOSR)-(A(RPOSR)*57.3*DT(TAEROR+1)+DT(TAEROR))/DKHC
      A(RPOSL)=A(RPOSL)-(A(RPOSL)*57.3*DT(TAEROL+1)+DT(TAEROL))/DKHC
      A(RPOSRP)=A(RPOSR)
      A(RPOSLP)=A(RPOSL)
C
C  ---------- INITIALIZE CONTROL ACTUATOR VARIABLES ---------
C
C
      A(COML)=DT(VC)
      A(COMS)=DT(VC)
      A(COML+3)=DT(VC+1)
      A(COMS+3)=DT(VC+1)
      DT(10)=A(VFP)
      DT(11)=A(VFP+1)
      DT(9)=A(RPOSR)*57.3
      DT(8)=A(RPOSL)*57.3
      CALL PDUMP(A(1),A(150),1)
      RETURN
 3000 CONTINUE
C     IF N=0 INDICATES THE RUDDER, N=1 INDICATES THE SPEEDBRAKE.
      DO 3010 I=1,3
      K=I
      IF(A(PSC+K-1).GT.SWIT)GO TO 3020
 3010 CONTINUE
 3020 IF(K.NE.SYS)SWIT=1300.
      SYS=K
      IF(SWIT.GT.1300.)SWIT=SWIT-50.
      N=0
      AEROT1=DT(TAEROR)+DT(TAEROR+1)*A(RPOSR)*57.3
      AEROT2=DT(TAEROL)+DT(TAEROL+1)*A(RPOSL)*57.3
 3050 IF (N.GE.2) GO TO 3200
C
C  ---------- SERVO AMPLIFIER ----------
C
      A(COMN+3*N)=DT(VC+N)
      A(COML+3*N)=DYNAG(A(COMS+3*N),A(COML+3*N),A(KCOM))
      A(COMS+3*N)=A(COMN+3*N)
      DT(17+N)=A(COML+3*N)
      VA=A(COML+3*N)-A(VFP+N)
```

6.107.7 (Continued)

```
      EYE=VA*DKA
      CALL XLIMIT(EYE,TM,FI,-DIL,DIL)
C
C   ---------- TORQUE MOTOR CONSTANTS ----------
C
      TM=EYE*DKTM
      TC=HYSTLI(TM,DH,A(KHYST+5*N))
      TXPSS=A(TXPS+N)
      A(TXPS+N)=A(XP3+N)*DKXPS
      TXPSP=2.0*A(TXPS+N)-TXPSS
      EPST=TC-A(TP1P+N)-TXPSP
      IF(ABS(EPST).GT..064973) EPST=SIGN(.064973,EPST)
      A(K3+N)=ABS(EPST*DKQS)
      A(K1+N)=DQLOSS
C
C   ---------- POWER SPOOL DISPLACEMENT ----------
C
      A(XYQS+2*N+1)=EPST*DKQS
      A(XQI+N)=DYNAM(A(XYQS+2*N),A(XQI+N),A(KFLOW))
      FLOW=A(XQI+N)*SQRABS(A(PSC+SYS-1)-SIGN(1.0,A(XQI))*A(P1P+N))
      A(XQS+2*N+1)=FLOW/DAPS
      A(XPS+N)=DYNAM(A(XQS+2*N),A(XPS+N),A(KINT))
      CALL XLIMIT(A(XPS+N),A(XQS+2*N+1),A(KSSGN),-XIL,XIL)
      IF(A(KSSGN).EQ.0.0) GO TO 3060
      A(XQS+2*N+1)=0.0
      A(K3+N)=0.0
 3060 CONTINUE
      A(XQS+2*N)=A(XQS+2*N+1)
      A(XYQS+2*N)=A(XYQS+2*N+1)
      P1S=A(P1+N)
      A(P1+N)=A(FB+N)/(4.0*DAPS)
      A(FB+N)=0.0
      A(P1P+N)=2.0*A(P1+N)-P1S
      A(TP1P+N)=A(P1P+N)*DKPI
      NXV=A(XPS+N)/VLVOL
      IF(NXV) 3100,3110,3120
C   -------- XPS LESS THAN ZERO --------
C
 3100 XVLV=A(XPS+N)+VLVOL
      GO TO 3130
C   --------- XPS LESS THAN VALVE OVERLAP -------
 3110 XVLV=0.0
      GO TO 3130
C   --------- XPS GREATER THAN ZERO ---------
 3120 XVLV=A(XPS+N)-VLVOL
 3130 A(K5+N)=ABS(DKOP*XVLV)
      DT(14+N)=XVLV
      N=N+1
      GO TO 3050
C
```

6.107.7 (Continued)

```
C  --------- PRESSURE DROP BETWEEN CONNECTIONS ----------
C
 3200 CONTINUE
      DO 3300 I=1,3
      ITER=0
      MA=I-1
C     THE VARIABLES IN THE A ARRAY WILL BE USED IN SETS OF THREE
C     USING ONE POSITION FOR EACH SYSTEM.  THE POSITION WILL BE
C     DETERMINED BY THE CONSTANT MA.
      L1=L(2*I-1)
      L2=L(2*I)
      PS=A(PSC+MA)
C     A(PSC+MA) IS THE PRESSURE DROP ACROSS THE CONNECTIONS FROM
C     THE PREVIOUS TIME STEP.
      QOLD=A(QRS+MA)
      IF (I.EQ.SYS) GO TO 3220
C     SYS IS THE ACTIVE CONTROL SYSTEM
      DO 3210 K=1,4
      CT(K)=0.0
 3210 PL(K)=0.0
      GO TO 3235
 3220 DO 3230 K=1,4
      PL(K)=0.0
 3230 CT(K)=A(K1+K-1)*4.0
      PL(3)=A(P1)*SIGN(1.0,A(XPS))
      PL(4)=A(P1+1)*SIGN(1.0,A(XPS+1))
 3235 CONTINUE
      CT(5)=A(K5)
      CT(6)=A(K6)
      QVR=A(VMTR+MA)*RADIS
      QVSB=A(VMTSB+MA)*SBMDIS
      PS=A(PSC+MA)
      QR=A(QRV+MA)
      QSB=A(QSBV+MA)
      SXPSR=SIGN(1.0,A(XPS))
      SXPSSB=SIGN(1.0,A(XPS+1))
 3240 PL(5)=(QR-QVR)*SXPSR/DKPQ
      PL(6)=(QSB-QVSB)*SXPSSB/DKPQ
      ITER=ITER+1
      CK=0.0
      DO 3260 K=1,6
 3260 CK=CK+CT(K)*SQRABS(1-PL(K)/PS)
      IF(CK.NE.0.0) GO TO 3265
      QNEW=(C(L1)-C(L2))/(Z(L1)+Z(L2)+DNLK)
      PS=C(L1)-QNEW*Z(L1)-C(L2)-QNEW*Z(L2)
      QR=0.0
      QSB=0.0
      GO TO 3274
 3265 CONTINUE
      X=((Z(L1)+Z(L2))*CK**2.0)/2.0
```

6.107.7 (Continued)

```
      Y=4.0*(C(L1)-C(L2))/(CK*(Z(L1)+Z(L2)))**2.0
      TPRES=1.0+ABS(Y)
      QNEW=X*(-1.0+SQRT(TPRES))*SIGN(1.0,Y)
      PSN=C(L1)-QNEW*Z(L1)-C(L2)-QNEW*Z(L2)
      IF(ABS(PS-PSN).LT..1.OR.ITER.GT.25) GO TO 3274
      PS=(PSN+PS)/2.
      QR=0.0
      IF(A(K5).EQ.0.0) GO TO 3262
      A1=1/(A(K5)**2.)
      B1=1/DKPQ
      C1=ABS(PS+QVR*SXPSR/DKPQ)
      QR=(-B1+SQRT(B1**2.+4.*A1*C1))/(2.*A1)
      QR=QR*SIGN(1.0,(PS+QVR*SXPSR/DKPQ)*A(XPS))
 3262 QSB=0.0
      IF(A(K6).EQ.0.0) GO TO 3240
      A2=1/(A(K6)**2)
      B2=1/DKPQ
      C2=ABS(PS+QVSB*SXPSSB/DKPQ)
      QSB=(-B2+SQRT(B2**2.+4.*A2*C2))/(2.*A2)
      QSB=QSB*SIGN(1.0,(PS+QVSB*SXPSSB/DKPQ)*A(XPS+1))
      GO TO 3240
 3274 CONTINUE
      A(PSC+MA)=PS
      A(QRV+MA)=QR
      A(QSBV+MA)=QSB
      Q(L1)=QNEW
      Q(L2)=-QNEW
      A(QRS+MA)=QNEW
      P(L1)=C(L1)-Q(L1)*Z(L1)
      P(L2)=C(L2)-Q(L2)*Z(L2)
      A(PMLR+MA)=(QR-A(VMTR+MA)*RMDIS)*250.
      A(PMLSB+MA)=(QSB-A(VMTSB+MA)*SBMDIS)*250.
      MTORR=A(PMLR+MA)*RMDIS
      MTORSB=A(PMLSB+MA)*SBMDIS
C
C  ---------- MOTOR FRICTION AND DAMPING EFFECTS ---------
C
C
      FDR=MTORR-A(TR+MA)
      FDSB=MTORSB-A(TSB+MA)
      IF(A(PSC+MA).LT.1300.)BF=(1300.-A(PSC+MA))/50.+1
      CALL CFRIC(FDR,FFR,A(VTRP+MA),A(VTR+MA),A(FAR+MA),A(FARO+MA)
     1,FRIC,SR,BF)
      CALL CFRIC(FDSB,FFSB,A(VTSBP+MA),A(VTSB+MA),A(FASB+MA),A(FASBO+MA
     1,FRIC,SR,BF)
      BF=1
      A(FARO+MA)=A(FAR+MA)
      A(FASBO+MA)=A(FASB+MA)
      A(FAR+MA)=FDR-A(VTRP+MA)*(BM/DJMR)+FFR
      A(FASB+MA)=FDSB-A(VTSBP+MA)*(BM/DJMSB)+FFSB
```

6.107.7 (Continued)

```
      A(VTRO+MA)=A(VTR+MA)
      A(VTSBO+MA)=A(VTSB+MA)
      A(VTR+MA)=A(VTRO+MA)+DELT2*(A(FAR+MA)+A(FARO+MA))
      A(VTSB+MA)=A(VTSBO+MA)+DELT2*(A(FASB+MA)+A(FASBO+MA))
      A(VTRP+MA)=2.0*A(VTR+MA)-A(VTRO+MA)
      A(VTSBP+MA)=2.0*A(VTSB+MA)-A(VTSBO+MA)
      A(VMTR+MA)=A(VTR+MA)/DJMR
      A(VMTSB+MA)=A(VTSB+MA)/DJMSB
 3300 CONTINUE
      DO 3310 I=1,3
      A(FB)=A(FB)+A(XPS)*DKB*(A(PSC+I-1)-A(PMLR+I-1)*SXPSR)
      A(FB+1)=A(FB+1)+A(XPS+1)*DKB*(A(PSC+I-1)-A(PMLSB+I-1)*SXPSSB)
 3310 CONTINUE
C
C  --------- RUDDER AND SPEEDBRAKE LOAD DYNAMICS
C
      A(VRT)=A(VRT+1)
      A(VSBT)=A(VSBT+1)
      A(VRT+1)=A(VMTR)+A(VMTR+1)+A(VMTR+2)
      A(VSBT+1)=A(VMTSB)+A(VMTSB+1)+A(VMTSB+2)
      A(POSR)=DYNAM(A(VRT),A(POSR),A(KINT))
      A(POSSB)=DYNAM(A(VSBT),A(POSSB),A(KINT))
      RIP=A(POSR)*DKGR
      SBIP=A(POSSB)*DKGSB
      A(VFP)=(RIP/GH)*57.3*A(DKP)
      A(VFP+1)=(2.0*SBIP/GH)*57.3*A(DKP+1)
      SPOSIR=RIP-SBIP
      SPOSIL=RIP+SBIP
      SPOSIR=SPOSIR/GH
      SPOSIL=SPOSIL/GH
      SPOSFR=SPOSIR-A(RPOSRP)
      SPOSFL=SPOSIL-A(RPOSLP)
      STR=SPOSFR*DFHC
      STL=SPOSFL*DFHC
      A(FSTR)=STR-AEROT1
      A(FSTL)=STL-AEROT2
      CALL CFRIC(A(FSTR),FFRP,A(VRPP),A(VRP),A(FRP+1),A(FRPO),
     1FRICP,SRP,BFP)
      CALL CFRIC(A(FSTL),FFLP,A(VLPP),A(VLP),A(FLP+1),A(FLPO),
     1FRICP,SRP,BFP)
      A(FRPO)=A(FRP+1)
      A(FLPO)=A(FLP+1)
      A(FRP+1)=A(FSTR)-A(VRPP)*(BP/DJP)+FFRP
      A(FLP+1)=A(FSTL)-A(VLPP)*(BP/DJP)+FFLP
      A(VRPO)=A(VRP)
      A(VLPO)=A(VLP)
      A(VRP)=DYNAM(A(FRP),A(VRP),A(KINT))
      A(VLP)=DYNAM(A(FLP),A(VLP),A(KINT))
      A(VRPP)=2.0*A(VRP)-A(VRPO)
      A(VLPP)=2.0*A(VLP)-A(VLPO)
```

6.107.7 (Continued)

```
      A(RP1)=A(RP1+1)
      A(RP2)=A(RP2+1)
      A(RP1+1)=A(VRP)/DJP
      A(RP2+1)=A(VLP)/DJP
      A(RPOSRS)=A(RPOSR)
      A(RPOSLS)=A(RPOSL)
      A(RPOSR)=DYNAM(A(RP1),A(RPOSR),A(KINT))
      A(RPOSL)=DYNAM(A(RP2),A(RPOSL),A(KINT))
      A(RPOSRP)=2.0*A(RPOSR)-A(RPOSRS)
      A(RPOSLP)=2.0*A(RPOSL)-A(RPOSLS)
      DT(16)=(A(RPOSL)-A(RPOSR))*57.3
C
C  ------------ GEAR EFFICIENCY CALCULATION -------------
C  ------------ HINGE GEAR EFFICIENCY -------------------
C
      TRP=STR/GH
      TLP=STL/GH
      IF(STR*A(VRP).GE.0.0) GO TO 3410
      TRPH=TRP/EFFH
      GO TO 3420
 3410 TRPH=TRP*EFFH
 3420 IF(STL*A(VLP).GE.0.0) GO TO 3430
      TLPH=TLP/EFFH
      GO TO 3440
 3430 TLPH=TLP*EFFH
C
C  ------------ MIXER GEAR EFFICIENCY ---------------
C
 3440 TGMR=TRPH+TLPH
      TGMSB=TLPH-TRPH
      TMR=(TGMR/GMR)/GS
      TMSB=(TGMSB/GMS)/GS
C
C  ------------ SUMMER GEAR EFFICIENCY ------------------
C
      DO 3490 K=1,3
      NA=K-1
      IF(TMR*A(VMTR+NA).GE.0.0) GO TO 3450
      A(TR+NA)=TMR/EFFG
      GO TO 3460
 3450 A(TR+NA)=TMR*EFFG
 3460 IF(TMSB*A(VMTSB+NA).GE.0.0) GO TO 3470
      A(TSB+NA)=TMSB/EFFG
      GO TO 3480
 3470 A(TSB+NA)=TMSB*EFFG
 3480 CONTINUE
 3490 CONTINUE
      DT(9)=A(RPOSR)*57.3
      DT(8)=A(RPOSL)*57.3
      DT(10)=A(VFP)
```

6.107.7 (Continued)

```
      DT(11)=A(VFP+1)
      DT(12)=A(VRT+1)*60./(2.*3.14159)
      DT(13)=A(VSBT+1)*60./(2.*3.14159)
      RETURN
      END
```

## 7.0 OUTPUT SUBROUTINES

The output subroutines comprising, store, graph and scaled are currently dedicated to producing print plots of the data calculated by the program.

Current options allow maximum or minimum calculated values to be substituted for plot values in event these max or min values occurred between plot intervals. This assures that the max or min values calculated are reflected in the output plots. Another option allows tabulation of all calculated values for each plot variable.

## 7.1 SUBROUTINE STORE

Subroutine STORE, which is called by HYTR, reads output requirements and stores data required for output plots.

### 7.1.1 Math Model

Not applicable.

### 7.1.2 Assumptions

Not applicable.

### 7.1.3 Computation Methods

Section 1000

Section 1000 reads in all the plot information for line and component plots.

Section 2000

This section first performs a test to determine if the current time step is also a plot time, if so line or component data is stored. If it is not time to store but the MAX/MIN option has been exercised, tests are made to determine if the current calculated value is less than or greater than (depending on which option was exercised) the previous value stored, if so the stored value is replaced by the current calculated value. If the LIST option has been exercised every calculated plot variable is printed. Once all or a max of 101 points have been stored, GRAPH is called to plot the points. A test is then performed to determine if more than 101 points are to be plotted, if so the additional points (up to 101) are calculated and stored as before. GRAPH is again called to plot these points. This procedures is repeated until all points have been plotted.

### 7.1.4 Approximations

Not applicable.

### 7.1.5 Limitations

Not applicable.

### 7.1.6 Variable Name

| Variable | Description | Units |
|---|---|---|
| I | Counter | |
| INDEX | Line Number Associated with Pressure and/or Flow Plots | |
| IPLT | Number of Plots Required along Line INDEX | |
| IPTS | Dummy Variable | |
| J | Counter | |
| LIST | Input Integer Value 0 (No List) of 1 (List of all Points) | |
| LPT | Distance in DELX's, of Required Plots from Upstream End of Line | |
| M | Counter | |
| $M_X$TREM | Dummy Variable | |
| N | Counter | |
| NABSQ | Not Used | |
| NISTEP | Counter | |
| NLPLTC | Number of Line Plot Points | |
| NOGRAF | Not Used | |
| NOMSG | Not Used | |
| NOSTOP | Not Used | |
| NPT | Dummy Variable | |
| NXTREM | Input Integer Value 0 (Normal Plot), +1 (Plot with Max Values) or -1 (Plot with Min Values) | |

7.1.6 (Continued)

| Variable | Description | Units |
|---|---|---|
| N1 | Counter | |
| Y | Dummy Variable | |
| YY( ) | Array Used to Store Line Positions of Required Plots | |

## 7.1.7 Subroutine Listing

```
      SUBROUTINE STORE
C**** REVISED AUGUST 5, 1975 ****
      DOUBLE PRECISION DD
      COMMON NTELPL,NTOLPL,IPT,IPOINT,NPTS,INEL,KNEL,NTOPL,NLPLT(61,3),
     1 VSTORE(1)
      COMMON/SUB/PARA(150,9),PM(1500),QM(1500),P(300),Q(300),C(300)
     1,Z(300),RHO(20),S2ORHO(20),VISC(20),BULK(20),TEMP(20),PVAP(20)
     2,ATPRES,T,DELT,TFINAL,PLTDEL,PI,TITLE(20),LEGH,ICON
     3,KTEMP(99),LSTART(150),NLPT(150),LTYPE(99),NC(99),INX,INZ
     4,INV,ISTEP,NLINE,NLL,IND,IENTR,MNLINE,MNEL,MNLLG,MNNODE,MNPLOT
     5,MNLPTS,MDS
      DIMENSION YY(10),DD(1400),ITITLE(40),IN(40),IY(40),IIC(40),IC(40),
     1IITITLE(40),ICHAR(8)
      COMMON/COMPD/D(4500),L(1500),LE(99,4)
      EQUIVALENCE(DD(1),D(1))
      DATA ICHAR/4HLINE,4HDIST,4HCOMP,4HVAR ,2HIN,2H P,2H Q,2H  /
      IF(IENTR) 1000, 1000,2000
 1000 CONTINUE
      IPT=0
      DISTEP=0
      IPTS=MPTS
      IF(NPTS.GT.101) MPTS=101
C
      NTOLPL=0
      READ(5,270) NLPLTC,NTELPL,NXTREM,LIST,NOSTOP,NOMSG,NOGRAF,NASSQ
      IF(NLPLTC.EQ.0) GO TO 142
      DO 140 I=1,NLPLTC
      READ(5,320) INDEX,IPLT,(YY(J),J=1,IPLT)
      DO 130 J=1,IPLT
      J=1
      IF(YY(J).LT.0.0) J=-1
      LPT=(YY(J)*J/PARA(INDEX,5) +0.5)
      IF(LPT.GE.NLPT(INDEX)) LPT =NLPT(INDEX)-1
      NTOLPL=NTOLPL+1
      NLPLT(NTOLPL,2) = INDEX
      NLPLT(NTOLPL,3) = LPT
      NLPLT(NTOLPL,1) = (LSTART(INDEX)+LPT)*J
  130 CONTINUE
  140 CONTINUE
  142 CONTINUE
      NTOPL=NTOLPL+NTELPL
      IF(NTOLPL.GT.MNPLOT) NTOLPL=MNPLOT
C
      IF(NTELPL.EQ.0) GO TO 144
      READ(5,270) ((NLPLT(I+NTOLPL,2),NLPLT(I+NTOLPL,3)),I=1,NTELPL)
  144 CONTINUE
      IF(NTOLPL+NTELPL.GT.MNPLOT) NTELPL=MNPLOT-NTOLPL
      IF(NTOLPL+NTELPL.NE.NTOPL) WRITE(6,520)
      NTOPL=NTOLPL+NTELPL
      IF(NTELPL.EQ.0) GO TO 1250
```

7.1.7 (Continued)

```
      LPT=NTOLPL+1
      DO 1200 I=LPT,NTOPL
      NPT=NLPLT(I,2)
      N  =NLPLT(I,3)
      IF(N) 1150,1180,1160
 1150 N1=-LE(NPT,3)+N+1
      GO TO 1170
 1160 N1=LE(NPT,2)+N-1
 1170 NLPLT(I,1)=N1
      GO TO 1200
 1180 NLPLT(I,1)=1
 1200 CONTINUE
C
      WRITE(6,1601)
      WRITE(6,1603)
      IF(NXTREM) 5,12,10
    5 WRITE(6,1606)
      GO TO 1220
   10 WRITE(6,1607)
      GO TO 1220
   12 WRITE(6,1608)
      GO TO 1220
 1220 WRITE(6,1603)
 1250 JJ=0
      II=NTOPL
      DO 1300 I=1,II
      J=I
      JJ=JJ+1
   15 N1=NLPLT(I,1)
      NPT=NLPLT(I,2)
      N=NLPLT(I,3)
      IF(I.GT.NTOLPL) GO TO 2500
      ITITLE(JJ)=ICHAR(1)
      IN(JJ)=NPT
      IY(JJ)=N*PARM(NPT,5)
      IITITLE(JJ)=ICHAR(2)
      NI=0
      IF(N1.LT.0) NI=1
      IC(JJ)=ICHAR(6+NI)
      IIC(JJ)=ICHAR(5)
      GO TO 1700
 2500 ITITLE(JJ)=ICHAR(3)
      IITITLE(JJ)=ICHAR(4)
      IN(JJ)=NPT
      IY(JJ)=N
      IC(JJ)=ICHAR(8)
      IIC(JJ)=ICHAR(9)
 1700 IF(JJ.LT.10.AND.I.LT.NTOPL) GO TO 1300
      III=I-JJ+1
      WRITE(6,1500) ((JJJ),JJJ=III,J)
```

7.1.7 (Continued)

```
      IF(I.GT.NTOLPL) GO TO 75
      WRITE(6,1600) ((ITITLE(JJJ),IN(JJJ),IC(JJJ)),JJJ=1,JJ)
      WRITE(6,1600) ((IITITLE(JJJ),IY(JJJ),IIC(JJJ)),JJJ=1,JJ)
      GO TO 16
   75 WRITE(6,1604) ((ITITLE(JJJ),IN(JJJ),IC(JJJ)),JJJ=1,JJ)
      WRITE(6,1604) ((IITITLE(JJJ),IY(JJJ),IIC(JJJ)),JJJ=1,JJ)
   16 WRITE(6,1605)
      JJ=0
 1300 CONTINUE
      WRITE(6,1602)
      WRITE(6,1603)
 2000 CONTINUE
C
      IF(ISTEP.EQ.NISTEP) GO TO 2010
      IF(NXTREM.EQ.0.AND.LIST.EQ.0) RETURN
      NXTREM=NXTREM
      GO TO 2020
 2005 NISTEP=NISTEP-IPOINT
 2010 NXTREM=0
      IPT=IPT+1
      NISTEP=NISTEP+IPOINT
      VSTORE(IPT)=T
 2020 NPT=IPT
      N1=0
      DO 2200 I=1,NTOPL
      NPT=NPT+NPTS
      N=NLPLT(I,1)
      IF(I.GT.NTOLPL) GO TO 2050
      IF(N) 2030,2150,2040
 2030 Y=QM(-N)
      GO TO 2080
 2040 Y=PM(N)
      GO TO 2080
 2050 IF(N) 2060,2150,2070
 2060 Y=QQ(-N)
      GO TO 2080
 2070 Y=Q(N)
 2080 IF(NXTREM)2090,2095,2100
 2095 IF(ISTEP+IPOINT.EQ.NISTEP) GO TO 2110
      GO TO 2120
 2090 IF(VSTORE(NPT).GT.Y) GO TO 2110
      GO TO 2120
 2100 IF(VSTORE(NPT).GE.Y) GO TO 2120
 2110 VSTORE(NPT)=Y
 2120 IF(LIST.EQ.0) GO TO 2200
      IF(I.EQ.1) WRITE(6,2211) T
 2130 N1=N1+1
      YY(N1)=Y
      IF(N1.NE.10) GO TO 2200
      WRITE(6,2210) YY
```

7.1.7 (Continued)

```
      N1=0
      GO TO 2200
 2150 WRITE(6,2220) I
      Y=T
      GO TO 2080
 2200 CONTINUE
C     IF(N1*LIST.NE.0)WRITE(6,2210) (YY(I),I=1,N1)
      IF(IPT.NE.NPTS) RETURN
      INV=NABS2
      CALL GRAPH
      IF(IPTS-NPTS) 2300,2350,2310
 2300 NPTS=IPTS
      IPT=0
      GO TO 2005
 2310 NPTS=101
      IPTS=IPTS-100
      IPT=0
      GO TO 2005
 2350 CONTINUE
      RETURN
  270 FORMAT (16I5)
  320 FORMAT (2I5,7F10.0)
  520 FORMAT(5X,42HTOO MANY PLOTS REQUESTED MAX NUMBER IS 60     )
 2210 FORMAT(5X,10E12.5)
 2211 FORMAT(5X,25HDATA CALCULATED AT TIME =,F8.4)
 2220 FORMAT(5X,45HVALUE OF N IN 2000 SECTION OF COMP IS ZERO I=  ,I5)
 1500 FORMAT(5X,10(9HGRAPH ,I4,2H  ))
 1600 FORMAT(5X,10(A4,I4,A2,2H  ))
 1601 FORMAT(1H1,42X,35HVARIABLES SELECTED FOR OUTPUT PLOTS)
 1602 FORMAT(1H1,53X,13HHYTRAN OUTPUT)
 1603 FORMAT(1H0)
 1604 FORMAT(5X,10(A4,I6,A2))
 1605 FORMAT(1H )
 1606 FORMAT(28X,71HVALUES PLOTTED REPRESENT MINIMUM VALUES CALCULATED I
     1N THE TIME INTERVAL)
 1607 FORMAT(28X,71HVALUES PLOTTED REPRESENT MAXIMUM VALUES CALCULATED I
     2N THE TIME INTERVAL)
 1608 FORMAT(21X,75HVALUES PLOTTED REPRESENT THE ACTUAL VALUES CALCULATE
     3D AT EACH PLOT INTERVAL)
      END
```

## 7.2 SUBROUTINE GRAPH

Subroutine graph produces print plots of the output data stored in VSTORE ( ).

Most computers will have their own version of this subroutine which could be used if necessary. However, since the plotted output is such an integral part of HYTRAN, this subroutine has been added to avoid the problems involved in changing from one computer to another.

7.2.1 Theory - Not applicable.

7.2.2 Assumptions - Not applicable.

7.2.3 Limitations

The program is executed once for each plot,up to the total number of plots NTOPL. The DO 901 J = 1, NTOPL controls this loop.

The first section which sets the X scale, is only executed on the first pass, when J = 1.

The program currently uses TIME (1) as XMIN and TIME (NPTS) as XMAX.

In the second section a DO loop is used to find the maximum and minimum values of the Y data to be plotted, using the functions AMAX1 (YMAY, VSTORE, (1+IADD))and AMIN1 (YMIN, VSTORE (1+IADD)).

With the maximum and minimum values established, a check is made to see if they are equal, if they are, 25 is added to YMAX, and YMIN is set at 50 less than that, to avoid a fruitless search for a suitable scale.

Subroutine SCALED is then called to obtain a preferred scale for the Y axis, and returns with values for YMAX and YMIN.

The next section finds the type of plot and sets the plot character, P, Q or C and the data to be written at the bottom of the output plot.

The routine then starts the output plot section by going to the top of a new page, and proceeds to plot the output data, line by line until the plot is complete. A more detailed explanation of this section will be added at a later date.

At the bottom of the plot a descriptive line is written which gives the line number and distance along the line for line pressure or flow plots or the variable number and the component number if it is a component data plot.

The next printed line is the title of the run, which was inputted on the first data card.

When all the plots have been completed program control returns to HYTR.

7.2.4 Approximations - Not applicable.

7.2.5 Limitations

The basic limitation of a print plot is the number of points that can be plotted on a single page graph and the resulting inaccuracy in reading the graph. To an extent these limitations can beovercome by use of the MAX/MIN and LIST options noted in Section 2.4 of Volume I of this report.

7.2.6 Variable Names

| Variable | Description | Units |
|---|---|---|
| AVS | Absolute value of VS | |
| DIST | Distance of Plot Point Down a Line | IN |
| I | Counter | |
| IADD | Address J*NPTS | |
| ICHAR | Plot character | |
| ICHAR( ) | X and Y Axis Write Characters | |
| ISP | Counter | |
| ISPACE( ) | Temporary Variable for Writing X and Y Axis Scales | |
| ITEST | Counter | |
| J | Counter Indicating Plot Number | |
| L | Dummy Variable | |
| LINE | Integer Counter for Plot Line Number | |
| NABSQ | Integer value 1 or 0 Used as Indicator | |
| NCHAR | Dummy Variable Representing Plot Character | |
| NVAR | Dummy Variable Representing Point at which Line Plot is taken or Component Number | |
| SP | Column Number Nearest to the Ith Value of X-Variable | |
| VS | Dummy Variable | |

| Variable | Description | Units |
|---|---|---|
| XAX | Temporary Variable for Writing X Axis Scale Values | |
| XDELTA | Distance Between Stored Points on X Axis | |
| XMAX | Last (Largest) X Axis Value | |
| XMIN | First (Lowest) X Axis Value | |
| XSCALE | X Scale Range | |
| Y | Temporary Variable (Y Axis Scale Value) | |
| YDELTA | Distance Between Stored Points on the Y Axis | |
| YLAST | Last Y Axis Scale Value | |
| YLO | Lowest Value in Search Range | |
| YMAX | Maximum Value to be Plotted | |
| YMIN | Minimum Value to be Plotted | |
| YUP | Highest Value in Search Range | |

## 7.2.7 Subroutine Listing

```
      SUBROUTINE GRAPH
C**** REVISED AUGUST 5, 1975 ****
      COMMON NTELPL,NTOLPL,IPT,IPOINT,NPTS,INEL,KNEL,NTOPL,NLPLT(61,3),
     1 VSTORE(1)
      COMMON/SUB/PARM(150,9),PM(1500),QM(1500),P(300),Q(300),C(300)
     1,X(300),RHO(20),S2ORHO(20),VISC(20),BULK(20),TEMP(20),PVAP(20)
     2,ATPRES,T,DELT,TFINAL,PLTDEL,PI,TITLE(20),LEGN,ICON
     3,XTEMP(99),LSTART(150),NLPT(150),LTYPE(99),NC(99),INX,INZ
     4,INV,ISTEP,NLINE,NEL,IND,IENTR,MNLINE,MNEL,MNLEG,MNNODE,MNPLOT
     5,MNLPTS,MDS
      DIMENSION ISPACE(101),XAX(6),ICHART(8)
      DATA ICHART/1HO,1HP,1HC,1HI,1H-,1H+,1H ,1H*/
      DATA ITEST,XSCALE/0,0.0/
C-----BEGIN OUTER LOOP. FIND X PARAMETERS ON FIRST PASS ONLY
    1 DO 901 J=1, NTOPL
      IADD=J*NPTS
      IF(J.NE.1)GO TO 2
      XMAX=VSTORE(NPTS)
      XMIN=VSTORE(1)
      ITEST=ITEST+1
      IF(ITEST.NE.1) XMAX=XMIN+XSCALE
      XSCALE=XMAX-XMIN
C     CALL SCALED(XMAX,XMIN)
      XDELTA=(XMAX-XMIN)/100.
C-----FIND Y PARAMETERS
    2 YMAX=VSTORE(1+IADD)
      YMIN=YMAX
      DO 902 I=2, NPTS
      YMAX=AMAX1(YMAX,VSTORE(I+IADD))
  902 YMIN=AMIN1(YMIN,VSTORE(I+IADD))
      NABSQ=0
      IF(J.GT.NTOLPL.OR.NLPLT(J,1).GE.0) GO TO 905
      IF(INV.LO.0) GO TO 905
      IF(YMIN.GT.0) GO TO 905
      NABSQ=1
      IF(ABS(YMIN).GT.YMAX) YMAX=ABS(YMIN)
  905 IF(YMAX.GE.YMIN)GO TO 9020
      YMAX=YMAX + 25.
      YMIN=YMAX - 50.
      GO TO 9025
 9020 AMAX = (YMAX+YMIN)*.001
      IF((YMAX-YMIN).GT.AMAX) GO TO 9025
      YMAX = YMAX+AMAX
      YMIN = YMIN-AMAX
 9025 CALL SCALED(YMAX,YMIN)
      YDELTA=(YMAX-YMIN)/50.
C-----FIND LINE/COMPONENT NUMBER, TYPE OF PLOT, OUTPUT DATA
      L=NLPLT(J,2)
      IF (J.GT.NTOLPL) GO TO 5
      IF(NLPLT(J,1)) 3,5,4
```

7.2.7 (Continued)

```
      3 ICHAR=ICHART(1)
        DIST=NLPLT(J,3)*PARM(L,5)
        GO TO 6
      4 ICHAR=ICHART(2)
        DIST=NLPLT(J,3)*PARM(L,5)
        GO TO 6
      5 ICHAR=ICHART(3)
        NVAR=NLPLT(J,3)
C-----GO TO TOP OF NEXT PAGE
      6 WRITE(6,601)
C-----LOOP FOR EACH PLOT LINE.
        Y=YMAX + YDELTA
      7 DO 907 LINE=1, 51
        YLAST=Y
        Y=Y-YDELTA
        YUP=Y+YDELTA/2.
        YLO=Y-YDELTA/2.
C-----FIRST + LAST CHAR. ON LINE = *I*
        ISPACE(1)=ICHART(4)
        ISPACE(101)=ICHART(4)
C-----FIRST + LAST LINES ALL *-*, EXCEPT *+* IN 11,21,31,41,...,81,+91
        IF(LINE.NE.1 .AND. LINE.NE.51)GO TO 11
      9 DO 909 ISP=2,100
        IF((ISP-1).EQ.(ISP-1)/10*10)GO TO 10
        ISPACE(ISP)=ICHART(5)
        GO TO 909
     10 ISPACE(ISP)=ICHART(6)
    909 CONTINUE
        GO TO 14
C-----INITIALIZE COL. 2-100 ON LINES 2-50 TO * *, OR *+----+-* IF AXIS
     11 IF(Y.LE.0. .AND.YLAST.GT.0.)GO TO 13
     12 DO 912 ISP=2, 100
    912 ISPACE(ISP)=ICHART(7)
        GO TO 14
     13 DO 913 ISP=2,100
        ISPACE(ISP)=ICHART(5)
    913 IF((ISP-1).EQ.(ISP-1)/10*10)ISPACE(ISP)=ICHART(6)
C---- SEARCH Y-VALUE ARRAY FOR THOSE IN RANGE YLO.LT.VALUE.GE.YUP
     14 DO 914 I=1, NPTS
        VS=VSTORE(I+IADD)
        NCHAR=ICHAR
        IF(VS.GT.YLO .AND. VS.LE.YUP)GO TO 145
        IF(NABSQ.NE.1) GO TO 914
        AVS=ABS(VS)
        IF(AVS.LT.YLO.OR.AVS.GT.YUP) GO TO 914
        NCHAR=ICHART(8)
C-----FIND COLUMN NUMBER NEAREST TO I-TH VALUE OF X-VARIABLE WHEN SCALED
    145 SP=(VSTORE(I)-XMIN)/XDELTA + 1
        IF(SP-AINT(SP).GT.0.50) SP=SP + 0.50
        ISP=SP
```

7.2.7 (Continued)

```
C-----CHECK ISP. IF LT 0 OR GT 102, ERROR; IF 0, ADD 1; IF 102, SUBT. 1
      IF(ISP) 914, 15, 16
   15 ISP=1
      GO TO 18
   16 IF(ISP-102)18,17,914
   17 ISP=101
   18 ISPACE(ISP)=NCHAR
  914 CONTINUE
C-----LINES 1,11,21,31,41,+51 HAVE Y-VALUES; THESE LINES, PLUS LINES 6,
C     16,26,... ALSO HAVE *+* IN COL. 1+101 IF EMPTY
      IF((LINE-1).NE.(LINE-1)/5*5)GO TO 19
      IF(ISPACE(1).NE.ICHAR) ISPACE(1)=ICHART(6)
      IF(ISPACE(101).NE.ICHAR)ISPACE(101)=ICHART(6)
      IF((LINE-1).NE.(LINE-1)/10*10)GO TO 19
C-----WRITE OUT PLOT LINE, CONTINUE
      WRITE(6,602)Y, ISPACE
      GO TO 907
   19 WRITE(6,603)ISPACE
  907 CONTINUE
C-----CALCULATE + PRINT X-AXIS VALUES
   20 DO 920 I=1, 6
  920 XAX(I)=XMIN + (I-1)*20.*XDELTA
      WRITE(6,604) XAX
C-----WRITE LOWER TITLES + VALUES, REENTER OUTER LOOP
      IF (J.GT.NTOLPL) GO TO 23
      IF(NLPLT(J,1))21,23,22
   21 WRITE(6,605) J,DIST,L
      GO TO 900
   22 WRITE(6,606) J,DIST,L
      GO TO 900
   23 WRITE(6,607) J,NVAR,L
  900 CONTINUE
      WRITE(6,608)TITLL
  901 CONTINUE
 1000 WRITE(6,601)
      WRITE(6,610)
      WRITE(6,611)
      IF(T.LT.TFINAL-DELT)RETURN
      DO 1250 J=1,NTOPL
      L=NLPLT(J,2)
      IF(J.GT.NTOLPL) GO TO 50
      DIST=NLPLT(J,3)*PARM(L,5)
      IF(NLPLT(J,1)) 25,1250,30
   25 WRITE(6,605) J,DIST,L
      GO TO 1250
   30 WRITE(6,606) J,DIST,L
      GO TO 1250
   50 NVAR=NLPLT(J,3)
      WRITE(6,607) J,NVAR,L
 1250 CONTINUE
```

7.2.7 (Continued)

```
601 FORMAT(1H1)
602 FORMAT(1X,15X,F12.4,1X,101A1)
603 FORMAT(1X,28X,101A1)
604 FORMAT(1X,23X,5(F9.3,11X),F9.3)
605 FORMAT(1X,28X,6HGRAPH ,I3,1X,53H  FLOW (CU.IN/SEC) VS. TIME (SEC.)
   + FOR A DISTANCE OF ,F8.2,26H INCHES ALONG LINE NUMBER ,I5)
606 FORMAT(1X,28X,6HGRAPH ,I3,1X,53H  PRESSURE  (PSIA) VS. TIME (SEC).
   +FOR A DISTANCE OF ,F8.2,26H INCHES ALONG LINE NUMBER ,I5)
607 FORMAT(1X,28X,6HGRAPH ,I3,1X,18H  VARIABLE NUMBER ,I3,21H OF COMPO
   +NENT NUMBER ,I3,38H VS. TIME (SEC.).  THE VARIABLE IS ---  )
608 FORMAT(1X,28X,20A4)
610 FORMAT(1H0,65X,27HHYTRAN PROGRAM OUTPUT PLOTS)
611 FORMAT(1H0)
    RETURN
    END
```

## 7.3 SUBROUTINE SCALED

The subroutine SCALED is used by GRAPH to obtain a preferred scale for the X and Y axis of the print plot graphs.

The number of divisions on the X axis = 100 and the number of divisions on the Y axis = 50, a preferred scale system was chosen which would give a difference between RMAX and RMIN of either 1.0*10**N, 2.0*10**N or 5.0*10**N where N can be +ve or -ve.

The graph data MAX and MIN is centered between RMAX and RMIN unless either RMAX or RMIN can be set to zero.

An overriding requirement is that the scales should be at some reasonable number for easy reading hence with a range of 5000, the MIN can be set at intervals of 500, or range/10. This sometimes leads to a larger scale being used than would expected from the actual range.

The goal however was graphical readability and scalability without the need to resort to a calculator to find the value of a point, and in meeting this goal we have payed some penalty in the size of the actual graph.

7.3.1 Theory Not applicable.

7.3.2 Assumptions - Not applicable.

7.3.3 Computation - To be added later.

7.3.4 Approximation - Not applicable.

7.3.5 Limitations

In its present form SCALED gives inconsistent answers for small values of RMAX and PMIN, and is not currently used to scale the X axis.

7.3.6 Variable Names

| NAME | DESCRIPTION | DIMENSION |
|---|---|---|
| AMAX | Maximum value to be plotted | - |
| AMIN | Minimum value to be plotted | - |
| IBOT | Variable used to calculated Y axis scale values. | - |
| IEMAX | Variable used to calculate Y axis scale values | - |
| IEXP | Variable used to calculate Y axis scale values | - |
| ITOP | Variable used to calculate Y axis scale values | - |
| J | Integer counter | - |
| MANT | Variable used to calculate Y axis scale values | - |
| RANGE | Range of values to be plotted | - |
| RMAX | Maximum Y axis scale value | - |
| RMIN | Minimum Y axis scale value | - |
| SCALE(-) | Scale factors for Y axis | - |

## 7.3.7 Subroutine Listing

```
      SUBROUTINE SCALED(RMAX,RMIN)
      DIMENSION SCALE(6)
      DATA SCALE/.5,1.,2.,5.,10.,20./
C-----FIND THE RANGE OF VALUES *RANGE*, AND PLACE ACTUAL MAX AND MIN
C-----POINTS IN *AMAX* AND *AMIN*
      RANGE=RMAX-RMIN
      AMAX=RMAX
      AMIN=RMIN
C-----FIND AN INTEGER EXPONENT *IEXP* AND BASE *MANT* SUCH THAT THE
C-----VALUE OF MANT**IEXP IS .GE. RANGE
      IEXP=ALOG10(RANGE)
      MANT=RANGE/10.**IEXP
      IF(RANGE.GT.MANT*10.**IEXP)MANT=MANT+1
C-----USING MANT, SELECT ONE OF THE PREFERRED SCALES
      IF(MANT.GT.10)GO TO 70
      IF(MANT.LT.1) GO TO 70
      GO TO(80,90,100,100,100,110,110,110,110,70), MANT
   70 MANT=1
      IEXP=IEXP+1
   80 J=2
      GO TO 120
   90 J=3
      GO TO 120
  100 J=4
      GO TO 120
  110 J=5
C-----SET *IEMAX* EQUAL TO THE EXPONENT OF 10. CORRESPONDING TO RMAX
  120 IF(RMAX.EQ.0.) GO TO 121
      IEMAX=ALOG10(ABS(RMAX))
C-----USE AMAX AND IEMAX TO FIND A POSSIBLE MAXIMUM VALUE FOR THE
C-----SCALE.  PLACE THE VALUE IN RMAX, AND COMPARE WITH THE ACTUAL
C-----MAXIMUM POINT.
      RMAX=INT(ABS(AMAX)/10.**IEMAX)*10.**IEMAX*SIGN(1.0,AMAX)
  121 IF(RMAX.GE.AMAX)GO TO 130
C-----IF RMAX IS .LT. ACTUAL MAX POINT, INCREASE IT BY 5
C-----PERCENT AND RECHECK--REPEAT AS NECESSARY
      RMAX=RMAX+.05*SCALE(J)*10.**IEXP
      GO TO 121
C-----SET THE SCALE'S MINIMUM BY SUBTRACTING SCALE(J)**IEXP FROM RMAX
C-----IF THE ACTUAL MINIMUM *AMIN* LIES WITHIN THE RANGE NOW DEFINED
C-----BY RMAX AND RMIN, CONTINUE.
  130 RMIN=RMAX-SCALE(J)*10.**IEXP
      IF(RMIN.LE.AMIN)GO TO 150
C-----GO TO THE NEXT LARGEST SCALE, RECALCULATE RMIN, AND RECHECK
      J=J+1
      IF(J.LT.5.5)GO TO 130
      J=1
      IEXP=IEXP+1
      GO TO 130
C-----IF THE SCALE'S MIN IS .LT. ZERO, BUT THE ACTUAL MIN IS POSITIVE,
```

7.3.7 (Continued)

```
C ----SHIFT THE SCALE UP SO THAT THE SCALE BEGINS AT ZERO
  150 IF(RMIN*AMIN.GT.0.)GO TO 170
      RMIN=0.
  160 RMAX=SCALE(J)*10.**IEXP
C-----DUE TO THE SHIFT, IT MAY BE POSSIBLE TO DECREASE THE SCALE TO
C-----THE NEXT SMALLEST SIZE
      IF(AMAX.GT.1.000001*SCALE(J-1)*10.**IEXP) RETURN
      J=J-1
      IF(J.GT.1.5) GO TO 160
      J=4
      IEXP=IEXP-1
      GO TO 160
C-----IF RMIN IS POSITIVE AND NEAR ZERO, SHIFT THE SCALE DOWN TO 0. MIN
  170 IF(RMIN.LT.0.)GO TO 175
      IF(RMIN.GT..1*RMAX)GO TO 180
      RMIN=0.
      RMAX=SCALE(J)*10.**IEXP
C-----IF THE SHIFT DOWN CAUSES RMAX TO LIE BELOW THE ACTUAL MAX,
C-----INCREASE THE SCALE RANGE TO THE NEXT LARGEST
      IF(RMAX.LT.AMAX)RMAX=SCALE(J+1)*10.**IEXP
      RETURN
C-----IF RMAX IS NEGATIVE AND NEAR ZERO, SHIFT THE SCALE UP TO 0. MAX
  175 IF(RMAX.GE.0.) GO TO 180
      IF(-RMAX.GT.-0.1*RMIN) GO TO 180
      RMAX=0.
      RMIN=-SCALE(J)*10.**IEXP
C-----IF THE SHIFT UP CAUSES RMIN TO LIE ABOVE THE ACTUAL MIN,
C-----INCREASE THE SCALE RANGE TO THE NEXT LARGEST
      IF(RMIN.GT.AMIN) RMIN=-SCALE(J+1)*10.**IEXP
      RETURN
C-----CENTER THE SCALE ABOUT THE ACTUAL RANGE OF POINTS
  180 ITOP=(RMAX-AMAX)/(.05*SCALE(J)*10.**IEXP)
      IBOT=(AMIN-RMIN)/(.05*SCALE(J)*10.**IEXP)
      IDIF=(ITOP-IBOT)/2
      IF(IDIF.EQ.0) RETURN
      RMIN=RMIN-IDIF*.05*SCALE(J)*10.**IEXP
      RMAX=RMIN+SCALE(J)*10.**IEXP
      RETURN
      END
```

## 8.0 UTILITY SUBROUTINES

The utility subroutines have been added to avoid some of the annoying problems encountered when a program is transferred to another system which may have similar but incompatible library routines.

INTERP and LAGRAN both of which started as library routines, have been modified to cut running costs wherever possible.

A skilled user can probably replace these routines with local library routines and operate efficiently.

## 8.1 INTERP SUBROUTINE

The INTERP subroutine provides interpolation for continuous or discontinuous functions of the form Y = f(X). INTERP is a shortened version of a MCAUTO library functional subroutine named DISCOT.

INTERP uses two other subroutines, DISER1 and LAGRAN, to derive the dependent variable from tabulated data input by the programmer or already existing in the program subroutine. Subroutine DISER1 gives the data points around the X variable. Lagrange's interpolation formula is used in the subroutine LAGRAN to obtain a Y value. For an X value lying outside the range of the tabulated data, the Y value will be extrapolated. Fluid viscosities are calculated using a modified Walther equation.

8.1.1 Solution Method. The INTERP subroutine provides the necessary control parameters to DISER1 and LAGRAN to yield a dependent variable. The subroutine arguments are:

Subroutine INTERP (X, TABX, TABY, NC, NY, Y, IND)

where:

X - Argument of function Y = f(X)

TABX - X array of independent variables in ascending order

TABY - Y array of dependent variables in ascending order

NC - Control word

| | |
|---|---|
| Tens Digit - | Degree of interpolation |
| Units Digit - | = 1 Walther equation |
| | = 0 LAGRAN interpolation |

NY - Number of data points in the Y array

Y - Dependent variable

IND - Error indicator

0 = Normal interpolation

1 = Extrapolation outside range of data points.

8.1.2 Assumptions. Not applicable

8.1.3 Computations. The degree of interpolation will be decoded from the control word NC in the INTERP subroutine argument and passed to DISER1. The error indicator IND is set to zero. On finding the data point closest to the X value from DISER1, it is entered into the LAGRAN subroutine argument. If the modified Walther equation is to be used for a viscosity calculation, IDX will be set equal to -1.

8.1.4 Approximations. Not applicable

8.1.5 Limitations. The X and Y data points must be entered in an ascending order. When tabulating a discontinuous function the independent variable (X) at the point of discontinuity is repeated, i.e.,

$X_1, X_2, X_3, X_3, X_4, X_5$

$Y_1, Y_2, Y_3, Y_4, Y_5, Y_6$

Thus for discontinuous functions there must be K + 1 points above and below the discontinuity, where K is the degree of interpolation.

8.1.6 INTERP Variable Names.

| Variables | Description | Dimensions |
|---|---|---|
| IDX | Degree of interpolation | - |
| IND | Solution indicator | - |
| | = 0 Normal interpolation | |
| | = 1 Extrapolation outside of data range | |
| NC | Control word | - |
| NPX | Dummy array | - |
| NPX1 | Location of data point X, Y for interpolation | - |
| NY | Number of Y data points | - |
| TABX | X array of data points | - |
| TABY | Y array of data points | - |
| X, XA | Independent variable | - |
| Y | Dependent variable | - |

8.1.7 Subroutine Listing

```
      SUBROUTINE INTERP(X,TABX,TABY,NC,NY,Y,IND)
      DIMENSION TABX(1),TABY(1),NPX(9)
      IDX=(NC-(NC/100)*100)/10
      IND=0
      XA=X
      CALL DIGLST(XA,TABX,1,NY,IDX,NPX,IND)
      NPY1=NPY(1)
      IF((NC-IDX*10).EQ.1)IDX=-1
      IF((NC-IDX*10).EQ.2)IDX=-2
      CALL LAGRA (XA,TABX(NPY1),TABY(NPY1),IDX+1,Y)
      RETURN
      END
```

## 8.2 DISEk1 SUBROUTINE

The subroutine DISER1 will return the array location of the lower bound value of the interval in which the independent variable lies. DISER1 is a modification of a MCAUTO library subroutine named DISSER.

The arguments for the DISER1 subroutine are as follows:

Subroutine DISER1 (XA, TAB, 1, NX, 1D, NPX, IND)

XA - Independent variable

TAB - X array

I - Tabulated data location

NX - Number of points in the independent array

ID - Degree of interpolation

NPX - Location of lower bound for data point XA, in the TAB array

IND - Indicator

8.2.1 Solution Method. Not applicable

8.2.2 Assumptions. Not applicable

8.2.3 Computations. On entry of the independent variable, XA, and the tabulated data form the TABX array, DISER1 will find the tabulated data values that bound XA, and return the smaller one to the calling program. If XA were to lie outside the lower end of the data, DISER1 would return the first data point as the lower bound. Should XA lie outside the upper tabulated value, the second from the last data point location will be returned by DISER1.

8.2.4 Approximations. Not applicable

8.2.5 Limitations. Not applicable

### 8.2.6 DISER1 Variable Names

| Variable | Description | Dimensions |
|---|---|---|
| IND | Solution indicator | - |
| I, ID, IT, J, NLOC, NLOW, NPB, NPT, NPU, NPX, NUPP, NY, NXX | Integer counters | - |
| TAB | Array of independent variables | - |
| XA | Independent variable | - |

## 8.2.7 Subroutine Listing

```
      SUBROUTINE DISER1(XA,TAB,I,NX,ID,NPX,IND)
      DIMENSION TAB(1)
      IF(XA-TAB(I))71,73,72
71    IND=IND+1
      NPX=I
      RETURN
73    XA=TAB(I)
      NPX=I
      RETURN
72    J=I+NX-1
      IF(XA-TAB(J))1,77,76
76    IND=IND+1
      NPX=J-ID
      RETURN
77    XA=TAB(J)
      NPX=J-ID
      RETURN
1     NPT=ID+1
      NP2=NPT/2
      NPU=NPT-NP2
      IF(NX-NPT)4,5,10
4     ID=NX-1
      GO TO 1
5     NPX=I
      RETURN
10    NLOW=I+NP2
      NUPP=I+NX-(NPU+1)
      IF(NX-20)18,18,11
11    NXX=NX/2+I
      IF(XA-TAB(NXX))12,17,13
12    NXX=NXX-NX/4
      IF(XA-TAB(NXX))12,17,17
13    NXX=NXX+NX/4
      IF(XA-TAB(NXX))14,17,17
14    NLOW=NXX-NX/4
      GO TO 18
17    NLOW=NXX
18    DO 19 II=NLOW,NUPP
      NLOC=II
      IF(TAB(II)-XA)19,20,20
19    CONTINUE
      NPX=NUPP-NP2+1
      RETURN
20    NPX=NLOC-NP2
      RETURN
      END
```

## 8.3 LAGRAN SUBROUTINE

The LAGRAN subroutine will interpolate or extrapolate a data point from two known tabulated values. In addition, LAGRAN will calculate viscosity using a modified Walther equation. The LAGRAN subroutine arguments are:

Subroutine LAGRAN (XA, X, Y, N, ANS)

XA - Independent variable

X - X array

Y - Y array

N - Degree of interpolation

ANS - Dependent varaible

8.3.1 Math Model. LAGRANGES interpolation equation is used in this subroutine to calculate the dependent variable. The LAGRANGE formula is:

$$P(x) = \sum_{i=0}^{m} L_i(x)\, y_i \qquad (1)$$

Where:

$L_i(x)$ is the Lagrange multiplier function.

$$L_i(x) = \frac{(x-x_0)\,(x-x_1)\cdots(x-x_{i-1})\,(x-x_{i+1})\cdots(x-x_n)}{(x_i-x_0)\,(x_i-x_1)\cdots(x_i-x_{i-1})\,(x_i-x_{i+1})\cdots(x_i-x_n)} \qquad (2)$$

The LAGRANGE equation will generate a polynomial between two data points. The degree or the polynominal will be that specified by the index value N. The dependent variable will be returned as ANS in the subroutine argument.

A modified version of the Walther equation is used in the calculation of viscosity. The ASTM charts are based on this equation.

$$\text{LOG}\,[\text{LOG}\,(\nu+c)\,] = A\ \text{LOG}\ {}^{\circ}R + B \qquad (3)$$

Where:

c = a constant

°R = Temperature, °RANKINE

$\nu$ = Viscosity, cSt

A,B = Constants for each fluid

LOG = Log to the base 10

The ASTM chart expresses c as a constant varying from 0.75 at 0.4 cSt to 0.6 at 1.5 cSt and above.

8.3.2 Assumptions. The Lagrangian equation generated by the subroutine will only use the data points around the dependent variable to generate a polynomial for interpolation. The last or first set of two data points will be used for extrapolation. The equation used to determine the viscosity uses a constant factor that is applicable to viscosity values of 2 centistokes or more.

8.3.3 Computation. The procedure LAGRAN will perform whether it be interpolation or the viscosity calculation, will always be recognized by testing the N argument in the subroutine statement. If N is equal to zero, then the viscosity will be calculated using the modified Walther equation. Otherwise N will specify the degree of interpolation to be used by the Lagrange formula. Both results will be returned to the calling program through the variable named ANS. The LAGRAN interpolation is a direct application of equation (1) to the given data.

Before evaluating the viscosity equation (3) for the viscosity value at XA temperature, the constants A and B must be calculated. They are solved using the data points that surround the dependent variable, or the first or last set of two data points if the dependent variable lies outside the range of the tabulated data. With the constants calculated for this fluid the viscosity can be computed from Equation (3).

8.3.4 Approximations. In the viscosity calculation, 0.6 was used as a constant factor for all ranges of viscosity. See reference 9.6 for a more thorough discussion.

8.3.5 Limitations. Since the LAGRANGE method only uses two data points to interpolate it can become inaccurate for remotely spaced tabulated data points. Any degree of interpolation greater than two can lead to erroneous results.

For the viscosity equation, any computed value of viscosity less than 2 centistokes cannot be considered accurate, and should be weighed in the final results.

8.3.6 LAGRAN Subroutine Variable Names

| Variable | Description | Dimensions |
|---|---|---|
| A | Constant for viscosity | - |
| ANS | Dependent variable | - |
| B | Constant for viscosity | - |
| I,J | Integer counters | - |
| N | Method of solution<br>N = 0 Viscosity calculation<br>N > 0 Degree of interpolation | - |
| P/QD | Lagrange partial product | - |
| P1 | LOG LOG of (Y (1) + C) | cSt |
| P2 | LOG LOG of (Y (2) + C) | cSt |
| T1 | LOG of T (1) | °R |
| T2 | LOG of T (2) | °R |
| X | X-array | - |
| XA | Independent variable | - |
| Y | Y-array | - |

### 8.3.7 Subroutine Listing

```
      SUBROUTINE LAGRAN(XA,X,Y,N,ANS)
      DIMENSION X(1),Y(1)
      IF(N.EQ.-1) GO TO 20
      IF(N.EQ.0)GO TO 10
      SUM=0.0
      DO 3 I=1,N
      PROD=Y(I)
      DO 2 J=1,N
      A=X(I)-X(J)
      IF(A) 1,2,1
    1 B=(XA-X(J))/A
      PROD=PROD*B
    2 CONTINUE
    3 SUM=SUM+PROD
      ANS=SUM
      RETURN
C     VISCOSITY CALCULATION
   10 CONTINUE
      A1=0.
      IF(Y(1).LE.2.)A1=EXP(-1.47-1.84*Y(1)-.51*Y(1)**2)
      A2=0.
      IF(Y(2).LE.2.)A2=EXP(-1.47-1.84*Y(2)-.51*Y(2)**2)
      P1=ALOG10(ALOG10(Y(1)+.7+A1))
      P2=ALOG10(ALOG10(Y(2)+.7+A2))
      T1=ALOG10(X(1)+460.)
      T2=ALOG10(X(2)+460.)
      B=(P1-P2)/(T2-T1)
      A=P1+B*T1
      Z=10**(10**(A-B*ALOG10(XA+460.)))
      IF(Z.LE.2.7)GO TO 11
      ANS=Z-.7
      RETURN
   11 ANS=(Z-.7)-EXP(-.7187-3.295*(Z-.7)+.5110*(Z-.7)**2
     +-.3193*(Z-.7)**3)
      RETURN
C     FLUIDE CALCULATION
   20 CONTINUE
      P1=ALOG10(ALOG10(Y(1)+.6))
      P2=ALOG10(ALOG10(Y(2)+.6))
      T1=ALOG10(X(1)+460.)
      T2=ALOG10(X(2)+460.)
      B=(P1-P2)/(T2-T1)
      A=P1+B*T1
      V0=10**(10**(A-B*ALOG10(XA+460.)))-.6
      T5=10**((.125980+A)/B)
      T1000=10**((-.477159+A)/B)
      S=ALOG10((T5)/100.+1.)-ALOG10((T1000)/100.+1)
      DELX=65.10979*S
      S=6.65/DELX
      ALPHA=3.23523-11.3896*S+13.1735*S*S-4.8831*S*S*S
```

8.3.7 (Continued)

```
      BETA=-5.33425+19.9521*S-23.9448*S*S+10.155*S*S*S
      CHI=3.35452-13.1273*S+17.1712*S*S-7.6551*S*S*S
      ANS=ALPHA+BETA*ALOG10(V0)+CHI*(ALOG10(V0))**2
      IF(ANS.LT.0.)ANS=0
      RETURN
      END
```

## 8.4 SIMULT SUBROUTINE

SIMULT is a Fortran library subroutine (Reference 9.3) that solves systems of N linear algebraic equations with N unknowns. SIMULT employs Gaussian elimination and positioning for size using the largest pivotal divisor as the solution process.

8.4.1 Solution Method - A system of linear equations may be written:

$$\begin{aligned} a_{11}x_1 &+ a_{12}x_2 + \ldots + a_{1n}x_m = b_1 \\ a_{21}x_1 &+ a_{22}x_2 + \ldots + a_{2n}x_m = b_2 \\ \cdot \quad & \quad \cdot \quad \ldots \quad \cdot \quad \cdot \\ \cdot \quad & \quad \cdot \quad \ldots \quad \cdot \quad \cdot \\ a_{m1}x_1 &+ a_{m2}x_2 + \ldots + a_{mm}x_n = b_m \end{aligned} \qquad (1)$$

Rewriting in matrix form:

$$\begin{bmatrix} a_{11} & a_{12} & \ldots & a_{1m} \\ a_{21} & a_{22} & \ldots & a_{2m} \\ \cdot & \cdot & \ldots & \cdot \\ \cdot & \cdot & \ldots & \cdot \\ a_{m1} & a_{m2} & & a_{mm} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \cdot \\ \cdot \\ x_m \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \\ \cdot \\ \cdot \\ b_m \end{bmatrix} \qquad (2)$$

Equation (2) may be further simplified by writing:

$$AX = B \qquad (3)$$

where

A = M * M Matrix of coefficients

B = M Matrix of Constants

X = M Matrix of M unknowns in the system.

The solution of a set of simultaneous linear equations as in (1) is by Gaussian elimination using pivoting. Each stage of elimination consists of interchanging rows when necessary to avoid division by zero or small elements. The forward solutio to obtain variable M is done in M stages. The back solution for the other variables is calculated by successive substitutions. Final solution values are developed in matrix B, with variable 1 in B(1), variable 2 in B(2),..., variable M in B(M). If no pivot can be found exceeding a tolerance of 0.0, the matrix is considered singular. The arguments for SIMULT are as follows:

Subroutine SIMULT (CALC1, CALC2, M, J)

where:

CALC1 = A

CALC2 = B

M = number of equations

J = solution indicator

J = 1 when no solution can be found - equations are singular

J = 0 for a normal solution

Both the original CALC1 and CALC2 Matrices are destroyed in the computation. The answers are returned through the CALC2 Matrix.

8.4.2 Assumptions - The basic assumption used in the solution of simultaneous linear equations involves the ability to actually linearize the complex mathematical system that is being described. If this can reasonably be done then a set of equations as in (1) may be written and solved.

8.4.3 Computation - Gaussian elimination and positioning for size using the largest pivotal divisor is used. Positioning for size or pivoting will ordinarily reduce some of the roundoff errors in the solution and may actually allow some ill-conditioned systems to be solved. See Appendix D SSFAN Technical Manual (MDC A3059 Vol II) for a more thorough discussion of this method.

8.4.4 Approximations - The approximations are inherent in the use of the Gaussian elimination procedure as described in Appendix D of the SSFAN Technical Manual.

8.4.5 Limitations - If no equation in the set (1) is a linear combination of the others, the system of equations is said to be linearly independent and a unique solution exists for the unknowns. A system of equations are homogeneous if each $b_i$ in B (EQN 2) is equal to zero. The Gaussian elimination method will provide a unique solution to equation (3) when the corresponding homogenous system has only the solution X = 0. Both systems AX = B and AX = 0 as well as the coefficient matrix A are then termed non-singular. When AX = 0 has solutions other than zero, the two systems and matrix A are termed singular. This results in AX = B either having no solution or an infinite number of solutions.

### 8.4.6 SIMULT Subroutine Variable Names

| Variables | Description | Dimensions |
|---|---|---|
| A | N*N Matrix of Coefficients | -- |
| B | N matrix of constants | -- |
| BIGA | Largest element | -- |
| IA,IB,IC,IJ<br>IMAX,IQS,IT<br>IX,IXJ,IXJX,<br>I1,I2,J,JJ,JJX,<br>JX,JY,K,NY | Integer counters | -- |
| N | Number of unknowns | -- |
| SAVE | Temporary storage location | -- |
| TOL | Tolerance | -- |

## 8.4.7 Subroutine Listing

```
      SUBROUTINE SIMULT(A,B,N,KS)
C**** REVISED SEPTEMBER 3,1974 ****
      DIMENSION A(1),B(1)
      TOL=0.0
      KS=0
      JJ=-N
      DO 65 J=1,N
      JY=J+1
      JJ=JJ+N+1
      BIGA=0
      IT=JJ-J
      DO 30 I=J,N
      IJ=IT+I
      IF(ABS(BIGA)-ABS(A(IJ)))20,30,30
   20 BIGA=A(IJ)
      IMAX=I
   30 CONTINUE
      IF(ABS(BIGA)-TOL)35,35,40
   35 KS=1
      RETURN
   40 I1=J+N*(J-2)
      IT=IMAX-J
      DO 50 K=J,N
      I1=I1+N
      I2=I1+IT
      SAVE=A(I1)
      A(I1)=A(I2)
      A(I2)=SAVE
   50 A(I1)=A(I1)/BIGA
      SAVE=B(IMAX)
      B(IMAX)=B(J)
      B(J)=SAVE/BIGA
      IF(J-N)55,70,55
   55 IQS=N*(J-1)
      DO 65 IX=JY,N
      IXJ=IQS+IX
      IT=J-IX
      DO 60 JX=JY,N
      IXJX=N*(JX-1)+IX
      JJX=IXJX+IT
   60 A(IXJX)=A(IXJX)-(A(IXJ)*A(JJX))
   65 B(IX)=B(IX)-(B(J)*A(IXJ))
   70 NY=N-1
      IT=N*N
      DO 80 J=1,NY
      IA=IT-J
      IB=N-J
      IC=N
      DO 80 K=1,J
      B(IB)=B(IB)-A(IA)*B(IC)
```

8.4.7 (Continued)

```
      IA=IA-N
   30 IC=IC-1
      RETURN
      END
```

## 8.5 SBRAN SUBROUTINE

The SBRAN subroutine is a special utility program which enables the programmer to input lines without having to separate them with a component or end them with a component.

### 8.5.1 MATH MODEL

Not applicable.

### 8.5.2 ASSUMPTIONS

Not applicable.

### 8.5.3 COMPUTATION

1000 SECTION

All the connection data for the lines is checked in the LC array. Should a line be dead-ended, a minus one will be in the appropriate address of the LC array. The number of dead ends are counted and the address of the downstream end of the line number is stored in the LL array.

Next the lines are checked to see if they are connected to each other. This occurs when a one is found in the downstream address of one line and the upstream address of another in the LC array. The address is stored in the LL array and a counter, NDENDI, is incremented for each line.

If the sequences of -1 -1, 0, 0, or +1, +1 are not followed for the line data, the program will stop execution.

3000 SECTION

For a dead-ended line the address of the lines pressure and flow data is found in the LL array. The flow in the line is set to zero and the end pressure is set to the characteristic pressure.

The LL array also contains the addresses of the P, Q, Z and L variables for any lines that are connected.

The flow is computed as

$Q(I)=(C(I)-C(JJ))/(Z(I)+Z(JJ)$

$Q(JJ)=-Q(I)$

where:

I denotes the upstream line end point

JJ denotes the downstream lines beginning

The pressures are calculated easily as

$P(I)=C(I)-Q(I)*Z(I)$

$P(JJ)=P(I)$

8.5.4 APPROXIMATIONS

Not applicable.

8.5.5 LIMITATIONS

SBRAN does not take into account the pressure drops or rises that result from any changes in cross sectional area going from one line to another.

8.5.6 VARIABLE NAMES

| Variable | Description |
|---|---|
| I | Downstream Line Address |
| JJ | Upstream Line Address |
| KFAIL | Error Indicator |
| NLINE | Number of Lines |
| NDENDS | Number of Dead-Ended Lines |
| NDUND1 | Counter for Connected Lines |

## 8.5.7 Subroutine Listing

```
      SUBROUTINE SBRAN
C *** REVISED DECEMBER 5, 1975 ***
      COMMON/SUB/PAR1(150,9),PM(1500),QM(1500),P(300),Q(300),C(300)
     1,Z(300),RHO(20),S2ORHO(20),VISC(20),BULK(20),TEMP(20),PVAP(20)
     2,ATPRES,T,DELT,TFINAL,PLTDEL,PI,TITLE(20),LEGN,ICON
     3,XTEMP(99),LSTART(150),NLPT(150),LTYPE(99),NC(99),INX,INZ
     4,INV,ISTOP,NLINE,NEL,IND,IENTR,MNLINE,MNEL,MNLEG,MNMODL,MNPLOT
     5,MNLPTS,NDS
      DIMENSION LL(55),LC(1)
      EQUIVALENCE (LC(1),C(1))
      IF(IENTR)1000,2000,3000
 1000 CONTINUE
      J=0
      DO 100 I=1,NLINE
      JJ=I*2
      IF(LC(JJ).NE.-1) GO TO 100
      J=J+1
      LL(J)=JJ
  100 CONTINUE
      NDLPTS=J
      NFAIL=0
      J=NLINE-1
      DO 200 I=1,J
      JJ=I*2
      IF(LC(JJ).EQ.-1) GO TO 100
      IF(LC(JJ).NE.LC(JJ+1)) GO TO 150
      IF(LC(JJ).EQ.0) GO TO 200
      J=J+1
      LL(J)=JJ
      GO TO 200
  150 IR=I+1
      WRITE(6,4000) I,IR
 4000 FORMAT(5X,I3,5X,I3)
      NFAIL=1
      GO TO 200
  100 IF(LC(JJ+1).NE.0) GO TO 150
  200 CONTINUE
      NDLPTS=NDLPTS+1
      IF(NFAIL.EQ.1)WRITE(6,999)
  999 FORMAT(10X,32HPROGRAM STOP IN SUBROUTINE SBRAN)
      IF(NFAIL.EQ.1) STOP 6002
 2000 RETURN
 3000 CONTINUE
      IF(NDLPTS.EQ.0) GO TO 3025
      DO 3020 J=1,NDLPTS
      I=LL(J)
      Q(I)=0.0
 3020 P(I)=C(I)
 3025 IF(NDLPTS.EQ.J) GO TO 3060
      DO 3050 J=NDLPT1,J
```

8.5.7 (Continued)

```
      I=LL(M)
      JJ=I+1
      Q(I)=(C(I)-C(JJ))/(Z(I)+Z(JJ))
      Q(JJ)=-Q(I)
      P(I)=C(I)-Q(I)*Z(I)
      P(JJ)=P(I)
 3050 CONTINUE
 3060 RETURN
      END
```

## 8.6 SUBROUTINE XLIMIT

XLIMIT is a utility subroutine which provides the calling program with information to determine if a limit has been reached. The subroutine is typically used for components with mechanical movement and returns position and velocity data.

### 8.6.1 MATH MODEL

Not applicable.

### 8.6.2 ASSUMPTIONS

Not applicable.

### 8.6.3 COMPUTATION METHOD

Minimum (POSMIN) and maximum (POSMAX) limits are input along with the current values of position (POS) and velocity (VEL) from the calling program. Initially the sign is set to zero and the position is compared against POSMAX.

If POS is greater than or equal to POSMAX, the position is set to POSMAX and ASIGN is set to 1. Should VEL be greater than zero it is zeroed and a return is made to the calling program.

If POS is less than POSMAX it is checked against POSMIN. When POS is less than or equal to POSMIN, POS is set to POSMIN, ASIGN equals -1 and the velocity is zeroed if it is less than zero.

Should POS be greater than POSMIN a return is made to the calling program without any position or velocity changes.

### 8.6.4 APPROXIMATIONS

Not applicable.

### 8.6.5 LIMITATIONS

Not applicable.

### 8.6.6 VARIABLE NAMES

| Variable | Description |
|---|---|
| ASIGN | Sign (-1, 0, 1) |
| POS | Position |
| POSMIN | Minimum Position |
| POSMAX | Maximum Position |
| VEL | Velocity |

### 8.6.7 Subroutine Listing

```
      SUBROUTINE XLIMIT(POS,VEL,ASIGN,POSMIN,POSMAX)
      ASIGN=0.0
      IF(POS-POSMAX) 20,10,10
   10 POS=POSMAX
      ASIGN=1.0
      IF(VEL.GT.0.0) GO TO 40
      GO TO 50
   20 IF(POS-POSMIN) 30,30,50
   30 POS=POSMIN
      ASIGN=-1.0
      IF(VEL.GE.0.0) GO TO 50
   40 VEL=0.0
   50 RETURN
      END
```

8.6.7 (Continued)

```
      SUBROUTINE DLIMIT(POS,VEL,ASIGN,POSMIN,POSMAX)
      DOUBLE PRECISION POS,VEL,ASIGN
      ASIGN=0.0D0
      IF(POS-POSMAX) 20,10,10
   10 POS=POSMAX
      ASIGN=1.0D0
      IF(VEL.GT.0.0D0) GO TO 40
      GO TO 50
   20 IF(POS-POSMIN) 30,30,50
   30 POS=POSMIN
      ASIGN=-1.0D0
      IF(VEL.GE.0.0D0) GO TO 50
   40 VEL=0.0D0
   50 RETURN
      END
```

8.6.7 (Continued)

```
      FUNCTION SQRABS(X)
      IF(X)100,200,300
  100 SQRABS=-SQRT(-X)
      GO TO 400
  200 SQRABS=0.0
      GO TO 400
  300 SQRABS=SQRT(X)
  400 RETURN
      END
```

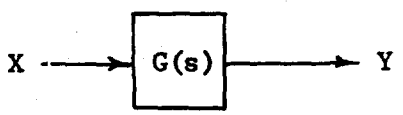
## 8.7 SUBROUTINE TUSTIN/FUNCTION DYNAM

The Tustin subroutine provides coefficients to approximate difference forms of continuous transfer functions for digital simulation. The Z-plane equivalents of the integrator, lag, or lag and washout for S-plane continuous functions can be selected.

Function DYNAM performs the actual computation for the chosen transfer function.

### 8.7.1 MATH MODEL

For a general function G(s) with input X and output Y,




The output at any time Yn can be approximated by a difference equation

$$Y_n = C_1 Y_{N-1} + C_2 X_N + C_3 X_{N-1}$$

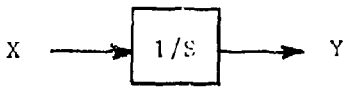
where $C_1$, $C_2$, $C_3$ are constants for a particular set of time constants ($\tau$) and clock time ($\Delta T$) and need only be computed one time, in the initialization section of the program.

The derivation of the integrator, lag and lag and washout are presented below.

# DIGITAL SIMULATION OF CONTINUOUS TRANSFER FUNCTIONS USING TUSTIN'S METHOD

Substitute $\frac{\Delta T}{2}\left[\frac{1+\zeta^{-1}}{1-\zeta^{-1}}\right]$ for $1/S$

1. INTEGRATOR




$$\frac{Y}{X} = \frac{1}{S} = \frac{\Delta T}{2}\left[\frac{1+\zeta^{-1}}{1-\zeta^{-1}}\right]$$

$$Y\left[1-\zeta^{-1}\right] = X\ \frac{\Delta T}{2}\left[1+\zeta^{-1}\right]$$

$$Y - Y_{-1} = \frac{\Delta T}{2}\left[X + X_{-1}\right]$$

$$Y = Y_{-1} + \frac{\Delta T}{2}\left[X + X_{-1}\right]$$

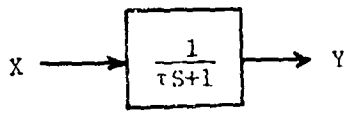
For $Y = C_1 Y_{-1} + C_2 X + C_3 X_{-1}$

$C_1 = 1$

$C_2 = \Delta T/2$

$C_3 = \Delta T/2$

2. LAG




$$\frac{Y}{X} = \frac{1}{\tau s+1} = \frac{1}{\tau\left[\frac{1}{s/s}\right] + 1}$$

$$= \frac{1}{\tau\left[\frac{2}{\Delta T}\left(\frac{1-\zeta^{-1}}{1+\zeta^{-1}}\right)\right] + 1}$$

$$\frac{Y}{X} = \frac{\left(\Delta T/2\right)/\tau\left)\left(1+\zeta^{-1}\right)\right.}{\left(1-\zeta^{-1}\right) + \left((\Delta t/2)/\tau\right)\left(1+\zeta^{-1}\right)}$$

$$\frac{Y}{X} = \frac{\left((\Delta T/2)/\tau\right)\left(1+\zeta^{-1}\right)}{\left(1+\Delta T/2)/\tau\right) - \left(1-\Delta T/2)/\tau\right)\zeta^{-1}}$$

$$\frac{Y}{X} = \frac{\left[\left(\Delta T/2\right)/\tau\right)\Big/\left(1+\left(\Delta T/2\right)/\tau\right)\right]\left(1+\zeta^{-1}\right)}{1 - \left[\frac{\left(1-\left(\Delta T/2\right)/\tau\right)}{\left(1+\left(\Delta T/2\right)/\tau\right)}\right]\zeta^{-1}}$$

$$\frac{Y}{X} = \frac{\frac{1}{\left(\tau\left(\Delta T/2\right)+1\right)}\left(1+\zeta^{-1}\right)}{1 - \left[\frac{\left(\tau/\left(\Delta T/2\right)-1\right)}{\left(\tau/\left(\Delta T/2\right)+1\right)}\right]\zeta^{-1}}$$

$$Y \left\{ 1 - \left[ \frac{\tau(\Delta T/2) - 1}{\tau/(\Delta T/2) + 1} \right] \zeta^{-1} \right\} = X \left[ \frac{1}{\tau/(\Delta T/2) + 1} \right] (1 + \zeta^{-1})$$

$$Y - \left[ \frac{\tau/(\Delta T/2) - 1}{\tau/(\Delta T/2) + 1} \right] Y_{-1} = \frac{1}{\tau/(\Delta T/2) + 1} (X + X_{-1})$$

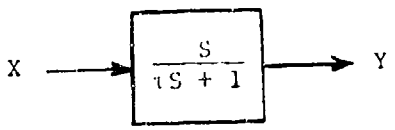
For $Y = C_1 Y_{-1} + C_2 X + C_3 X_{-1}$

$$C_1 = \frac{\tau/(\Delta T/2) - 1}{\tau/(\Delta T/2) + 1}$$

$$C_2 = \frac{1}{\tau/(\Delta T/2) + 1}$$

$$C_3 = C_2$$

3. LAG AND WASHOUT




$$\frac{Y}{X} = \frac{S}{\tau S+1} = \frac{1}{\tau + \frac{1}{S}}$$

$$= \frac{1}{\tau + \frac{\Delta T}{2}\left[\frac{1 + \zeta^{-1}}{1 - \zeta^{-1}}\right]}$$

$$\frac{Y}{X} = \frac{(1 - \zeta^{-1})}{\tau(1 - \zeta^{-1}) + \frac{\Delta T}{2}(1 + \zeta^{-1})}$$

$$\frac{Y}{X} = \frac{(1 - \zeta^{-1})}{(1 + \frac{\Delta T}{2}) - (1 - \frac{\Delta T}{2})\zeta^{-1}}$$

$$\frac{Y}{X} = \frac{\frac{1}{\tau + (\Delta T/2)}(1 - \zeta^{-1})}{1 - \left[\frac{\tau - (\Delta T/2)}{\tau + (\Delta T/2)}\right]\zeta^{-1}}$$

$$Y - \left[\frac{\tau/(\Delta T/2) - 1}{1/(\Delta T/2) +}\right]Y_{-1} = \left[\frac{1/(\Delta T/2)}{1/(\Delta T/2) + 1}\right](X - X_{-1})$$

For $Y = C_1 Y_{-1} + C_2 X + C_3 X_{-1}$

$$C_1 = \frac{\tau/(\Delta T/2) - 1}{\tau/(\Delta T/2) + 1}$$

$$C_2 = \frac{1/(\Delta T/2)}{\tau/(\Delta T/2) + 1}$$

$$C_3 = -C_2$$

8.7.2 ASSUMPTIONS

Not applicable.

8.7.3 COMPUTATION METHODS

Given below are the values of the constants for the three transfer functions:

1 - Integrator

2 - LAG

3 - LAG and Washout

1. INTEGRATOR - $G(s)=1/s$

$C_1 = 1$

$C_2 = \Delta T/2$

$C_3 = \Delta T/2$

2. LAG - $G(s)=\frac{1}{\tau s+1}$

$C_1 = \tau - (\Delta T/2) \quad / \quad \tau + \Delta T/2)$

$C_2 = (\ t/2) \quad / \quad \tau + (DT/2)$

$C_3 = C_2$

3. LAG and WASHOUT - $G(s)=\frac{s}{\tau s+1}$

$C_1 = \tau - (\Delta T/2) \quad / \quad \tau + (\Delta T/2)$

$C_2 = 1 \ / \ \tau + (\Delta T/2)$

$C_3 = -C_2$

8.7.4 APPROXIMATIONS

Not applicable.

8.7.5 LIMITATIONS

The sample frequency must be 10 to 15 times larger than the largest system frequency. Should lower sample frequencies (larger time steps) be

selected discrepancies will arise between the Z-plane and S-plane frequency responses. With a fairly large sample frequency the Z-plane response is almost identical to the S-plane frequency response and continuous design techniques may be used. However if the sample frequency is 15 to 40 times the highest system frequency, there are numerical difficulties with the Z-transform technique which will cause the frequency response to deviate from the continuous case.

8.7.6 Variable Names

Not applicable.

### 8.7.7 Subroutine Listing

```
      SUBROUTINE TUSTIN(J,TAU,C,DT)
C
C     COEFFICIENTS FOR DIGITAL SIMULATION OF CONTINUOUS TRANSFER FUNCTIO
C
C     FUNCTION IN THE FORM (APPROXIMATE DIFFERENCE FORM)
C
C              Y(N) = C1*Y(N-1) + C2*X(N) + C3*X(N-1)
C
C     J = 1    INTEGRATOR
C     J = 2    LAG
C     J = 3    LAG AND WASHOUT
C
      DIMENSION C(3)
      DT2 = DT/2.0
      IF (I.GT.1) GO TO 10
C
C     ----- INTEGRATOR
C
      C(1) = 1.0
      C(2) = DT2
      C(3) = DT2
      GO TO 50
   10 CONTINUE
      TAUP = TAU+DT2
      TAUM = TAU-DT2
      GO TO (20,20,30),J
C
C     ----- LAG
C
   20 CONTINUE
      C(1) = TAUM/TAUP
      C(2) = DT2/TAUP
      C(3) = C(2)
      GO TO 50
C
C     ----- LAG AND WASHOUT
C
   30 CONTINUE
      C(1) = TAUM/TAUP
      C(2) = 1.0/TAUP
      C(3) = -C(2)
   50 CONTINUE
      RETURN
      END
```

**8.7.7 (Continued)**

```
      FUNCTION DYNAM(X,Y,C)
C
C     CALCULATION OF CONTINUOUS TRANSFER FUNCTIONS BY TUSTINS METHOD
C
C     FUNCTION IN THE FORM (APPROXIMATE DIFFERENCE FORM)
C
C                Y(N) = C1*Y(N-1) + C2*X(N) + C3*X(N-1)
C
      DIMENSION X(2),C(3)
      DYNAM = C(1)*Y+C(2)*X(2)+C(3)*X(1)
      RETURN
      END
```

## 8.8 SUBROUTINE LUCUP

Subroutine LUCUP provides linear interpolation of data points for a three dimensional table. Two coordinate points are input and LUCUP returns the third value.

### 8.8.1 Subroutine Listing

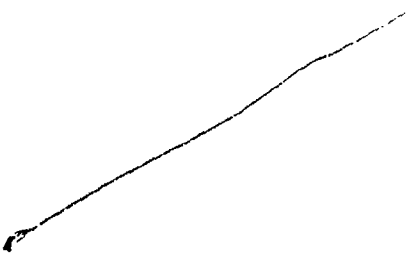
```
      SUBROUTINE LOCUP(ND,NA,X,Z,XA,ZR,K,IE,NEXTR)
      DIMENSION X(1),Z(1),NA(1),XA(1)
      NT=NA(1)
      L2=NA(1)
      T=XA(1)
      P=XA(2)
      IT=1
      IF(T-X(1))80,90,50
   50 DO 60 I=2,L2
      IF(T-X(I))70,90,60
   60 CONTINUE
      IT=L2
      KODE=1
      GO TO 100
   70 IT=I
      KODE=1
      GO TO 100
   80 IT=2
      KODE=1
      GO TO 100
   90 IT=I
      KODE=2
      GO TO 110
  100 CONTINUE
      RATIOT=(T-X(IT-1))/(X(IT)-X(IT-1))
  110 CONTINUE
      L1=L2+2
      L2=L2+NA(2)
      I=L1-1
      IF(P-X(L1-1))130,190,150
  150 DO 160 I=L1,L2
      IF(P-X(I))170,190,160
  160 CONTINUE
      IP=L2
      KODEP=1
      GO TO 200
  170 IP=I
      KODEP=1
      GO TO 200
  130 IP=L1
      KODEP=1
      GO TO 200
  190 IP=I
      KODEP=2
      GO TO 210
  200 CONTINUE
      RATIOP=(P-X(IP-1))/(X(IP)-X(IP-1))
  210 CONTINUE
      IP=IP-NT
      IF(KODE.EQ.1.AND.KODEP.EQ.2)GO TO 400
```

**8.8.1 (Continued)**

```
      IF(KODL.EQ.2.AND.KODEP.EQ.1)GO TO 600
      IF(KODL.EQ.2.AND.KODEP.EQ.2)GO TO 800
      IZ=NT*(IP-2)+(IT-1)
      T1=Z(IZ)+RATIOT*(Z(IZ+1)-Z(IZ))
      IZ=IZ+NT
      T2=Z(IZ)+RATIOT*(Z(IZ+1)-Z(IZ))
      ZR=T1+RATIOP*(T2-T1)
      GO TO 1000
  400 CONTINUE
      IZ=NT*(IP-1)+(IT-1)
      ZR=Z(IZ)+RATIOT*(Z(IZ+1)-Z(IZ))
      GO TO 1000
  600 CONTINUE
      IZ=NT*(IP-2)+IT
      IZT=IZ+NT
      ZR=Z(IZ)+RATIOP*(Z(IZT)-Z(IZ))
      GO TO 1000
  800 CONTINUE
      IZ=NT*(IP-1)+IT
      ZR=Z(IZ)
 1000 CONTINUE
      RETURN
      END
```

## 8.9 FUNCTION HYSTLI

HYSTLI is a function subprogram which determines values of torque motor output within the aysteresis limits specified by the calling program.



### 8.9.1 Subroutine Listing

```
      FUNCTION HYSTLI(X,H,T)
      DIMENSION T(5)
C
C     TORQUE MOTOR HYSTERESIS
      IF(T(1).NE.0.0) GO TO 50
      T(3)= H
      T(2)= -H
      T(4) = 0.0
      T(5)= 0.0
   50 CONTINUE
C
C     T(1) = +1     X MOVING IN POSITIVE DIRECTION
C     T(1) = -1     X MOVING IN NEGATIVE DIRECTION
C
      IF(T(1)) 200,100,100
C
C     X MOVING IN POSITIVE DIRECTION
C
  100 CONTINUE
      IF (X-T(5)) 120,110,110
  110 CONTINUE
      IF (X.LT.T(3)) GO TO 300
      T(4) = X-H
      GO TO 300
  120 T(1) = -1
      T(2) = T(4)-H
      GO TO 220
C
C     X MOVING IN NEGATIVE DIRECTION
C
  200 CONTINUE
      IF (X-T(5)) 220,220,210
  210 T(1) = 1
      T(3) = T(4)+H
      GO TO 110
  220 CONTINUE
      IF (X.GT.T(2)) GO TO 300
      T(4) = X+H
  300 CONTINUE
      T(5) = X
      HYSTLI = T(4)
      RETURN
      END
```

## 8.10 CFRIC/CFRIC2 SUBROUTINES

CFRIC/CFRIC2 provide for the inclusion of coulomb friction into the simulation of a mechanical system. Coulomb friction is that which occurs between a block sitting on any typical dry surface. If the block is in motion (sliding on the surface), the friction force is assumed constant and opposite to the velocity. If the block is at rest, the friction is equal to the applied force (but with opposite sign) bounded by the stiction ratio times the running friction value.

### 8.10.1 Subroutine Listing

```
      SUBROUTINE CFRIC(FDRIVE,FF,XD,XDS,XDD,XDDS,FRIC,SR,BF)
      IF(XD.NE.0.)GO TO 25
      FF = -FDRIVE
      IF(ABS(FF).GT.FRIC*SR*BF)FF=SIGN(FRIC,-FDRIVE)*SR*BF
      GO TO 55
   25 CONTINUE
      NPXD=SIGN(1.,XD)
      NPXDS=SIGN(1.,XDS)
      IF(NPXD.EQ.NPXDS)GO TO 50
      IF(ABS(FDRIVE).GT.FRIC*SR*BF)GO TO 50
      XD=0.
      XDS=0.
      XDD=0.
      XDDS=0.
      FF=-FDRIVE
      GO TO 55
   50 CONTINUE
      FF=SIGN(FRIC,-XD)*BF
   55 CONTINUE
      RETURN
      END
```

**8.10.1 (Continued)**

```
      SUBROUTINE CFRIC2(FDRIVE,FF,XD,XDS,XDD,XDDS,V1,V2,FRIC,SR,BF)
      IF(XD.NE.0.)GO TO 25
      FF = -FDRIVE
      IF(ABS(FF).GT.FRIC*SR*BF)FF=SIGN(FRIC,-FDRIVE)*SR*BF
      GO TO 55
   25 CONTINUE
      NPXD=SIGN(1.,XD)
      NPXDS=SIGN(1.,XDS)
      IF(NPXD.EQ.NPXDS)GO TO 50
      IF(ABS(FDRIVE).GT.FRIC*SR*BF)GO TO 50
      XD=0.
      XDS=0.
      XDD=0.
      XDDS=0.
      V1=0.
      V2=0.
      FF=-FDRIVE
      GO TO 55
   50 CONTINUE
      FF=SIGN(FRIC,-XD)*BF
   55 CONTINUE
      RETURN
      END
```

## 9.0 REFERENCES


1. R. J. Krane, A. Reiff, "A Method of Characteristics Solution for the Equations Governing the Unsteady Flow of Liquids in Closed Systems," 30 August 1966 - NASA Accession Number N68-10219, CR #89954.
2. John F. Blackburn, Gerhard Reethof, J. Lowen Shearer, "Fluid Power Control". MIT press third printing April 1969, Student Edition.
3. F. B. Hildebrand, "Introduction to Numerical Analysis". McGraw-Hill, 1956.
4. Arun K. Trikha, "An Efficient Method for Simulating Frequency Dependent Friction in Transient Liquid Flow", ASME Journal of Fluids Engineering, March 1975.
5. Victor L. Streeter, E. Benjamin Wylie, "Hydraulic Transients". McGraw-Hill Book Co., Inc., New York, 1967.
6. Headquarters, U.S. Army Material Command, "Engineering Design Handbook - Hydraulic Fluids". AMCP 706-123, April 1971, pages 3-9.