D.O. 123
NAS8-36955

Users Manual for Program NYQUIST

Liquid Rocket Nyquist Plots

Developed for Use on a PC Computer

Research Institute
The University of Alabama in Huntsville

124

Users Manual for Program NYQUIST

Liquid Rocket Nyquist Plots

Developed for Use on a PC Computer

Wilbur C. Armstrong

June 1992

Table of Contents

1.0 Introduction

The piping in a liquid rocket can assume complex configurations due to multiple tanks, multiple engines, and structures that must be piped around. The capability to handle some of these complex configurations have been incorporated into the NYQUIST code. The capability to modify the input on line has been implemented.

The configurations allowed include multiple tanks, multiple engines, the splitting of a pipe into unequal segments going to different (or the same) engines. This program will handle the following type elements

Straight pipes
Bends
Inline accumulators
Tuned stub accumulators
Helmholtz resonators
Parallel resonators
Pumps
Split pipes
Multiple tanks
Multiple engines

The code is too large to compile as one program using Microsoft FORTRAN V, therefore the code was broken into two segments: NYQUIST1.FOR and NYQUIST2.FOR. These are compiled separately and then linked together. The final run code is not too large (≈344,000 bytes).

2.0 Input Description

NYQUIST uses the following files: ENG.RLN, LOX.RLN, FUEL.RLN, and CONST.RLN. All files are in free format, therefore each of the following records will give the same results.

```
Record 1:   1.000000E-01 6219.000000 2.670000 2.330E-03 -315.0000

Record 2:   0.1  6219.0  2.67  0.00233 -315.0

Record 3:   1.E-01
            6219.0
            2.67
            2.33E-3
           -315.0
```

The file assignments are given in the following table:

Unit	File Name	File Type	Description
9	ENG.RLN	Input	Engine data
10	LOX.RLN	Input	LOX tanks & lines data
11	FUEL.RLN	Input	Fuel tanks & lines data
12	CONST.RLN	Input	Chamber data
13	SURF.ERR	Output	Convergence error information
14	NYQ.OUT	Output	K() values
15	(LOX)	Work	Temporary file with LOX data
16	(FUEL)	Work	Temporary file with fuel data
17	(RESULT)	Work	Temporary file for results

2.1 Description of file ENG.RLN

```
Card # 1
   number of engines
 Card # 2
    total flow in engine (lbm/sec),
    chamber pressure (lbf/ft^2),
    pressure drop across orifice (lbf/ft^2)
 Read card # 2 "number of engines" times
```

2.2 Description of files LOX.RLN or FUEL.RLN

```
Card # 1
   title
Card # 2
   number of tanks
 Card # 3
    volume (ft^3),
    mass flow (lbm/sec),
    bulk modulus (lbf/ft^2),
    density (lbm/ft^3)
```

```
 Read card # 3 "number of tanks" times
Card # 4
   number of lines leaving tank
 Card # 5
    tank number,
    engine number (0 if split follows)
 Card # 6
    number of segments,
    number of unique splits
  Card # 7
     section type,
     pipe1,
     pipe2,
     pipe3,
     pipe4,
     pipe5
   Read card # 7 "number of segments" times
 if split > 0
   Card # 8
      number of segments,
      number of identical lines,
      engine number
    Card # 9
       section type,
       pipe1,
       pipe2,
       pipe3,
       pipe4,
       pipe5
     Read card # 9 "number of segments" times
    Read card # 8-9 "number of splits" times
   Read card # 5-9 "number of lines" times
```

where

type	name	PIPE1	PIPE2	PIPE3	PIPE4	PIPE5
0	bend	radius	angle	diameter	end len.	
1	straight	length	diameter			
2	inline	length	diameter			
3	tuned	length	diameter			
4	Helmholtz	length	diameter	volume		
5	parallel	length	diameter	volume		
6	pump	length	diameter	$dp/d\mathring{m}$	L	C
7	manifold	volume	bulk mod.			

Dimensions:

radius, length, diameter, end length	- ft
angle	- deg
volume	- ft^3
$dp/d\mathring{m}$ (non-dimensionalized by $\bar{\dot{m}}/\bar{p}_c$)	- non-dimensional
L	- sec
C	- sec
bulk modulus	- lbf/ft^2

2.3 Description of file CONST.RLN

Card # 1
transport lag (sec),
characteristic velocity (ft/sec),
mixture ratio,
characteristic time constant (sec),
change in velocity with mixture ratio (ft/sec)
Read card # 1 "number of engines" times

3.0 Output Description

3.1 Output Files

Output from the program is a file (NYQ.OUT) which may be printed and various graphs under the control of the user. The print file contains the following:

Title, time, and date
for no LOX of FUEL lines
FREQ. K1(R) K1(I)
for no FUEL line
FREQ. K1(R) K1(I) ENG. K2(R) K2(I)
for no LOX line
FREQ. K1(R) K1(I) ENG. K3(R) K3(I)
for both LOX and FUEL lines
FREQ. K1(R) K1(I)
ENG. K2(R) K2(I) K3(R) K3(I) K4(R) K4(I)

Also, if a split pipe is analyzed, a file (SURF.ERR) is created if any point fails to converge within the specified number of iterations. This file contains:

Title, time, and date

jw = after iterations has error of % in line
I= J= |G|= |GOLD|=

3.2 Graphs Available

The graphs available are

1. Plot of the FUEL and/or LOX piping layouts for a specific engine. These are plotted upon request.
2. The following plots are available upon request:
   a. Nyquist plot of K(jw)
   b. Nyquist plot of K(jw,Gox) if LOX line present
   c. Nyquist plot of K(jw,Gf) if FUEL line present
   d. Nyquist plot of K(jw,Gox,Gf) if LOX & FUEL lines present
   e. Phase-Gain plot of K(jw)
   f. Phase-Gain plot of K(jw,Gox) if LOX line present
   g. Phase-Gain plot of K(jw,Gf) if FUEL line present
   h. Phase-Gain plot of K(jw,Gox,Gf) if LOX & FUEL lines present

4.0 Sample Run

The sample run consists of two lox tanks and four engines, two of the engines and lines going to them are identical. The total mass flow from each tank is the same, however the line from the first tank is split with half the mass flow going to engine # 2 and the other half split into two identical engines # 1.

4.1 Input for Sample Run

Sample ENG.RLN file:

```
3
853.5       4.502040E+05      1.610532E+06
1707.0      4.502040E+05      1.610532E+06
3414.0      4.502040E+05      1.610532E+06
```

Sample LOX.RLN file:

```
Sample Run
2
1.956300E+04   2928.0   1.185883E+07   71.4
1.956300E+04   2928.0   1.185883E+07   71.4
2
1 0
13 2
1   15.0     1.416   0.0     0.0   0.0
0   35.0     45.0    1.416   0.0   0.0
1   30.0     1.416   0.0     0.0   0.0
0   3.5      135.0   1.416   0.0   0.0
1   15.0     1.416   0.0     0.0   0.0
1   20.641   1.416   0.0     0.0   0.0
1   20.558   1.416   0.0     0.0   0.0
1   20.558   1.416   0.0     0.0   0.0
1   8.541    1.416   0.0     0.0   0.0
1   6.383    1.416   0.0     0.0   0.0
```

```
0   4.25      90.0      1.416     0.0     0.0
1   9.33      1.416     0.0       0.0     0.0
0   3.33      80.0      1.416     0.0     0.0
5   1  1
1   3.53      0.708     0.0       0.0     0.0
1   12.2      0.708     0.0       0.0     0.0
0   1.28      35.0      0.708     0.0     0.0
1   12.2      0.708     0.0       0.0     0.0
7   13.5   1.183346E+07 0.0       0.0     0.0
5   1 2
1   3.53      1.00126   0.0       0.0     0.0
1   12.2      1.00126   0.0       0.0     0.0
0   1.28      35.0      1.00126   0.0     0.0
1   12.2      1.00126   0.0       0.0     0.0
7   13.5   1.183346E+07 0.0       0.0     0.0
2 3
18 0
1   15.0      1.416     0.0       0.0     0.0
0   35.0      45.0      1.416     0.0     0.0
1   30.0      1.416     0.0       0.0     0.0
0   3.5       135.0     1.416     0.0     0.0
1   15.0      1.416     0.0       0.0     0.0
1   20.641    1.416     0.0       0.0     0.0
1   20.558    1.416     0.0       0.0     0.0
1   20.558    1.416     0.0       0.0     0.0
1   8.541     1.416     0.0       0.0     0.0
1   6.383     1.416     0.0       0.0     0.0
0   4.25      90.0      1.416     0.0     0.0
1   9.33      1.416     0.0       0.0     0.0
0   3.33      80.0      1.416     0.0     0.0
1   3.53      1.416     0.0       0.0     0.0
1   12.2      1.416     0.0       0.0     0.0
0   1.28      35.0      1.416     0.0     0.0
1   12.2      1.416     0.0       0.0     0.0
7   13.5   1.183346E+07 0.0       0.0     0.0
```

Sample CONST.RLN file:

```
0.1  6219.0  2.67  2.330000E-03 -315.0
0.1  6219.0  2.67  2.330000E-03 -315.0
0.1  6219.0  2.67  2.330000E-03 -315.0
```

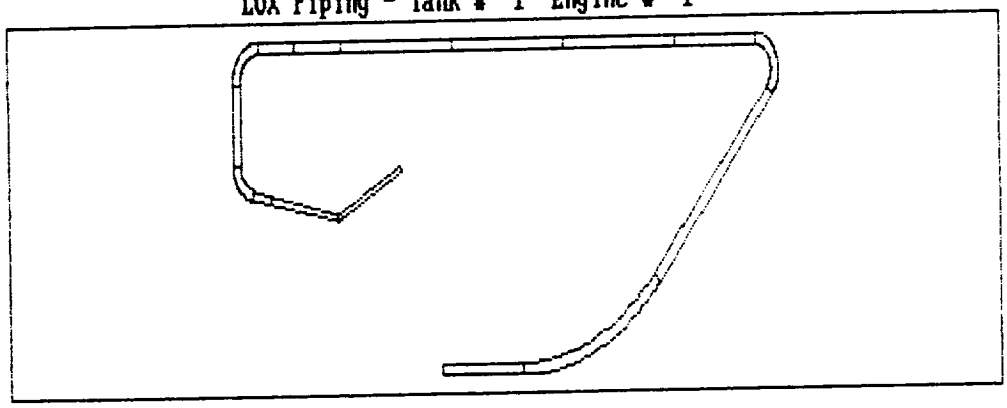
4.2 Walkthrough of Sample Run

```
        Good Morning and Welcome to NYQUIST!!

    Program NYQUIST provides stability predictions
              of feedline systems

          To send a plot to the printer

        The computer MUST be in GRAPHICS mode

    Hit PrScn to send the current plot to the printer
```

```
If you want frequency in rad/sec, hit enter.
If you want it in Hertz, enter "H". h
Do you have FUEL data? n
Do you have LOX data? y
Is the engine data on file ENG.RLN? (Y/N) y
Is the lox file name LOX.RLN? (Y/N) y
Max. no. of iterations is set at  20
Do you wish to change it? n
Do you wish to modify lox line data?   n
  Are the following variables in a file? (Y/N)

      VARIABLES
 TRANSPORT LAG
 CHARACTERISTIC ROCKET VELOCITY
 MIXTURE RATIO
 CHARACTERISTIC TIME CONSTANT
 CHANGE IN VELOCITY WITH MIXTURE RATIO

y
 Is the name of the file CONST.RLN? (Y/N) y
  Enter 20 character title
Sample Run
  Enter range of frequencies in Hertz
  Low freq, high freq, #pts
    1001 = Maximum number of points
    1  40  40
  The following LOX lines may be plotted

 Line #    Tank #    Engine #

    1         1         1
    2         1         2
    3         2         3

 Enter line # to be plotted, 0 will end plot   1
```

Sample Run 10:53AM 12-12-91

LOX Piping - Tank # 1 Engine # 1



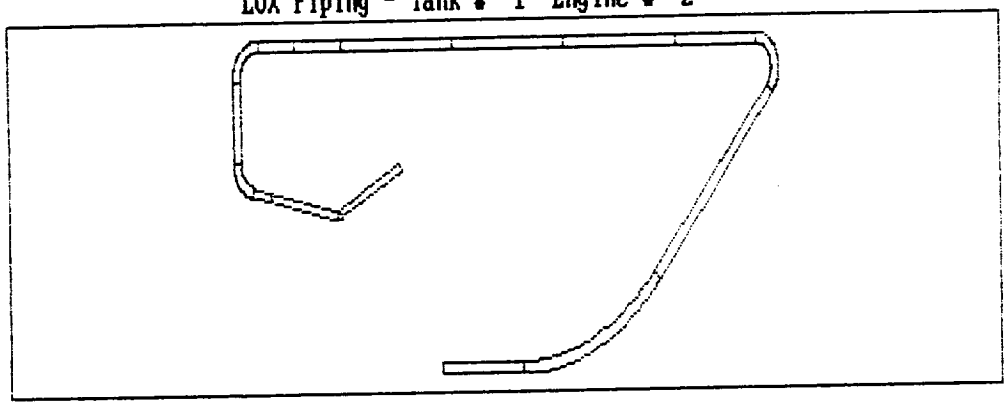
The following LOX lines may be plotted

Line #	Tank #	Engine #
1	1	1
2	1	2
3	2	3

Enter line # to be plotted, 0 will end plot **2**

Sample Run 10:53AM 12-12-91

LOX Piping - Tank # 1 Engine # 2

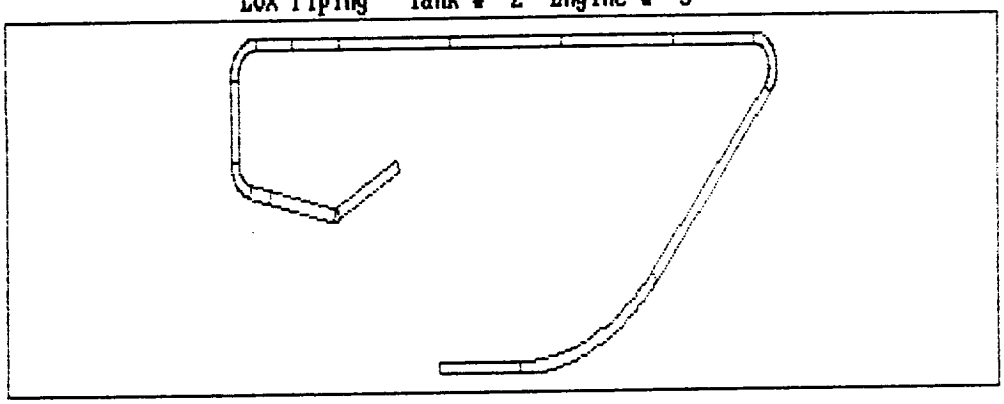


The following LOX lines may be plotted

Line #	Tank #	Engine #
1	1	1
2	1	2
3	2	3

Enter line # to be plotted, 0 will end plot 3

Sample Run 10:53AM 12-12-91
LOX Piping - Tank # 2 Engine # 3



The following LOX lines may be plotted

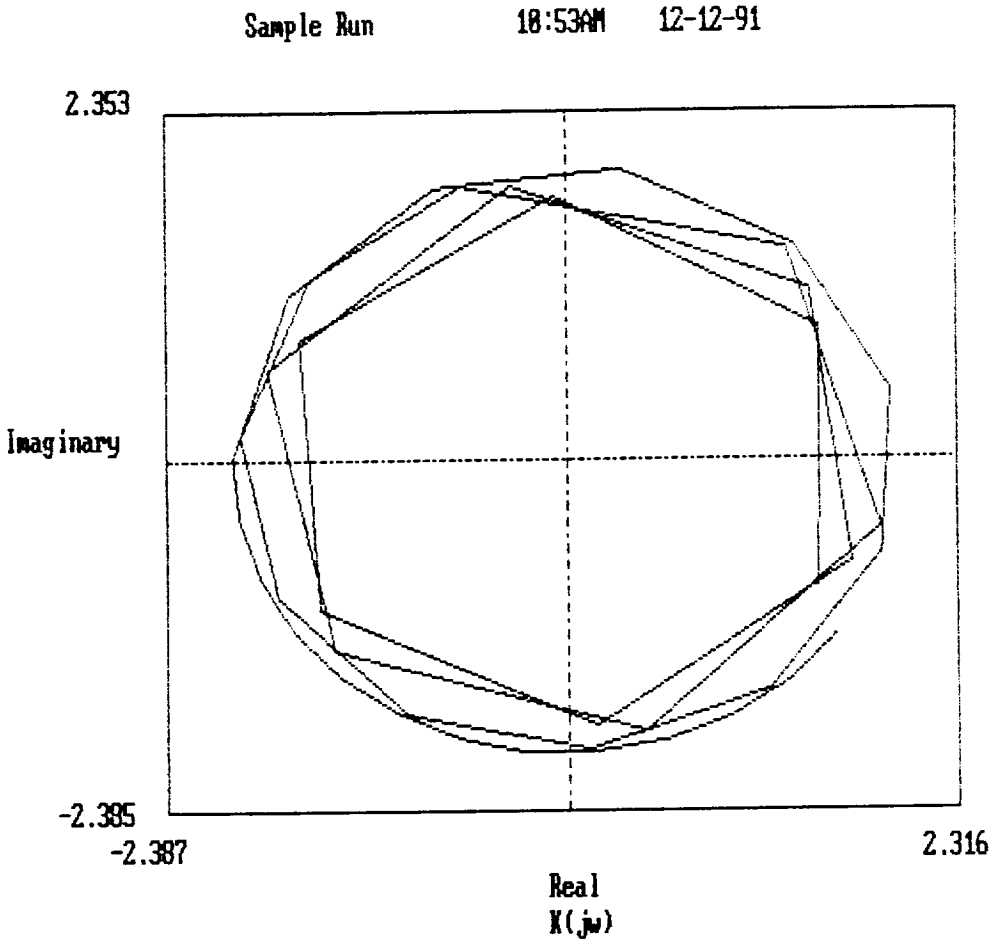
Line #	Tank #	Engine #
1	1	1
2	1	2
3	2	3

Enter line # to be plotted, 0 will end plot **4**
You did not enter a valid line #. Try again

Enter line # to be plotted, 0 will end plot **0**
Please wait while computations proceed.
Enter graph selection

1 Nyquist plot independent of fuel or lox.
2 Nyquist plot independent of fuel.
5 Phase-Gain plot independent of fuel or lox.
6 Phase-Gain plot independent of fuel.
9 End plots.

1


Sample Run
18:53AM
12-12-91
2.353
Imaginary
-2.385
-2.387
2.316
Real
K(jw)

Enter graph selection

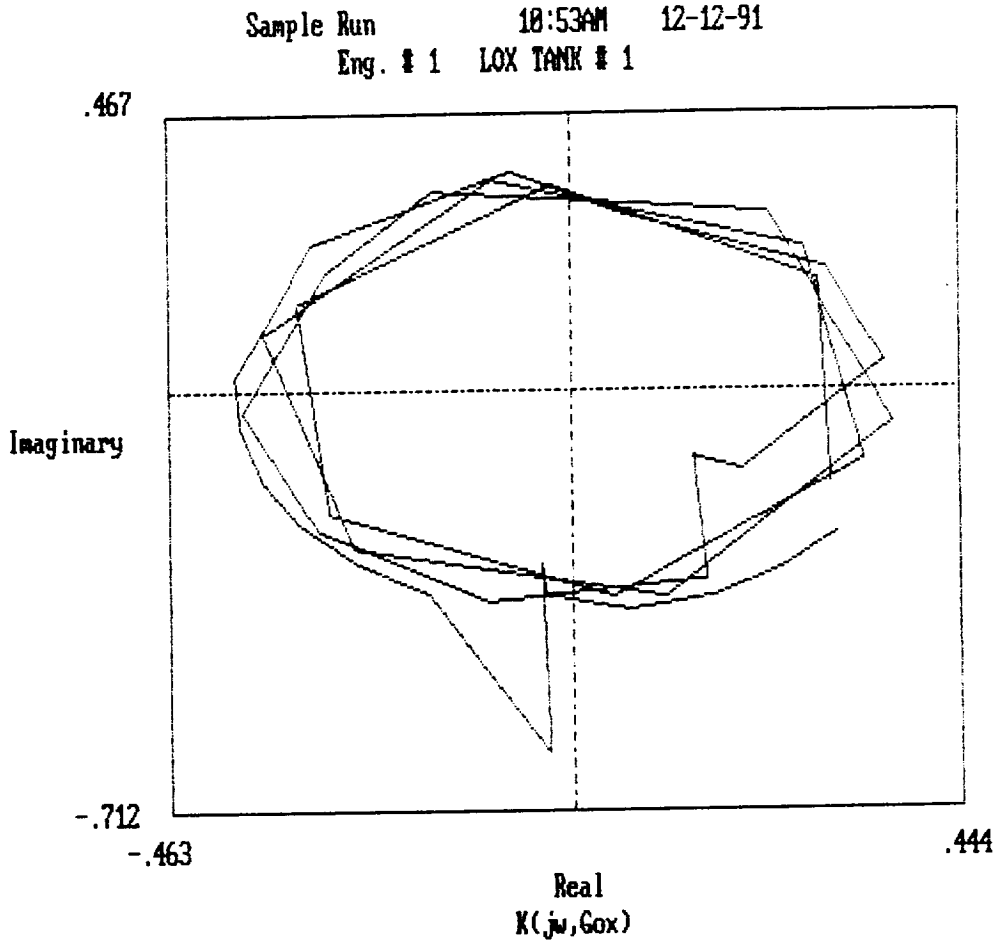
1 Nyquist plot independent of fuel or lox.
2 Nyquist plot independent of fuel.
5 Phase-Gain plot independent of fuel or lox.
6 Phase-Gain plot independent of fuel.
9 End plots.

2

The following LOX lines are available

Line #	Tank #	Engine #
1	1	1
2	1	2
3	2	3

Enter line # to be plotted **1**


Sample Run 10:53AM 12-12-91
Eng. # 1 LOX TANK # 1
.467
Imaginary
-.712
-.463
.444
Real
K(jw,Gox)

Enter graph selection

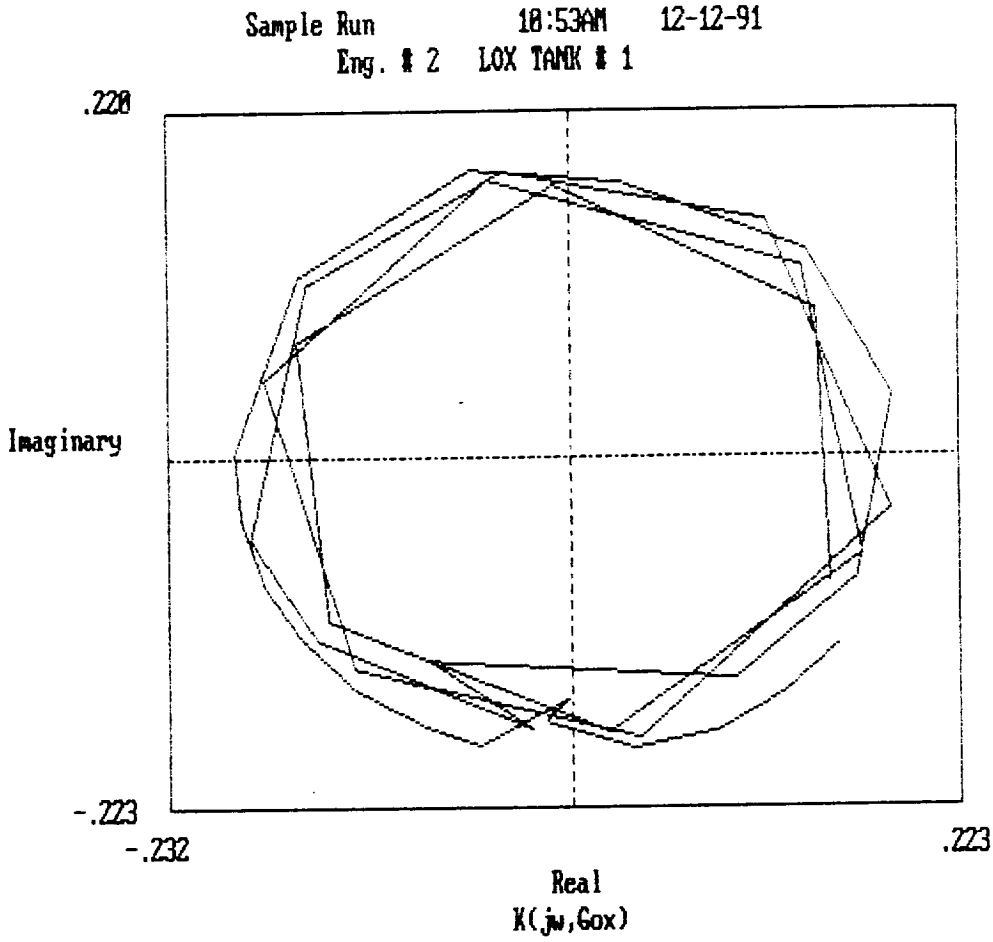
1 Nyquist plot independent of fuel or lox.
2 Nyquist plot independent of fuel.
5 Phase-Gain plot independent of fuel or lox.
6 Phase-Gain plot independent of fuel.
9 End plots.

2

The following LOX lines are available

Line #	Tank #	Engine #
1	1	1
2	1	2
3	2	3

Enter line # to be plotted **2**


Sample Run 18:53AM 12-12-91
Eng. # 2 LOX TANK # 1
.220
Imaginary
-.223
-.232
.223
Real
K(jw,Gox)

Enter graph selection

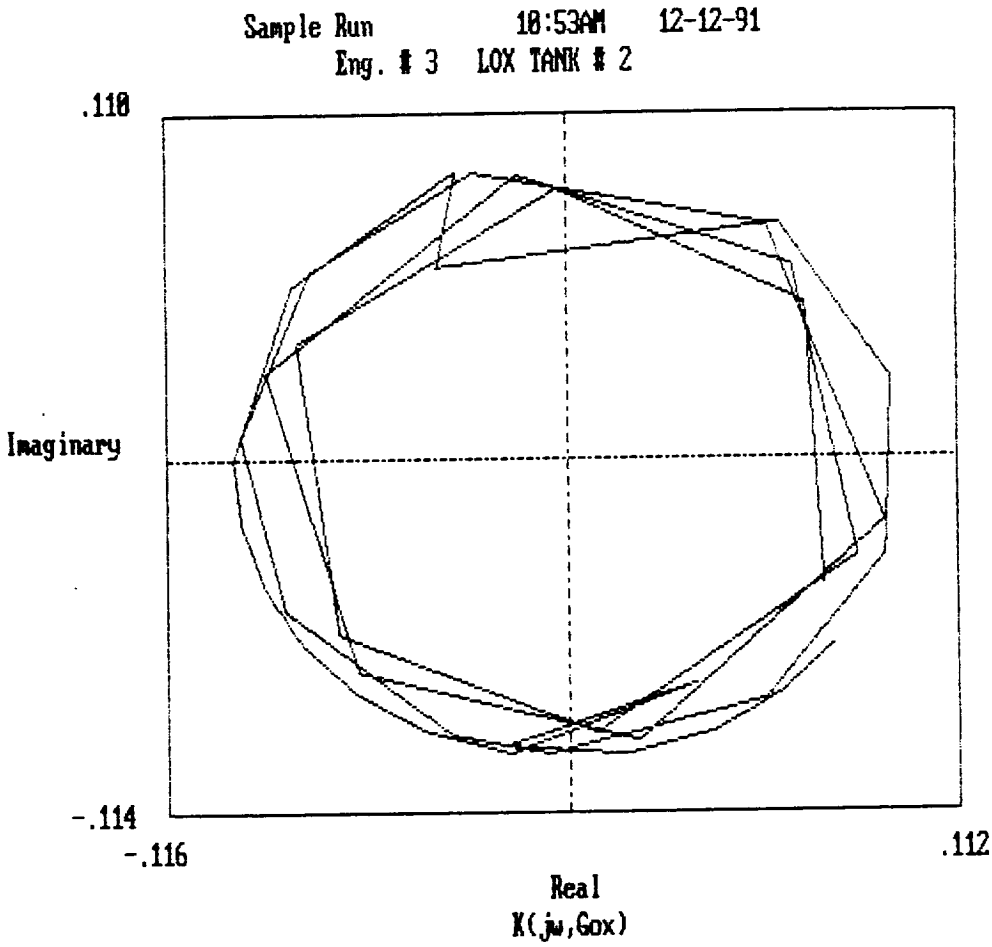
1 Nyquist plot independent of fuel or lox.
2 Nyquist plot independent of fuel.
5 Phase-Gain plot independent of fuel or lox.
6 Phase-Gain plot independent of fuel.
9 End plots.

2

The following LOX lines are available

Line #	Tank #	Engine #
1	1	1
2	1	2
3	2	3

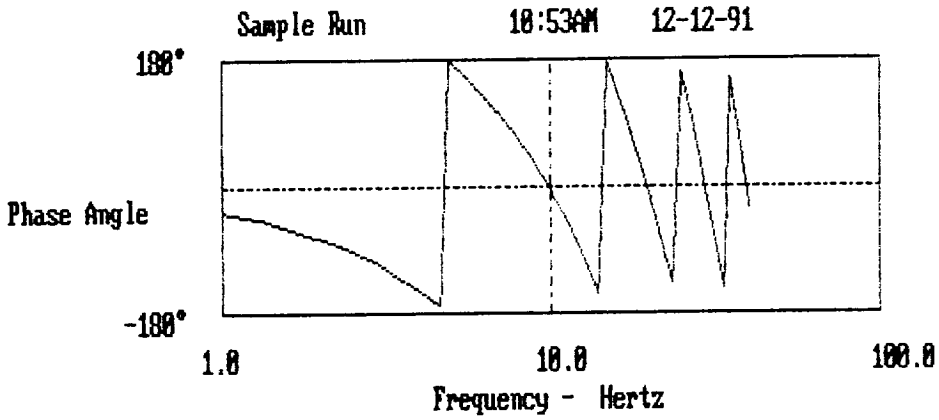
Enter line # to be plotted **3**

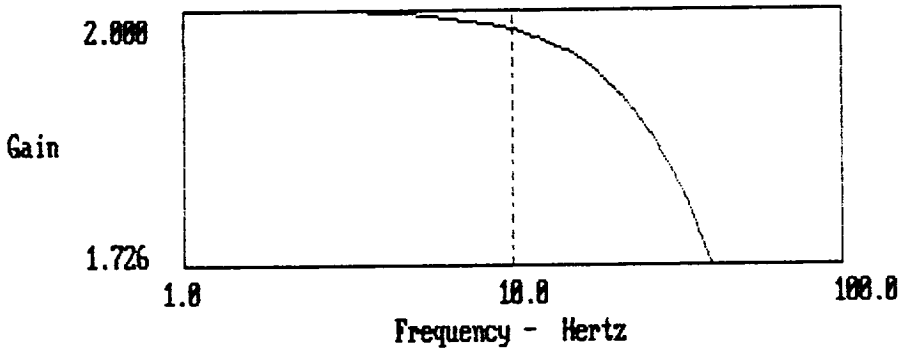

Sample Run 10:53AM 12-12-91
Eng. # 3 LOX TANK # 2
.110
Imaginary
-.114
-.116
.112
Real
K(jw,Gox)

Enter graph selection

1 Nyquist plot independent of fuel or lox.
2 Nyquist plot independent of fuel.
5 Phase-Gain plot independent of fuel or lox.
6 Phase-Gain plot independent of fuel.
9 End plots.

5


Sample Run
10:53AM
12-12-91
180°
Phase Angle
-180°
1.0
10.0
100.0
Frequency - Hertz


2.000
Gain
1.726
1.0
10.0
100.0
Frequency - Hertz

Enter graph selection

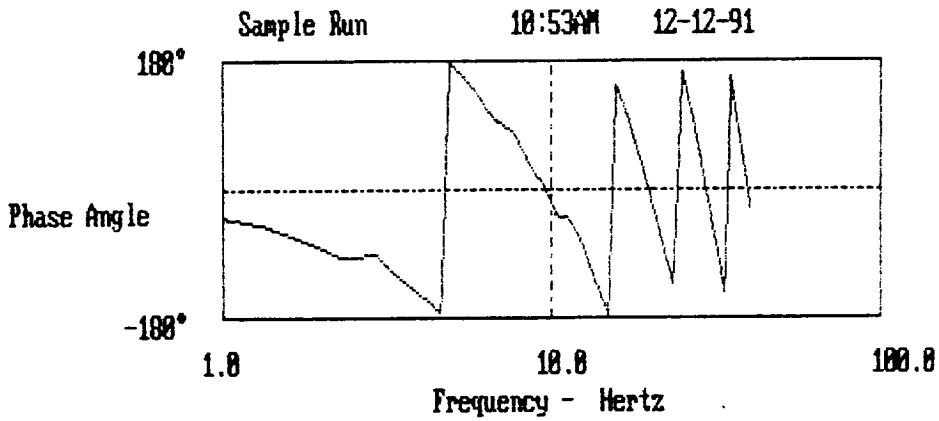
1 Nyquist plot independent of fuel or lox.
2 Nyquist plot independent of fuel.
5 Phase-Gain plot independent of fuel or lox.
6 Phase-Gain plot independent of fuel.
9 End plots.

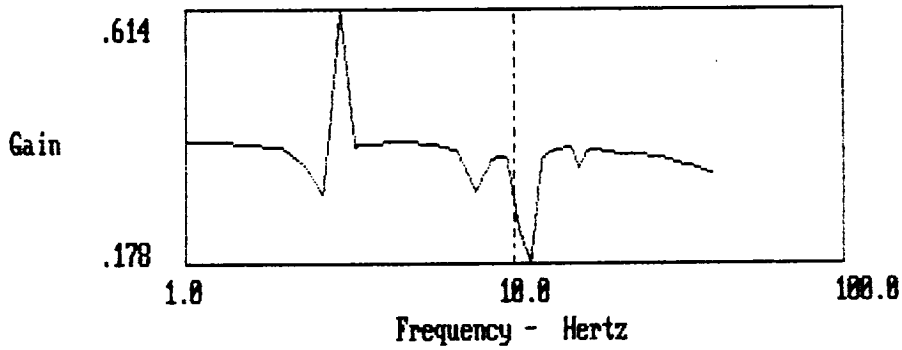
6

The following LOX lines are available

Line #	Tank #	Engine #
1	1	1
2	1	2
3	2	3

Enter line # to be plotted **1**





Eng. # 1 LOX TANK # 1

Enter graph selection

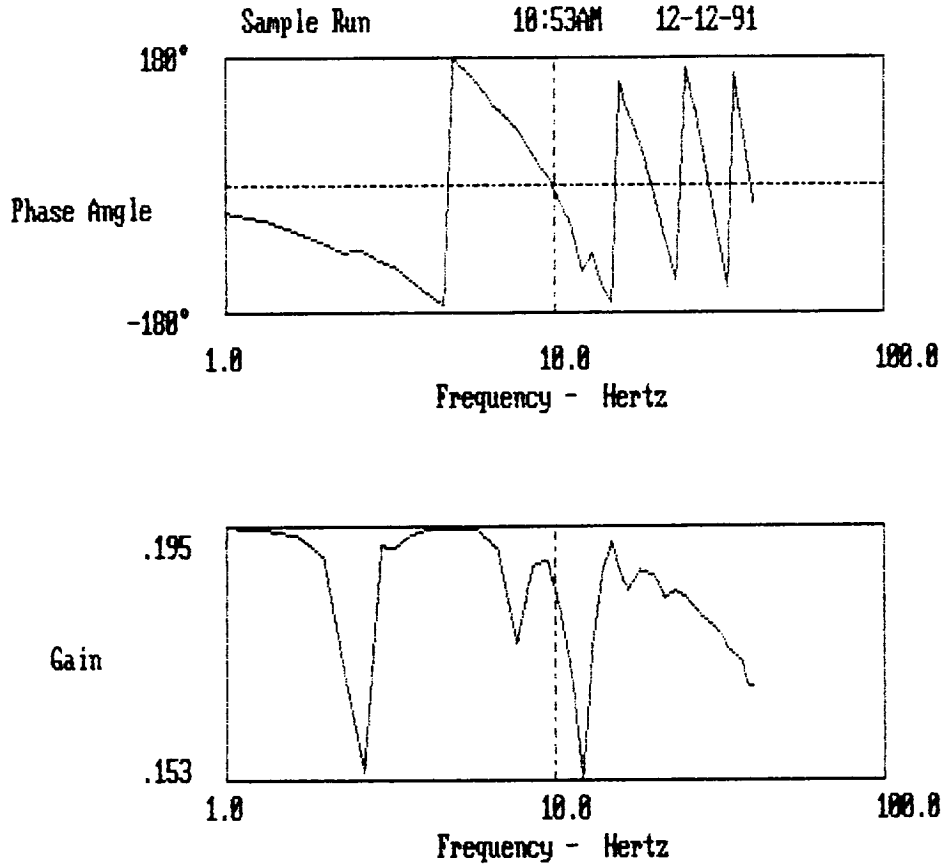
1 Nyquist plot independent of fuel or lox.
2 Nyquist plot independent of fuel.
5 Phase-Gain plot independent of fuel or lox.
6 Phase-Gain plot independent of fuel.
9 End plots.

6

The following LOX lines are available

Line #	Tank #	Engine #
1	1	1
2	1	2
3	2	3

Enter line # to be plotted **2**



Eng. # 2 LOX TANK # 1

Enter graph selection

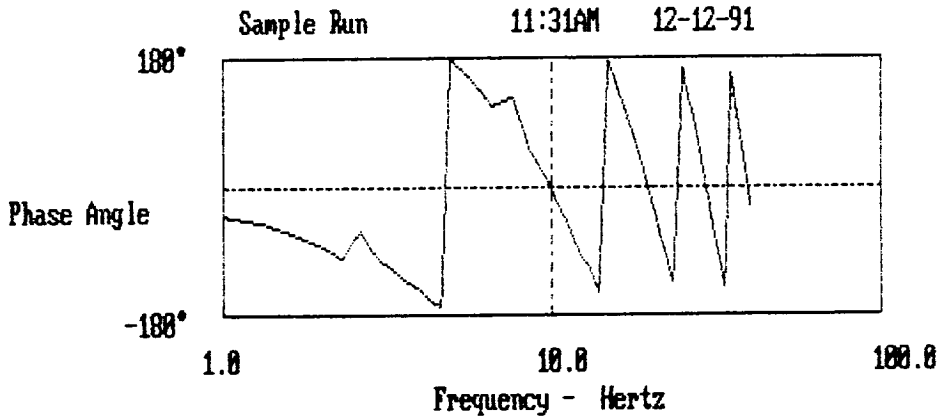
1 Nyquist plot independent of fuel or lox.
2 Nyquist plot independent of fuel.
5 Phase-Gain plot independent of fuel or lox.
6 Phase-Gain plot independent of fuel.
9 End plots.

6

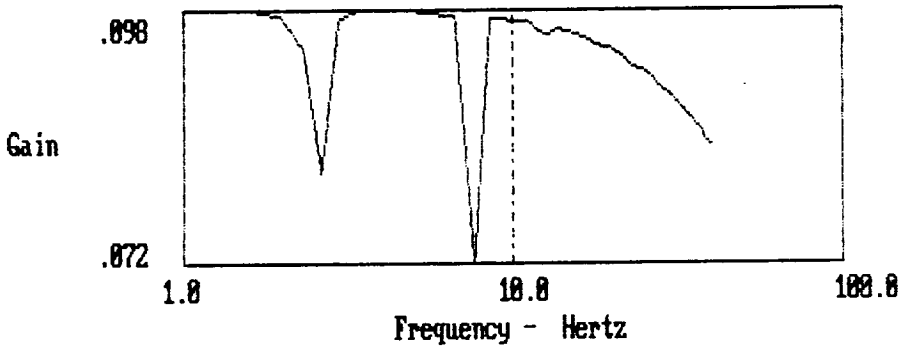
The following LOX lines are available

Line #	Tank #	Engine #
1	1	1
2	1	2
3	2	3

Enter line # to be plotted **3**



Eng. # 3 LOX TANK # 2



K(jw,Gox)

```
Enter graph selection

  1   Nyquist plot independent of fuel or lox.
  2   Nyquist plot independent of fuel.
  5   Phase-Gain plot independent of fuel or lox.
  6   Phase-Gain plot independent of fuel.
  9   End plots.

9
 Enter E to exit,
       F to run new frequency range,
       C to run a new case,
       N to read new files. e
```

4.3 Output for Sample Run

NYQ.OUT File

Sample Run 10:53AM 12-12-91

FREQ.	K1(R)	K1(I)	ENG.	K2(R)	K2(I)
1.0000E+00	1.6005E+00	-1.1990E+00	1	3.0390E-01	-2.4404E-01
1.0000E+00	1.6005E+00	-1.1990E+00	2	1.5277E-01	-1.2105E-01
1.0000E+00	1.6005E+00	-1.1990E+00	3	7.6444E-02	-6.0587E-02
1.3250E+00	1.3168E+00	-1.5048E+00	1	2.4068E-01	-3.0540E-01
1.3250E+00	1.3168E+00	-1.5048E+00	2	1.2167E-01	-1.5172E-01
1.3250E+00	1.3168E+00	-1.5048E+00	3	6.1054E-02	-7.5997E-02
1.6500E+00	9.7593E-01	-1.7451E+00	1	1.6259E-01	-3.5084E-01
1.6500E+00	9.7593E-01	-1.7451E+00	2	8.3069E-02	-1.7467E-01
1.6500E+00	9.7593E-01	-1.7451E+00	3	4.1882E-02	-8.7884E-02
1.9750E+00	5.9263E-01	-1.9093E+00	1	6.9614E-02	-3.7361E-01
1.9750E+00	5.9263E-01	-1.9093E+00	2	3.6472E-02	-1.8630E-01
1.9750E+00	5.9263E-01	-1.9093E+00	3	1.8233E-02	-9.5210E-02
2.3000E+00	1.8365E-01	-1.9904E+00	1	-3.2350E-02	-3.4500E-01
2.3000E+00	1.8365E-01	-1.9904E+00	2	-1.5258E-02	-1.6837E-01
2.3000E+00	1.8365E-01	-1.9904E+00	3	-1.7430E-02	-9.2128E-02
2.6250E+00	-2.3320E-01	-1.9849E+00	1	-3.5376E-02	-2.9597E-01
2.6250E+00	-2.3320E-01	-1.9849E+00	2	-2.2322E-03	-1.5421E-01
2.6250E+00	-2.3320E-01	-1.9849E+00	3	3.6578E-02	-7.2733E-02
2.9500E+00	-6.3973E-01	-1.8930E+00	1	-2.7266E-02	-6.1385E-01
2.9500E+00	-6.3973E-01	-1.8930E+00	2	-5.1827E-02	-1.8444E-01
2.9500E+00	-6.3973E-01	-1.8930E+00	3	-1.6659E-02	-9.4921E-02
3.2750E+00	-1.0183E+00	-1.7187E+00	1	-1.6445E-01	-3.4257E-01
3.2750E+00	-1.0183E+00	-1.7187E+00	2	-8.3960E-02	-1.7178E-01
3.2750E+00	-1.0183E+00	-1.7187E+00	3	-4.2356E-02	-8.7415E-02
3.6000E+00	-1.3523E+00	-1.4698E+00	1	-2.5095E-01	-2.9235E-01
3.6000E+00	-1.3523E+00	-1.4698E+00	2	-1.2426E-01	-1.4770E-01
3.6000E+00	-1.3523E+00	-1.4698E+00	3	-6.1878E-02	-7.5100E-02
3.9250E+00	-1.6273E+00	-1.1570E+00	1	-3.1388E-01	-2.2656E-01
3.9250E+00	-1.6273E+00	-1.1570E+00	2	-1.5501E-01	-1.1622E-01
3.9250E+00	-1.6273E+00	-1.1570E+00	3	-7.7171E-02	-5.9359E-02

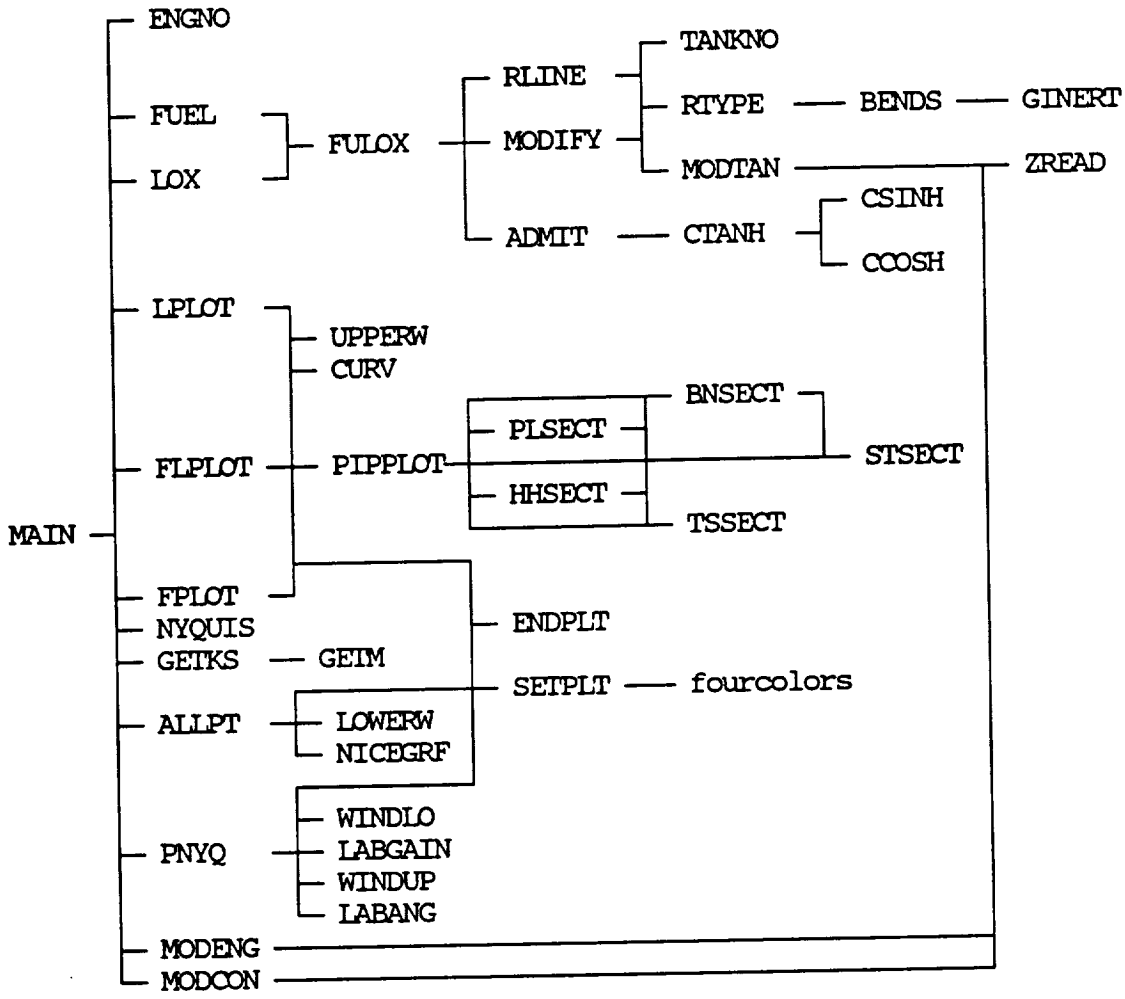
4.2500E+00	-1.8314E+00	-7.9403E-01	1	-3.5779E-01	-1.4976E-01
4.2500E+00	-1.8314E+00	-7.9403E-01	2	-1.7715E-01	-7.9292E-02
4.2500E+00	-1.8314E+00	-7.9403E-01	3	-8.8310E-02	-4.1012E-02
4.5750E+00	-1.9557E+00	-3.9676E-01	1	-3.8254E-01	-6.5826E-02
4.5750E+00	-1.9557E+00	-3.9676E-01	2	-1.9036E-01	-3.8747E-02
4.5750E+00	-1.9557E+00	-3.9676E-01	3	-9.5089E-02	-2.0895E-02
4.9000E+00	-1.9948E+00	1.7516E-02	1	-3.8760E-01	2.1285E-02
4.9000E+00	-1.9948E+00	1.7516E-02	2	-1.9432E-01	3.5422E-03
4.9000E+00	-1.9948E+00	1.7516E-02	3	-9.7336E-02	1.0008E-04
5.8000E+00	-1.6588E+00	1.1044E+00	1	-2.9920E-01	2.4542E-01
5.8000E+00	-1.6588E+00	1.1044E+00	2	-1.5734E-01	1.1401E-01
5.8000E+00	-1.6588E+00	1.1044E+00	3	-8.0045E-02	5.5188E-02
6.7000E+00	-7.8406E-01	1.8295E+00	1	-6.6755E-02	3.6850E-01
6.7000E+00	-7.8406E-01	1.8295E+00	2	-5.5539E-02	1.8308E-01
6.7000E+00	-7.8406E-01	1.8295E+00	3	-3.2324E-02	9.1370E-02
7.6000E+00	3.4342E-01	1.9578E+00	1	5.4292E-02	2.9949E-01
7.6000E+00	3.4342E-01	1.9578E+00	2	2.9180E-02	1.7313E-01
7.6000E+00	3.4342E-01	1.9578E+00	3	-3.8203E-02	6.1255E-02
8.5000E+00	1.3559E+00	1.4493E+00	1	2.9148E-01	2.0582E-01
8.5000E+00	1.3559E+00	1.4493E+00	2	1.3521E-01	1.3081E-01
8.5000E+00	1.3559E+00	1.4493E+00	3	6.1280E-02	7.4722E-02
9.4000E+00	1.9244E+00	4.7142E-01	1	3.5810E-01	4.5479E-02
9.4000E+00	1.9244E+00	4.7142E-01	2	1.8464E-01	3.9720E-02
9.4000E+00	1.9244E+00	4.7142E-01	3	9.3258E-02	2.5367E-02
1.0300E+01	1.8656E+00	-6.5608E-01	1	1.9654E-01	-1.3761E-01
1.0300E+01	1.8656E+00	-6.5608E-01	2	1.6347E-01	-7.7551E-02
1.0300E+01	1.8656E+00	-6.5608E-01	3	9.1121E-02	-3.1760E-02
1.1200E+01	1.2012E+00	-1.5660E+00	1	1.3862E-01	-1.1178E-01
1.1200E+01	1.2012E+00	-1.5660E+00	2	9.4130E-02	-1.4168E-01
1.1200E+01	1.2012E+00	-1.5660E+00	3	5.6674E-02	-7.7824E-02
1.2100E+01	1.4954E-01	-1.9637E+00	1	1.5268E-01	-3.2401E-01
1.2100E+01	1.4954E-01	-1.9637E+00	2	-8.2018E-02	-1.2902E-01
1.2100E+01	1.4954E-01	-1.9637E+00	3	-4.7040E-03	-9.5223E-02
1.3000E+01	-9.4578E-01	-1.7221E+00	1	-9.4360E-02	-3.5742E-01
1.3000E+01	-9.4578E-01	-1.7221E+00	2	-2.2622E-02	-1.7364E-01
1.3000E+01	-9.4578E-01	-1.7221E+00	3	-3.3698E-02	-8.8757E-02
1.3900E+01	-1.7289E+00	-9.2304E-01	1	-2.9358E-01	-2.3433E-01
1.3900E+01	-1.7289E+00	-9.2304E-01	2	-1.4705E-01	-1.1621E-01
1.3900E+01	-1.7289E+00	-9.2304E-01	3	-8.2148E-02	-4.8763E-02
1.4800E+01	-1.9471E+00	1.7121E-01	1	-3.7734E-01	-3.4540E-02
1.4800E+01	-1.9471E+00	1.7121E-01	2	-1.8504E-01	-5.0988E-02
1.4800E+01	-1.9471E+00	1.7121E-01	3	-9.5133E-02	6.5637E-03
1.5700E+01	-1.5329E+00	1.2039E+00	1	-2.8284E-01	1.9258E-01
1.5700E+01	-1.5329E+00	1.2039E+00	2	-1.5246E-01	1.0871E-01
1.5700E+01	-1.5329E+00	1.2039E+00	3	-7.4668E-02	5.8910E-02
1.6600E+01	-6.2440E-01	1.8404E+00	1	-1.5822E-01	3.3608E-01
1.6600E+01	-6.2440E-01	1.8404E+00	2	-3.7439E-02	1.8014E-01
1.6600E+01	-6.2440E-01	1.8404E+00	3	-2.6974E-02	9.0839E-02
1.8271E+01	1.3114E+00	1.4188E+00	1	2.2481E-01	3.0025E-01
1.8271E+01	1.3114E+00	1.4188E+00	2	1.1217E-01	1.5026E-01
1.8271E+01	1.3114E+00	1.4188E+00	3	5.7285E-02	7.4394E-02
1.9943E+01	1.8610E+00	-4.7155E-01	1	3.6837E-01	-6.1085E-02
1.9943E+01	1.8610E+00	-4.7155E-01	2	1.8380E-01	-3.4382E-02

```
1.9943E+01  1.8610E+00 -4.7155E-01    3   9.1216E-02 -2.1258E-02
2.1614E+01  4.7185E-01 -1.8475E+00    1   1.0911E-01 -3.5085E-01
2.1614E+01  4.7185E-01 -1.8475E+00    2   4.0099E-02 -1.7829E-01
2.1614E+01  4.7185E-01 -1.8475E+00    3   2.0366E-02 -9.0745E-02
2.3286E+01 -1.3869E+00 -1.2884E+00    1  -2.5132E-01 -2.6953E-01
2.3286E+01 -1.3869E+00 -1.2884E+00    2  -1.2542E-01 -1.3496E-01
2.3286E+01 -1.3869E+00 -1.2884E+00    3  -6.1509E-02 -6.8410E-02
2.4957E+01 -1.7812E+00  5.9693E-01    1  -3.5324E-01  9.5945E-02
2.4957E+01 -1.7812E+00  5.9693E-01    2  -1.7609E-01  5.0354E-02
2.4957E+01 -1.7812E+00  5.9693E-01    3  -8.7505E-02  2.7218E-02
2.6629E+01 -3.2594E-01  1.8347E+00    1  -8.9997E-02  3.5150E-01
2.6629E+01 -3.2594E-01  1.8347E+00    2  -4.6848E-02  1.7520E-01
2.6629E+01 -3.2594E-01  1.8347E+00    3  -1.4011E-02  8.9818E-02
2.8300E+01  1.4421E+00  1.1551E+00    1   2.6690E-01  2.4154E-01
2.8300E+01  1.4421E+00  1.1551E+00    2   1.3238E-01  1.2181E-01
2.8300E+01  1.4421E+00  1.1551E+00    3   6.4856E-02  6.2118E-02
2.9971E+01  1.6901E+00 -7.0565E-01    1   3.3526E-01 -1.2302E-01
2.9971E+01  1.6901E+00 -7.0565E-01    2   1.6711E-01 -6.3096E-02
2.9971E+01  1.6901E+00 -7.0565E-01    3   8.3234E-02 -3.2460E-02
3.1643E+01  1.8974E-01 -1.8048E+00    1   4.9959E-02 -3.5060E-01
3.1643E+01  1.8974E-01 -1.8048E+00    2   2.1814E-02 -1.7575E-01
3.1643E+01  1.8974E-01 -1.8048E+00    3   8.0807E-03 -8.8172E-02
3.3314E+01 -1.4781E+00 -1.0230E+00    1  -2.7942E-01 -2.0948E-01
3.3314E+01 -1.4781E+00 -1.0230E+00    2  -1.4051E-01 -1.0339E-01
3.3314E+01 -1.4781E+00 -1.0230E+00    3  -6.7146E-02 -5.5897E-02
3.4986E+01 -1.5916E+00  7.9723E-01    1  -3.1573E-01  1.4451E-01
3.4986E+01 -1.5916E+00  7.9723E-01    2  -1.5757E-01  7.2957E-02
3.4986E+01 -1.5916E+00  7.9723E-01    3  -7.8591E-02  3.6920E-02
3.6657E+01 -6.5268E-02  1.7611E+00    1  -1.9983E-02  3.4369E-01
3.6657E+01 -6.5268E-02  1.7611E+00    2  -8.1432E-03  1.7247E-01
3.6657E+01 -6.5268E-02  1.7611E+00    3  -2.6680E-03  8.5946E-02
3.8329E+01  1.4970E+00  8.9502E-01    1   2.8223E-01  1.8719E-01
3.8329E+01  1.4970E+00  8.9502E-01    2   1.4092E-01  9.3133E-02
3.8329E+01  1.4970E+00  8.9502E-01    3   6.8416E-02  5.0007E-02
4.0000E+01  1.4893E+00 -8.7212E-01    1   2.9607E-01 -1.6029E-01
4.0000E+01  1.4893E+00 -8.7212E-01    2   1.4794E-01 -8.0328E-02
4.0000E+01  1.4893E+00 -8.7212E-01    3   7.3729E-02 -4.0632E-02
```

SURF.ERR File

```
Sample Run              10:53AM    12-12-91

jw =     3.0  after 20 iterations has error of  18.520%  in LOX  line
           I=  3   J=  2   |G|=  2.2871E-03   |GOLD|=  1.9298E-03
jw =    33.3  after 20 iterations has error of  68.872%  in LOX  line
           I=  1   J=  2   |G|=  2.0232E-02   |GOLD|=  1.1981E-02
jw =    36.7  after 20 iterations has error of   2.041%  in LOX  line
           I=  4   J=  3   |G|=  6.6627E-02   |GOLD|=  6.5294E-02
jw =    38.3  after 20 iterations has error of 456.116%  in LOX  line
           I=  2   J=  3   |G|=  3.2994E-02   |GOLD|=  5.9329E-03
```

5.0 Flow Diagram



6.0 Variable Description

Variables in Commons

/ARCCON/

XC	REAL*4	x coordinate of curve center
YC	REAL*4	y coordinate of curve center
RAD	REAL*4	radius of bend
ANG	REAL*4	angle of bend in radians
ANGLE	REAL*4	angle of bend in degrees

/COMMQQ/

SCREEN	CHAR*22	screen atributes for plotting

/EPARAM/

MENG	INTEGER*2	number of engines
TFLOW(25)	REAL*4	total flow rate of engine (lbm/sec)
PCHMB(25)	REAL*4	chamber pressure (lbf/ft^2)
DPROR(25)	REAL*4	pressure drop across orifices (lbf/ft^2)
PMRAT(25)	REAL*4	chamber pressure/total mass flow

/FACTOR/

SFAC	REAL*4	factor for frequency

/FOPIPE/

PIPE1F(75,25)	REAL*4	first parameter of pipe description
PIPE2F(75,25)	REAL*4	second parameter of pipe description
PIPE3F(75,25)	REAL*4	third parameter of pipe description
PIPE4F(75,25)	REAL*4	fourth parameter of pipe description
PIPE5F(75,25)	REAL*4	fifth parameter of pipe description

/FPARAM/

MLINEF	INTEGER*2	number of lines from tank
SPLITF(25)	REAL*4	number of unique lines from pipe split
AF(25)	REAL*4	speed of sound in the fluid (ft/sec)
CMANF(25)	REAL*4	manifold capacitance
CTANKF(25)	REAL*4	tank capacitance
DENSF(25)	REAL*4	density of fluid (lbm/ft^3)
KMANF(25)	REAL*4	bulk modulus of manifold (lbf/ft^2)
KTANKF(25)	REAL*4	bulk modulus of tank (lbf/ft^2)
LFLOWF(25)	REAL*4	flow rate through pipe (lbm/sec)
VOLF(25)	REAL*4	volume of tank (ft^3)
VOLMFF(25)	REAL*4	volume of manifold (ft^3)
AREAF(75,25)	REAL*4	area of pipe section (ft^2)
DIAF(75,25)	REAL*4	diameter of pipe section (ft)
LF(75,25)	REAL*4	length of pipe section (ft)
PINDF(75,25)	REAL*4	inductance of pipe section
PCAPF(75,25)	REAL*4	capacitance of pipe section
AVGKF(25)	REAL*4	average bulk modulus
SEGMNF(25)	INTEGER*2	number of pipe sections
SECINF(75,25)	INTEGER*2	pipe section type
NOLINF(25)	INTEGER*2	number of identical lines
IENGF(25)	INTEGER*2	engine number

ITANKF(25)	INTEGER*2	tank number
LOPOLF(25)	INTEGER*2	previous maximum number of iterations
LOPENF(25)	INTEGER*2	maximum number of iterations for split pipe

/NOCOL/

NCOLS	INTEGER*2	number of text columns
NMODE	INTEGER*2	graphics mode

/OPARAM/

MLINEO	INTEGER*2	number of lines from tank
SPLITO(25)	REAL*4	number of unique lines from pipe split
AO(25)	REAL*4	speed of sound in the fluid (ft/sec)
CMANO(25)	REAL*4	manifold capacitance
CTANKO(25)	REAL*4	tank capacitance
DENSO(25)	REAL*4	density of fluid (lbm/ft^3)
KMANO(25)	REAL*4	bulk modulus of manifold (lbf/ft^2)
KTANKO(25)	REAL*4	bulk modulus of tank (lbf/ft^2)
LFLOWO(25)	REAL*4	flow rate through pipe (lbm/sec)
VOLO(25)	REAL*4	volume of tank (ft^3)
VOLMFO(25)	REAL*4	volume of manifold (ft^3)
AREAO(75,25)	REAL*4	area of pipe section (ft^2)
DIAO(75,25)	REAL*4	diameter of pipe section (ft)
LO(75,25)	REAL*4	length of pipe section (ft)
PINDO(75,25)	REAL*4	inductance of pipe section
PCAPO(75,25)	REAL*4	capacitance of pipe section
AVGKO(25)	REAL*4	average bulk modulus
SEGMNO(25)	INTEGER*2	number of pipe sections
SECTNO(75,25)	INTEGER*2	pipe section type
NOLINO(25)	INTEGER*2	number of identical lines
IENGO(25)	INTEGER*2	engine number
ITANKO(25)	INTEGER*2	tank number
LOPOLO(25)	INTEGER*2	previous maximum number of iterations
LOPENO(25)	INTEGER*2	maximum number of iterations for split pipe

/PIPPXY/

X	REAL*4	x location of current centerline
XH	REAL*4	x location of current upper pipe
XL	REAL*4	x location of current lower pipe
Y	REAL*4	y location of current centerline
YH	REAL*4	y location of current upper pipe
YL	REAL*4	y location of current lower pipe
XMIN	REAL*4	minimum x value of piping layout
XMAX	REAL*4	maximum x value of piping layout
YMIN	REAL*4	minimum y value of piping layout
YMAX	REAL*4	maximum y value of piping layout
SINA	REAL*4	sine of current pipe direction
COSA	REAL*4	cosine of current pipe direction

/SETUP/

PIPE1(150)	REAL*4	current first parameter of pipe description
PIPE2(150)	REAL*4	current second parameter of pipe description
PIPE3(150)	REAL*4	current third parameter of pipe description
PIPE4(150)	REAL*4	current fourth parameter of pipe description

NENGF(25)	INTEGER*2	engine for a fuel line
NTANKF(25)	INTEGER*2	tank for a fuel line
NLINEF(25)	INTEGER*2	section number for a fuel line
NSPF(25)	INTEGER*2	starting fuel line number
ILINEF	INTEGER*2	current fuel line
NENGO(25)	INTEGER*2	engine for a lox line
NTANKO(25)	INTEGER*2	tank for a lox line
NLINEO(25)	INTEGER*2	section number for a lox line
NSPO(25)	INTEGER*2	starting lox line number
ILINEO	INTEGER*2	current lox line
SEGMN	INTEGER*2	current number of pipe sections
SECTN(150)	INTEGER*2	current pipe section type

/TANK/

MTANK	INTEGER*2	number of tanks

/WCAOUT/

NAMLIN(2)	CHAR*24	name of files containing pipe description

/WCATIT/

TITLE	CHAR*40	title for plots
TITLF	CHAR*20	title from pipe file
IHR	INTEGER*2	hour code run
IMIN	INTEGER*2	minute code run
AP	CHAR*2	AM or PM
IYR	INTEGER*2	yesr code run
IMON	INTEGER*2	month code run
IDAY	INTEGER*2	day code run

/WORK1/

K1R(1001)	REAL*4	real part of K(jw)
K1C(1001)	REAL*4	complex part of K(jw)
K2R(1001)	REAL*4	real part of K(jw,GOX)
K2C(1001)	REAL*4	complex part of K(jw,GOX)
K3R(1001)	REAL*4	real part of K(jw,GF)
K3C(1001)	REAL*4	complex part of K(jw,GF)
K4R(1001)	REAL*4	real part of K(jw,GOX,GF)
K4C(1001)	REAL*4	complex part of K(jw,GOX,GF)
DUMMY1(3392)	REAL*4	dummy array for spacing
DUMMY2(8397)	REAL*4	dummy array for spacing
X(1001)	REAL*4	frequency array
YR(1001)	REAL*4	real part of Nyquist
YC(1001)	REAL*4	complex part of Nyquist
G(0:75,25)	COMPLEX*8	admittance looking toward tank
ZT(0:75,25)	COMPLEX*8	impedance looking toward tank
ZG(0:75,25)	COMPLEX*8	impedance looking toward engine

/WORK2/

ZO(75,25)	REAL*4	characteristic impedance
POINT(8,200)	REAL*4	description of plot element
DUMMY3(175)	REAL*4	dummy array for spacing
ITYPE(200)	INTEGER*2	type plot element

PROGRAM NYQ
Logic portion of code

Commons EPARAM FACTOR FPARAM NOCOL OPARAM SETUP WCAOUT
WCATIT WORK1

Local Variables

AM	CHAR*2	'AM'
ANS	CHAR*1	response to question
ANS1	CHAR*2	response to question
CSTAR(25)	REAL*4	characteristic rocket velocity (ft/sec)
DCDR(25)	REAL*4	change in velocity with mixture ratio (ft/sec)
GF(25)	COMPLEX*8	admittance of fuel line looking toward tank
GOX(25)	COMPLEX*8	admittance of lox line looking toward tank
HFREQ	REAL*4	maximum frequency requested
I	INTEGER*2	do loop index
IFUEL	INTEGER*2	flag indicating presence of fuel line
IGONE	INTEGER*2	flag for FUEL & LOX routines
ILOX	INTEGER*2	flag indicating presence of lox line
ISEC	INTEGER*2	second code run
I100	INTEGER*2	hundredth of second code run
J	INTEGER*2	do loop index
JUNIT	INTEGER*2	unit number of engine data file
K	INTEGER*2	do loop index
KW(1001)	REAL*4	frequency array
LFREQ	REAL*4	minimum frequency requested
NAMENG	CHAR*24	name of engine data file
NOXY	INTEGER*2	counter
NOXYP	INTEGER*2	pointer
NPTS	INTEGER*2	flag to switch step size
PM	CHAR*2	'PM'
PTS	INTEGER*2	number of frequencies
RBAR(25)	REAL*4	mixture ratio
S	COMPLEX*8	complex frequency
SSIZE1	REAL*4	parameter to pack frequencies toward low end
SSIZE2	REAL*4	parameter to pack frequencies toward low end
SSIZE3	REAL*4	parameter to pack frequencies toward low end
TAUT(25)	REAL*4	transport lag (sec)
THETAC(25)	REAL*4	characteristic time constant (sec)
VARI	CHAR*24	name of input file
W	REAL*4	oscillatory part of frequency

SUBROUTINE ALLPT
Supervises Nyquist plot

Commons NOCOL

Variables in Argument List

GHOLD(1001)	REAL*4	complex part of K()
ING	INTEGER*2	engine number
ITF	INTEGER*2	fuel tank number
ITO	INTEGER*2	lox tank number
ITYPE	INTEGER*2	which K()
PTS	INTEGER*2	number of values to plot

WHOLD(1001)	REAL*4	real part of K()
	Local Variables	
DUMWIL	INTEGER*2	intermediate variable
ENGINK	CHAR*38	intermediate variable
I	INTEGER*2	do loop index
IMAX	REAL*8	maximum value of complex part
IMMIN	REAL*8	minimum value of complex part
RMAX	REAL*8	maximum value of real part
RMIN	REAL*8	minimum value of real part
S	CHAR*4	intermediate variable
X	REAL*8	x value of point to be plotted
XY	CHAR*16	intermediate variable
Y	REAL*8	y value of point to be plotted

SUBROUTINE CURV
Draws circular arc

Commons ARCCON

	Variables in Argument List	
A1	REAL*8	starting angle for arc
A2	REAL*8	ending angle for arc
	Local Variables	
ANG1	REAL*4	starting angle for arc
ANG2	REAL*4	ending angle for arc
DA	REAL*4	incremental angle for plot
DTH	REAL*4	total angle to plot
DUMWIL	INTEGER*2	intermediate variable
I	INTEGER*2	do loop index
N	INTEGER*2	number of points to plot
T	REAL*4	current angle
XP	REAL*8	x location of point to plot
XY	CHAR*16	intermediate variable
YP	REAL*8	y location of point to plot

SUBROUTINE ENDPLT
Closes plot routines

	Local Variables	
DUMMY	INTEGER*2	intermediate variable

SUBROUTINE FLPLOT
Supervises plot of piping

Commons EPARAM FOPIPE FPARAM OPARAM SETUP

	Variables in Argument List	
ILOX	INTEGER*2	flag indicating presence of lox line
	Local Variables	
I	INTEGER*2	do loop index
IPF	INTEGER*2	fuel line pointer
IPLOTF	INTEGER*2	fuel line plot pointer

IPLOTO	INTEGER*2	lox line plot pointer
IPO	INTEGER*2	lox line pointer
J	INTEGER*2	do loop index
K	INTEGER*2	pointer
L	INTEGER*2	do loop index
NOXY	INTEGER*2	intermediate variable
NOXYP	INTEGER*2	intermediate variable

LOGICAL FUNCTION fourcolors
Determines type of graphics monitor

Commons COMMQQ

Local Variables

DUMMY	INTEGER*2	intermediate variable

SUBROUTINE FPLOT
Determines fuel line to be plotted

Commons EPARAM FOPIPE FPARAM SETUP

Variables in Argument List

ILOX	INTEGER*2	flag indicating presence of lox line

Local Variables

I	INTEGER*2	do loop index
IPF	INTEGER*2	fuel line pointer
IPLOTF	INTEGER*2	fuel line plot pointer
J	INTEGER*2	pointer
K	INTEGER*2	pointer
L	INTEGER*2	do loop index

SUBROUTINE LABANG
Labels phase angle plot

Commons COMMQQ FACTOR NOCOL WCATIT

Variables in Argument List

XMAX	REAL*8	maximum x value for phase angle plot
XMIN	REAL*8	minimum x value for phase angle plot
YMAX	REAL*8	maximum y value for phase angle plot
YMIN	REAL*8	minimum y value for phase angle plot

Local Variables

DUMMY	REAL*4	intermediate variable
DUMWIL	INTEGER*2	intermediate variable
HI	REAL*4	intermediate variable
I	INTEGER*2	do loop index
IDEL	INTEGER*2	intermediate variable
IHI	INTEGER*2	intermediate variable
ILO	INTEGER*2	intermediate variable
ILOC	INTEGER*2	intermediate variable
IMAX	INTEGER*2	intermediate variable
ROW	INTEGER*2	intermediate variable
ROWS	INTEGER*2	intermediate variable

S	CHAR*4	intermediate variable
XHI	CHAR*7	label for x tick marks
XP	REAL*8	x point for plot
XY	CHAR*16	intermediate variable
YHI	CHAR*6	' 180°' upper phase angle label
YLO	CHAR*6	' -180°' lower phase angle label
YP	REAL*8	y point for plot

SUBROUTINE LABGAIN
Labels gain plot

Commons COMMQQ FACTOR NOCOL WCATIT

Variables in Argument List

ITYPE	INTEGER*2	which K()
XMAX	REAL*8	maximum x value for gain plot
XMIN	REAL*8	minimum x value for gain plot
YMAX	REAL*8	maximum y value for gain plot
YMIN	REAL*8	minimum y value for gain plot

Local Variables

DUMMY	REAL*4	intermediate variable
DUMWIL	INTEGER*2	intermediate variable
HI	REAL*4	intermediate variable
I	INTEGER*2	do loop index
IDEL	INTEGER*2	intermediate variable
IHI	INTEGER*2	intermediate variable
ILO	INTEGER*2	intermediate variable
ILOC	INTEGER*2	intermediate variable
IMAX	INTEGER*2	intermediate variable
ROW	INTEGER*2	intermediate variable
ROWS	INTEGER*2	intermediate variable
S	CHAR*4	intermediate variable
XHI	CHAR*7	label for x tick marks
XP	REAL*8	x point for plot
XY	CHAR*16	intermediate variable
YHI	CHAR*6	' 180°' upper phase angle label
YLO	CHAR*6	' -180°' lower phase angle label
YP	REAL*8	y point for plot

SUBROUTINE LOWERW
Sets up lower plotting window

Commons COMMQQ NOCOL

Variables in Argument List

XMAX	REAL*8	maximum x value for Nyquist plot
XMIN	REAL*8	minimum x value for Nyquist plot
YMAX	REAL*8	maximum y value for Nyquist plot
YMIN	REAL*8	minimum y value for Nyquist plot

Local Variables

COLS	INTEGER*2	number of text columns
DUMMY	INTEGER*2	intermediate variable
ROWS	INTEGER*2	number of text rows

XLEN	REAL*8	intermediate variable
XWIDTH	INTEGER*2	number of x pixels
YHEIGHT	INTEGER*2	number of y pixels
YLEN	REAL*8	intermediate variable

SUBROUTINE LPLOT
Determines lox line to be plotted

Commons EPARAM FOPIPE OPARAM SETUP

Variables in Argument List

ILOX	INTEGER*2	flag indicating presence of lox line

Local Variables

I	INTEGER*2	do loop index
IPLOTO	INTEGER*2	lox line plot pointer
IPO	INTEGER*2	lox line pointer
J	INTEGER*2	pointer
K	INTEGER*2	pointer
L	INTEGER*2	do loop index

SUBROUTINE NICEGRF
Plots Nyquist curve

Commons COMMQQ FACTOR NOCOL WCATIT

Variables in Argument List

IMAX	REAL*8	maximum value of complex part
IMMIN	REAL*8	minimum value of complex part
ITYPE	INTEGER*2	which K()
RMAX	REAL*8	maximum value of real part
RMIN	REAL*8	minimum value of real part

Local Variables

DUMMY	REAL*4	intermediate variable
ROW	INTEGER*2	intermediate variable
ROWS	INTEGER*2	intermediate variable
S	CHAR*4	intermediate variable
XHI	CHAR*6	label for maximum x value
XLO	CHAR*6	label for minimum x value
XMAX	REAL*8	maximum x value
XMIN	REAL*8	minimum x value
YHI	CHAR*6	label for maximum y value
YLO	CHAR*6	label for minimum y value
YMAX	REAL*8	maximum y value
YMIN	REAL*8	minimum y value

SUBROUTINE PIPPLOT
Supervises plot of piping layout

Commons ARCCON PIPPXY WORK2

Variables in Argument List

IENG	INTEGER*2	engine number
ILOX	INTEGER*2	flag for fuel or lox

ITANK	INTEGER*2	tank number
PIPE1(75)	REAL*4	first parameter of pipe description
PIPE2(75)	REAL*4	second parameter of pipe description
PIPE3(75)	REAL*4	third parameter of pipe description
PIPE4(75)	REAL*4	fourth parameter of pipe description
R	CHAR*1	flag for fuel or lox
SECTN(75)	INTEGER*2	pipe section type
SEGMN	INTEGER*2	number of pipe sections
	Local Variables	
DUMWIL	INTEGER*2	intermediate variable
I	INTEGER*2	do loop index
J	INTEGER*2	do loop index
XRANGE	REAL*4	range of x values
X0	REAL*4	intermediate variable
X1	REAL*4	intermediate variable
X2	REAL*4	intermediate variable
X3	REAL*4	intermediate variable
YRANGE	REAL*4	range of y values
Y0	REAL*4	intermediate variable
Y1	REAL*4	intermediate variable
Y2	REAL*4	intermediate variable
Y3	REAL*4	intermediate variable

SUBROUTINE PLSECT

Computes plot coordinates for parallel resonator

Commons ARCCON	PIPPXY	
	Variables in Argument List	
DIA	REAL*4	diameter of parallel segment (ft)
ITYPE(200)	INTEGER*2	type plot element
J	INTEGER*2	pointer to element
LEN	REAL*4	length of parallel segment (ft)
POINT(8,200)	REAL*4	description of plot element
VOL	REAL*4	volume of bypassed segment (ft^3)
	Local Variables	
ANGOLD	REAL*4	intermediate variable
ANGSAV	REAL*4	intermediate variable
COSOLD	REAL*4	intermediate variable
DIAM	REAL*4	intermediate variable
PDIA	REAL*4	intermediate variable
PLEN	REAL*4	intermediate variable
RADIUS	REAL*4	intermediate variable
SIDE	REAL*4	intermediate variable
SINOLD	REAL*4	intermediate variable
TURN	REAL*4	intermediate variable
XHC	REAL*4	intermediate variable
XHOLD	REAL*4	intermediate variable
XHSAV	REAL*4	intermediate variable
XLC	REAL*4	intermediate variable
XLOLD	REAL*4	intermediate variable
XLSAV	REAL*4	intermediate variable
XOLD	REAL*4	intermediate variable

XSAV	REAL*4	intermediate variable
YHC	REAL*4	intermediate variable
YHOLD	REAL*4	intermediate variable
YHSAV	REAL*4	intermediate variable
YLC	REAL*4	intermediate variable
YLOLD	REAL*4	intermediate variable
YLSAV	REAL*4	intermediate variable
YOLD	REAL*4	intermediate variable
YSAV	REAL*4	intermediate variable

SUBROUTINE PNYQ
Plots gain and phase angle

Commons COMMQQ NOCOL WORK1

Variables in Argument List

ING	INTEGER*2	engine number
ITF	INTEGER*2	fuel tank number
ITO	INTEGER*2	lox tank number
ITYPE	INTEGER*2	type plot element
KC(PTS)	REAL*4	complex part of K()
KR(PTS)	REAL*4	real part of K()
KW(PTS)	REAL*4	frequency
PTS	INTEGER*2	number of points

Local Variables

DUMWIL	INTEGER*2	intermediate variable
ENGINK	CHAR*38	intermediate variable
I	INTEGER*2	do loop index
ROWS	REAL*4	intermediate variable
S	CHAR*4	intermediate variable
XHI	REAL*8	intermediate variable
XLO	REAL*8	intermediate variable
XMAX	REAL*8	maximum x value
XMIN	REAL*8	minimum x value
XP	REAL*8	x point to plot
XY	CHAR*16	intermediate variable
YMAXC	REAL*8	maximum phase angle
YMAXR	REAL*8	maximum amplitude
YMINC	REAL*8	minimum phase angle
YMINR	REAL*8	minimum amplitude
YP	REAL*8	y point to plot

SUBROUTINE SETPLT
Sets up the plot environment

Commons COMMQQ NOCOL WCAPAS

SUBROUTINE UPPERW
Sets up upper plotting window

Commons COMMQQ NOCOL WCATIT

Variables in Argument List

IENG	INTEGER*2	engine number
ILOX	INTEGER*2	flag indicating presence of lox line
ITANK	INTEGER*2	tank number
R	CHAR*1	flag for fuel of lox
X00	REAL*4	minimum value of x for piping layout window
X11	REAL*4	maximum value of x for piping layout window
Y00	REAL*4	minimum value of y for piping layout window
Y11	REAL*4	maximum value of y for piping layout window

Local Variables

ADDX	REAL*4	intermediate variable
ADDY	REAL*4	intermediate variable
COLS	INTEGER*2	number of text columns
DUMMY	INTEGER*2	intermediate variable
FULOX	CHAR*36	plot identification
HALFY	REAL*4	intermediate variable
PICX	REAL*4	intermediate variable
PICY	REAL*4	intermediate variable
ROWS	INTEGER*2	number of text rows
S	CHAR*4	intermediate variable
XRANG	REAL*4	intermediate variable
XRAT	REAL*4	intermediate variable
XWIDTH	INTEGER*2	number ox x pixels
X0	REAL*8	minimum x value
X1	REAL*8	maximum x value
YHEIGHT	INTEGER*2	number of y pixels
YRANG	REAL*4	intermediate variable
YRAT	REAL*4	intermediate variable
Y0	REAL*8	minimum y value
Y1	REAL*8	maximum y value

SUBROUTINE WINDLO(XMIN,XMAX,YMIN,YMAX)

Sets up gain window

Commons COMMQQ NOCOL

Variables in Argument List

XMAX	REAL*8	maximum x value
XMIN	REAL*8	minimum x value
YMAX	REAL*8	maximum y value
YMIN	REAL*8	minimum y value

Local Variables

COLS	INTEGER*2	number of text columns
DUMMY	INTEGER*2	intermediate variable
HALFY	INTEGER*2	intermediate variable
ROWS	INTEGER*2	number of text rows
XLEN	REAL*8	intermediate variable
XMAXP	REAL*8	maximum x value
XMINP	REAL*8	minimum x value
XWIDTH	INTEGER*2	number of x pixels
YHEIGHT	INTEGER*2	number of y pixels
YLEN	REAL*8	intermediate variable
YMAXP	REAL*8	maximum y value

YMINP	REAL*8	minimum y value

SUBROUTINE WINDUP
Sets up phase angle window

Commons COMMQQ	NOCOL	
	Variables in Argument List	
XMAX	REAL*8	maximum x value
XMIN	REAL*8	minimum x value
YMAX	REAL*8	maximum y value
YMIN	REAL*8	minimum y value
	Local Variables	
COLS	INTEGER*2	number of text columns
DUMMY	INTEGER*2	intermediate variable
HALFY	INTEGER*2	intermediate variable
ROWS	INTEGER*2	number of text rows
XLEN	REAL*8	intermediate variable
XMAXP	REAL*8	maximum x value
XMINP	REAL*8	minimum x value
XWIDTH	INTEGER*2	number of x pixels
YHEIGHT	INTEGER*2	number of y pixels
YLEN	REAL*8	intermediate variable
YMAXP	REAL*8	maximum y value
YMINP	REAL*8	minimum y value

SUBROUTINE ADMIT
Determines admittance looking toward tank

Commons FACTOR	WCATIT WORK1 WORK2	
	Variables in Argument List	
A	REAL*4	speed of sound in the fluid (ft/sec)
AREA(75,25)	REAL*4	area of pipe section (ft^2)
CMAN(25)	REAL*4	manifold capacitance
CTANK	REAL*4	tank capacitance
DPROR(25)	REAL*4	pressure drop across orifices (lbf/ft^2)
GADM(25)	COMPLEX*8	admittance looking toward tank
IENG(25)	INTEGER*2	engine number
ILINE	INTEGER*2	line number
IP	INTEGER*2	current pipe section
ITLIN	INTEGER*2	flag for fuel or lox
L(75,25)	REAL*4	length of pipe section (ft)
LFLOW	REAL*4	flow rate through pipe (lbm/sec)
LOPEND	INTEGER*2	maximum number of iterations for split pipe
NOLINE(25)	INTEGER*2	number of identical lines
PCAP(75,25)	REAL*4	capacitance of pipe section
PIND(75,25)	REAL*4	inductance of pipe section
PMRAT(25)	REAL*4	chamber pressure/total mass flow
S	COMPLEX*8	current frequency
SECIN(75,25)	INTEGER*2	pipe section type
SEGMN(25)	INTEGER*2	number of pipe sections
SPLIT	REAL*4	number of unique lines from pipe split

TFLOW(25)	REAL*4	total flow rate of engine (lbm/sec)
	Local Variables	
CAPM	COMPLEX*8	intermediate variable
CAPN	COMPLEX*8	intermediate variable
CFAC	COMPLEX*8	intermediate variable
ERRP	REAL*4	convergence error
GDIF	REAL*4	maximum difference in admittance
GOLD(0:75,25)	COMPLEX*8	previous addmittance
GRAV	REAL*4	gravitational constant (lbm-ft/lbf-sec^2)
I	INTEGER*2	do loop index
IE	INTEGER*2	current engine number
IOPEN	INTEGER*2	flag indicating if SURF.ERR is open
IWG	INTEGER*2	first index of maximum error
J	INTEGER*2	do loop index
JWG	INTEGER*2	second index of maximum error
K	INTEGER*2	do loop index
KLOOP	INTEGER*2	do loop index
LOPHI	INTEGER*2	intermediate variable
RATPM	REAL*4	intermediate variable
RHS	COMPLEX*8	intermediate variable
TCOUNT	REAL*4	intermediate variable
TL	REAL*4	length/speed of sound
TMASS	REAL*4	intermediate variable
TYPEL(2)	CHAR*13	intermediate array
WG	REAL*4	intermediate variable
WGOLD	REAL*4	intermediate variable
ZGEFF	COMPLEX*8	effective impedance for calculations
ZLP	REAL*4	intermediate variable
ZOEFF	REAL*4	effective ZO for calculations
ZOR(25)	REAL*4	intermediate variable
ZTEFF	COMPLEX*8	effective Zt for calculations
ZTOP	REAL*4	intermediate variable

SUBROUTINE BENDS

Computes effective straight pipe for bend

	Variables in Argument List	
DIME	REAL*4	effective diameter (ft)
PIPE1	REAL*4	radius of bend (ft)
PIPE2	REAL*4	angle of bend (degrees)
PIPE3	REAL*4	diameter of bend (ft)
PIPE4	REAL*4	length of end straight segments (ft)
VALUE	REAL*4	effective length (ft)
	Local Variables	
GAMMA	REAL*4	intermediate variable
LBEND	REAL*4	intermediate variable
RATIO	REAL*4	intermediate variable
Y	REAL*4	intermediate variable

SUBROUTINE BNSECT

Computes plot coordinates for a bend

Commons ARCCON PIPPXY

Variables in Argument List

ITYPE(200)	INTEGER*2	type plot element
J	INTEGER*2	pointer to element
PIPE1	REAL*4	first parameter of pipe description
PIPE2	REAL*4	second parameter of pipe description
PIPE3	REAL*4	third parameter of pipe description
PIPE4	REAL*4	fourth parameter of pipe description
POINT(8,200)	REAL*4	description of plot element

Local Variables

DIA	REAL*4	intermediate variable
HOLD	REAL*4	intermediate variable
RANG	REAL*4	intermediate variable
SLENTH	REAL*4	intermediate variable
X0	REAL*4	intermediate variable
X1	REAL*4	intermediate variable
X2	REAL*4	intermediate variable
X3	REAL*4	intermediate variable
Y0	REAL*4	intermediate variable
Y1	REAL*4	intermediate variable
Y2	REAL*4	intermediate variable
Y3	REAL*4	intermediate variable

COMPLEX FUNCTION CCOSH

Evaluates the complex hyperbolic cosine

Variables in Argument List

S	COMPLEX*8	current frequency

Local Variables

COSHI	REAL*4	intermediate variable
COSHR	REAL*4	intermediate variable
LAMDA	REAL*4	real part of complex frequency
MU	REAL*4	complex part of complex frequency

COMPLEX FUNCTION CSINH

Evaluates the complex hyperbolic sine

Variables in Argument List

S	COMPLEX*8	current frequency

Local Variables

LAMDA	REAL*4	real part of complex frequency
MU	REAL*4	complex part of complex frequency
SINHI	REAL*4	intermediate variable
SINHR	REAL*4	intermediate variable

COMPLEX FUNCTION CTANH

Evaluates the complex hyperbolic tangent

Variables in Argument List

S	COMPLEX*8	current frequency

SUBROUTINE ENGNO
Reads engine parameters

Commons EPARAM

Variables in Argument List

IUNIT	INTEGER*2	unit number of engine file

Local Variables

I	INTEGER*2	do loop index

SUBROUTINE FUEL
Handles fuel piping logic

Commons EPARAM FOPIPE FPARAM WCAOUT

Variables in Argument List

GF(25)	COMPLEX*8	admittance looking toward tank
IGONE	INTEGER*2	flag for subroutine fuel or lox
IUNIT	INTEGER*2	unit number of fuel data file
IUNITP	INTEGER*2	unit number of fuel work file
S	COMPLEX*8	current frequency

Local Variables

ANS	CHAR*1	response to question
FUELIN	CHAR*24	name of fuel data file

SUBROUTINE FULOX
Handles read, modify, and admittance calls for fuel and lox

Commons EPARAM

Variables in Argument List

A(25)	REAL*4	speed of sound in the fluid (ft/sec)
AREA(75,25)	REAL*4	area of pipe section (ft^2)
AVGK(25)	REAL*4	average bulk modulus
CMAN(25)	REAL*4	manifold capacitance
CTANK(25)	REAL*4	tank capacitance
DENS(25)	REAL*4	density of fluid (lbm/ft^3)
DIA(75,25)	REAL*4	diameter of pipe section (ft)
GF(25)	COMPLEX*8	admittance looking toward tank
IENG(25)	INTEGER*2	engine number
IGONE	INTEGER*2	flag for subroutine fuel or lox
ITANK(25)	INTEGER*2	tank number
ITLIN	INTEGER*2	flag indication fuel or lox
IUNIT	INTEGER*2	unit number of piping data file
IUNITP	INTEGER*2	unit number of working file
KMAN(25)	REAL*4	bulk modulus of manifold (lbf/ft^2)
KTANK(25)	REAL*4	bulk modulus of tank (lbf/ft^2)
L(75,25)	REAL*4	length of pipe section (ft)
LFLOW(25)	REAL*4	flow rate through pipe (lbm/sec)
LOPEND(25)	INTEGER*2	maximum number of iterations for split pipe
LOPOLD(25)	INTEGER*2	previous maximum number of iterations
MLINE	INTEGER*2	number of lines from tank
NOLINE(25)	INTEGER*2	number of identical lines
PCAP(75,25)	REAL*4	capacitance of pipe section

PIND(75,25)	REAL*4	inductance of pipe section
PIPE1(75,25)	REAL*4	first parameter of pipe description
PIPE2(75,25)	REAL*4	second parameter of pipe description
PIPE3(75,25)	REAL*4	third parameter of pipe description
PIPE4(75,25)	REAL*4	fourth parameter of pipe description
PIPE5(75,25)	REAL*4	fifth parameter of pipe description
S	COMPLEX*8	current frequency
SECTN(75,25)	INTEGER*2	pipe section type
SEGMN(25)	INTEGER*2	number of pipe sections
SPLIT(25)	REAL*4	number of unique lines from pipe split
VOL(25)	REAL*4	volume of tank (ft^3)
VOLMF(25)	REAL*4	volume of manifold (ft^3)
	Local Variables	
ANS	CHAR*1	response to question
I	INTEGER*2	do loop index
ILINE	INTEGER*2	current line number
IP	INTEGER*2	current segment number
IT	INTEGER*2	current tank number
QUEST1(2)	CHAR*40	question array
QUEST2(2)	CHAR*48	question array
QUEST3(2)	CHAR*40	question array
TITL	CHAR*20	title from data file

SUBROUTINE GETKS

Determines Nyquist equation to be plotted

	Variables in Argument List	
I	INTEGER*2	fuel line number
J	INTEGER*2	lox line number
K	INTEGER*2	engine number
K1C(1001)	REAL*4	complex part of K(jw)
K1R(1001)	REAL*4	real part of K(jw)
K2C(1001)	REAL*4	complex part of K(jw,GOX)
K2R(1001)	REAL*4	real part of K(jw,GOX)
K3C(1001)	REAL*4	complex part of K(jw,GF)
K3R(1001)	REAL*4	real part of K(jw,GF)
K4C(1001)	REAL*4	complex part of K(jw,GOX,GF)
K4R(1001)	REAL*4	real part of K(jw,GOX,GF)
N	INTEGER*2	number of points
	Local Variables	
C1K(25)	REAL*4	work array for complex part of K(jw)
C2K(25)	REAL*4	work array for complex part of K(jw,GOX)
C3K(25)	REAL*4	work array for complex part of K(jw,GF)
C4K(25)	REAL*4	work array for complex part of K(jw,GOX,GF)
L	INTEGER*2	do loop index
M	INTEGER*2	pointer
R1K(25)	REAL*4	work array for real part of K(jw)
R2K(25)	REAL*4	work array for real part of K(jw,GOX)
R3K(25)	REAL*4	work array for real part of K(jw,GF)
R4K(25)	REAL*4	work array for real part of K(jw,GOX,GF)

SUBROUTINE GEIM

Determines location of data to be plotted

Commons SETUP

	Variables in Argument List	
I	INTEGER*2	fuel line pointer
J	INTEGER*2	lox line pointer
M	INTEGER*2	plot pointer
	Local Variables	
II	INTEGER*2	do loop index
JJ	INTEGER*2	do loop index

SUBROUTINE GINERT

Evaluates curve fit of inertance of bends

	Variables in Argument List	
BEND	REAL*4	angle of bend (degrees)
X	REAL*4	ratio of inner to outer radius
Y	REAL*4	inertance
	Local Variables	
A	REAL*4	intermediate variable
B(3)	REAL*4	coefficient array for inertance fit

SUBROUTINE HHSECT

Computes plot coordinates for Helmholtz resonator

Commons PIPPXY

	Variables in Argument List	
DIA	REAL*4	diameter of opening (ft)
ITYPE(200)	INTEGER*2	type plot element
J	INTEGER*2	pointer to element
LEN	REAL*4	length of opening (ft)
POINT(8,200)	REAL*4	description of plot element
VOL	REAL*4	volume of reservoir (ft^3)
	Local Variables	
COSOLD	REAL*4	intermediate variable
DIAM	REAL*4	intermediate variable
SIDE	REAL*4	intermediate variable
SINOLD	REAL*4	intermediate variable
XC	REAL*4	intermediate variable
XHOLD	REAL*4	intermediate variable
XLOLD	REAL*4	intermediate variable
XOLD	REAL*4	intermediate variable
YC	REAL*4	intermediate variable
YHOLD	REAL*4	intermediate variable
YLOLD	REAL*4	intermediate variable
YOLD	REAL*4	intermediate variable

SUBROUTINE LOX

Handles fuel piping logic

Commons EPARAM FOPIPE OPARAM WCAOUT

Variables in Argument List

GOX(25)	COMPLEX*8	admittance looking toward tank
IGONE	INTEGER*2	flag for subroutine fuel or lox
IUNIT	INTEGER*2	unit number of lox data file
IUNITP	INTEGER*2	unit number of lox work file
S	COMPLEX*8	current frequency

Local Variables

ANS	CHAR*1	response to question
LOXIN	CHAR*24	name of lox data file

SUBROUTINE MODCON

Modifies CONST.RLN parameters

Variables in Argument List

CSTAR(25)	REAL*4	characteristic rocket velocity (ft/sec)
DCDR(25)	REAL*4	change in velocity with mixture ratio (ft/sec)
IUNIT	INTEGER*2	unit number for CONST data
MENG	INTEGER*2	number of engines
RBAR(25)	REAL*4	mixture ratio
TAUT(25)	REAL*4	transport lag (sec)
THETAC(25)	REAL*4	characteristic time constant (sec)
VARI	CHAR*24	name of CONST data file

Local Variables

ANS	CHAR*1	response to question
I	INTEGER*2	pointer
II	INTEGER*2	do loop index
J	INTEGER*2	do loop index
NAME	CHAR*8	name of variable to be modified
VALUE	REAL*4	value of variable to be modified
VARL(5)	CHAR*8	array of names (lower case)
VARU(5)	CHAR*8	array of names (upper case)

SUBROUTINE MODENG

Modifies engine parameters

Commons EPARAM

Variables in Argument List

IUNIT	INTEGER*2	unit number of engine data file
NAMENG	CHAR*24	name of engine data file

Local Variables

ANS	CHAR*1	response to question
I	INTEGER*2	pointer
II	INTEGER*2	do loop index
J	INTEGER*2	do loop index
NAME	CHAR*8	name of variable to be modified
VALUE	REAL*4	value of variable to be modified
VARL(3)	CHAR*8	array of names (lower case)
VARU(3)	CHAR*8	array of names (upper case)

SUBROUTINE MODIFY

Allows modifications to input data

Commons EPARAM TANK WCAOUT

Variables in Argument List

A(25)	REAL*4	speed of sound in the fluid (ft/sec)
AREA(75,25)	REAL*4	area of pipe section (ft^2)
AVGK(25)	REAL*4	average bulk modulus
CMAN(25)	REAL*4	manifold capacitance
CTANK(25)	REAL*4	tank capacitance
DENS(25)	REAL*4	density of fluid (lbm/ft^3)
DIA(75,25)	REAL*4	diameter of pipe section (ft)
IENG(25)	INTEGER*2	engine number
ITANK(25)	INTEGER*2	tank number
IUNIT	INTEGER*2	unit number of fuel or lox file
KMAN(25)	REAL*4	bulk modulus of manifold (lbf/ft^2)
KTANK(25)	REAL*4	bulk modulus of tank (lbf/ft^2)
L(75,25)	REAL*4	length of pipe section (ft)
LFLOW(25)	REAL*4	flow rate through pipe (lbm/sec)
LOPEND(25)	INTEGER*2	maximum number of iterations for split pipe
LOPOLD(25)	INTEGER*2	previous maximum number of iterations
MLINE	INTEGER*2	number of lines from tank
NOLINE(25)	INTEGER*2	number of identical lines
PCAP(75,25)	REAL*4	capacitance of pipe section
PIND(75,25)	REAL*4	inductance of pipe section
PIPE1(75,25)	REAL*4	first parameter of pipe description
PIPE2(75,25)	REAL*4	second parameter of pipe description
PIPE3(75,25)	REAL*4	third parameter of pipe description
PIPE4(75,25)	REAL*4	fourth parameter of pipe description
PIPE5(75,25)	REAL*4	fifth parameter of pipe description
SECIN(75,25)	INTEGER*2	pipe section type
SEGMN(25)	INTEGER*2	number of pipe sections
SPLIT(25)	REAL*4	number of unique lines from pipe split
TITL	CHAR*20	title from input file
VOL(25)	REAL*4	volume of tank (ft^3)
VOLMF(25)	REAL*4	volume of manifold (ft^3)

Local Variables

ANS	CHAR*1	response to question
I	INTEGER*2	pointer
II	INTEGER*2	do loop index
III	INTEGER*2	do loop index
IP	INTEGER*2	pointer to current segment
IPP	INTEGER*2	do loop index
ISEGMN	INTEGER*2	intermediate variable
IT	INTEGER*2	current tank number
J	INTEGER*2	do loop index
K	INTEGER*2	do loop index
M	INTEGER*2	do loop index
NAMNAM	INTEGER*2	pointer, fuel or lox
SECT	REAL*4	type of segment

SUBROUTINE MODTAN
Modifies tank parameters

Variables in Argument List

A(25)	REAL*4	speed of sound in the fluid (ft/sec)
CTANK(25)	REAL*4	tank capacitance
DENS(25)	REAL*4	density of fluid (lbm/ft^3)
KTANK(25)	REAL*4	bulk modulus of tank (lbf/ft^2)
LFLOW(25)	REAL*4	flow rate through pipe (lbm/sec)
MTANK	INTEGER*2	number of tanks
VOL(25)	REAL*4	volume of tank (ft^3)

Local Variables

ANS	CHAR*1	response to question
GRAV	REAL*4	gravitational constant (lbm-ft/lbf-sec^2)
I	INTEGER*2	pointer
II	INTEGER*2	do loop index
J	INTEGER*2	do loop index
NAME	CHAR*8	name of variable to be modified
VALUE	REAL*4	value of variable to be modified
VARL(4)	CHAR*8	array of names (lower case)
VARU(4)	CHAR*8	array of names (upper case)

SUBROUTINE NYQUIS
Computes the K()'s

Commons EPARAM FACTOR SETUP

Variables in Argument List

CSTAR(25)	REAL*4	characteristic rocket velocity (ft/sec)
DCDR(25)	REAL*4	change in velocity with mixture ratio (ft/sec)
GF(25)	COMPLEX*8	admittance of fuel line looking toward tank
GOX(25)	COMPLEX*8	admittance of lox line looking toward tank
IFUEL	INTEGER*2	flag indicating presence of fuel line
ILOX	INTEGER*2	flag indicating presence of lox line
RBAR(25)	REAL*4	mixture ratio
S	COMPLEX*8	complex frequency
TAUT(25)	REAL*4	transport lag (sec)
THETAC(25)	REAL*4	characteristic time constant (sec)

Local Variables

I	INTEGER*2	do loop index
J	INTEGER*2	do loop index
K	INTEGER*2	current engine number
KG1(25)	COMPLEX*8	K(jw)
KG2	COMPLEX*8	K(jw,GOX)
KG3	COMPLEX*8	K(jw,GF)
K1C(25)	REAL*4	complex part of K(jw)
K2C(25)	REAL*4	complex part of K(jw,GOX)
K3C(25)	REAL*4	complex part of K(jw,GF)
K4C(25)	REAL*4	complex part of K(jw,GOX,GF)
K1R(25)	REAL*4	real part of K(jw)
K2R(25)	REAL*4	real part of K(jw,GOX)
K3R(25)	REAL*4	real part of K(jw,GF)
K4R(25)	REAL*4	real part of K(jw,GOX,GF)

L	INTEGER*2	pointer

SUBROUTINE RLINE
Reads fuel or lox file.

Commons EPARAM TANK

Variables in Argument List

A(25)	REAL*4	speed of sound in the fluid (ft/sec)
AREA(75,25)	REAL*4	area of pipe section (ft^2)
AVGK(25)	REAL*4	average bulk modulus
CMAN(25)	REAL*4	manifold capacitance
CTANK(25)	REAL*4	tank capacitance
DENS(25)	REAL*4	density of fluid (lbm/ft^3)
DIA(75,25)	REAL*4	diameter of pipe section (ft)
IENG(25)	INTEGER*2	engine number
ITANK(25)	INTEGER*2	tank number
IUNIT	INTEGER*2	unit number of fuel or lox file
KMAN(25)	REAL*4	bulk modulus of manifold (lbf/ft^2)
KTANK(25)	REAL*4	bulk modulus of tank (lbf/ft^2)
L(75,25)	REAL*4	length of pipe section (ft)
LFLOW(25)	REAL*4	flow rate through pipe (lbm/sec)
LOPEND(25)	INTEGER*2	maximum number of iterations for split pipe
LOPOLD(25)	INTEGER*2	previous maximum number of iterations
MLINE	INTEGER*2	number of lines from tank
NOLINE(25)	INTEGER*2	number of identical lines
PCAP(75,25)	REAL*4	capacitance of pipe section
PIND(75,25)	REAL*4	inductance of pipe section
PIPE1(75,25)	REAL*4	first parameter of pipe description
PIPE2(75,25)	REAL*4	second parameter of pipe description
PIPE3(75,25)	REAL*4	third parameter of pipe description
PIPE4(75,25)	REAL*4	fourth parameter of pipe description
PIPE5(75,25)	REAL*4	fifth parameter of pipe description
SECTN(75,25)	INTEGER*2	pipe section type
SEGMN(25)	INTEGER*2	number of pipe sections
SPLIT(25)	REAL*4	number of unique lines from pipe split
TITL	CHAR*20	title from input file
VOL(25)	REAL*4	volume of tank (ft^3)
VOLMF(25)	REAL*4	volume of manifold (ft^3)

Local Variables

ANS	CHAR*1	response to question
DIVAVG	REAL*4	intermediate variable
I	INTEGER*2	do loop index
IE	INTEGER*2	current engine number
IT	INTEGER*2	current tank number
J	INTEGER*2	do loop index
M	INTEGER*2	pointer
MM	INTEGER*2	do loop index

SUBROUTINE RTYPE
Stores values for different types of piping

Variables in Argument List

AREA	REAL*4	area of pipe section (ft^2)
AVGK	REAL*4	average bulk modulus
CMAN	REAL*4	manifold capacitance
DENS	REAL*4	density of fluid (lbm/ft^3)
DIA	REAL*4	diameter of pipe section (ft)
KMAN	REAL*4	bulk modulus of manifold (lbf/ft^2)
L	REAL*4	length of pipe section (ft)
PCAP	REAL*4	capacitance of pipe section
PIND	REAL*4	inductance of pipe section
PIPE1	REAL*4	first parameter of pipe description
PIPE2	REAL*4	second parameter of pipe description
PIPE3	REAL*4	third parameter of pipe description
PIPE4	REAL*4	fourth parameter of pipe description
PIPE5	REAL*4	fifth parameter of pipe description
SECTN	INTEGER*2	pipe section type
VOLMF	REAL*4	volume of manifold (ft^3)

Local Variables

AREAB	REAL*4	area of pipe
DIME	REAL*4	diameter of pipe
GRAV	REAL*4	gravitational constant (lbm-ft/lbf-sec^2)
PI	REAL*4	mathematical constant
VALUE	REAL*4	length of pipe

SUBROUTINE STSECT

Computes plot coordinates for a straight section

Commons PIPPXY

Variables in Argument List

DIA	REAL*4	diameter of segment (ft)
ITYPE(200)	INTEGER*2	type plot element
J	INTEGER*2	pointer to element
LEN	REAL*4	length of segment (ft)
POINT(8,200)	REAL*4	description of plot element

SUBROUTINE TANKNO

Reads tank parameters

Variables in Argument List

A(25)	REAL*4	speed of sound in the fluid (ft/sec)
CTANK(25)	REAL*4	tank capacitance
DENS(25)	REAL*4	density of fluid (lbm/ft^3)
IUNIT	INTEGER*2	unit number of fuel or lox file
KTANK(25)	REAL*4	bulk modulus of tank (lbf/ft^2)
LFLOW(25)	REAL*4	flow rate through pipe (lbm/sec)
MTANK	INTEGER*2	number of tanks
VOL(25)	REAL*4	volume of tank (ft^3)

Local Variables

GRAV	REAL*4	gravitational constant (lbm-ft/lbf-sec^2)
I	INTEGER*2	do loop index

SUBROUTINE TSSECT
Computes plot coordinates for a tuned stub

Commons PIPPXY

	Variables in Argument List	
DIA	REAL*4	diameter of tuned stub (ft)
ITYPE(200)	INTEGER*2	type plot element
J	INTEGER*2	pointer to element
LEN	REAL*4	length of tuned stub
POINT(8,200)	REAL*4	description of plot element
	Local Variables	
DIAM	REAL*4	intermediate variable

SUBROUTINE ZREAD
Reads input for input modification

	Variables in Argument List	
NAME(8)	CHAR*1	name of input variable
VALUE	REAL*4	value of input variable
	Local Variables	
BLK	CHAR*1	' '
CARD(80)	CHAR*1	card image
CEND(3)	CHAR*1	'E','N','D'
COMMA	CHAR*1	','
CTIT(5)	CHAR*1	'T','I','T','L','E'
DCARD	CHAR*80	card image
E	CHAR*1	'E'
FRACT	REAL*4	fractional part of number
I	INTEGER*2	do loop index
ICOUNT	INTEGER*2	position counter
ID	INTEGER*2	position counter
II	INTEGER*2	position counter
J	INTEGER*2	do loop index
JJ	INTEGER*2	position counter
LE	CHAR*1	'e'
LEND(3)	CHAR*1	'e','n','d'
LTIT(5)	CHAR*1	't','i','t','l','e'
MINUS	CHAR*1	'-'
NUMBER(10)	CHAR*1	'0','1','2','3','4','5','6','7','8','9'
PERIOD	CHAR*1	'.'
PLUS	CHAR*1	'+'
POUND	CHAR*1	'#'
QUEST	CHAR*1	'?'
SIGN	REAL*4	sign of number or exponent
WHOLE	REAL*4	whole part of number

7.0 Program Listing

```
C
C     PROGRAM NYQUIST  03-24-92
C
C        Program to calculate values for a Nyquist plot using no fuel or
C        lox lines, fuel line only, lox line only, or fuel and lox lines
C
C        This program will handle the following type elements
C
C              Straight pipes
C              Bends
C              Inline accumulators
C              Tuned stub accumulators
C              Helmholtz resonators
C              Parallel resonators
C              Pumps
C              Split pipes
C              Multiple tanks
C              Multiple engines
C
$LARGE
      INCLUDE 'FGRAPH.FI'
      INCLUDE 'FGRAPH.FD'
      COMMON /NOCOL/NCOLS,NMODE
      INTEGER*2 NCOLS,NMODE
      INTEGER*2 IHR,IMIN,ISEC,I100,IYR,IMON,IDAY
      CHARACTER*2 AM,PM,AP
      COMMON /EPARAM/MENG,TFLOW(25),PCHMB(25),DPROR(25),PMRAT(25)
      INTEGER SEGMNF(25),SECINF(75,25),NOLINF(25),IENGF(25),ITANKF(25),
     *        LOPOLF(25),LOPENF(25)
      REAL KMANF(25),KTANKF(25),LFLOWF(25),LF(75,25)
      COMMON /FPARAM/MLINEF,SPLITF(25),AF(25),CMANF(25),CTANKF(25),
     *               DENSF(25),KMANF,KTANKF,LFLOWF,VOLF(25),VOLMFF(25),
     *               AREAF(75,25),DIAF(75,25),LF,PINDF(75,25),
     *               PCAPF(75,25),AVGKF(25),
     *               SEGMNF,SECINF,NOLINF,IENGF,ITANKF,LOPOLF,LOPENF
      INTEGER SEGMNO(25),SECINO(75,25),NOLINO(25),IENGO(25),ITANKO(25),
     *        LOPOLO(25),LOPENO(25)
      REAL KMANO(25),KTANKO(25),LFLOWO(25),LO(75,25)
      COMMON /OPARAM/MLINEO,SPLITO(25),AO(25),CMANO(25),CTANKO(25),
     *               DENSO(25),KMANO,KTANKO,LFLOWO,VOLO(25),VOLMFO(25),
     *               AREAO(75,25),DIAO(75,25),LO,PINDO(75,25),
     *               PCAPO(75,25),AVGKO(25),
     *               SEGMNO,SECINO,NOLINO,IENGO,ITANKO,LOPOLO,LOPENO
      COMPLEX S,GF(25),GOX(25)
      REAL K1R(1001),K2R(1001),K3R(1001),K1C(1001),K2C(1001),K3C(1001)
      REAL K4R(1001),K4C(1001),KW(1001)
      COMMON /WORK1/K1R,K1C,K2R,K2C,K3R,K3C,K4R,K4C,DUMMY1(3392)
      REAL LFREQ,TAUT(25),CSTAR(25),RBAR(25),THETAC(25),DCDR(25)
      INTEGER PTS
      CHARACTER*1 ANS
```

```
      CHARACTER*2 ANS1
      EQUIVALENCE (ANS,ANS1(1:1))
      CHARACTER*24 NAMLIN(2),NAMENG
      CHARACTER*40 TITLE
      CHARACTER*20 TITLF
      CHARACTER*24 VARI
      COMMON /WCATIT/TITLE,TITLF,IHR,IMIN,AP,IYR,IMON,IDAY
      COMMON /WCAOUT/NAMLIN
      COMMON /FACTOR/SFAC
      INTEGER SEGMN,SECIN(150)
      COMMON /SETUP/PIPE1(150),PIPE2(150),PIPE3(150),PIPE4(150),
     *              NENGF(25),NTANKF(25),NLINEF(25),NSPF(25),ILINEF,
     *              NENGO(25),NTANKO(25),NLINEO(25),NSPO(25),ILINEO,
     *              SEGMN,SECIN
      DATA AM/'AM'/,PM/'PM'/
    1 FORMAT(A20,1X,I2.2,':',I2.2,A2,4X,I2.2,'-',I2.2,'-',I2.2)
      OPEN(17,FORM='UNFORMATTED')
      OPEN(UNIT=14,FILE='NYQ.OUT')
      CALL GETTIM(IHR,IMIN,ISEC,I100)
      CALL GETDAT(IYR,IMON,IDAY)
      IYR=IYR-1900
      CALL CLEARSCREEN(0)
      WRITE(*,'(10X,A)')
     *'╔═══════════════════════════════════════════════════════════╗'
      WRITE(*,'(10X,A)')
     *'║                                                           ║'
      IF(IHR.LT.12)  THEN
       WRITE(*,'(10X,A)')
     *'║              Good Morning and Welcome to NYQUIST!!        ║'
       AP=AM
      ELSE
       WRITE(*,'(10X,A)')
     *'║             Good Afternoon and Welcome to NYQUIST!!       ║'
       AP=PM
       IF(IHR.GT.12)  IHR=IHR-12
      ENDIF
      WRITE(*,'(10X,A)')
     *'║                                                           ║'
      WRITE(*,'(10X,A)')
     *'║        Program NYQUIST provides stability predictions     ║'
      WRITE(*,'(10X,A)')
     *'║                       of feedline systems                 ║'
      WRITE(*,'(10X,A)')
     *'║                                                           ║'
      WRITE(*,'(10X,A)')
     *'║                  To send a plot to the printer            ║'
      WRITE(*,'(10X,A)')
     *'║                                                           ║'
      WRITE(*,'(10X,A)')
     *'║              The computer MUST be in GRAPHICS mode        ║'
      WRITE(*,'(10X,A)')
     *'║                                                           ║'
      WRITE(*,'(10X,A)')
```

```
*'||      Hit PrScn to send the current plot to the printer          ||'
 WRITE(*,'(10X,A)')                                                  ||'
*'||                                                                 ||'
 WRITE(*,'(10X,A)')                                                  ||'
*'|L=================================================================='
 WRITE(*,*)' '
 SFAC=1.0
 WRITE(*,'(A)')' If you want frequency in rad/sec, hit enter.'
 WRITE(*,'(A\)')' If you want it in Hertz, enter "H". '
 READ(*,'(A)')ANS
 IF(ANS.EQ.'H'.OR.ANS.EQ.'h')  SFAC=6.283185
21 CONTINUE
 WRITE(*,'(A\)')' Do you have FUEL data? '
 READ(*,'(A)')ANS
 IF(ANS .EQ.'N' .OR. ANS .EQ. 'n') THEN
  IFUEL=1
 ELSE
  IFUEL=0
 ENDIF
 WRITE(*,'(A\)')' Do you have LOX data? '
 READ(*,'(A)')ANS
 IF(ANS .EQ.'N' .OR. ANS .EQ. 'n') THEN
  ILOX=1
 ELSE
  ILOX=0
 ENDIF
 IF(IFUEL.EQ.0.OR.ILOX.EQ.0)  THEN
  WRITE(*,'(A\)')' Is the engine data on file ENG.RLN? (Y/N) '
  READ(*,'(A)')ANS
  IF(ANS.NE.'N'.AND.ANS.NE.'n')  THEN
   NAMENG='ENG.RLN'
  ELSE
   WRITE(*,'(A\)')'  Enter name of file with the engine data '
   READ(*,'(A)')NAMENG
  ENDIF
  OPEN(UNIT=9,FILE=NAMENG)
  JUNIT=9
  CALL ENGNO(JUNIT)
 ELSE
  MENG=1
 ENDIF
 IF(IFUEL.EQ.0)  THEN
  IGONE=2
  CALL FUEL(S,GF,11,16,IGONE)
 ENDIF
 IF(ILOX.EQ.0)  THEN
  IGONE=2
  CALL LOX(S,GOX,10,15,IGONE)
 ENDIF
 IGONE=0
 WRITE(*,*)'  Are the following variables in a file? (Y/N) '
 WRITE(*,*)' '
 WRITE(*,*)'       VARIABLES   '
```

```
      WRITE(*,*)' TRANSPORT LAG'
      WRITE(*,*)' CHARACTERISTIC ROCKET VELOCITY'
      WRITE(*,*)' MIXTURE RATIO '
      WRITE(*,*)' CHARACTERISTIC TIME CONSTANT '
      WRITE(*,*)' CHANGE IN VELOCITY WITH MIXTURE RATIO '
      WRITE(*,*)' '
      READ(*,'(A)')ANS
      IF(ANS .EQ. 'N' .OR. ANS .EQ. 'n') THEN
       WRITE(*,*)'         File BUFFY.DOG will be created '
       OPEN(UNIT=12,FILE='BUFFY.DOG')
       VARI='BUFFY.DOG'
       WRITE(*,*)'Enter values for VARIABLES as listed above.'
       DO 24 I=1,MENG
   22   CONTINUE
        WRITE(*,'('' for engine #'',I2)')I
        READ(*,*,ERR=23)TAUT(I),CSTAR(I),RBAR(I),THETAC(I),DCDR(I)
        WRITE(12,'(1P5E15.5)')TAUT(I),CSTAR(I),RBAR(I),THETAC(I),DCDR(I)
        GO TO 24
   23  CONTINUE
       WRITE(*,*)'  Enter numeric values only.  Please try again !!'
       GO TO 22
   24 CONTINUE
      ELSE
       WRITE(*,'(A\)')' Is the name of the file CONST.RLN? (Y/N) '
       READ(*,'(A)')ANS
       IF(ANS.EQ.'N'.OR.ANS.EQ.'n')  THEN
        WRITE(*,'(A\)')'  Enter name of file with VARIABLES data '
        READ(*,'(A)')VARI
       ELSE
        VARI='CONST.RLN'
       ENDIF
       OPEN(UNIT=12,FILE=VARI)
       DO 25 I=1,MENG
        READ(12,*)TAUT(I),CSTAR(I),RBAR(I),THETAC(I),DCDR(I)
   25  CONTINUE
      ENDIF
   26 CONTINUE
      WRITE(*,*)' Enter 20 character title'
      READ(*,'(A)')TITLF
      WRITE(TITLE,1)TITLF,IHR,IMIN,AP,IMON,IDAY,IYR
   27 CONTINUE
      REWIND 17
   28 CONTINUE
      IF(SFAC.EQ.1.0)  THEN
       WRITE(*,*)' Enter range of frequencies in rad/sec '
      ELSE
       WRITE(*,*)' Enter range of frequencies in Hertz '
      ENDIF
      WRITE(*,*)' Low freq, high freq, #pts'
      WRITE(*,*)'   1001 = Maximum number of points'
      READ(*,*,ERR=29)LFREQ,HFREQ,PTS
      IF(LFREQ.LE.0.0)  LFREQ=1.0E-5
      IF(PTS.LE.1)  GO TO 49
```

```
      GO TO 30
   29 CONTINUE
      WRITE(*,*)'  Enter numeric values only. Please try again !!'
      GO TO 28
   30 CONTINUE
      NPTS=PTS/3
      IF(NPTS.GT.1)  THEN
       SSIZE1=0.1*(HFREQ-LFREQ)/(NPTS-1)
       SSIZE2=0.3*(HFREQ-LFREQ)/NPTS
       IF(3*NPTS.EQ.PTS)  THEN
        SSIZE3=0.6*(HFREQ-LFREQ)/NPTS
       ELSEIF(3*NPTS.EQ.PTS-1)  THEN
        SSIZE3=0.6*(HFREQ-LFREQ)/(NPTS+1)
       ELSEIF(3*NPTS.EQ.PTS-2)  THEN
        SSIZE3=0.6*(HFREQ-LFREQ)/(NPTS+2)
       ENDIF
      ELSE
       SSIZE1=(HFREQ-LFREQ)/(PTS-1)
       NPTS=PTS
      ENDIF
       IF(IFUEL.EQ.0.OR.ILOX.EQ.0)  THEN
        IF(IFUEL.NE.0)  THEN
         CALL LPLOT(ILOX)
        ELSEIF(ILOX.NE.0)  THEN
         CALL FPLOT(ILOX)
        ELSE
         CALL FLPLOT(ILOX)
        ENDIF
       ENDIF
       WRITE(*,*)'  Please wait while computations proceed. '
      W=LFREQ
      WRITE(14,'(1X,A/)')TITLE
      IF(IFUEL.NE.0.AND.ILOX.NE.0)
     * WRITE(14,'(/4X,''FREQ.'',7X,''K1(R)'',7X,''K1(I)'')')
      IF(IFUEL.EQ.0.AND.ILOX.NE.0)  WRITE(14,'(/4X,''FREQ.'',7X,
     * ''K1(R)'',7X,''K1(I)''5X,''ENG.'',4X,''K3(R)'',7X,''K3(I)''/)')
      IF(IFUEL.NE.0.AND.ILOX.EQ.0)  WRITE(14,'(/4X,''FREQ.'',7X,
     * ''K1(R)'',7X,''K1(I)''5X,''ENG.'',4X,''K2(R)'',7X,''K2(I)''/)')
      IF(IFUEL.EQ.0.AND.ILOX.EQ.0)  THEN
       WRITE(14,'(/4X,''FREQ.'',7X,''K1(R)'',7X,''K1(I)'')')
       WRITE(14,'(5X,''ENG.'',2X,''K2(R)'',7X,''K2(I)'',7X,''K3(R)'',7X,
     *          ''K3(I)'',7X,''K4(R)'',7X,''K4(I)''/)')
      ENDIF
      DO 31 K=1,PTS
       IF(K.LE.NPTS)  THEN
        IF(K.GT.1)  W=W+SSIZE1
       ELSEIF(K.GT.2*NPTS)  THEN
        W=W+SSIZE3
       ELSE
        W=W+SSIZE2
       ENDIF
       IF(K.EQ.PTS)  THEN
        W=HFREQ
```

```
      ENDIF
      KW(K)=W
      S=CMPLX(0.0,SFAC*W)
      IF(IFUEL.EQ.0)  CALL FUEL(S,GF,11,16,IGONE)
      IF(ILOX.EQ.0)  CALL LOX(S,GOX,10,15,IGONE)
      CALL NYQUIS(GF,GOX,S,TAUT,CSTAR,RBAR,DCDR,THETAC,IFUEL,ILOX)
   31 CONTINUE
   32 CONTINUE
     WRITE(*,*)'   Enter graph selection '
     WRITE(*,*)' '
     WRITE(*,*)'   1   Nyquist plot independent of fuel or lox. '
     IF(ILOX.EQ.0)
    * WRITE(*,*)'   2   Nyquist plot independent of fuel.'
     IF(IFUEL.EQ.0)
    * WRITE(*,*)'   3   Nyquist plot independent of lox.'
     IF(ILOX.EQ.0.AND.IFUEL.EQ.0)
    * WRITE(*,*)'   4   Nyquist plot with fuel and lox.'
     WRITE(*,*)'   5   Phase-Gain plot independent of fuel or lox. '
     IF(ILOX.EQ.0)
    * WRITE(*,*)'   6   Phase-Gain plot independent of fuel.'
     IF(IFUEL.EQ.0)
    * WRITE(*,*)'   7   Phase-Gain plot independent of lox.'
     IF(ILOX.EQ.0.AND.IFUEL.EQ.0)
    * WRITE(*,*)'   8   Phase-Gain plot with fuel and lox.'
     WRITE(*,*)'   9   End plots.'
     WRITE(*,*)' '
     ANS1(2:2)=' '
     READ(*,'(A)')ANS1
     IF(ANS.EQ.'9')  GO TO 49
     IF(ANS1(2:2).NE.' '.OR.ANS.EQ.'0')  THEN
      WRITE(*,*)' Number must be between 1 and 9, TRY AGAIN'
      GO TO 32
     ENDIF
     IF(ILOX.EQ.1)  THEN
      IF(ANS.EQ.'2'.OR.ANS.EQ.'4'.OR.ANS.EQ.'6'.OR.ANS.EQ.'8')  THEN
       WRITE(*,*)' No LOX file, do not use 2,4,6,8'
       GO TO 32
      ENDIF
     ENDIF
     IF(IFUEL.EQ.1)  THEN
      IF(ANS.EQ.'3'.OR.ANS.EQ.'4'.OR.ANS.GE.'7')  THEN
       WRITE(*,*)' No FUEL file, do not use 3,4,7,8'
       GO TO 32
      ENDIF
     ENDIF
     CALL GETTIM(IHR,IMIN,ISEC,I100)
     CALL GETDAT(IYR,IMON,IDAY)
     IYR=IYR-1900
     IF(IHR.LT.12)  THEN
      AP=AM
     ELSE
      AP=PM
      IF(IHR.GT.12)  IHR=IHR-12
```

```
      ENDIF
      IF(ANS.EQ.'1')  THEN
        K=1
       CALL GETKS(PTS,K,0,0,K1R,K2R,K3R,K4R,K1C,K2C,K3C,K4C)
       CALL ALLPT(K1R,K1C,PTS,1,NTANKF(1),NTANKO(1),K)
      ELSEIF(ANS.EQ.'2') THEN
       IF(ILINEO.EQ.1)  THEN
        J=1
       ELSE
        WRITE(*,*)' The following LOX lines are available'
        WRITE(*,'(/'' Line #    Tank #   Engine #''/)')
        DO 33 J=1,ILINEO
         WRITE(*,'(I5,I10,I11)')J,NTANKO(J),NENGO(J)
33      CONTINUE
34      CONTINUE
        WRITE(*,'(/'' Enter line # to be plotted ''\)')
        READ(*,*)J
        IF(J.LE.0.OR.J.GT.ILINEO)  THEN
         WRITE(*,*)' Line # invalid, try again'
         GO TO 34
        ENDIF
       ENDIF
       K=NENGO(J)
       CALL GETKS(PTS,K,0,J,K1R,K2R,K3R,K4R,K1C,K2C,K3C,K4C)
       CALL ALLPT(K2R,K2C,PTS,2,NTANKF(1),NTANKO(J),K)
      ELSEIF(ANS.EQ.'3') THEN
       IF(ILINEF.EQ.1)  THEN
        I=1
       ELSE
        WRITE(*,*)' The following FUEL lines are available'
        WRITE(*,'(/'' Line #    Tank #   Engine #''/)')
        DO 35 I=1,ILINEF
         WRITE(*,'(I5,I10,I11)')I,NTANKF(I),NENGF(I)
35      CONTINUE
36      CONTINUE
        WRITE(*,'(/'' Enter line # to be plotted ''\)')
        READ(*,*)I
        IF(I.LE.0.OR.I.GT.ILINEF)  THEN
         WRITE(*,*)' Line # invalid, try again'
         GO TO 36
        ENDIF
       ENDIF
       K=NENGF(I)
       CALL GETKS(PTS,K,I,0,K1R,K2R,K3R,K4R,K1C,K2C,K3C,K4C)
       CALL ALLPT(K3R,K3C,PTS,3,NTANKF(I),NTANKO(1),K)
      ELSEIF(ANS.EQ.'4') THEN
       IF(ILINEF.EQ.1)  THEN
        I=1
       ELSE
        WRITE(*,*)' The following FUEL lines are available'
        WRITE(*,'(/'' Line #    Tank #   Engine #''/)')
        DO 37 I=1,ILINEF
         WRITE(*,'(I5,I10,I11)')I,NTANKF(I),NENGF(I)
```

```
37    CONTINUE
38    CONTINUE
      WRITE(*,'(/'' Enter line # to be plotted ''\)')
      READ(*,*)I
      IF(I.LE.0.OR.I.GT.ILINEF)  THEN
       WRITE(*,*)' Line # invalid, try again'
       GO TO 38
      ENDIF
     ENDIF
     K=NENGF(I)
     IF(ILINEO.EQ.0)  THEN
      J=1
     ELSE
      WRITE(*,*)' The following LOX lines are available'
      WRITE(*,'(/'' Line #    Tank #   Engine #''/)')
      NOXY=0
      DO 39 J=1,ILINEO
       IF(NENGO(J).NE.K)  GO TO 39
       WRITE(*,'(I5,I10,I11)')J,NTANKO(J),NENGO(J)
       NOXY=NOXY+1
       NOXYP=J
39    CONTINUE
      IF(NOXY.EQ.1)  THEN
       J=NOXYP
      ELSE
40     CONTINUE
       WRITE(*,'(/'' Enter line # to be plotted ''\)')
       READ(*,*)J
       IF(J.LE.0.OR.J.GT.ILINEO.OR.NENGO(J).NE.K)  THEN
        WRITE(*,*)' Line # invalid, try again'
        GO TO 40
       ENDIF
      ENDIF
     ENDIF
     CALL GETKS(PTS,K,I,J,K1R,K2R,K3R,K4R,K1C,K2C,K3C,K4C)
     IF(K.NE.0)  CALL ALLPT(K4R,K4C,PTS,4,NTANKF(I),NTANKO(J),K)
    ELSEIF(ANS.EQ.'5') THEN
     K=1
     CALL GETKS(PTS,K,0,0,K1R,K2R,K3R,K4R,K1C,K2C,K3C,K4C)
     CALL PNYQ(K1R,K1C,KW,PTS,1,NTANKF(1),NTANKO(1),K)
    ELSEIF(ANS.EQ.'6') THEN
     IF(ILINEO.EQ.1)  THEN
      J=1
     ELSE
      WRITE(*,*)' The following LOX lines are available'
      WRITE(*,'(/'' Line #    Tank #   Engine #''/)')
      DO 41 J=1,ILINEO
       WRITE(*,'(I5,I10,I11)')J,NTANKO(J),NENGO(J)
41    CONTINUE
42    CONTINUE
      WRITE(*,'(/'' Enter line # to be plotted ''\)')
      READ(*,*)J
      IF(J.LE.0.OR.J.GT.ILINEO)  THEN
```

```
      WRITE(*,*)' Line # invalid, try again'
      GO TO 42
     ENDIF
    ENDIF
    K=NENGO(J)
    CALL GETKS(PTS,K,0,J,K1R,K2R,K3R,K4R,K1C,K2C,K3C,K4C)
    CALL PNYQ(K2R,K2C,KW,PTS,2,NTANKF(1),NTANKO(J),K)
   ELSEIF(ANS.EQ.'7') THEN
    IF(ILINEF.EQ.1)  THEN
     I=1
    ELSE
     WRITE(*,*)' The following FUEL lines are available'
     WRITE(*,'(/'' Line #    Tank #   Engine #''/)')
     DO 43 I=1,ILINEF
      WRITE(*,'(I5,I10,I11)')I,NTANKF(I),NENGF(I)
43   CONTINUE
44   CONTINUE
     WRITE(*,'(/'' Enter line # to be plotted ''\)')
     READ(*,*)I
     IF(I.LE.0.OR.I.GT.ILINEF)  THEN
      WRITE(*,*)' Line # invalid, try again'
      GO TO 44
     ENDIF
    ENDIF
    K=NENGF(I)
    CALL GETKS(PTS,K,I,0,K1R,K2R,K3R,K4R,K1C,K2C,K3C,K4C)
    CALL PNYQ(K3R,K3C,KW,PTS,3,NTANKF(I),NTANKO(1),K)
   ELSEIF(ANS.EQ.'8') THEN
    IF(ILINEF.EQ.1)  THEN
     I=1
    ELSE
     WRITE(*,*)' The following FUEL lines are available'
     WRITE(*,'(/'' Line #    Tank #   Engine #''/)')
     DO 45 I=1,ILINEF
      WRITE(*,'(I5,I10,I11)')I,NTANKF(I),NENGF(I)
45   CONTINUE
46   CONTINUE
     WRITE(*,'(/'' Enter line # to be plotted ''\)')
     READ(*,*)I
     IF(I.LE.0.OR.I.GT.ILINEF)  THEN
      WRITE(*,*)' Line # invalid, try again'
      GO TO 46
     ENDIF
    ENDIF
    K=NENGF(I)
    IF(ILINEO.EQ.1)  THEN
     J=1
    ELSE
     WRITE(*,*)' The following LOX lines are available'
     WRITE(*,'(/'' Line #    Tank #   Engine #''/)')
     NOXY=0
     DO 47 J=1,ILINEO
      IF(NENGO(J).NE.K)  GO TO 47
```

```
      WRITE(*,'(I5,I10,I11)')J,NTANKO(J),NENGO(J)
      NOXY=NOXY+1
      NOXYP=J
47    CONTINUE
      IF(NOXY.EQ.1)  THEN
       J=NOXYP
      ELSE
48     CONTINUE
       WRITE(*,'(/'' Enter line # to be plotted ''\)')
       READ(*,*)J
       IF(J.LE.0.OR.J.GT.ILINEO.OR.NENGO(J).NE.K)  THEN
        WRITE(*,*)' Line # invalid, try again'
        GO TO 48
       ENDIF
      ENDIF
     ENDIF
     CALL GETKS(PTS,K,I,J,K1R,K2R,K3R,K4R,K1C,K2C,K3C,K4C)
     IF(K.NE.0)  CALL PNYQ(K4R,K4C,KW,PTS,4,NTANKF(I),NTANKO(J),K)
    ENDIF
    GO TO 32
49  CONTINUE
    WRITE(*,*)' Enter E to exit,'
    WRITE(*,*)'       F to run new frequency range,'
    WRITE(*,*)'       C to run a new case, '
    WRITE(*,'(A\)')'        N to read new files. '
    READ(*,'(A)')ANS
    IF(ANS.EQ.'F'.OR.ANS.EQ.'f')  GO TO 27
    IF(ANS.EQ.'E'.OR.ANS.EQ.'e')  STOP
    IF(ANS.EQ.'C'.OR.ANS.EQ.'c') THEN
     IF(ILOX.EQ.0.OR.IFUEL.EQ.0) THEN
      WRITE(*,'(A\)')'    Do you wish to modify engine file.? '
      READ(*,'(A)')ANS
      IF(ANS.EQ.'Y'.OR.ANS.EQ.'y')  THEN
       CALL MODENG(JUNIT,NAMENG)
      ELSE
       WRITE(*,'(A\)')'    Do you wish to rewind engine file.? '
       READ(*,'(A)')ANS
       IF(ANS.EQ.'Y'.OR.ANS.EQ.'y')  REWIND JUNIT
       CALL ENGNO(JUNIT)
      ENDIF
     ENDIF
     WRITE(*,'(A\)')'    Do you wish to modify CONST file.? '
     READ(*,'(A)')ANS
     IF(ANS.EQ.'Y'.OR.ANS.EQ.'y')  THEN
      CALL MODCON(12,VARI,MENG,TAUT,CSTAR,RBAR,THETAC,DCDR)
     ELSE
      WRITE(*,'(A\)')'    Do you wish to rewind CONST file.? '
      READ(*,'(A)')ANS
      IF(ANS.EQ.'Y'.OR.ANS.EQ.'y')  REWIND 12
      DO 50 I=1,MENG
       READ(12,*)TAUT(I),CSTAR(I),RBAR(I),THETAC(I),DCDR(I)
50    CONTINUE
     ENDIF
```

```
      IF(IFUEL.EQ.0)  THEN
       IGONE=1
       CALL FUEL(S,GF,11,16,IGONE)
      ENDIF
      IF(ILOX.EQ.0)  THEN
       IGONE=1
       CALL LOX(S,GOX,10,15,IGONE)
      ENDIF
      IGONE=0
      GO TO 26
     ENDIF
     IF(ANS.EQ.'N'.OR.ANS.EQ.'n') THEN
      IF(IFUEL.EQ.0)  CLOSE(11)
      IF(ILOX.EQ.0)  CLOSE(10)
      IF(IFUEL.EQ.0.OR.ILOX.EQ.0)  CLOSE(9)
      CLOSE(12)
      IFUEL=1
      ILOX=1
      REWIND 17
      GO TO 21
     ENDIF
     WRITE(*,*)' You did not enter E, F, C, or N. Try again.'
     GO TO 49
     END
     SUBROUTINE ALLPT(WHOLD,GHOLD,PTS,ITYPE,ITF,ITO,ING)
C      Supervises Nyquist plot
     INCLUDE 'FGRAPH.FD'
     RECORD/WXYCOORD/XY
     RECORD/RCCOORD/S
     INTEGER*2 DUMWIL
     COMMON /NOCOL/NCOLS,NMODE
     REAL WHOLD(1001),GHOLD(1001)
     REAL*8 RMIN,RMAX,IMMIN,IMAX
     REAL*8 X,Y
     INTEGER PTS
     CHARACTER*38 ENGINK
   1 FORMAT(15X,'      ',17X)
   2 FORMAT(7X,'Eng. #',I2,3X,'LOX TANK #',I2,8X)
   3 FORMAT(7X,'Eng. #',I2,3X,'FUEL TANK #',I2,7X)
   4 FORMAT('Eng. #',I2,3X,'FUEL TANK #',I2,2X,'LOX TANK #',I2)
     CALL SETPLT
     RMAX=WHOLD(1)
     RMIN=WHOLD(1)
     IMAX=GHOLD(1)
     IMMIN=GHOLD(1)
     DO 21 I=2,PTS
      IF(WHOLD(I).GT.RMAX)  RMAX=WHOLD(I)
      IF(WHOLD(I).LT.RMIN)  RMIN=WHOLD(I)
      IF(GHOLD(I).GT.IMAX)  IMAX=GHOLD(I)
      IF(GHOLD(I).LT.IMMIN)  IMMIN=GHOLD(I)
  21 CONTINUE
     CALL LOWERW(RMIN,RMAX,IMAX,IMMIN)
     CALL NICEGRF(RMIN,RMAX,IMAX,IMMIN,ITYPE)
```

```
      IF(ITYPE.EQ.1)  WRITE(ENGINK,1)
      IF(ITYPE.EQ.2)  WRITE(ENGINK,2)ING,ITO
      IF(ITYPE.EQ.3)  WRITE(ENGINK,3)ING,ITF
      IF(ITYPE.EQ.4)  WRITE(ENGINK,4)ING,ITF,ITO
      IF(NMODE.EQ.6)  THEN
       CALL settextposition( 2, 1, s)
       CALL OUTTEXT(ENGINK)
      ELSE
       CALL settextposition( 2, 20, s)
       CALL OUTTEXT(ENGINK)
      ENDIF
      CALL SETLINESTYLE(62268)
      X=0.0
      Y=IMMIN
      CALL MOVETO_W(X,Y,XY)
      Y=IMAX
      DUMWIL=LINETO_W(X,Y)
      Y=0.0
      X=RMIN
      CALL MOVETO_W(X,Y,XY)
      X=RMAX
      DUMWIL=LINETO_W(X,Y)
      CALL SETLINESTYLE(65535)
      X=WHOLD(1)
      Y=GHOLD(1)
      CALL MOVETO_W(X,Y,XY)
      DO 22 I=2,PTS
       X=WHOLD(I)
       Y=GHOLD(I)
       DUMWIL=LINETO_W(X,Y)
   22 CONTINUE
      CALL ENDPLT
      RETURN
      END
      SUBROUTINE CURV(A1,A2)
C       Draws circular arc
      INCLUDE 'FGRAPH.FD'
      RECORD/WXYCOORD/XY
      INTEGER*2 DUMWIL
      COMMON /ARCCON/XC,YC,RAD,ANG,ANGLE
      REAL*8 XP,YP,A1,A2
      ANG1=A1
      ANG2=A2
      DTH=ANG2-ANG1
      IF(DTH.LT.0.0)  DTH=6.283185+DTH
      N=57.29578*DTH
      DA=DTH/(N-1)
      XP=XC+RAD*SIN(ANG1)
      YP=YC-RAD*COS(ANG1)
      CALL MOVETO_W(XP,YP,XY)
      DO 21 I=1,N-1
      T=ANG1+I*DA
      XP=XC+RAD*SIN(T)
```

```
      YP=YC-RAD*COS(T)
      DUMWIL=LINETO_W(XP,YP)
   21 CONTINUE
      RETURN
      END
      SUBROUTINE ENDPLT
C       Closes plot routines
      INCLUDE  'FGRAPH.FD'
      INTEGER*2             dummy
      READ (*,*)          ! Wait for ENTER key to be pressed
      dummy = setvideomode( $DEFAULTMODE )
      RETURN
      END
      SUBROUTINE FLPLOT(ILOX)
C       Supervises plot of piping
      COMMON /EPARAM/MENG,TFLOW(25),PCHMB(25),DPROR(25),PMRAT(25)
      INTEGER SEGMNF(25),SECTNF(75,25),NOLINF(25),IENGF(25),ITANKF(25),
     *        LOPOLF(25),LOPENF(25)
      REAL KMANF(25),KTANKF(25),LFLOWF(25),LF(75,25)
      COMMON /FPARAM/MLINEF,SPLITF(25),AF(25),CMANF(25),CTANKF(25),
     *               DENSF(25),KMANF,KTANKF,LFLOWF,VOLF(25),VOLMFF(25),
     *               AREAF(75,25),DIAF(75,25),LF,PINDF(75,25),
     *               PCAPF(75,25),AVGKF(25),
     *               SEGMNF,SECTNF,NOLINF,IENGF,ITANKF,LOPOLF,LOPENF
      COMMON /FOPIPE/PIPE1F(75,25),PIPE2F(75,25),PIPE3F(75,25),
     *              PIPE4F(75,25),PIPE5F(75,25)
      INTEGER SEGMNO(25),SECTNO(75,25),NOLINO(25),IENGO(25),ITANKO(25),
     *        LOPOLO(25),LOPENO(25)
      REAL KMANO(25),KTANKO(25),LFLOWO(25),LO(75,25)
      COMMON /OPARAM/MLINEO,SPLITO(25),AO(25),CMANO(25),CTANKO(25),
     *               DENSO(25),KMANO,KTANKO,LFLOWO,VOLO(25),VOLMFO(25),
     *               AREAO(75,25),DIAO(75,25),LO,PINDO(75,25),
     *               PCAPO(75,25),AVGKO(25),
     *               SEGMNO,SECTNO,NOLINO,IENGO,ITANKO,LOPOLO,LOPENO
      INTEGER SEGMN,SECTN(150)
      COMMON /SETUP/PIPE1(150),PIPE2(150),PIPE3(150),PIPE4(150),
     *              NENGF(25),NTANKF(25),NLINEF(25),NSPF(25),ILINEF,
     *              NENGO(25),NTANKO(25),NLINEO(25),NSPO(25),ILINEO,
     *              SEGMN,SECTN
      ILINEF=0
      IPF=1
      DO 22 I=1,MLINEF
       IF(SPLITF(I).EQ.0.0)  THEN
        ILINEF=ILINEF+1
        NENGF(ILINEF)=IENGF(IPF)
        NTANKF(ILINEF)=ITANKF(I)
        NLINEF(ILINEF)=IPF
        NSPF(ILINEF)=IPF
       ELSE
        DO 21 J=IPF+1,IPF+SPLITF(I)
         ILINEF=ILINEF+1
         NENGF(ILINEF)=IENGF(J)
         NTANKF(ILINEF)=ITANKF(I)
```

```
      NLINEF(ILINEF)=IPF
      NSPF(ILINEF)=J
21    CONTINUE
     ENDIF
     IPF=IPF+SPLITF(I)+1
22  CONTINUE
    ILINEO=0
    IPO=1
    DO 24 I=1,MLINEO
     IF(SPLITO(I).EQ.0.0)  THEN
      ILINEO=ILINEO+1
      NENGO(ILINEO)=IENGO(IPO)
      NTANKO(ILINEO)=ITANKO(I)
      NLINEO(ILINEO)=IPO
      NSPO(ILINEO)=IPO
     ELSE
      DO 23 J=IPO+1,IPO+SPLITO(I)
       ILINEO=ILINEO+1
       NENGO(ILINEO)=IENGO(J)
       NTANKO(ILINEO)=ITANKO(I)
       NLINEO(ILINEO)=IPO
       NSPO(ILINEO)=J
23     CONTINUE
     ENDIF
     IPO=IPO+SPLITO(I)+1
24  CONTINUE
25  CONTINUE
    IF(ILINEF.EQ.1)  THEN
     IPLOTF=1
    ELSE
     WRITE(*,*)' The following FUEL lines may be plotted'
     WRITE(*,'(/'' Line #    Tank #   Engine #''/)')
     DO 26 I=1,ILINEF
      WRITE(*,'(I5,I10,I11)')I,NTANKF(I),NENGF(I)
26   CONTINUE
27   CONTINUE
     WRITE(*,'(/'' Enter line # to be plotted, 0 will end plot ''\)')
     READ(*,*)IPLOTF
     IF(IPLOTF.LE.0)  RETURN
     IF(IPLOTF.GT.ILINEF)  THEN
      WRITE(*,*)'  You did not enter a valid line #. Try again'
      GO TO 27
     ENDIF
    ENDIF
    K=NENGF(IPLOTF)
    IF(ILINEO.EQ.1)  THEN
     IPLOTO=1
    ELSE
     WRITE(*,*)' The following LOX lines may be plotted'
     WRITE(*,'(/'' Line #    Tank #   Engine #''/)')
     NOXY=0
     DO 28 I=1,ILINEO
      IF(NENGO(I).NE.K)  GO TO 28
```

```
      WRITE(*,'(I5,I10,I11)')I,NTANKO(I),NENGO(I)
      NOXY=NOXY+1
      NOXYP=I
   28 CONTINUE
      IF(NOXY.EQ.1)  THEN
       IPLOTO=NOXYP
      ELSE
   29  CONTINUE
       WRITE(*,'(/'' Enter line # to be plotted, 0 will end plot ''\)')
       READ(*,*)IPLOTO
       IF(IPLOTO.LE.0)  RETURN
       IF(IPLOTO.GT.ILINEO.OR.NENGO(IPLOTO).NE.K)  THEN
        WRITE(*,*)'  You did not enter a valid line #. Try again'
        GO TO 29
       ENDIF
      ENDIF
     ENDIF
     CALL SETPLT
     J=NSPF(IPLOTF)
     I=NLINEF(IPLOTF)
     K=0
     SEGMN=0
     SEGMN=SEGMN+SEGMNF(I)
     REWIND 16
     READ(16)PIPE1F,PIPE2F,PIPE3F,PIPE4F,PIPE5F
     DO 30 L=1,SEGMNF(I)
      K=K+1
      SECIN(K)=SECINF(L,I)
      PIPE1(K)=PIPE1F(L,I)
      PIPE2(K)=PIPE2F(L,I)
      PIPE3(K)=PIPE3F(L,I)
      PIPE4(K)=PIPE4F(L,I)
   30 CONTINUE
     IF(I.NE.J)  THEN
      SEGMN=SEGMN+SEGMNF(J)
      DO 31 L=1,SEGMNF(J)
       K=K+1
       SECIN(K)=SECINF(L,J)
       PIPE1(K)=PIPE1F(L,J)
       PIPE2(K)=PIPE2F(L,J)
       PIPE3(K)=PIPE3F(L,J)
       PIPE4(K)=PIPE4F(L,J)
   31  CONTINUE
     ENDIF
     CALL PIPPLOT(SEGMN,SECIN,PIPE1,PIPE2,PIPE3,PIPE4,ILOX,
    *             NTANKF(IPLOTF),NENGF(IPLOTF),'A')
     J=NSPO(IPLOTO)
     I=NLINEO(IPLOTO)
     K=0
     SEGMN=0
     SEGMN=SEGMN+SEGMNO(I)
     REWIND 15
     READ(15)PIPE1F,PIPE2F,PIPE3F,PIPE4F,PIPE5F
```

```
      DO 32 L=1,SEGMNO(I)
       K=K+1
       SECIN(K)=SECINO(L,I)
       PIPE1(K)=PIPE1F(L,I)
       PIPE2(K)=PIPE2F(L,I)
       PIPE3(K)=PIPE3F(L,I)
       PIPE4(K)=PIPE4F(L,I)
   32 CONTINUE
      IF(I.NE.J)  THEN
       SEGMN=SEGMN+SEGMNO(J)
       DO 33 L=1,SEGMNO(J)
        K=K+1
        SECIN(K)=SECINO(L,J)
        PIPE1(K)=PIPE1F(L,J)
        PIPE2(K)=PIPE2F(L,J)
        PIPE3(K)=PIPE3F(L,J)
        PIPE4(K)=PIPE4F(L,J)
   33  CONTINUE
      ENDIF
      CALL PIPPLOT(SEGMN,SECIN,PIPE1,PIPE2,PIPE3,PIPE4,ILOX,
     *             NTANKO(IPLOTO),NENGO(IPLOTO),'B')
      IF(ILINEF.EQ.1.AND.ILINEO.EQ.1)  RETURN
      GO TO 25
      END
      LOGICAL FUNCTION fourcolors()
C       Determines type of graphics monitor
      INCLUDE  'FGRAPH.FD'
      INTEGER*2             dummy
      RECORD /videoconfig/ screen
      COMMON               screen
      CALL getvideoconfig( screen )
      SELECT CASE( screen.adapter )
         CASE( $OGA, $OCGA )
            dummy = setvideomode( $MRES4COLOR )
         CASE( $EGA, $OEGA )
            dummy = setvideomode( $ERESCOLOR )
         CASE( $VGA, $OVGA )
            dummy = setvideomode( $VRES16COLOR )
         CASE DEFAULT
            dummy = 0
      END SELECT
      CALL getvideoconfig( screen )
      fourcolors = .TRUE.
      IF( dummy .EQ. 0 ) fourcolors = .FALSE.
      END
      SUBROUTINE FPLOT(ILOX)
C       Determines fuel line to be plotted
      COMMON /EPARAM/MENG,TFLOW(25),PCHMB(25),DPROR(25),PMRAT(25)
      INTEGER SEGMNF(25),SECINF(75,25),NOLINF(25),IENGF(25),ITANKF(25),
     *        LOPOLF(25),LOPENF(25)
      REAL KMANF(25),KTANKF(25),LFLOWF(25),LF(75,25)
      COMMON /FPARAM/MLINEF,SPLITF(25),AF(25),CMANF(25),CTANKF(25),
     *               DENSF(25),KMANF,KTANKF,LFLOWF,VOLF(25),VOLMFF(25),
```

```
*                AREAF(75,25),DIAF(75,25),LF,PINDF(75,25),
*                PCAPF(75,25),AVGKF(25),
*                SEGMNF,SECINF,NOLINF,IENGF,ITANKF,LOPOLF,LOPENF
 COMMON /FOPIPE/PIPE1F(75,25),PIPE2F(75,25),PIPE3F(75,25),
*               PIPE4F(75,25),PIPE5F(75,25)
 INTEGER SEGMN,SECIN(150)
 COMMON /SETUP/PIPE1(150),PIPE2(150),PIPE3(150),PIPE4(150),
*               NENGF(25),NTANKF(25),NLINEF(25),NSPF(25),ILINEF,
*               NENGO(25),NTANKO(25),NLINEO(25),NSPO(25),ILINEO,
*               SEGMN,SECIN
 ILINEF=0
 IPF=1
 DO 22 I=1,MLINEF
  IF(SPLITF(I).EQ.0.0)  THEN
   ILINEF=ILINEF+1
   NENGF(ILINEF)=IENGF(IPF)
   NTANKF(ILINEF)=ITANKF(I)
   NLINEF(ILINEF)=IPF
   NSPF(ILINEF)=IPF
  ELSE
   DO 21 J=IPF+1,IPF+SPLITF(I)
    ILINEF=ILINEF+1
    NENGF(ILINEF)=IENGF(J)
    NTANKF(ILINEF)=ITANKF(I)
    NLINEF(ILINEF)=IPF
    NSPF(ILINEF)=J
21  CONTINUE
  ENDIF
  IPF=IPF+SPLITF(I)+1
22 CONTINUE
23 CONTINUE
  IF(ILINEF.EQ.1)  THEN
   IPLOTF=1
  ELSE
   WRITE(*,*)' The following FUEL lines may be plotted'
   WRITE(*,'(/'' Line #    Tank #   Engine #''/)')
   DO 24 I=1,ILINEF
    WRITE(*,'(I5,I10,I11)')I,NTANKF(I),NENGF(I)
24 CONTINUE
25 CONTINUE
   WRITE(*,'(/'' Enter line # to be plotted, 0 will end plot ''\)')
   READ(*,*)IPLOTF
   IF(IPLOTF.LE.0)  RETURN
   IF(IPLOTF.GT.ILINEF)  THEN
    WRITE(*,*)'  You did not enter a valid line #. Try again'
    GO TO 25
   ENDIF
  ENDIF
  CALL SETPLT
  J=NSPF(IPLOTF)
  I=NLINEF(IPLOTF)
  K=0
  SEGMN=0
```

```
      SEGMN=SEGMN+SEGMNF(I)
      REWIND 16
      READ(16)PIPE1F,PIPE2F,PIPE3F,PIPE4F,PIPE5F
      DO 26 L=1,SEGMNF(I)
       K=K+1
       SECIN(K)=SECINF(L,I)
       PIPE1(K)=PIPE1F(L,I)
       PIPE2(K)=PIPE2F(L,I)
       PIPE3(K)=PIPE3F(L,I)
       PIPE4(K)=PIPE4F(L,I)
   26 CONTINUE
      IF(I.NE.J)  THEN
       SEGMN=SEGMN+SEGMNF(J)
       DO 27 L=1,SEGMNF(J)
        K=K+1
        SECIN(K)=SECINF(L,J)
        PIPE1(K)=PIPE1F(L,J)
        PIPE2(K)=PIPE2F(L,J)
        PIPE3(K)=PIPE3F(L,J)
        PIPE4(K)=PIPE4F(L,J)
   27  CONTINUE
      ENDIF
      CALL PIPPLOT(SEGMN,SECIN,PIPE1,PIPE2,PIPE3,PIPE4,ILOX,
     *             NTANKF(IPLOTF),NENGF(IPLOTF),'A')
      IF(ILINEF.EQ.1)  RETURN
      GO TO 23
      END
      SUBROUTINE LABANG(XMIN,XMAX,YMIN,YMAX)
C       Labels phase angle plot
      INCLUDE  'FGRAPH.FD'
      RECORD/WXYCOORD/XY
      RECORD /videoconfig/ screen
      COMMON               screen
      CHARACTER*40 TITLE
      CHARACTER*20 TITLF
      INTEGER*2 IHR,IMIN,IYR,IMON,IDAY
      CHARACTER*2 AP
      COMMON /WCATIT/TITLE,TITLF,IHR,IMIN,AP,IYR,IMON,IDAY
      COMMON /NOCOL/NCOLS,NMODE
      COMMON /FACTOR/SFAC
      INTEGER*2 NCOLS
      INTEGER*2 row,rows
      INTEGER*2 DUMWIL
      RECORD/RCCOORD/S
      REAL*8 XMIN, XMAX, YMIN, YMAX, XP, YP
      CHARACTER*6 YLO,YHI
      CHARACTER*7 XHI
      DATA YLO/' -180°'/
      DATA YHI/'  180°'/
    1 FORMAT(F7.1)
    2 FORMAT(F7.2)
    3 FORMAT(F7.3)
    4 FORMAT(F7.4)
```

```
5 FORMAT(F7.5)
6 FORMAT(F7.6)
  rows     = screen.numtextrows
  IF(NMODE.EQ.6)  THEN
   CALL settextposition( 1, 1, s)
  ELSE
   CALL settextposition( 0, 20, s)
  ENDIF
  CALL OUTTEXT(TITLE)
  dummy = rectangle_w( $GBORDER, XMIN, YMIN, XMAX, YMAX )
  row=rows/4
  CALL SETTEXTPOSITION(row,1,s)
  IF(NCOLS.LE.40)  THEN
   CALL OUTTEXT('Angle')
  ELSE
   CALL OUTTEXT(' Phase Angle')
  ENDIF
  IF(NMODE.EQ.6)  THEN
   CALL SETTEXTPOSITION(rows/2-1,18,s)
   CALL OUTTEXT('freq')
  ELSE
   CALL SETTEXTPOSITION(rows/2-1,35,s)
   IF(SFAC.EQ.1.0)  THEN
    CALL OUTTEXT('Frequency - rad/sec')
   ELSE
    CALL OUTTEXT('Frequency -  Hertz ')
   ENDIF
  ENDIF
  CALL GETTEXTPOSITION(s)
  IF(NMODE.EQ.6)  THEN
   CALL SETTEXTPOSITION(3,1,s)
   CALL OUTTEXT(YHI)
   CALL SETTEXTPOSITION(s.row-3,1,s)
   CALL OUTTEXT(YLO)
   CALL GETTEXTPOSITION(s)
   ILOC=4
   IMAX=26
  ELSEIF(NMODE.EQ.16)  THEN
   CALL SETTEXTPOSITION(2,10,s)
   CALL OUTTEXT(YHI)
   CALL SETTEXTPOSITION(s.row-2,10,s)
   CALL OUTTEXT(YLO)
   CALL GETTEXTPOSITION(s)
   ILOC=13
   IMAX=54
  ELSE
   CALL SETTEXTPOSITION(2,10,s)
   CALL OUTTEXT(YHI)
   CALL SETTEXTPOSITION(s.row-2,10,s)
   CALL OUTTEXT(YLO)
   CALL GETTEXTPOSITION(s)
   ILOC=13
   IMAX=54
```

```
      ENDIF
      ILO=XMIN
      IHI=XMAX
      IDEL=IMAX/(IHI-ILO)
      row=s.row+1
      DO 21 I=ILO,IHI
       HI=10.0**I
       IF(HI.GE.0.1)  THEN
        WRITE(XHI,1)HI
       ELSEIF(HI.GE.0.01)  THEN
        WRITE(XHI,2)HI
       ELSEIF(HI.GE.0.001)  THEN
        WRITE(XHI,3)HI
       ELSEIF(HI.GE.0.0001)  THEN
        WRITE(XHI,4)HI
       ELSEIF(HI.GE.0.00001)  THEN
        WRITE(XHI,5)HI
       ELSE
        WRITE(XHI,6)HI
       ENDIF
       CALL SETTEXTPOSITION(row,ILOC,s)
       CALL OUTTEXT(XHI)
       ILOC=ILOC+IDEL
       IF(I.EQ.ILO.OR.I.EQ.IHI)  GO TO 21
       CALL SETLINESTYLE(62268)
       XP=I
       YP=YMIN
       CALL MOVETO_W(XP,YP,XY)
       YP=YMAX
       DUMWIL=LINETO_W(XP,YP)
       CALL SETLINESTYLE(65535)
   21 CONTINUE
      RETURN
      END
      SUBROUTINE LABGAIN(XMIN,XMAX,YMIN,YMAX,ITYPE)
C       Labels gain plot
      INCLUDE  'FGRAPH.FD'
      RECORD/WXYCOORD/XY
      RECORD /videoconfig/ screen
      COMMON               screen
      CHARACTER*40 TITLE
      CHARACTER*20 TITLF
      INTEGER*2 IHR,IMIN,IYR,IMON,IDAY
      CHARACTER*2 AP
      COMMON /WCATIT/TITLE,TITLF,IHR,IMIN,AP,IYR,IMON,IDAY
      COMMON /NOCOL/NCOLS,NMODE
      COMMON /FACTOR/SFAC
      INTEGER*2 NCOLS
      INTEGER*2 row,rows
      INTEGER*2 DUMWIL
      RECORD/RCCOORD/S
      REAL*8 XMIN, XMAX, YMIN, YMAX, XP, YP
      CHARACTER*6 YLO,YHI
```

```
  CHARACTER*7 XHI
1 FORMAT(F7.1)
2 FORMAT(F7.2)
3 FORMAT(F7.3)
4 FORMAT(F7.4)
5 FORMAT(F7.5)
6 FORMAT(F7.6)
7 FORMAT(F6.3)
  rows    = screen.numtextrows
  dummy = rectangle_w( $GBORDER, XMIN, YMIN, XMAX, YMAX )
  row=rows/4
  CALL SETTEXTPOSITION(row,5,s)
  CALL OUTTEXT('Gain ')
  IF(NMODE.EQ.6)  THEN
   CALL SETTEXTPOSITION(rows/2-1,18,s)
   CALL OUTTEXT('freq')
   CALL SETTEXTPOSITION(rows,16,s)
  ELSE
   CALL SETTEXTPOSITION(rows/2-1,35,s)
   IF(SFAC.EQ.1.0)  THEN
    CALL OUTTEXT('Frequency - rad/sec')
   ELSE
    CALL OUTTEXT('Frequency -  Hertz ')
   ENDIF
   CALL SETTEXTPOSITION(rows,39,s)
  ENDIF
  IF(ITYPE.EQ.1)  CALL OUTTEXT('    K(jw)    ')
  IF(ITYPE.EQ.2)  CALL OUTTEXT(' K(jw,Gox)   ')
  IF(ITYPE.EQ.3)  CALL OUTTEXT('  K(jw,Gf)   ')
  IF(ITYPE.EQ.4)  CALL OUTTEXT('K(jw,Gox,Gf)')
  WRITE(YLO,7)YMIN
  WRITE(YHI,7)YMAX
  CALL GETTEXTPOSITION(s)
  IF(NMODE.EQ.6)  THEN
   CALL SETTEXTPOSITION(3,1,s)
   CALL OUTTEXT(YHI)
   CALL SETTEXTPOSITION(s.row-3,1,s)
   CALL OUTTEXT(YLO)
   CALL GETTEXTPOSITION(s)
   ILOC=4
   IMAX=26
  ELSEIF(NMODE.EQ.16)  THEN
   CALL SETTEXTPOSITION(3,10,s)
   CALL OUTTEXT(YHI)
   CALL SETTEXTPOSITION(s.row-4,10,s)
   CALL OUTTEXT(YLO)
   CALL GETTEXTPOSITION(s)
   ILOC=13
   IMAX=54
  ELSE
   CALL SETTEXTPOSITION(2,10,s)
   CALL OUTTEXT(YHI)
   CALL SETTEXTPOSITION(s.row-3,10,s)
```

```
      CALL OUTTEXT(YLO)
      CALL GETTEXTPOSITION(s)
      ILOC=13
      IMAX=54
     ENDIF
     ILO=XMIN
     IHI=XMAX
     IDEL=IMAX/(IHI-ILO)
     row=s.row+1
     DO 21 I=ILO,IHI
      HI=10.0**I
      IF(HI.GE.0.1)  THEN
       WRITE(XHI,1)HI
      ELSEIF(HI.GE.0.01)  THEN
       WRITE(XHI,2)HI
      ELSEIF(HI.GE.0.001)  THEN
       WRITE(XHI,3)HI
      ELSEIF(HI.GE.0.0001)  THEN
       WRITE(XHI,4)HI
      ELSEIF(HI.GE.0.00001)  THEN
       WRITE(XHI,5)HI
      ELSE
       WRITE(XHI,6)HI
      ENDIF
      CALL SETTEXTPOSITION(row,ILOC,s)
      CALL OUTTEXT(XHI)
      ILOC=ILOC+IDEL
      IF(I.EQ.ILO.OR.I.EQ.IHI)  GO TO 21
      CALL SETLINESTYLE(62268)
      XP=I
      YP=YMIN
      CALL MOVETO_W(XP,YP,XY)
      YP=YMAX
      DUMWIL=LINETO_W(XP,YP)
      CALL SETLINESTYLE(65535)
   21 CONTINUE
     RETURN
     END
     SUBROUTINE LOWERW(XMIN,XMAX,YMAX,YMIN)
C      Sets up lower plotting window
     INCLUDE  'FGRAPH.FD'
     INTEGER*2             dummy
     INTEGER*2             xwidth, yheight, cols, rows
     RECORD /videoconfig/ screen
     COMMON                screen
     COMMON /NOCOL/NCOLS,NMODE
     INTEGER*2 NCOLS,NMODE
     REAL*8 XMIN, XMAX, YMIN, YMAX, XLEN, YLEN
     XLEN=0.1*(XMAX-XMIN)
     YLEN=0.1*(YMAX-YMIN)
     XMIN=XMIN-XLEN
     XMAX=XMAX+XLEN
     YMIN=YMIN-YLEN
```

```
      YMAX=YMAX+YLEN
      xwidth  = screen.numxpixels
      yheight = screen.numypixels
      cols    = screen.numtextcols
      rows    = screen.numtextrows
      IF(NMODE.EQ.6)  THEN
       CALL setviewport( 50, yheight - 30, xwidth - 20, 10 )
      ELSE
       CALL setviewport( 100, yheight - 50, xwidth - 50, 40 )
      ENDIF
       CALL settextwindow(  0, 1, rows, cols)
      dummy = setwindow(.TRUE.,XMIN,YMIN,XMAX,YMAX)
      CALL clearscreen( $GWINDOW )
      RETURN
      END
      SUBROUTINE LPLOT(ILOX)
C       Determines lox line to be plotted
      COMMON /EPARAM/MENG,TFLOW(25),PCHMB(25),DPROR(25),PMRAT(25)
      INTEGER SEGMNO(25),SECTNO(75,25),NOLINO(25),IENGO(25),ITANKO(25),
     *        LOPOLO(25),LOPENO(25)
      REAL KMANO(25),KTANKO(25),LFLOWO(25),LO(75,25)
      COMMON /OPARAM/MLINEO,SPLITO(25),AO(25),CMANO(25),CTANKO(25),
     *               DENSO(25),KMANO,KTANKO,LFLOWO,VOLO(25),VOLMFO(25),
     *               AREAO(75,25),DIAO(75,25),LO,PINDO(75,25),
     *               PCAPO(75,25),AVGKO(25),
     *               SEGMNO,SECTNO,NOLINO,IENGO,ITANKO,LOPOLO,LOPENO
      COMMON /FOPIPE/PIPE1O(75,25),PIPE2O(75,25),PIPE3O(75,25),
     *              PIPE4O(75,25),PIPE5O(75,25)
      INTEGER SEGMN,SECTN(150)
      COMMON /SETUP/PIPE1(150),PIPE2(150),PIPE3(150),PIPE4(150),
     *              NENGF(25),NTANKF(25),NLINEF(25),NSPF(25),ILINEF,
     *              NENGO(25),NTANKO(25),NLINEO(25),NSPO(25),ILINEO,
     *              SEGMN,SECTN
      ILINEO=0
      IPO=1
      DO 22 I=1,MLINEO
       IF(SPLITO(I).EQ.0.0)  THEN
        ILINEO=ILINEO+1
        NENGO(ILINEO)=IENGO(IPO)
        NTANKO(ILINEO)=ITANKO(I)
        NLINEO(ILINEO)=IPO
        NSPO(ILINEO)=IPO
       ELSE
        DO 21 J=IPO+1,IPO+SPLITO(I)
         ILINEO=ILINEO+1
         NENGO(ILINEO)=IENGO(J)
         NTANKO(ILINEO)=ITANKO(I)
         NLINEO(ILINEO)=IPO
         NSPO(ILINEO)=J
   21   CONTINUE
       ENDIF
       IPO=IPO+SPLITO(I)+1
   22 CONTINUE
```

```
 23 CONTINUE
    IF(ILINEO.EQ.1)  THEN
     IPLOTO=1
    ELSE
     WRITE(*,*)' The following LOX lines may be plotted'
     WRITE(*,'(/'' Line #    Tank #   Engine #''/)')
     DO 24 I=1,ILINEO
      WRITE(*,'(I5,I10,I11)')I,NTANKO(I),NENGO(I)
 24  CONTINUE
 25  CONTINUE
     WRITE(*,'(/'' Enter line # to be plotted, 0 will end plot ''\)')
     READ(*,*)IPLOTO
     IF(IPLOTO.LE.0)  RETURN
     IF(IPLOTO.GT.ILINEO)  THEN
      WRITE(*,*)'  You did not enter a valid line #. Try again'
      GO TO 25
     ENDIF
    ENDIF
    CALL SETPLT
    J=NSPO(IPLOTO)
    I=NLINEO(IPLOTO)
    K=0
    SEGMN=0
    SEGMN=SEGMN+SEGMNO(I)
    REWIND 15
    READ(15)PIPE1O,PIPE2O,PIPE3O,PIPE4O,PIPE5O
    DO 26 L=1,SEGMNO(I)
     K=K+1
     SECIN(K)=SECINO(L,I)
     PIPE1(K)=PIPE1O(L,I)
     PIPE2(K)=PIPE2O(L,I)
     PIPE3(K)=PIPE3O(L,I)
     PIPE4(K)=PIPE4O(L,I)
 26 CONTINUE
    IF(I.NE.J)  THEN
     SEGMN=SEGMN+SEGMNO(J)
     DO 27 L=1,SEGMNO(J)
      K=K+1
      SECIN(K)=SECINO(L,J)
      PIPE1(K)=PIPE1O(L,J)
      PIPE2(K)=PIPE2O(L,J)
      PIPE3(K)=PIPE3O(L,J)
      PIPE4(K)=PIPE4O(L,J)
 27  CONTINUE
    ENDIF
    CALL PIPPLOT(SEGMN,SECIN,PIPE1,PIPE2,PIPE3,PIPE4,ILOX,
   *             NTANKO(IPLOTO),NENGO(IPLOTO),'B')
    IF(ILINEO.EQ.1)  RETURN
    GO TO 23
    END
    SUBROUTINE NICEGRF(RMIN,RMAX,IMAX,IMMIN,ITYPE)
C     Plots Nyquist curve
    INCLUDE  'FGRAPH.FD'
```

```
      RECORD /videoconfig/ screen
      COMMON               screen
      CHARACTER*40 TITLE
      CHARACTER*20 TITLF
      INTEGER*2 IHR,IMIN,IYR,IMON,IDAY
      CHARACTER*2 AP
      COMMON /WCATIT/TITLE,TITLF,IHR,IMIN,AP,IYR,IMON,IDAY
      COMMON /NOCOL/NCOLS,NMODE
      COMMON /FACTOR/SFAC
      INTEGER*2 NCOLS,NMODE
      INTEGER*2 row,rows
      RECORD/RCCOORD/S
      REAL*8 IMMIN,IMAX,RMIN,RMAX
      REAL*8 XMIN, XMAX, YMIN, YMAX
      CHARACTER*6 YLO,YHI,XLO,XHI
    1 FORMAT(F6.3)
      rows    = screen.numtextrows
      XMIN=RMIN
      XMAX=RMAX
      YMIN=IMMIN
      YMAX=IMAX
      IF(NMODE.EQ.6)  THEN
       CALL settextposition( 0, 1, s)
       CALL OUTTEXT(TITLE)
      ELSE
       CALL settextposition( 0, 20, s)
       CALL OUTTEXT(TITLE)
      ENDIF
      dummy = rectangle_w( $GBORDER, XMIN, YMIN, XMAX, YMAX )
      row=rows/2
      CALL SETTEXTPOSITION(row,1,s)
      IF(NMODE.EQ.6)  THEN
       CALL OUTTEXT('Imag')
       CALL SETTEXTPOSITION(rows-1,16,s)
       CALL OUTTEXT('    Real')
       CALL SETTEXTPOSITION(rows,16,s)
      ELSE
       CALL OUTTEXT('Imaginary')
       CALL SETTEXTPOSITION(rows-1,39,s)
       CALL OUTTEXT('    Real')
       CALL SETTEXTPOSITION(rows,39,s)
      ENDIF
      IF(ITYPE.EQ.1)  CALL OUTTEXT('    K(jw)   ')
      IF(ITYPE.EQ.2)  CALL OUTTEXT(' K(jw,Gox)  ')
      IF(ITYPE.EQ.3)  CALL OUTTEXT('  K(jw,Gf)  ')
      IF(ITYPE.EQ.4)  CALL OUTTEXT('K(jw,Gox,Gf)')
      WRITE(YLO,1)YMIN
      WRITE(YHI,1)YMAX
      WRITE(XLO,1)XMIN
      WRITE(XHI,1)XMAX
      CALL GETTEXTPOSITION(s)
      IF(NMODE.EQ.6)  THEN
       CALL SETTEXTPOSITION(s.row-3,1,s)
```

```
      CALL OUTTEXT(YLO)
      CALL GETTEXTPOSITION(s)
      CALL SETTEXTPOSITION(s.row+1,4,s)
      CALL OUTTEXT(XLO)
      CALL GETTEXTPOSITION(s)
      CALL SETTEXTPOSITION(s.row,35,s)
      CALL OUTTEXT(XHI)
      CALL SETTEXTPOSITION(4,1,s)
      CALL OUTTEXT(YHI)
     ELSE
      CALL SETTEXTPOSITION(s.row-3,5,s)
      CALL OUTTEXT(YLO)
      CALL GETTEXTPOSITION(s)
      CALL SETTEXTPOSITION(s.row+1,9,s)
      CALL OUTTEXT(XLO)
      CALL GETTEXTPOSITION(s)
      CALL SETTEXTPOSITION(s.row,71,s)
      CALL OUTTEXT(XHI)
      CALL SETTEXTPOSITION(3,5,s)
      CALL OUTTEXT(YHI)
     ENDIF
     RETURN
     END
     SUBROUTINE PIPPLOT(SEGMN,SECIN,PIPE1,PIPE2,PIPE3,PIPE4,ILOX,
    *                   ITANK,IENG,R)
C      Supervises plot of piping layout
     INCLUDE 'FGRAPH.FD'
     RECORD/WXYCOORD/XY
     INTEGER*2 DUMWIL
     COMMON /ARCCON/XC,YC,RAD,ANG,ANGLE
     COMMON /PIPPXY/X,XH,XL,Y,YH,YL,XMIN,XMAX,YMIN,YMAX,SINA,COSA
     INTEGER*2 SEGMN,SECIN(75),ITYPE(200)
     REAL PIPE1(75),PIPE2(75),PIPE3(75),PIPE4(75)
     REAL*8 X0,X1,X2,X3,Y0,Y1,Y2,Y3
     COMMON /WORK2/ POINT(8,200),DUMMY3(175),ITYPE
     CHARACTER*1 R
     ANG=0.0
     ANGLE=0.0
     COSA=1.0
     SINA=0.0
     X=0.0
     XH=0.0
     XL=0.0
     Y=0.0
     IF(SECIN(1).EQ.0)  THEN
      YH=Y+0.5*PIPE3(1)
      YL=Y-0.5*PIPE3(1)
     ELSEIF(SECIN(1).GE.3.AND.SECIN(1).LE.5)  THEN
      IF(SECIN(2).EQ.0)  THEN
       YH=Y+0.5*PIPE3(2)
       YL=Y-0.5*PIPE3(2)
      ELSE
       YH=Y+0.5*PIPE2(2)
```

```
        YL=Y-0.5*PIPE2(2)
       ENDIF
      ELSE
       YH=Y+0.5*PIPE2(1)
       YL=Y-0.5*PIPE2(1)
      ENDIF
      J=0
      XMIN=0.0
      XMAX=0.0
      YMIN=AMIN1(Y,YL,YH)
      YMAX=AMAX1(Y,YL,YH)
      DO 21 I=1,SEGMN
       IF(SECIN(I).EQ.0)  THEN
C         bend
        CALL BNSECT(J,ITYPE,POINT,PIPE1(I),PIPE2(I),PIPE3(I),PIPE4(I))
       ELSEIF(SECIN(I).EQ.1)  THEN
C         straight section
        CALL STSECT(J,ITYPE,POINT,PIPE1(I),PIPE2(I))
       ELSEIF(SECIN(I).EQ.2)  THEN
C         inline accumulator
        CALL STSECT(J,ITYPE,POINT,PIPE1(I),PIPE2(I))
       ELSEIF(SECIN(I).EQ.3)  THEN
C         tuned stub accumulator
        CALL TSSECT(J,ITYPE,POINT,PIPE1(I),PIPE2(I))
       ELSEIF(SECIN(I).EQ.4)  THEN
C         helmholtz resonator
        CALL HHSECT(J,ITYPE,POINT,PIPE1(I),PIPE2(I),PIPE3(I))
       ELSEIF(SECIN(I).EQ.5)  THEN
C         parallel resonator
        CALL PLSECT(J,ITYPE,POINT,PIPE1(I),PIPE2(I),PIPE3(I))
       ELSEIF(SECIN(I).EQ.6)  THEN
C         pump
        CALL STSECT(J,ITYPE,POINT,PIPE1(I),PIPE2(I))
       ENDIF
   21 CONTINUE
      XRANGE=XMAX-XMIN
      YRANGE=YMAX-YMIN
      XMIN=XMIN-0.05*XRANGE
      XMAX=XMAX+0.05*XRANGE
      YMIN=YMIN-0.05*YRANGE
      YMAX=YMAX+0.05*YRANGE
      CALL UPPERW(XMIN,YMIN,XMAX,YMAX,ILOX,ITANK,IENG,R)
      DO 22 I=1,J
       IF(ITYPE(I).EQ.0)  THEN
C         bend
        XC=POINT(1,I)
        YC=POINT(2,I)
        X1=POINT(3,I)
        Y1=POINT(4,I)
        RAD=POINT(5,I)
        IF(X1.GT.Y1)  THEN
         X1=3.14159+X1
         Y1=3.14159+Y1
```

```
          CALL CURV(Y1,X1)
         ELSE
          CALL CURV(X1,Y1)
         ENDIF
        ELSE
C         all except bend
         X0=POINT(1,I)
         Y0=POINT(2,I)
         X1=POINT(3,I)
         Y1=POINT(4,I)
         X2=POINT(5,I)
         Y2=POINT(6,I)
         X3=POINT(7,I)
         Y3=POINT(8,I)
         CALL MOVETO_W(X0,Y0,XY)
         DUMWIL=LINETO_W(X1,Y1)
         CALL MOVETO_W(X2,Y2,XY)
         DUMWIL=LINETO_W(X3,Y3)
         CALL MOVETO_W(X0,Y0,XY)
         DUMWIL=LINETO_W(X2,Y2)
         CALL MOVETO_W(X1,Y1,XY)
         DUMWIL=LINETO_W(X3,Y3)
        ENDIF
   22 CONTINUE
        IF(R.EQ.'A')  THEN
         IF(ILOX.EQ.0)  RETURN
        ENDIF
        CALL ENDPLT
        RETURN
        END
        SUBROUTINE PLSECT(J,ITYPE,POINT,LEN,DIA,VOL)
C         Computes plot coordinates for parallel resonator
        COMMON /PIPPXY/X,XH,XL,Y,YH,YL,XMIN,XMAX,YMIN,YMAX,SINA,COSA
        COMMON /ARCCON/XC,YC,RAD,ANG,ANGLE
        REAL LEN,POINT(8,200)
        INTEGER*2 ITYPE(200)
        XOLD=X
        XHOLD=XH
        XLOLD=XL
        YOLD=Y
        YHOLD=YH
        YLOLD=YL
        ANGOLD=ANG
        ANGSAV=ANGLE
        SINOLD=SINA
        COSOLD=COSA
        DIAM=SQRT((XH-XL)**2+(YH-YL)**2)
        CALL STSECT(J,ITYPE,POINT,DIA,DIAM)
        XC=0.5*(XHOLD+XH)
        XHC=XHOLD
        XLC=XL
        YC=0.5*(YHOLD+YH)
        YHC=YHOLD
```

```
      YLC=YL
      PLEN=LEN-2.0*DIA
      PDIA=(VOL-2.0*DIA*DIAM)/PLEN
      CALL STSECT(J,ITYPE,POINT,PLEN,PDIA)
      CALL STSECT(J,ITYPE,POINT,DIA,DIAM)
      XSAV=X
      XHSAV=XH
      XLSAV=XL
      YSAV=Y
      YHSAV=YH
      YLSAV=YL
      SINA=COSOLD
      COSA=-SINOLD
      RADIUS=DIA
      TURN=-90.0
      SIDE=LEN-5.0*DIA
      ANG=ANG+1.5708
      ANGLE=ANGLE+90.0
      X=XC
      Y=YC
      XH=XHC
      XL=XLC
      YH=YHC
      YL=YLC
      CALL BNSECT(J,ITYPE,POINT,RADIUS,TURN,DIA,DIA)
      CALL STSECT(J,ITYPE,POINT,SIDE,DIA)
      CALL BNSECT(J,ITYPE,POINT,RADIUS,TURN,DIA,DIA)
      X=XSAV
      Y=YSAV
      XH=XHSAV
      XL=XLSAV
      YH=YHSAV
      YL=YLSAV
      ANG=ANGOLD
      ANGLE=ANGSAV
      SINA=SINOLD
      COSA=COSOLD
      RETURN
      END
      SUBROUTINE PNYQ(KR,KC,KW,PTS,ITYPE,ITF,ITO,ING)
C       Plots gain and phase angle
      INCLUDE 'FGRAPH.FD'
      INTEGER PTS
      REAL KR(PTS),KC(PTS),KW(PTS)
      COMMON /WORK1/DUMMY2(8397),X(1001),YR(1001),YC(1001)
      RECORD/WXYCOORD/XY
      RECORD /videoconfig/ screen
      COMMON               screen
      RECORD/RCCOORD/S
      COMMON /NOCOL/NCOLS,NMODE
      INTEGER*2 DUMWIL
      REAL*8 XMIN,XMAX,YMINR,YMAXR,YMINC,YMAXC,XP,YP,XLO,XHI
      CHARACTER*38 ENGINK
```

```
 1 FORMAT(15X,'        ',17X)
 2 FORMAT(7X,'Eng. #',I2,3X,'LOX TANK #',I2,8X)
 3 FORMAT(7X,'Eng. #',I2,3X,'FUEL TANK #',I2,7X)
 4 FORMAT('Eng. #',I2,3X,'FUEL TANK #',I2,2X,'LOX TANK #',I2)
   rows    = screen.numtextrows
   CALL SETPLT
   DO 21 I=1,PTS
    YR(I)=SQRT(KR(I)**2+KC(I)**2)
    YC(I)=57.29578*ATAN2(KC(I),KR(I))
    X(I)=ALOG10(KW(I))
21 CONTINUE
   YMINR=YR(1)
   YMAXR=YR(1)
   YMINC=-180.0
   YMAXC= 180.0
   XMIN=X(1)
   XMAX=X(1)
   DO 22 I=2,PTS
    IF(X(I).LT.XMIN)   XMIN=X(I)
    IF(X(I).GT.XMAX)   XMAX=X(I)
    IF(YR(I).LT.YMINR)   YMINR=YR(I)
    IF(YR(I).GT.YMAXR)   YMAXR=YR(I)
22 CONTINUE
   XLO=0.0
   XHI=0.0
   DO 23 I=-10,10
    IF(XMIN.GE.I)   XLO=I
    IF(XMAX.LE.I)   THEN
     XHI=I
     GO TO 24
    ENDIF
23 CONTINUE
24 CONTINUE
   IF(XLO.EQ.XHI)   XHI=XHI+1.0
   CALL WINDLO(XLO,XHI,YMINR,YMAXR)
   CALL LABGAIN(XLO,XHI,YMINR,YMAXR,ITYPE)
   IF(ITYPE.EQ.1)   WRITE(ENGINK,1)
   IF(ITYPE.EQ.2)   WRITE(ENGINK,2)ING,ITO
   IF(ITYPE.EQ.3)   WRITE(ENGINK,3)ING,ITF
   IF(ITYPE.EQ.4)   WRITE(ENGINK,4)ING,ITF,ITO
   IF(NMODE.EQ.6)   THEN
    CALL settextposition( ROWS/2+1, 1, s)
   ELSE
    CALL settextposition( ROWS/2+1, 26, s)
   ENDIF
   CALL OUTTEXT(ENGINK)
   CALL SETLINESTYLE(62268)
   IF(XMIN.LE.0.0.AND.XMAX.GE.0.0)   THEN
    XP=0.0
    YP=YMINR
    CALL MOVETO_W(XP,YP,XY)
    YP=YMAXR
    DUMWIL=LINETO_W(XP,YP)
```

```
      ENDIF
      IF(YMINR.LE.0.0.AND.YMAXR.GE.0.0)  THEN
       YP=0.0
       XP=XLO
       CALL MOVETO_W(XP,YP,XY)
       XP=XHI
       DUMWIL=LINETO_W(XP,YP)
      ENDIF
      CALL SETLINESTYLE(65535)
      XP=X(1)
      YP=YR(1)
      CALL MOVETO_W(XP,YP,XY)
      DO 25 I=2,PTS
       XP=X(I)
       YP=YR(I)
       DUMWIL=LINETO_W(XP,YP)
   25 CONTINUE
      CALL WINDUP(XLO,XHI,YMINC,YMAXC)
      CALL LABANG(XLO,XHI,YMINC,YMAXC)
      CALL SETLINESTYLE(62268)
      IF(XMIN.LE.0.0.AND.XMAX.GE.0.0)  THEN
       XP=0.0
       YP=YMINC
       CALL MOVETO_W(XP,YP,XY)
       YP=YMAXC
       DUMWIL=LINETO_W(XP,YP)
      ENDIF
      IF(YMINC.LE.0.0.AND.YMAXC.GE.0.0)  THEN
       YP=0.0
       XP=XLO
       CALL MOVETO_W(XP,YP,XY)
       XP=XHI
       DUMWIL=LINETO_W(XP,YP)
      ENDIF
      CALL SETLINESTYLE(65535)
      XP=X(1)
      YP=YC(1)
      CALL MOVETO_W(XP,YP,XY)
      DO 26 I=2,PTS
       XP=X(I)
       YP=YC(I)
       DUMWIL=LINETO_W(XP,YP)
   26 CONTINUE
      CALL ENDPLT
      RETURN
      END
      SUBROUTINE SETPLT
C       Sets up the plot environment
      INCLUDE 'FGRAPH.FD'
      RECORD /videoconfig/ screen
      COMMON               screen
      LOGICAL  fourcolors
      EXTERNAL fourcolors
```

```
      COMMON /NOCOL/NCOLS,NMODE
      INTEGER*2 NCOLS,NMODE
      IF( .NOT.fourcolors() ) THEN
       WRITE (*,*) ' This program requires a CGA, EGA, or',
     +             ' VGA graphics card.'
       STOP
      ENDIF
      NCOLS    = screen.numtextcols
      NMODE    = screen.mode
      RETURN
      END
      SUBROUTINE UPPERW(X00,Y00,X11,Y11,ILOX,ITANK,IENG,R)
C       Sets up upper plotting window
      INCLUDE  'FGRAPH.FD'
      RECORD/RCCOORD/S
      INTEGER*2             dummy
      INTEGER*2             xwidth, yheight, cols, rows
      RECORD /videoconfig/ screen
      COMMON                screen
      COMMON /NOCOL/NCOLS,NMODE
      INTEGER*2 NCOLS,NMODE
      CHARACTER*2 AP
      CHARACTER*40 TITLE
      CHARACTER*20 TITLF
      CHARACTER*36 FULOX
      COMMON /WCATIT/TITLE,TITLF,IHR,IMIN,AP,IYR,IMON,IDAY
      REAL*8 X0, X1, Y0, Y1
      CHARACTER*1 R
    1 FORMAT('FUEL Piping - Tank # ',I2,'  Engine # ',I2)
    2 FORMAT(' LOX Piping - Tank # ',I2,'  Engine # ',I2)
      xwidth  = screen.numxpixels
      yheight = screen.numypixels
      cols    = screen.numtextcols
      rows    = screen.numtextrows
      halfy   = yheight/2
      X0=X00
      Y0=Y00
      X1=X11
      Y1=Y11
      PICX=XWIDTH-20
      PICY=HALFY-30
      IF(NCOLS.LE.40)  PICY=HALFY-20
      XRANG=DABS(X1-X0)
      YRANG=DABS(Y1-Y0)
      XRAT=PICX/XRANG
      YRAT=PICY/YRANG
      IF(XRAT.LT.YRAT)  THEN
       YRAT=PICY/XRAT
       ADDY=0.5*(YRAT-YRANG)
       Y0=Y0-ADDY
       Y1=Y1+ADDY
      ELSE
       XRAT=PICX/YRAT
```

```
 ADDX=0.5*(XRAT-XRANG)
 X0=X0-ADDX
 X1=X1+ADDX
ENDIF
IF(R .EQ. 'A') THEN
 IF(NMODE.EQ.6)  THEN
  CALL setviewport( 10, halfy + 10, xwidth - 10, yheight - 10 )
  dummy = setwindow( .TRUE., X0-1.0, Y0-1.0, X1+1.0, Y1+1.0 )
  CALL settextwindow( (rows / 2 ) + 1, 1, rows, cols)
 ELSE
  CALL setviewport( 10, halfy + 10, xwidth - 10, yheight - 10 )
  dummy = setwindow( .TRUE., X0-1.0, Y0-1.0, X1+1.0, Y1+1.0 )
  CALL settextwindow( (rows / 2 ) + 1, 5, rows, cols - 5)
 ENDIF
 CALL clearscreen( $GWINDOW )
 IF(ILOX.EQ.0)  dummy = rectangle_w( $GBORDER, X0, Y0, X1, Y1 )
 IF(NMODE.EQ.6)  THEN
  CALL SETTEXTPOSITION(1,1,S)
 ELSE
  CALL SETTEXTPOSITION(1,20,S)
 ENDIF
  WRITE(FULOX,1)ITANK,IENG
  CALL OUTTEXT(FULOX)
ENDIF
IF(R.EQ.'B'.OR.ILOX.EQ.1)  THEN
 IF(NMODE.EQ.6)  THEN
  CALL setviewport( 10, 20, xwidth - 10, halfy  )
  dummy = setwindow( .TRUE., X0-1.0, Y0-1.0, X1+1.0, Y1+1.0 )
  CALL settextwindow(0 , 1, (rows / 2 ) , cols)
 ELSE
  CALL setviewport( 10, 25, xwidth - 10, halfy - 5 )
  dummy = setwindow( .TRUE., X0-1.0, Y0-1.0, X1+1.0, Y1+1.0 )
  CALL settextwindow(0 , 1, (rows / 2 ) , cols - 5)
 ENDIF
 CALL clearscreen( $GWINDOW )
 dummy = rectangle_w( $GBORDER, X0, Y0, X1, Y1 )
 IF(NMODE.EQ.6)  THEN
  CALL SETTEXTPOSITION(0,1,S)
 ELSE
  CALL SETTEXTPOSITION(0,20,S)
 ENDIF
 CALL OUTTEXT(TITLE)
 IF(NMODE.EQ.6)  THEN
  CALL SETTEXTPOSITION(2,1,S)
 ELSE
  CALL SETTEXTPOSITION(2,20,S)
 ENDIF
 IF(ILOX.EQ.0)  THEN
  WRITE(FULOX,2)ITANK,IENG
  CALL OUTTEXT(FULOX)
 ENDIF
ENDIF
RETURN
```

```
      END
      SUBROUTINE WINDLO(XMIN,XMAX,YMIN,YMAX)
C       Sets up gain window
      INCLUDE  'FGRAPH.FD'
      INTEGER*2             dummy
      INTEGER*2             xwidth, yheight, cols, rows, halfy
      RECORD /videoconfig/  screen
      COMMON                screen
      COMMON /NOCOL/NCOLS,NMODE
      INTEGER*2 NCOLS
      REAL*8 XMIN, XMAX, YMIN, YMAX, XLEN, YLEN
      REAL*8 XMINP, XMAXP, YMINP, YMAXP
      XLEN=0.1*(XMAX-XMIN)
      YLEN=0.1*(YMAX-YMIN)
      XMINP=XMIN-XLEN
      XMAXP=XMAX+XLEN
      YMINP=YMIN-YLEN
      YMAXP=YMAX+YLEN
      xwidth  = screen.numxpixels
      yheight = screen.numypixels
      cols    = screen.numtextcols
      rows    = screen.numtextrows
      halfy   = yheight/2
      IF(NCOLS.LE.40)  THEN
       CALL setviewport( 50, halfy + 10, xwidth - 20, yheight - 30 )
      ELSE
       CALL setviewport( 100, halfy + 10, xwidth - 50, yheight - 50 )
      ENDIF
      CALL settextwindow( (rows / 2 ) + 1, 1, rows, cols - 1)
      dummy = setwindow(.TRUE.,XMINP,YMINP,XMAXP,YMAXP)
      CALL clearscreen( $GWINDOW )
      RETURN
      END
      SUBROUTINE WINDUP(XMIN,XMAX,YMIN,YMAX)
C       Sets up phase angle window
      INCLUDE  'FGRAPH.FD'
      INTEGER*2             dummy
      INTEGER*2             xwidth, yheight, cols, rows, halfy
      RECORD /videoconfig/  screen
      COMMON                screen
      COMMON /NOCOL/NCOLS,NMODE
      INTEGER*2 NCOLS
      REAL*8 XMIN, XMAX, YMIN, YMAX, XLEN, YLEN
      REAL*8 XMINP, XMAXP, YMINP, YMAXP
      XLEN=0.1*(XMAX-XMIN)
      YLEN=0.1*(YMAX-YMIN)
      XMINP=XMIN-XLEN
      XMAXP=XMAX+XLEN
      YMINP=YMIN-YLEN
      YMAXP=YMAX+YLEN
      xwidth  = screen.numxpixels
      yheight = screen.numypixels
      cols    = screen.numtextcols
```

```
rows     = screen.numtextrows
halfy    = yheight/2
IF(NCOLS.LE.40)  THEN
 CALL setviewport( 50, 10, xwidth - 20, halfy - 30 )
ELSE
 CALL setviewport( 100, 10, xwidth - 50, halfy - 50 )
ENDIF
CALL settextwindow( 1, 1, (rows / 2 ) - 1, cols - 1)
dummy = setwindow(.TRUE.,XMINP,YMINP,XMAXP,YMAXP)
CALL clearscreen( $       GWINDOW )
RETURN
END
```

```
      PROGRAM NYQUIST2


      SUBROUTINE ADMIT(S,GADM,A,AREA,CMAN,CTANK,DPROR,L,LFLOW,PMRAT,
     *                 SEGMN,SECIN,SPLIT,LOPEND,PCAP,PIND,IENG,TFLOW,
     *                 NOLINE,IP,ILINE,ITLIN)
C       Determines admittance looking toward tank
      CHARACTER*40 TITLE
      CHARACTER*20 TITLF
      INTEGER*2 IHR,IMIN,IYR,IMON,IDAY
      CHARACTER*2 AP
      COMMON /WCATTT/TITLE,TITLF,IHR,IMIN,AP,IYR,IMON,IDAY
      INTEGER SEGMN(25),SECIN(75,25)
      INTEGER IENG(25),NOLINE(25)
      REAL AREA(75,25),PCAP(75,25),PIND(75,25),L(75,25),LFLOW,ZO(75,25),
     *     CMAN(25),DPROR(25),PMRAT(25),ZOR(25),TFLOW(25)
      COMPLEX G(0:75,25),ZT(0:75,25),ZG(0:75,25),GOLD(0:75,25),GADM(25),
     *        S,ZGEFF,ZTEFF
      COMMON /WORK1/G,ZT,ZG
      COMMON /WORK2/ZO
      COMMON /FACTOR/SFAC
      COMPLEX CTANH,RHS,CFAC,CAPN,CAPM
      CHARACTER*13 TYPEL(2)
      DATA TYPEL/' in FUEL line',' in LOX  line'/
      DATA GRAV/32.2/
      DATA IOPEN/0/
      ZTOP=A/GRAV
        TMASS=0.0
        TCOUNT=0.0
      DO 22 J=IP,IP+SPLIT
       GOLD(0,J)=0.0
       SECIN(SEGMN(J)+1,J)=0
       DO 21 I=1,SEGMN(J)
        GOLD(I,J)=0.0
        ZO(I,J)=0.0
        IF(SECIN(I,J).LE.2)  THEN
         ZO(I,J)=ZTOP/AREA(I,J)
        ELSEIF(SECIN(I,J).EQ.7)  THEN
         ZO(I,J)=0.0
        ELSE
         ZO(I,J)=SQRT(PIND(I,J)/PCAP(I,J))
        ENDIF
   21  CONTINUE
       IF(IENG(J).NE.0)  THEN
        IE=IENG(J)
        ZOR(J)=2.0*DPROR(IE)/LFLOW
        IF(J.EQ.IP.AND.SPLIT.EQ.0.0)  THEN
         TMASS=TFLOW(IE)
        ELSEIF(J.NE.IP)  THEN
         TMASS=TMASS+NOLINE(J)*TFLOW(IE)
         TCOUNT=TCOUNT+NOLINE(J)
        ENDIF
       ENDIF
```

```
   22 CONTINUE
      IF(TCOUNT.EQ.0.0)  TCOUNT=1.0
      G(0,IP)=CTANK*S
      G(0,IP)=G(0,IP)/TCOUNT
      ZT(0,IP)=1.0/G(0,IP)
      DO 31 KLOOP=1,LOPEND
       DO 25 J=IP,IP+SPLIT
        IF(J.NE.IP)  THEN
         G(0,J)=G(SEGMN(IP),IP)
         ZT(0,J)=1.0/G(0,J)
        ENDIF
        DO 24 I=1,SEGMN(J)
         ZGEFF=G(I-1,J)
         IF(SECIN(I,J).LE.1)  THEN
C             bend in pipe or straight section
          TL=L(I,J)/A
          IF(KLOOP.NE.1.AND.SPLIT.NE.0.AND.J.NE.IP.AND.I.EQ.1)  THEN
           ZGEFF=0.0
           DO 23 K=IP+1,IP+SPLIT
            IE=IENG(K)
            IF(K.EQ.J)  THEN
             ZGEFF=ZGEFF+(NOLINE(K)-1.0)/ZG(0,K)
            ELSE
             ZGEFF=ZGEFF+NOLINE(K)/ZG(0,K)
            ENDIF
   23       CONTINUE
           ZGEFF=G(SEGMN(IP),IP)+ZGEFF
          ENDIF
          G(I,J)=(1.0+CTANH(S*TL)/(ZGEFF*ZO(I,J)))/(1.0+ZGEFF*
     *           ZO(I,J)*CTANH(S*TL))
         ELSEIF(SECIN(I,J).EQ.2)  THEN
C             inline resonator
          G(I,J)=1.0+PCAP(I,J)*S/ZGEFF
         ELSEIF(SECIN(I,J).EQ.3)  THEN
C             tuned stub
          G(I,J)=1.0+CTANH(S*SQRT(PIND(I,J)*PCAP(I,J)))/(ZO(I,J)*
     *           ZGEFF)
         ELSEIF(SECIN(I,J).EQ.4)  THEN
C             helmholtz resonator
          G(I,J)=1.0+S*PCAP(I,J)/(1.0+PIND(I,J)*PCAP(I,J)*S**2)/ZGEFF
         ELSEIF(SECIN(I,J).EQ.5)  THEN
C             parallel resonator
          G(I,J)=PIND(I,J)*PCAP(I,J)*S**2+1.0
          G(I,J)=G(I,J)/(G(I,J)+PIND(I,J)*S*ZGEFF)
         ELSEIF(SECIN(I,J).EQ.6)  THEN
C             pump
          G(I,J)=(1.0+PCAP(I,J)*S/ZGEFF)/(1.0+(PIND(I,J)*S+
     *            AREA(I,J))*(PCAP(I,J)*S+ZGEFF))
         ELSEIF(SECIN(I,J).EQ.7)  THEN
          G(SEGMN(J),J)=1.0+CMAN(J)*S/ZGEFF
         ENDIF
         G(I,J)=G(I,J)*ZGEFF
         ZT(I,J)=1.0/G(I,J)
```

```
 24   CONTINUE
      IF(SPLIT.NE.0.0.AND.J.EQ.IP)  GO TO 25
      G(SEGMN(J)+1,J)=1.0/(1.0+ZOR(J)*G(SEGMN(J),J))
      G(SEGMN(J)+1,J)=G(SEGMN(J)+1,J)*G(SEGMN(J),J)
 25   CONTINUE
      IF(LOPEND.EQ.1.OR.SPLIT.EQ.0.0)  GO TO 31
      DO 28 J=IP+SPLIT,IP,-1
       IF(J.EQ.IP)  THEN
        LOPHI=SEGMN(J)
       ELSE
        ZG(SEGMN(J)-1,J)=ZOR(J)/(ZOR(J)*CMAN(J)*S+1.0)
        LOPHI=SEGMN(J)-2
       ENDIF
       IF(LOPHI.NE.0)  THEN
        DO 27 I=LOPHI,1,-1
         IF(I.EQ.LOPHI.AND.J.EQ.IP)  THEN
          ZG(I,J)=0.0
          ZTEFF=ZT(I-1,J)
          DO 26 K=IP+1,IP+SPLIT
           ZGEFF=ZG(1,K)
           ZOEFF=ZO(1,K)
           ZLP=L(1,K)
           TL=(L(I,J)+ZLP)/A
           CAPN=(ZOEFF-ZTEFF)/(ZOEFF+ZTEFF)
           CAPM=(ZOEFF-ZGEFF)/(ZOEFF+ZGEFF)
           CFAC=CEXP(-2.0*S*TL)
           RHS=(ZOEFF+ZGEFF)*(1.0-CAPN*CAPM*CFAC)*CEXP(S*ZLP/A)
           CFAC=CAPN*CFAC*CEXP(2.0*S*ZLP/A)
           ZG(0,K)=(RHS-ZOEFF*(1.0-CFAC))/(1.0+CFAC)
           ZG(I,J)=ZG(I,J)+NOLINE(K)/ZG(0,K)
 26       CONTINUE
          ZG(I,J)=1.0/ZG(I,J)
         ELSE
          ZGEFF=ZG(I+1,J)
          ZOEFF=ZO(I+1,J)
          ZLP=L(I+1,J)
          ZTEFF=ZT(I-1,J)
          IF(SECTN(I+1,J).LE.1)  THEN
C            bend in pipe or straight section
           TL=(L(I,J)+ZLP)/A
           CAPN=(ZOEFF-ZTEFF)/(ZOEFF+ZTEFF)
           CAPM=(ZOEFF-ZGEFF)/(ZOEFF+ZGEFF)
           CFAC=CEXP(-2.0*S*TL)
           RHS=(ZOEFF+ZGEFF)*(1.0-CAPN*CAPM*CFAC)*CEXP(S*ZLP/A)
           CFAC=CAPN*CFAC*CEXP(2.0*S*ZLP/A)
           ZG(I,J)=(RHS-ZOEFF*(1.0-CFAC))/(1.0+CFAC)
          ELSEIF(SECTN(I+1,J).EQ.2)  THEN
C            inline resonator
           ZG(I,J)=ZGEFF/(ZGEFF*PCAP(I+1,J)*S+1.0)
          ELSEIF(SECTN(I+1,J).EQ.3)  THEN
C            tuned stub
           ZG(I,J)=ZOEFF/CTANH(S*SQRT(PIND(I+1,J)*PCAP(I+1,J)))
           ZG(I,J)=(ZG(I,J)*ZGEFF)/(ZG(I,J)+ZGEFF)
```

```
          ELSEIF(SECTN(I+1,J).EQ.4)  THEN
C           helmholtz resonator
           ZG(I,J)=(1.0+PIND(I+1,J)*PCAP(I+1,J)*S**2)/(PCAP(I+1,J)*S)
           ZG(I,J)=(ZG(I,J)*ZGEFF)/(ZG(I,J)+ZGEFF)
          ELSEIF(SECTN(I+1,J).EQ.5)  THEN
C           parallel resonator
           ZG(I,J)=ZGEFF+PIND(I+1,J)*S/(PIND(I+1,J)*PCAP(I+1,J)*S**2+
     *          1.0)
          ELSEIF(SECTN(I+1,J).EQ.6)  THEN
C           pump
           ZG(I,J)=ZGEFF+PIND(I+1,J)*S-AREA(I+1,J)
           ZG(I,J)=ZG(I,J)/(1.0+ZG(I,J)*PCAP(I+1,J)*S)
          ENDIF
         ENDIF
   27    CONTINUE
        ENDIF
   28   CONTINUE
        ERRP=0.0
        DO 30 J=IP,IP+SPLIT
         DO 29 I=1,SEGMN(J)
          GDIF=CABS(GOLD(I,J))
          IF(GDIF.NE.0.0)  GDIF=ABS(GDIF-CABS(G(I,J)))/GDIF
          IF(GDIF.GT.ERRP)  THEN
           ERRP=GDIF
           WG=CABS(G(I,J))
           WGOLD=CABS(GOLD(I,J))
           IWG=I
           JWG=J
          ENDIF
          GOLD(I,J)=G(I,J)
   29    CONTINUE
   30   CONTINUE
        IF(KLOOP.GT.1.AND.ERRP.LT.0.001)  GO TO 32
   31 CONTINUE
        IF(LOPEND.EQ.1)  GO TO 32
        IF(IOPEN.EQ.0)  THEN
         OPEN(UNIT=13,FILE='SURF.ERR')
         WRITE(13,*)' '
         WRITE(13,*)' '
         WRITE(13,*)TITLE
         WRITE(13,*)' '
         IOPEN=1
        ENDIF
        WRITE(13,'('' jw ='',F8.1,''  after'',I3,'' iterations'',
     *           '' has error of'',F8.3,''% '',A)')
     *           AIMAG(S)/SFAC,LOPEND,100.0*ERRP,TYPEL(ITLIN)
        WRITE(13,'(10X,'' I='',I3,3X,''J='',I3,3X,''|G|='',1PE12.4,3X,
     *           ''|GOLD|='',E12.4)')IWG,JWG,WG,WGOLD
   32 CONTINUE
      DO 35 J=IP,IP+SPLIT
       IF(IENG(J).EQ.0.0)  THEN
        RATPM=0.0
        DO 33 I=IP+1,IP+SPLIT
```

```
        RATPM=RATPM+PMRAT(IENG(I))
   33   CONTINUE
        LOPHI=SEGMN(J)
       ELSE
        RATPM=PMRAT(IENG(J))
        LOPHI=SEGMN(J)+1
       ENDIF
       DO 34 I=0,LOPHI
        G(I,J)=RATPM*G(I,J)
   34  CONTINUE
       IF(J.EQ.IP.AND.SPLIT.NE.0.0)  GO TO 35
       GADM(ILINE)=G(LOPHI,J)
       ILINE=ILINE+1
   35 CONTINUE
      RETURN
      END
      SUBROUTINE BENDS(PIPE1,PIPE2,PIPE3,PIPE4,VALUE,DIME)
C       Computes effective straight pipe for bend
      REAL LBEND
      LBEND=0.0174533*PIPE1*ABS(PIPE2)
      RATIO=(PIPE1-0.5*PIPE3)/(PIPE1+0.5*PIPE3)
      CALL GINERT(ABS(PIPE2),RATIO,Y)
      GAMMA=(LBEND+Y*PIPE3)/LBEND
      VALUE=GAMMA*(LBEND+2.0*PIPE4)
      DIME=PIPE3/(GAMMA)**0.25
      RETURN
      END
      SUBROUTINE BNSECT(J,ITYPE,POINT,PIPE1,PIPE2,PIPE3,PIPE4)
C       Computes plot coordinates for a bend
      COMMON /PIPPXY/X,XH,XL,Y,YH,YL,XMIN,XMAX,YMIN,YMAX,SINA,COSA
      COMMON /ARCCON/XC,YC,RAD,ANG,ANGLE
      REAL POINT(8,200)
      INTEGER*2 ITYPE(200)
C          first straight section of bend
      IF(PIPE4.NE.0.0)  CALL STSECT(J,ITYPE,POINT,PIPE4,PIPE3)
C          curved section of bend
      IF(PIPE2.GE.0.0)  THEN
       XC=X-SINA*PIPE1
       YC=Y+COSA*PIPE1
       DIA= 0.5
      ELSE
       XC=X+SINA*PIPE1
       YC=Y-COSA*PIPE1
       DIA=-0.5
      ENDIF
      J=J+1
      ITYPE(J)=0
      POINT(1,J)=XC
      POINT(2,J)=YC
      POINT(3,J)=ANG
      ANG=ANG+0.01745329*PIPE2
      ANGLE=ANGLE+0.5*PIPE2
      RANG=0.01745329*ANGLE
```

```
COSA=COS(RANG)
SINA=SIN(RANG)
RAD=PIPE1-DIA*PIPE3
POINT(4,J)=ANG
POINT(5,J)=RAD
X0=XC-RAD
Y0=YC+RAD
X1=XC+RAD
Y1=YC-RAD
X2=XH
Y2=YH
SLENTH=2.0*RAD*SIN(0.00872665*ABS(PIPE2))
XH=X2+COSA*SLENTH
YH=Y2+SINA*SLENTH
X3=XH
Y3=YH
IF(DIA.LT.0.0)  THEN
 HOLD=X2
 X2=X3
 X3=HOLD
 HOLD=Y2
 Y2=Y3
 Y3=HOLD
ENDIF
RAD=PIPE1+DIA*PIPE3
X0=XC-RAD
Y0=YC+RAD
X1=XC+RAD
Y1=YC-RAD
X2=XL
Y2=YL
SLENTH=2.0*RAD*SIN(0.00872665*ABS(PIPE2))
XL=X2+COSA*SLENTH
YL=Y2+SINA*SLENTH
X3=XL
Y3=YL
IF(DIA.LT.0.0)  THEN
 HOLD=X2
 X2=X3
 X3=HOLD
 HOLD=Y2
 Y2=Y3
 Y3=HOLD
ENDIF
J=J+1
ITYPE(J)=0
POINT(1,J)=POINT(1,J-1)
POINT(2,J)=POINT(2,J-1)
POINT(3,J)=POINT(3,J-1)
POINT(4,J)=POINT(4,J-1)
POINT(5,J)=RAD
SLENTH=2.0*PIPE1*SIN(0.00872665*ABS(PIPE2))
X=X+COSA*SLENTH
```

```
      Y=Y+SINA*SLENTH
      XMIN=AMIN1(X,XL,XH,XMIN)
      XMAX=AMAX1(X,XL,XH,XMAX)
      YMIN=AMIN1(Y,YL,YH,YMIN)
      YMAX=AMAX1(Y,YL,YH,YMAX)
C          last straight section of bend
      ANGLE=ANGLE+0.5*PIPE2
      RANG=0.01745329*ANGLE
      COSA=COS(RANG)
      SINA=SIN(RANG)
      J=J+1
      ITYPE(J)=1
      POINT(1,J)=XH
      POINT(2,J)=YH
      POINT(3,J)=XL
      POINT(4,J)=YL
      X=X+COSA*PIPE4
      XH=X-0.5*SINA*PIPE3
      XL=X+0.5*SINA*PIPE3
      Y=Y+SINA*PIPE4
      YH=Y+0.5*COSA*PIPE3
      YL=Y-0.5*COSA*PIPE3
      POINT(5,J)=XH
      POINT(6,J)=YH
      POINT(7,J)=XL
      POINT(8,J)=YL
      XMIN=AMIN1(X,XL,XH,XMIN)
      XMAX=AMAX1(X,XL,XH,XMAX)
      YMIN=AMIN1(Y,YL,YH,YMIN)
      YMAX=AMAX1(Y,YL,YH,YMAX)
      RETURN
      END
      COMPLEX FUNCTION CCOSH(S)
C       Evaluates the complex hyperbolic cosine
      COMPLEX S
      REAL LAMDA, MU
      LAMDA=REAL(S)
      MU=AIMAG(S)
      COSHR=COSH(LAMDA)*COS(MU)
      COSHI=SINH(LAMDA)*SIN(MU)
      CCOSH=CMPLX(COSHR,COSHI)
      RETURN
      END
      COMPLEX FUNCTION CSINH(S)
C       Evaluates the complex hyperbolic sine
      COMPLEX S
      REAL LAMDA, MU
      LAMDA=REAL(S)
      MU=AIMAG(S)
      SINHR=SINH(LAMDA)*COS(MU)
      SINHI=COSH(LAMDA)*SIN(MU)
      CSINH=CMPLX(SINHR,SINHI)
      RETURN
```

```
      END
      COMPLEX FUNCTION CTANH(S)
C       Evaluates the complex hyperbolic tangent
      COMPLEX CCOSH,CSINH,S
      CTANH=CSINH(S)/CCOSH(S)
      RETURN
      END
      SUBROUTINE ENGNO(IUNIT)
C       Reads engine parameters
      COMMON /EPARAM/MENG,TFLOW(25),PCHMB(25),DPROR(25),PMRAT(25)
      READ(IUNIT,*)MENG
      IF(MENG.GT.25)  THEN
       WRITE(*,*)'  Number of engines must be less than 25'
       STOP
      ENDIF
      IF(MENG.LE.0)  MENG=1
      DO 21 I=1,MENG
       READ(IUNIT,*)TFLOW(I),PCHMB(I),DPROR(I)
       PMRAT(I)=PCHMB(I)/TFLOW(I)
   21 CONTINUE
      RETURN
      END
      SUBROUTINE FUEL(S,GF,IUNIT,IUNITP,IGONE)
C       Handles fuel piping logic
      COMPLEX GF(25),S
      COMMON /EPARAM/MENG,TFLOW(25),PCHMB(25),DPROR(25),PMRAT(25)
      INTEGER SEGMN(25),SECIN(75,25),NOLINE(25),IENG(25),ITANK(25),
     *        LOPOLD(25),LOPEND(25)
      REAL KMAN(25),KTANK(25),LFLOW(25),L(75,25)
      COMMON /FPARAM/MLINE,SPLIT(25),A(25),CMAN(25),CTANK(25),
     *               DENS(25),KMAN,KTANK,LFLOW,VOL(25),VOLMF(25),
     *               AREA(75,25),DIA(75,25),L,PIND(75,25),
     *               PCAP(75,25),AVGK(25),
     *               SEGMN,SECIN,NOLINE,IENG,ITANK,LOPOLD,LOPEND
      COMMON /FOPIPE/PIPE1(75,25),PIPE2(75,25),PIPE3(75,25),
     *              PIPE4(75,25),PIPE5(75,25)
      CHARACTER*24 FUELIN,NAMLIN(2)
      COMMON /WCAOUT/NAMLIN
      CHARACTER*1 ANS
      IF(IGONE.EQ.2) THEN
       WRITE(*,'(A\)')' Is the fuel file name FUEL.RLN? (Y/N) '
       READ(*,'(A)')ANS
       IF(ANS.NE.'N'.AND.ANS.NE.'n')  THEN
        OPEN(UNIT=IUNIT,FILE='FUEL.RLN')
        NAMLIN(1)='FUEL.RLN'
       ELSE
        WRITE(*,'(A\)')'  Enter name of file with fuel line data '
        READ(*,'(A)')FUELIN
        OPEN(IUNIT,FILE=FUELIN)
        NAMLIN(1)=FUELIN
       ENDIF
       OPEN(IUNITP,FORM='UNFORMATTED')
      ENDIF
```

```
      CALL FULOX(S,GF,SEGMN,SECIN,PIPE1,PIPE2,PIPE3,PIPE4,PIPE5,
     *  A,AREA,AVGK,CMAN,CTANK,DENS,DIA,IENG,IGONE,ITANK,
     *  IUNIT,IUNITP,KMAN,KTANK,L,LOPEND,LOPOLD,LFLOW,MLINE,NOLINE,PCAP,
     *  PIND,SPLIT,VOL,VOLMF,1)
      RETURN
      END
      SUBROUTINE FULOX(S,GF,SEGMN,SECIN,PIPE1,PIPE2,PIPE3,PIPE4,PIPE5,
     *  A,AREA,AVGK,CMAN,CTANK,DENS,DIA,IENG,IGONE,ITANK,
     *  IUNIT,IUNITP,KMAN,KTANK,L,LOPEND,LOPOLD,LFLOW,MLINE,NOLINE,PCAP,
     *  PIND,SPLIT,VOL,VOLMF,ITLIN)
C       Handles read, modify, and admittance calls for fuel and lox
      COMMON /EPARAM/MENG,TFLOW(25),PCHMB(25),DPROR(25),PMRAT(25)
      INTEGER SEGMN(25),SECIN(75,25),NOLINE(25),IENG(25),ITANK(25),
     *        LOPOLD(25),LOPEND(25)
      REAL KMAN(25),KTANK(25),LFLOW(25),L(75,25)
      REAL SPLIT(25),A(25),CMAN(25),CTANK(25),
     *               DENS(25),VOL(25),VOLMF(25),
     *               AREA(75,25),DIA(75,25),PIND(75,25),
     *               PCAP(75,25),AVGK(25)
      REAL PIPE1(75,25),PIPE2(75,25),PIPE3(75,25),
     *     PIPE4(75,25),PIPE5(75,25)
      COMPLEX GF(25),S
      CHARACTER*20 TITL
      CHARACTER*1 ANS
      CHARACTER*40 QUEST1(2)
      CHARACTER*48 QUEST2(2)
      CHARACTER*40 QUEST3(2)
      DATA QUEST1/' Do you wish to modify fuel line data?  ',
     *           ' Do you wish to modify lox line data?   '/
      DATA QUEST2/' Do you wish to modify current fuel line data?  ',
     *           ' Do you wish to modify current lox line data?   '/
      DATA QUEST3/' Do you wish to rewind fuel line file?  ',
     *           ' Do you wish to rewind lox line file?   '/
      IF(IGONE.EQ.2)  THEN
       CALL RLINE(TITL,SEGMN,SECIN,PIPE1,PIPE2,PIPE3,
     *  PIPE4,PIPE5,L,AREA,DIA,PIND,PCAP,LOPEND,LOPOLD,SPLIT,IUNIT,
     *  A,CMAN,CTANK,DENS,KMAN,KTANK,LFLOW,VOL,VOLMF,NOLINE,IENG,ITANK,
     *  AVGK,MLINE)
       REWIND IUNITP
       WRITE(IUNITP)PIPE1,PIPE2,PIPE3,PIPE4,PIPE5
       WRITE(*,'(A\)')QUEST1(ITLIN)
       READ(*,'(A)')ANS
       IF(ANS .EQ. 'Y' .OR. ANS .EQ. 'y')  THEN
        CALL MODIFY(TITL,SEGMN,SECIN,PIPE1,PIPE2,PIPE3,
     *  PIPE4,PIPE5,L,AREA,DIA,PIND,PCAP,LOPEND,LOPOLD,SPLIT,IUNIT,
     *  A,CMAN,CTANK,DENS,KMAN,KTANK,LFLOW,VOL,VOLMF,NOLINE,IENG,ITANK,
     *  AVGK,MLINE)
        REWIND IUNITP
        WRITE(IUNITP)PIPE1,PIPE2,PIPE3,PIPE4,PIPE5
       ENDIF
      ELSEIF(IGONE.EQ.0)  THEN
       IP=1
       ILINE=1
```

```
      DO 21 I=1,MLINE
       IT=ITANK(I)
       CALL ADMIT(S,GF,A(IT),AREA,CMAN,CTANK(IT),DPROR,
     *            L,LFLOW(IT),PMRAT,SEGMN,SECIN,
     *            SPLIT(I),LOPEND(I),PCAP,PIND,IENG,TFLOW,
     *            NOLINE,IP,ILINE,ITLIN)
       IP=IP+SPLIT(I)+1
   21  CONTINUE
       RETURN
      ELSEIF(IGONE .EQ. 1)  THEN
       WRITE(*,'(A\)')QUEST2(ITLIN)
       READ(*,'(A)')ANS
       IF(ANS .EQ. 'Y' .OR. ANS .EQ. 'y')  THEN
        CALL MODIFY(TITL,SEGMN,SECIN,PIPE1,PIPE2,PIPE3,
     *  PIPE4,PIPE5,L,AREA,DIA,PIND,PCAP,LOPEND,LOPOLD,SPLIT,IUNIT,
     *  A,CMAN,CTANK,DENS,KMAN,KTANK,LFLOW,VOL,VOLMF,NOLINE,IENG,ITANK,
     *  AVGK,MLINE)
        REWIND IUNITP
        WRITE(IUNITP)PIPE1,PIPE2,PIPE3,PIPE4,PIPE5
       ELSE
        WRITE(*,'(A\)')QUEST3(ITLIN)
        READ(*,'(A)')ANS
        IF(ANS .EQ. 'Y' .OR. ANS .EQ. 'y')  REWIND IUNIT
        CALL RLINE(TITL,SEGMN,SECIN,PIPE1,PIPE2,PIPE3,
     *  PIPE4,PIPE5,L,AREA,DIA,PIND,PCAP,LOPEND,LOPOLD,SPLIT,IUNIT,
     *  A,CMAN,CTANK,DENS,KMAN,KTANK,LFLOW,VOL,VOLMF,NOLINE,IENG,ITANK,
     *  AVGK,MLINE)
        REWIND IUNITP
        WRITE(IUNITP)PIPE1,PIPE2,PIPE3,PIPE4,PIPE5
        WRITE(*,*)QUEST1(ITLIN)
        WRITE(*,'(A\)')'  if not, press enter key. '
        READ(*,'(A)')ANS
        WRITE(*,*)' '
        IF(ANS .EQ. 'Y' .OR. ANS .EQ. 'y') THEN
         CALL MODIFY(TITL,SEGMN,SECIN,PIPE1,PIPE2,PIPE3,
     *   PIPE4,PIPE5,L,AREA,DIA,PIND,PCAP,LOPEND,LOPOLD,SPLIT,IUNIT,
     *   A,CMAN,CTANK,DENS,KMAN,KTANK,LFLOW,VOL,VOLMF,NOLINE,IENG,ITANK,
     *   AVGK,MLINE)
         REWIND IUNITP
         WRITE(IUNITP)PIPE1,PIPE2,PIPE3,PIPE4,PIPE5
        ENDIF
       ENDIF
       IGONE=0
      ENDIF
      RETURN
      END
      SUBROUTINE GETKS(N,K,I,J,K1R,K2R,K3R,K4R,K1C,K2C,K3C,K4C)
C       Determines Nyquist equation to be plotted
      REAL K1R(1001),K1C(1001),K2R(1001),K2C(1001),K3R(1001),K3C(1001),
     *     K4R(1001),K4C(1001)
      REAL R1K(25),C1K(25),R2K(25),C2K(25),R3K(25),C3K(25),
     *     R4K(25),C4K(25)
      REWIND 17
```

```
      M=0
      IF(I.NE.0.AND.J.NE.0)  THEN
       CALL GETM(I,J,M)
       IF(M.EQ.0)  THEN
        K=0
        RETURN
       ENDIF
      ENDIF
      DO 21 L=1,N
       READ(17)R1K,C1K,R2K,C2K,R3K,C3K,R4K,C4K
       K1R(L)=R1K(K)
       K1C(L)=C1K(K)
       IF(J.NE.0)  THEN
        K2R(L)=R2K(J)
        K2C(L)=C2K(J)
       ENDIF
       IF(I.NE.0)  THEN
        K3R(L)=R3K(I)
        K3C(L)=C3K(I)
       ENDIF
       IF(M.NE.0)  THEN
        K4R(L)=R4K(M)
        K4C(L)=C4K(M)
       ENDIF
   21 CONTINUE
      RETURN
      END
      SUBROUTINE GETM(I,J,M)
C       Determines location of data to be plotted
      INTEGER SEGMN,SECTN(150)
      COMMON /SETUP/PIPE1(150),PIPE2(150),PIPE3(150),PIPE4(150),
     *              NENGF(25),NTANKF(25),NLINEF(25),NSPF(25),ILINEF,
     *              NENGO(25),NTANKO(25),NLINEO(25),NSPO(25),ILINEO,
     *              SEGMN,SECTN
      DO 22 II=1,ILINEF
       DO 21 JJ=1,ILINEO
        IF(NENGF(II).NE.NENGO(JJ))  GO TO 21
        M=M+1
        IF(II.EQ.I.AND.JJ.EQ.J)  RETURN
   21  CONTINUE
   22 CONTINUE
      WRITE(*,*)' Somethings WRONG! Plot will be bypassed.'
      M=0
      RETURN
      END
      SUBROUTINE GINERT(BEND,X,Y)
C       Evaluates curve fit of inertance of bends
      DIMENSION B(3)
      DATA B/0.0,0.7877014E-02,-0.2814679E-04/
      A=B(1)+(B(2)+B(3)*BEND)*BEND
      Y=A*(X-1.0)**2
      RETURN
      END
```

```
      SUBROUTINE HHSECT(J,ITYPE,POINT,LEN,DIA,VOL)
C       Computes plot coordinates for Helmholtz resonator
      COMMON /PIPPXY/X,XH,XL,Y,YH,YL,XMIN,XMAX,YMIN,YMAX,SINA,COSA
      REAL LEN,POINT(8,200)
      INTEGER*2 ITYPE(200)
      XOLD=X
      XHOLD=XH
      XLOLD=XL
      YOLD=Y
      YHOLD=YH
      YLOLD=YL
      SINOLD=SINA
      COSOLD=COSA
      DIAM=SQRT((XH-XL)**2+(YH-YL)**2)
      CALL TSSECT(J,ITYPE,POINT,LEN,DIA)
      XC=0.5*(XOLD+X)
      YC=0.5*(YOLD+Y)
      XOLD=X
      YOLD=Y
      SINA=COSOLD
      COSA=-SINOLD
      X=XC+COSA*(LEN+0.5*DIAM)
      Y=YC+SINA*(LEN+0.5*DIAM)
      SIDE=VOL**0.3333333
      CALL STSECT(J,ITYPE,POINT,SIDE,SIDE)
      X=XOLD
      Y=YOLD
      SINA=SINOLD
      COSA=COSOLD
      DIAM=SQRT((XHOLD-XLOLD)**2+(YHOLD-YLOLD)**2)
      XH=X-0.5*SINA*DIAM
      XL=X+0.5*SINA*DIAM
      YH=Y+0.5*COSA*DIAM
      YL=Y-0.5*COSA*DIAM
      RETURN
      END
      SUBROUTINE LOX(S,GOX,IUNIT,IUNITP,IGONE)
C       Handles fuel piping logic
      COMPLEX GOX(25),S
      COMMON /EPARAM/MENG,TFLOW(25),PCHMB(25),DPROR(25),PMRAT(25)
      INTEGER SEGMN(25),SECTN(75,25),NOLINE(25),IENG(25),ITANK(25),
     *        LOPOLD(25),LOPEND(25)
      REAL KMAN(25),KTANK(25),LFLOW(25),L(75,25)
      COMMON /OPARAM/MLINE,SPLIT(25),A(25),CMAN(25),CTANK(25),
     *               DENS(25),KMAN,KTANK,LFLOW,VOL(25),VOLMF(25),
     *               AREA(75,25),DIA(75,25),L,PIND(75,25),
     *               PCAP(75,25),AVGK(25),
     *               SEGMN,SECTN,NOLINE,IENG,ITANK,LOPOLD,LOPEND
      COMMON /FOPIPE/PIPE1(75,25),PIPE2(75,25),PIPE3(75,25),
     *              PIPE4(75,25),PIPE5(75,25)
      CHARACTER*24 LOXIN,NAMLIN(2)
      COMMON /WCAOUT/NAMLIN
      CHARACTER*1 ANS
```

```
      IF(IGONE.EQ.2) THEN
       WRITE(*,'(A\)')' Is the lox file name LOX.RLN? (Y/N) '
       READ(*,'(A)')ANS
       IF(ANS.NE.'N'.AND.ANS.NE.'n')  THEN
        OPEN(UNIT=IUNIT,FILE='LOX.RLN')
        NAMLIN(2)='LOX.RLN'
       ELSE
        WRITE(*,'(A\)')'  Enter name of file with lox line data '
        READ(*,'(A)')LOXIN
        OPEN(IUNIT,FILE=LOXIN)
        NAMLIN(2)=LOXIN
       ENDIF
       OPEN(IUNITP,FORM='UNFORMATTED')
      ENDIF
      CALL FULOX(S,GOX,SEGMN,SECIN,PIPE1,PIPE2,PIPE3,PIPE4,PIPE5,
     *  A,AREA,AVGK,CMAN,CTANK,DENS,DIA,IENG,IGONE,ITANK,
     *  IUNIT,IUNITP,KMAN,KTANK,L,LOPEND,LOPOLD,LFLOW,MLINE,NOLINE,PCAP,
     *  PIND,SPLIT,VOL,VOLMF,2)
      RETURN
      END
      SUBROUTINE MODCON(IUNIT,VARI,MENG,TAUT,CSTAR,RBAR,THETAC,DCDR)
C       Modifies CONST.RLN parameters
      REAL TAUT(25),CSTAR(25),RBAR(25),THETAC(25),DCDR(25)
      CHARACTER*24 VARI
      CHARACTER*8 NAME
      CHARACTER*1 ANS
      CHARACTER*8 VARL(5),VARU(5)
      DATA VARL/'taut    ','cstar   ','rbar    ','thetac  ','dcdr    '/
      DATA VARU/'TAUT    ','CSTAR   ','RBAR    ','THETAC  ','DCDR    '/
      DO 25 J=1,MENG
       WRITE(*,'(A,I3,A\)')' Do you wish to change parameters for engine
     * #',                  J,'? '
       READ(*,'(A)')ANS
       IF(ANS.NE.'Y'.AND.ANS.NE.'y')  GO TO 25
   21  CONTINUE
       WRITE(*,*)' '
       WRITE(*,*)'   VARIABLE NAMES AND VALUES'
       WRITE(*,*)' '
       WRITE(*,'(A,1PE15.5)')' TAUT - transport lag                    ',
     *                        TAUT(J)
       WRITE(*,'(A,1PE15.5)')' CSTAR - characteristic rocket velocity ',
     *                        CSTAR(J)
       WRITE(*,'(A,1PE15.5)')' RBAR - mixture ratio                    ',
     *                        RBAR(J)
       WRITE(*,'(A,1PE15.5)')' THETAC - characteristic time constant   ',
     *                        THETAC(J)
       WRITE(*,'(A,1PE15.5)')' DCDR - d(velocity)/d(mixture ratio)     ',
     *                        DCDR(J)
       WRITE(*,*)' '
       WRITE(*,*)'   Enter variable name and new value, or'
       WRITE(*,*)'         # to print variable names & values, or'
       WRITE(*,*)'         END when all changes have been made'
       WRITE(*,*)' '
```

```
 22  CONTINUE
     WRITE(*,'(A\)')'   Enter variable name and new value, END, or # '
     CALL ZREAD(NAME,VALUE)
     IF(NAME.EQ.'#')  GO TO 21
     IF(NAME.EQ.'END'.OR.NAME.EQ.'end')  GO TO 25
     DO 23 II=1,5
      I=II
      IF(NAME.EQ.VARU(I).OR.NAME.EQ.VARL(I))  GO TO 24
 23  CONTINUE
     WRITE(*,*)'     Invalid name, try again'
     GO TO 21
 24  CONTINUE
     IF(I.EQ. 1)  TAUT(J)=VALUE
     IF(I.EQ. 2)  CSTAR(J)=VALUE
     IF(I.EQ. 3)  RBAR(J)=VALUE
     IF(I.EQ. 4)  THETAC(J)=VALUE
     IF(I.EQ. 5)  DCDR(J)=VALUE
     GO TO 22
 25 CONTINUE
    WRITE(*,'(A\)')' Do you wish to save these changes? Y or N '
    READ(*,'(A)')ANS
    IF(ANS.NE.'Y'.AND.ANS.NE.'y')  RETURN
    WRITE(*,'(A,A,A\)')' Do you wish to use file ',VARI,
   *                   '? Y or N '
    READ(*,'(A)')ANS
    IF(ANS.NE.'Y'.AND.ANS.NE.'y')  THEN
     WRITE(*,'(A\)')' Enter name of file to use '
     READ(*,'(A)')VARI
     CLOSE(UNIT=IUNIT)
     OPEN(UNIT=IUNIT,FILE=VARI)
    ELSE
     WRITE(*,'(A,A,A\)')' Do you wish to rewind ',VARI,
   *                    '? Y or N '
     READ(*,'(A)')ANS
     IF(ANS.EQ.'Y'.OR.ANS.EQ.'y')  REWIND IUNIT
    ENDIF
    DO 26 J=1,MENG
     WRITE(IUNIT,'(1P5E15.5)')TAUT(J),CSTAR(J),RBAR(J),THETAC(J),
   *                          DCDR(J)
 26 CONTINUE
    RETURN
    END
    SUBROUTINE MODENG(IUNIT,NAMENG)
C     Modifies engine parameters
    COMMON /EPARAM/MENG,TFLOW(25),PCHMB(25),DPROR(25),PMRAT(25)
    CHARACTER*24 NAMENG
    CHARACTER*8 NAME
    CHARACTER*1 ANS
    CHARACTER*8 VARL(3),VARU(3)
    DATA VARL/'tflow   ','pchmb   ','dpror   '/
    DATA VARU/'TFLOW   ','PCHMB   ','DPROR   '/
    DO 25 J=1,MENG
     WRITE(*,'(A,A,I3,A\)')' Do you wish to change flow conditions ',
```

```
*                 'for engine #',J,'? '
      READ(*,'(A)')ANS
      IF(ANS.NE.'Y'.AND.ANS.NE.'y')  GO TO 25
21    CONTINUE
      WRITE(*,*)' '
      WRITE(*,*)'   VARIABLE NAMES AND VALUES'
      WRITE(*,*)' '
      WRITE(*,'(A,1PE15.5)')' TFLOW - total mass flow (lbm/sec)      ',
     *                       TFLOW(J)
      WRITE(*,'(A,1PE15.5)')' PCHMB - chamber pressure (lbf/ft^2)    ',
     *                       PCHMB(J)
      WRITE(*,'(A,1PE15.5)')' DPROR - orfice pressure drop (lbf/ft^2)',
     *                       DPROR(J)
      WRITE(*,*)' '
      WRITE(*,*)'   Enter variable name and new value, or'
      WRITE(*,*)'         # to print variable names & values, or'
      WRITE(*,*)'         END when all changes have been made'
      WRITE(*,*)' '
22    CONTINUE
      WRITE(*,'(A\)')'   Enter variable name and new value, END, or # '
      CALL ZREAD(NAME,VALUE)
      IF(NAME.EQ.'#')  GO TO 21
      IF(NAME.EQ.'END'.OR.NAME.EQ.'end')  GO TO 25
      DO 23 II=1,3
       I=II
       IF(NAME.EQ.VARU(I).OR.NAME.EQ.VARL(I))  GO TO 24
23    CONTINUE
      WRITE(*,*)'     Invalid name, try again'
      GO TO 21
24    CONTINUE
      IF(I.EQ. 1)  TFLOW(J)=VALUE
      IF(I.EQ. 2)  PCHMB(J)=VALUE
      IF(I.EQ. 3)  DPROR(J)=VALUE
      PMRAT(J)=PCHMB(J)/TFLOW(J)
      GO TO 22
25    CONTINUE
     WRITE(*,'(A\)')' Do you wish to save these changes? Y or N '
     READ(*,'(A)')ANS
     IF(ANS.NE.'Y'.AND.ANS.NE.'y')  RETURN
     WRITE(*,'(A,A,A\)')' Do you wish to use file ',NAMENG,
    *                   '? Y or N '
     READ(*,'(A)')ANS
     IF(ANS.NE.'Y'.AND.ANS.NE.'y')  THEN
      WRITE(*,'(A\)')' Enter name of file to use '
      READ(*,'(A)')NAMENG
      CLOSE(UNIT=IUNIT)
      OPEN(UNIT=IUNIT,FILE=NAMENG)
     ELSE
      WRITE(*,'(A,A,A\)')' Do you wish to rewind ',NAMENG,
     *                   '? Y or N '
      READ(*,'(A)')ANS
      IF(ANS.EQ.'Y'.OR.ANS.EQ.'y')  REWIND IUNIT
     ENDIF
```

```
      WRITE(IUNIT,'(I5)')MENG
      DO 26 J=1,MENG
       WRITE(IUNIT,'(1P3E15.5)')TFLOW(J),PCHMB(J),DPROR(J)
   26 CONTINUE
      RETURN
      END
      SUBROUTINE MODIFY(TITL,SEGMN,SECTN,PIPE1,PIPE2,PIPE3,
     *  PIPE4,PIPE5,L,AREA,DIA,PIND,PCAP,LOPEND,LOPOLD,SPLIT,IUNIT,
     *  A,CMAN,CTANK,DENS,KMAN,KTANK,LFLOW,VOL,VOLMF,NOLINE,IENG,ITANK,
     *  AVGK,MLINE)
C       Allows modifications to input data
      COMMON /EPARAM/MENG,TFLOW(25),PCHMB(25),DPROR(25),PMRAT(25)
      COMMON /TANK/MTANK
      CHARACTER*24 NAMLIN(2)
      COMMON /WCAOUT/NAMLIN
      REAL SPLIT(25),AVGK(25)
      REAL AREA(75,25),DIA(75,25),L(75,25),PIND(75,25),
     *     PCAP(75,25)
      REAL PIPE1(75,25),PIPE2(75,25),PIPE3(75,25),PIPE4(75,25),
     *     PIPE5(75,25)
      INTEGER SEGMN(25),SECTN(75,25)
      INTEGER ITANK(25),IENG(25),LOPOLD(25),LOPEND(25),NOLINE(25)
      REAL A(25),CTANK(25),DENS(25),KTANK(25),CMAN(25),KMAN(25),
     *     LFLOW(25),VOL(25),VOLMF(25)
      CHARACTER*20 TITL
      CHARACTER*1 ANS
    1 FORMAT(1PE15.6)
    2 FORMAT(I5,1P5E15.6)
    3 FORMAT(' This segment is a bend of',1PE13.5,' deg and radius of',
     *       E13.5)
    4 FORMAT(' This segment is straight ',1PE13.5,' diameter pipe ',
     *       E13.5,' ft. long')
    5 FORMAT(' This segment is a manifold with',1PE13.5,' vol.',
     *       E13.5,' bulk modulus')
    6 FORMAT(' This segment is a pump with length =',1PE13.5,' dia =',
     *       E13.5/5X,'dp/dm =',E13.5,' capacitance =',E13.5,
     *       ' inductance =',E13.5)
    7 FORMAT(' This segment is a tuned pipe ',1PE13.5,' long & dia =',
     *       E13.5)
    8 FORMAT(' This segment is a Helmholtz resonator with'/5X,'length ='
     *       ,1PE13.5,' dia =',E13.5,' and vol =',E13.5)
    9 FORMAT(' This segment is a parallel resonator with'/5X,'length =',
     *       1PE13.5,' dia =',E13.5,' and vol =',E13.5)
   10 FORMAT(' This segment is a',1PE13.5,' long inline acc. with',
     *       ' diameter of',E13.5)
      IF(IUNIT.EQ.11)  THEN
       NAMNAM=1
      ELSE
       NAMNAM=2
      ENDIF
      WRITE(*,'(A\)')' Do you wish to change tank parameters? '
      READ(*,'(A)')ANS
      IF(ANS.EQ.'Y'.OR.ANS.EQ.'y')  THEN
```

```
      CALL MODTAN(MTANK,VOL,LFLOW,KTANK,DENS,A,CTANK)
      ENDIF
      WRITE(*,'(A\)')' Do you wish to change the pipe layout? '
      READ(*,'(A)')ANS
      IF(ANS.NE.'Y'.AND.ANS.NE.'y')  GO TO 28
      IP=0
      DO 27 M=1,MLINE
       IP=IP+1
       IT=ITANK(M)
       DO 26 IPP=IP,IP+SPLIT(M)
        I=0
        ISEGMN=SEGMN(IPP)
        DO 25 II=1,SEGMN(IPP)
         I=I+1
         IF(SECIN(I,IPP).EQ.0)  THEN
          WRITE(*,3)PIPE2(I,IPP),PIPE1(I,IPP)
         ELSEIF(SECIN(I,IPP).EQ.1)  THEN
          WRITE(*,4)PIPE2(I,IPP),PIPE1(I,IPP)
         ELSEIF(SECIN(I,IPP).EQ.2)  THEN
          WRITE(*,10)PIPE1(I,IPP),PIPE2(I,IPP)
         ELSEIF(SECIN(I,IPP).EQ.3)  THEN
          WRITE(*,7)PIPE1(I,IPP),PIPE2(I,IPP)
         ELSEIF(SECIN(I,IPP).EQ.4)  THEN
          WRITE(*,8)PIPE1(I,IPP),PIPE2(I,IPP),PIPE3(I,IPP)
         ELSEIF(SECIN(I,IPP).EQ.5)  THEN
          WRITE(*,9)PIPE1(I,IPP),PIPE2(I,IPP),PIPE3(I,IPP)
         ELSEIF(SECIN(I,IPP).EQ.6)  THEN
          WRITE(*,6)PIPE1(I,IPP),PIPE2(I,IPP),PIPE3(I,IPP),
     *              PIPE4(I,IPP),PIPE5(I,IPP)
         ELSEIF(SECIN(I,IPP).EQ.7)  THEN
          WRITE(*,5)PIPE1(I,IPP),PIPE2(I,IPP)
         ENDIF
         WRITE(*,'(A\)')' You may keep (K), modify (Y), delete (D),',
     *          ' add before (B), or add after (A)? '
         READ(*,'(A)')ANS
         IF(ANS.EQ.'A'.OR.ANS.EQ.'a')  THEN
          I=I+1
          DO 21 III=ISEGMN,I,-1
           PIPE1(III+1,IPP)=PIPE1(III,IPP)
           PIPE2(III+1,IPP)=PIPE2(III,IPP)
           PIPE3(III+1,IPP)=PIPE3(III,IPP)
           PIPE4(III+1,IPP)=PIPE4(III,IPP)
           PIPE5(III+1,IPP)=PIPE5(III,IPP)
           L(III+1,IPP)=L(III,IPP)
           DIA(III+1,IPP)=DIA(III,IPP)
           AREA(III+1,IPP)=AREA(III,IPP)
           PCAP(III+1,IPP)=PCAP(III,IPP)
           PIND(III+1,IPP)=PIND(III,IPP)
           SECIN(III+1,IPP)=SECIN(III,IPP)
21       CONTINUE
          ISEGMN=ISEGMN+1
          GO TO 24
         ELSEIF(ANS.EQ.'B'.OR.ANS.EQ.'b')  THEN
```

```
            DO 22 III=ISEGMN,I,-1
             PIPE1(III+1,IPP)=PIPE1(III,IPP)
             PIPE2(III+1,IPP)=PIPE2(III,IPP)
             PIPE3(III+1,IPP)=PIPE3(III,IPP)
             PIPE4(III+1,IPP)=PIPE4(III,IPP)
             PIPE5(III+1,IPP)=PIPE5(III,IPP)
             L(III+1,IPP)=L(III,IPP)
             DIA(III+1,IPP)=DIA(III,IPP)
             AREA(III+1,IPP)=AREA(III,IPP)
             PCAP(III+1,IPP)=PCAP(III,IPP)
             PIND(III+1,IPP)=PIND(III,IPP)
             SECTN(III+1,IPP)=SECTN(III,IPP)
   22       CONTINUE
            ISEGMN=ISEGMN+1
            GO TO 24
           ELSEIF(ANS.EQ.'D'.OR.ANS.EQ.'d')  THEN
            DO 23 III=I,ISEGMN
             PIPE1(III,IPP)=PIPE1(III+1,IPP)
             PIPE2(III,IPP)=PIPE2(III+1,IPP)
             PIPE3(III,IPP)=PIPE3(III+1,IPP)
             PIPE4(III,IPP)=PIPE4(III+1,IPP)
             PIPE5(III,IPP)=PIPE5(III+1,IPP)
             L(III,IPP)=L(III+1,IPP)
             DIA(III,IPP)=DIA(III+1,IPP)
             AREA(III,IPP)=AREA(III+1,IPP)
             PCAP(III,IPP)=PCAP(III+1,IPP)
             PIND(III,IPP)=PIND(III+1,IPP)
             SECTN(III,IPP)=SECTN(III+1,IPP)
   23       CONTINUE
            I=I-1
            ISEGMN=ISEGMN-1
            GO TO 25
           ELSEIF(ANS.NE.'Y'.AND.ANS.NE.'y')  THEN
            GO TO 25
           ENDIF
   24      CONTINUE
            WRITE(*,*)' Specify 0 for BEND,           1 for STRAIGHT pipe,'
            WRITE(*,*)'         2 for INLINE ACCUM.,  3 for TUNED STUB,'
            WRITE(*,*)'         4 for HELMHOLTZ RES., 5 for PARALLEL RES.'
            WRITE(*,*)'         6 for PUMP,           7 for MANIFOLD'
            READ(*,*) SECT
            IF(SECT.LT.0.OR.SECT.GT.7)  GO TO 24
            SECTN(I,IPP)=SECT
            IF(SECT.EQ.0)  THEN
C                bend in pipe
             WRITE(*,*)' RADIUS of bend along CL(ft), ANGLE of bend(deg),'
             WRITE(*,*)' DIAMETER(ft), and LENGTH(ft) beyond bend of pipe'
             READ(*,*)PIPE1(I,IPP),PIPE2(I,IPP),PIPE3(I,IPP),PIPE4(I,IPP)
             CALL RTYPE(SECTN(I,IPP),PIPE1(I,IPP),PIPE2(I,IPP),
     *                  PIPE3(I,IPP),PIPE4(I,IPP),PIPE5(I,IPP),L(I,IPP),
     *                  AREA(I,IPP),DIA(I,IPP),PIND(I,IPP),PCAP(I,IPP),
     *                  AVGK(M),DENS(IT),CMAN(IPP),KMAN(IPP),VOLMF(IPP))
            ELSEIF(SECT.EQ.1)  THEN
```

```
C           straight section
            WRITE(*,*)' Specify LENGTH (ft) and DIAMETER (ft) of segment'
            READ(*,*) PIPE1(I,IPP),PIPE2(I,IPP)
            CALL RTYPE(SECTN(I,IPP),PIPE1(I,IPP),PIPE2(I,IPP),
     *                 PIPE3(I,IPP),PIPE4(I,IPP),PIPE5(I,IPP),L(I,IPP),
     *                 AREA(I,IPP),DIA(I,IPP),PIND(I,IPP),PCAP(I,IPP),
     *                 AVGK(M),DENS(IT),CMAN(IPP),KMAN(IPP),VOLMF(IPP))
          ELSEIF(SECT.EQ.2)  THEN
C           inline accumulator
            WRITE(*,*)' Specify LENGTH (ft) & DIAMETER (ft) of',
     *                ' accumulator '
            READ(*,*) PIPE1(I,IPP),PIPE2(I,IPP)
            CALL RTYPE(SECTN(I,IPP),PIPE1(I,IPP),PIPE2(I,IPP),
     *                 PIPE3(I,IPP),PIPE4(I,IPP),PIPE5(I,IPP),L(I,IPP),
     *                 AREA(I,IPP),DIA(I,IPP),PIND(I,IPP),PCAP(I,IPP),
     *                 AVGK(M),DENS(IT),CMAN(IPP),KMAN(IPP),VOLMF(IPP))
          ELSEIF(SECT.EQ.3)  THEN
C           tuned stub
            WRITE(*,*)' Specify LENGTH (ft) & DIAMETER (ft) of tuned stub'
            READ(*,*)PIPE1(I,IPP),PIPE2(I,IPP)
            CALL RTYPE(SECTN(I,IPP),PIPE1(I,IPP),PIPE2(I,IPP),
     *                 PIPE3(I,IPP),PIPE4(I,IPP),PIPE5(I,IPP),L(I,IPP),
     *                 AREA(I,IPP),DIA(I,IPP),PIND(I,IPP),PCAP(I,IPP),
     *                 AVGK(M),DENS(IT),CMAN(IPP),KMAN(IPP),VOLMF(IPP))
          ELSEIF(SECT.EQ.4)  THEN
C           helmholtz resonator
            WRITE(*,*)' Specify LENGTH (ft), DIAMETER (ft) ,VOLUME ',
     *                '(ft^3) of Helmholtz Resonator'
            READ(*,*)PIPE1(I,IPP),PIPE2(I,IPP),PIPE3(I,IPP)
            CALL RTYPE(SECTN(I,IPP),PIPE1(I,IPP),PIPE2(I,IPP),
     *                 PIPE3(I,IPP),PIPE4(I,IPP),PIPE5(I,IPP),L(I,IPP),
     *                 AREA(I,IPP),DIA(I,IPP),PIND(I,IPP),PCAP(I,IPP),
     *                 AVGK(M),DENS(IT),CMAN(IPP),KMAN(IPP),VOLMF(IPP))
          ELSEIF(SECT.EQ.5)  THEN
C           parallel resonator
            WRITE(*,*)' Specify LENGTH (ft), DIAMETER (ft) ,VOLUME ',
     *                '(ft^3) of Parallel Resonator'
            READ(*,*)PIPE1(I,IPP),PIPE2(I,IPP),PIPE3(I,IPP)
            CALL RTYPE(SECTN(I,IPP),PIPE1(I,IPP),PIPE2(I,IPP),
     *                 PIPE3(I,IPP),PIPE4(I,IPP),PIPE5(I,IPP),L(I,IPP),
     *                 AREA(I,IPP),DIA(I,IPP),PIND(I,IPP),PCAP(I,IPP),
     *                 AVGK(M),DENS(IT),CMAN(IPP),KMAN(IPP),VOLMF(IPP))
          ELSEIF(SECT.EQ.6)  THEN
C           pump
            WRITE(*,*)' Specify LENGTH (ft), DIAMETER (ft) ,dp/dm, CAP.',
     *                ' & IND. of pump'
            READ(*,*)PIPE1(I,IPP),PIPE2(I,IPP),PIPE3(I,IPP),
     *               PIPE4(I,IPP),PIPE5(I,IPP)
            CALL RTYPE(SECTN(I,IPP),PIPE1(I,IPP),PIPE2(I,IPP),
     *                 PIPE3(I,IPP),PIPE4(I,IPP),PIPE5(I,IPP),L(I,IPP),
     *                 AREA(I,IPP),DIA(I,IPP),PIND(I,IPP),PCAP(I,IPP),
     *                 AVGK(M),DENS(IT),CMAN(IPP),KMAN(IPP),VOLMF(IPP))
          ELSEIF(SECT.EQ.7)  THEN
```

```
C            manifold
          WRITE(*,*)' Specify VOLUME (ft^3) and BULK MODULUS (lbf/ft^2)'
          READ(*,*) PIPE1(I,IPP),PIPE2(I,IPP)
          CALL RTYPE(SECTN(I,IPP),PIPE1(I,IPP),PIPE2(I,IPP),
     *               PIPE3(I,IPP),PIPE4(I,IPP),PIPE5(I,IPP),L(I,IPP),
     *               AREA(I,IPP),DIA(I,IPP),PIND(I,IPP),PCAP(I,IPP),
     *               AVGK(M),DENS(IT),CMAN(IPP),KMAN(IPP),VOLMF(IPP))
        ENDIF
   25   CONTINUE
        SEGMN(IPP)=ISEGMN
   26   CONTINUE
        IF(SPLIT(M).NE.0.0)  THEN
         WRITE(*,'(A,I3)')' Maximun no. of iterations is set at ',
     *                    LOPOLD(M)
         WRITE(*,'(A\)')' Do you wish to change it? '
         READ(*,'(A)')ANS
         IF(ANS.EQ.'Y'.OR.ANS.EQ.'y')  THEN
          WRITE(*,'(A\)')' Enter maximum no. of iterations '
          READ(*,*)LOPOLD(M)
         ENDIF
         LOPEND(M)=LOPOLD(M)
         IP=IP+SPLIT(M)
        ENDIF
   27 CONTINUE
   28 CONTINUE
      WRITE(*,'(A\)')' Do you wish to save these changes? Y or N '
      READ(*,'(A)')ANS
      IF(ANS.NE.'Y'.AND.ANS.NE.'y')  RETURN
      WRITE(*,'(A,A,A\)')' Do you wish to use file ',NAMLIN(NAMNAM),
     *                   '? Y or N '
      READ(*,'(A)')ANS
      IF(ANS.NE.'Y'.AND.ANS.NE.'y')  THEN
       WRITE(*,'(A\)')' Enter name of file to use '
       READ(*,'(A)')NAMLIN(NAMNAM)
       CLOSE(UNIT=IUNIT)
       OPEN(UNIT=IUNIT,FILE=NAMLIN(NAMNAM))
      ELSE
       WRITE(*,'(A,A,A\)')' Do you wish to rewind ',NAMLIN(NAMNAM),
     *                    '? Y or N '
       READ(*,'(A)')ANS
       IF(ANS.EQ.'Y'.OR.ANS.EQ.'y')  REWIND IUNIT
      ENDIF
      IP=0
      WRITE(IUNIT,'(A)')TITL
      WRITE(IUNIT,2)MTANK
      DO 29 M=1,MTANK
       WRITE(IUNIT,1)VOL(M)
       WRITE(IUNIT,1)LFLOW(M)
       WRITE(IUNIT,1)KTANK(M)
       WRITE(IUNIT,1)DENS(M)
   29 CONTINUE
      WRITE(IUNIT,2)MLINE
      DO 33 M=1,MLINE
```

```
      IP=IP+1
      WRITE(IUNIT,2)ITANK(M)
      WRITE(IUNIT,2)IENG(IP)
      WRITE(IUNIT,2)SEGMN(IP)
      WRITE(IUNIT,2)SPLIT(M)
      DO 30 J=1,SEGMN(IP)
       WRITE(IUNIT,2)SECIN(J,IP),PIPE1(J,IP),PIPE2(J,IP),PIPE3(J,IP),
     *                PIPE4(J,IP),PIPE5(J,IP)
   30 CONTINUE
      IF(SPLIT(M).EQ.0)  GO TO 33
      DO 32 K=1,SPLIT(M)
       IP=IP+1
       WRITE(IUNIT,2)SEGMN(IP)
       WRITE(IUNIT,2)NOLINE(IP)
       WRITE(IUNIT,2)IENG(IP)
       DO 31 J=1,SEGMN(IP)
        WRITE(IUNIT,2)SECIN(J,IP),PIPE1(J,IP),PIPE2(J,IP),PIPE3(J,IP),
     *                PIPE4(J,IP),PIPE5(J,IP)
   31  CONTINUE
   32 CONTINUE
   33 CONTINUE
      RETURN
      END
      SUBROUTINE MODTAN(MTANK,VOL,LFLOW,KTANK,DENS,A,CTANK)
C       Modifies tank parameters
      REAL VOL(25),LFLOW(25),KTANK(25),DENS(25),A(25),CTANK(25)
      CHARACTER*1 ANS
      CHARACTER*8 NAME
      CHARACTER*8 VARL(4),VARU(4)
      DATA VARL/'vol     ','lflow   ','ktank   ','dens    '/
      DATA VARU/'VOL     ','LFLOW   ','KTANK   ','DENS    '/
      DATA GRAV/32.2/
      DO 25 J=1,MTANK
       WRITE(*,'(A,I3,A\)')' Do you wish to change parameters for tank #
     *',J,'? '
       READ(*,'(A)')ANS
       IF(ANS.NE.'Y'.AND.ANS.NE.'y')  GO TO 25
   21  CONTINUE
       WRITE(*,*)' '
       WRITE(*,*)'   VARIABLE NAMES AND VALUES'
       WRITE(*,*)' '
       WRITE(*,'(A,1PE15.5)')' VOL - volume (ft^3)                    ',
     *                        VOL(J)
       WRITE(*,'(A,1PE15.5)')' LFLOW - mass flow in pipe (lbm/sec^2) ',
     *                        LFLOW(J)
       WRITE(*,'(A,1PE15.5)')' KTANK - tank bulk modulus (lbf/ft^2)  ',
     *                        KTANK(J)
       WRITE(*,'(A,1PE15.5)')' DENS - density )lbm/ft^3)             ',
     *                        DENS(J)
       WRITE(*,*)' '
       WRITE(*,*)'   Enter variable name and new value, or'
       WRITE(*,*)'         # to print variable names & values, or'
       WRITE(*,*)'         END when all changes have been made'
```

```
      WRITE(*,*)' '
   22 CONTINUE
      WRITE(*,'(A\)')'   Enter variable name and new value, END, or # '
      CALL ZREAD(NAME,VALUE)
      IF(NAME.EQ.'#')  GO TO 21
      IF(NAME.EQ.'END'.OR.NAME.EQ.'end')  GO TO 25
      DO 23 II=1,4
       I=II
       IF(NAME.EQ.VARU(I).OR.NAME.EQ.VARL(I))  GO TO 24
   23 CONTINUE
      WRITE(*,*)'     Invalid name, try again'
      GO TO 21
   24 CONTINUE
      IF(I.EQ. 1)  VOL(J)=VALUE
      IF(I.EQ. 2)  LFLOW(J)=VALUE
      IF(I.EQ. 3)  KTANK(J)=VALUE
      IF(I.EQ. 4)  DENS(J)=VALUE
      A(J)=SQRT(GRAV*KTANK(J)/DENS(J))
      CTANK(J)=DENS(J)*VOL(J)/KTANK(J)
      GO TO 22
   25 CONTINUE
      RETURN
      END
      SUBROUTINE NYQUIS(GF,GOX,S,TAUT,CSTAR,RBAR,DCDR,THETAC,IFUEL,ILOX)
C       Computes the K()'s
      COMPLEX GF(25),GOX(25),KG1(25),KG2,KG3,S
      REAL THETAC(25),RBAR(25),CSTAR(25),DCDR(25),TAUT(25)
      REAL K1R(25),K2R(25),K3R(25),K1C(25),K2C(25),K3C(25),
     *     K4R(25),K4C(25)
      COMMON /EPARAM/MENG,TFLOW(25),PCHMB(25),DPROR(25),PMRAT(25)
      INTEGER SEGMN,SECIN(150)
      COMMON /SETUP/PIPE1(150),PIPE2(150),PIPE3(150),PIPE4(150),
     *              NENGF(25),NTANKF(25),NLINEF(25),NSPF(25),ILINEF,
     *              NENGO(25),NTANKO(25),NLINEO(25),NSPO(25),ILINEO,
     *              SEGMN,SECIN
      COMMON /FACTOR/SFAC
      DO 21 I=1,MENG
       KG1(I)=2.0*CEXP(-S*TAUT(I))/(THETAC(I)*S +1.0)
       K1C(I)=AIMAG(KG1(I))
       K1R(I)=REAL(KG1(I))
   21 CONTINUE
      IF(ILOX.EQ.0)  THEN
       DO 22 I=1,ILINEO
        K=NENGO(I)
        KG2=0.5*KG1(K)*((1.0+(1.0+RBAR(K))*DCDR(K)/CSTAR(K))*GOX(I))
        K2C(I)=AIMAG(KG2)
        K2R(I)=REAL(KG2)
   22  CONTINUE
      ENDIF
      IF(IFUEL.EQ.0)  THEN
       DO 23 I=1,ILINEF
        K=NENGF(I)
        KG3=0.5*KG1(K)*((1.0-RBAR(K)*(1.0+RBAR(K))*DCDR(K)/CSTAR(K))*
```

```
     *      GF(I))
            K3C(I)=AIMAG(KG3)
            K3R(I)=REAL(KG3)
   23     CONTINUE
          ENDIF
          IF(ILOX.EQ.0.AND.IFUEL.EQ.0)  THEN
           L=0
           DO 25 I=1,ILINEF
            DO 24 J=1,ILINEO
             IF(NENGF(I).EQ.NENGO(J))  THEN
              L=L+1
              K4C(L)=K2C(J)+K3C(I)
              K4R(L)=K2R(J)+K3R(I)
             ENDIF
   24       CONTINUE
   25      CONTINUE
          ENDIF
          WRITE(17)K1R,K1C,K2R,K2C,K3R,K3C,K4R,K4C
          DO 26 I=1,MENG
           IF(IFUEL.NE.0.AND.ILOX.NE.0)
     *     WRITE(14,'(1P3E12.4)')AIMAG(S)/SFAC,K1R(I),K1C(I)
           IF(IFUEL.EQ.0.AND.ILOX.NE.0)  WRITE(14,'(1P3E12.4,I5,1X,1P2E12.4)
     *')  AIMAG(S)/SFAC,K1R(I),K1C(I),I,K3R(I),K3C(I)
           IF(IFUEL.NE.0.AND.ILOX.EQ.0)  WRITE(14,'(1P3E12.4,I5,1X,1P2E12.4)
     *')  AIMAG(S)/SFAC,K1R(I),K1C(I),I,K2R(I),K2C(I)
           IF(IFUEL.EQ.0.AND.ILOX.EQ.0)  THEN
            WRITE(14,'(1P3E12.4)')AIMAG(S)/SFAC,K1R(I),K1C(I)
            WRITE(14,'(I7,1P6E12.4)')I,K2R(I),K2C(I),K3R(I),K3C(I),K4R(I),
     *     K4C(I)
           ENDIF
   26     CONTINUE
          RETURN
          END
          SUBROUTINE RLINE(TITL,SEGMN,SECIN,PIPE1,PIPE2,PIPE3,
     *     PIPE4,PIPE5,L,AREA,DIA,PIND,PCAP,LOPEND,LOPOLD,SPLIT,IUNIT,
     *     A,CMAN,CTANK,DENS,KMAN,KTANK,LFLOW,VOL,VOLMF,NOLINE,IENG,ITANK,
     *     AVGK,MLINE)
C         Reads fuel or lox file.
          COMMON /EPARAM/MENG,TFLOW(25),PCHMB(25),DPROR(25),PMRAT(25)
          COMMON /TANK/MTANK
          REAL SPLIT(25),AVGK(25)
          REAL AREA(75,25),DIA(75,25),L(75,25),PIND(75,25),
     *     PCAP(75,25)
          REAL PIPE1(75,25),PIPE2(75,25),PIPE3(75,25),PIPE4(75,25),
     *     PIPE5(75,25)
          INTEGER SEGMN(25),SECIN(75,25)
          INTEGER ITANK(25),IENG(25),LOPOLD(25),LOPEND(25),NOLINE(25)
          REAL A(25),CMAN(25),CTANK(25),DENS(25),KMAN(25),KTANK(25),
     *     LFLOW(25),VOL(25),VOLMF(25)
          CHARACTER*20 TITL
          CHARACTER*1 ANS
          READ(IUNIT,'(A)')TITL
          CALL TANKNO(MTANK,VOL,LFLOW,KTANK,DENS,A,CTANK,IUNIT)
```

```
      READ(IUNIT,*)MLINE
      IF(MLINE.GT.25)  THEN
       WRITE(*,*)' Number of lines must be less than 25'
       STOP
      ENDIF
      IF(MLINE.LE.0)  MLINE=1
      DO 20 M=1,25
       IENG(M)=0
   20 CONTINUE
      M=0
      DO 24 MM=1,MLINE
       M=M+1
       READ(IUNIT,*)ITANK(MM),IENG(M)
       IF(ITANK(MM).GT.MTANK)  THEN
        WRITE(*,*)'  Invalid tank number.'
        STOP
       ENDIF
       IF(IENG(M).GT.MENG)  THEN
        WRITE(*,*)'  Invalid engine number.'
        STOP
       ENDIF
       IT=ITANK(MM)
       IE=IENG(M)
       LOPOLD(MM)=20
       LOPEND(MM)=1
       AVGK(MM)=0.0
       DIVAVG=0.0
       READ(IUNIT,*)SEGMN(M),SPLIT(MM)
       DO 21 I=1,SEGMN(M)
        READ(IUNIT,*)SECIN(I,M),PIPE1(I,M),PIPE2(I,M),PIPE3(I,M),
     *               PIPE4(I,M),PIPE5(I,M)
        IF(SECIN(I,M).NE.7)  GO TO 21
        AVGK(MM)=AVGK(MM)+PIPE2(I,M)
        DIVAVG=DIVAVG+1
   21 CONTINUE
       IF(SPLIT(MM).EQ.0)  THEN
        AVGK(MM)=KTANK(IT)
        NOLINE(M)=1
        GO TO 24
       ENDIF
C             split pipe
       DO 23 J=1,SPLIT(MM)
        M=M+1
        READ(IUNIT,*)SEGMN(M),NOLINE(M),IENG(M)
        IF(IENG(M).GT.MENG)  THEN
         WRITE(*,*)'  Invalid engine number.'
         STOP
        ENDIF
        IE=IENG(M)
        IF(NOLINE(M).EQ.0)  NOLINE(M)=1
        DO 22 I=1,SEGMN(M)
         READ(IUNIT,*)SECIN(I,M),PIPE1(I,M),PIPE2(I,M),PIPE3(I,M),
     *               PIPE4(I,M),PIPE5(I,M)
```

```
      IF(SECIN(I,M).NE.7)  GO TO 22
      AVGK(MM)=AVGK(MM)+PIPE2(I,M)*NOLINE(M)
      DIVAVG=DIVAVG+NOLINE(M)
   22  CONTINUE
   23 CONTINUE
      WRITE(*,'(A,I3)')' Max. no. of iterations is set at ',
     *                  LOPOLD(MM)
      WRITE(*,'(A\)')' Do you wish to change it? '
      READ(*,'(A)')ANS
      IF(ANS.EQ.'Y'.OR.ANS.EQ.'y')  THEN
       WRITE(*,'(A\)')' Enter maximum no. of iterations '
       READ(*,*)LOPOLD(MM)
      ENDIF
      LOPEND(MM)=LOPOLD(MM)
      IF(DIVAVG.LE.0.0)  DIVAVG=1.0
      AVGK(MM)=KTANK(IT)+AVGK(MM)/DIVAVG
   24 CONTINUE
      M=0
      DO 28 MM=1,MLINE
       M=M+1
       IT=ITANK(MM)
       IE=IENG(M)
       DO 25 I=1,SEGMN(M)
        CALL RTYPE(SECIN(I,M),PIPE1(I,M),PIPE2(I,M),
     *             PIPE3(I,M),PIPE4(I,M),PIPE5(I,M),L(I,M),AREA(I,M),
     *             DIA(I,M),PIND(I,M),PCAP(I,M),AVGK(MM),DENS(IT),
     *             CMAN(M),KMAN(M),VOLMF(M))
   25 CONTINUE
       IF(SPLIT(MM).EQ.0)  GO TO 28
       DO 27 J=1,SPLIT(MM)
        M=M+1
        IE=IENG(M)
        DO 26 I=1,SEGMN(M)
         CALL RTYPE(SECIN(I,M),PIPE1(I,M),PIPE2(I,M),
     *              PIPE3(I,M),PIPE4(I,M),PIPE5(I,M),L(I,M),AREA(I,M),
     *              DIA(I,M),PIND(I,M),PCAP(I,M),AVGK(MM),DENS(IT),
     *              CMAN(M),KMAN(M),VOLMF(M))
   26 CONTINUE
   27 CONTINUE
   28 CONTINUE
      RETURN
      END
      SUBROUTINE RTYPE(SECIN,PIPE1,PIPE2,PIPE3,PIPE4,PIPE5,L,AREA,DIA,
     *                 PIND,PCAP,AVGK,DENS,CMAN,KMAN,VOLMF)
C       Stores values for different types of piping
      INTEGER SECIN
      REAL L,KMAN
      DATA GRAV/32.2/,PI/3.141593/
      IF(SECIN.EQ.0)  THEN
       CALL BENDS(PIPE1,PIPE2,PIPE3,PIPE4,VALUE,DIME)
         AREAB=0.785398*DIME**2
         L=VALUE
         AREA=AREAB
```

```
      DIA=DIME
     ELSEIF(SECTN.EQ.1)  THEN
C          straight section
      VALUE=PIPE1
      DIME=PIPE2
      AREAB=0.785398*DIME**2
      L=VALUE
      AREA=AREAB
      DIA=DIME
     ELSEIF(SECTN.EQ.2)  THEN
C          inline accumulator
C        PIPE1 - LEN
C        PIPE2 - DIA
C        PIPE3 - DEN
C        PIPE4 - K
      L=PIPE1
      DIA=PIPE2
      AREA=0.25*PI*PIPE2**2
      IF(PIPE3.EQ.0.0)  PIPE3=DENS
      IF(PIPE4.EQ.0.0)  PIPE4=AVGK
      PCAP=PIPE3*L*AREA/PIPE4
     ELSEIF(SECTN.EQ.3)  THEN
C          tuned stub - suppresses omega = (PI/2)/(L*SQRT(PIND*PCAP))
C        PIPE1 - LEN
C        PIPE2 - DIA
C        PIPE3 - DEN
C        PIPE4 - K
      L=PIPE1
      DIA=PIPE2
      AREA=0.25*PI*DIA**2
      IF(PIPE3.EQ.0.0)  PIPE3=DENS
      IF(PIPE4.EQ.0.0)  PIPE4=AVGK
      PCAP=PIPE3*L*AREA/PIPE4
      PIND=L/(AREA*GRAV)
     ELSEIF(SECTN.EQ.4.OR.SECTN.EQ.5)  THEN
C          helmholtz resonator or parallel resonator
C            suppresses omega = 1/SQRT(PIND*PCAP)
C        PIPE1 - LEN
C        PIPE2 - DIA
C        PIPE3 - VOL
C        PIPE4 - DEN
C        PIPE5 - K
      L=PIPE1
      DIA=PIPE2
      AREA=PIPE3
      IF(PIPE4.EQ.0.0)  PIPE4=DENS
      IF(PIPE5.EQ.0.0)  PIPE5=AVGK
      PCAP=PIPE4*AREA/PIPE5
      PIND=L/(0.25*PI*DIA**2*GRAV)
     ELSEIF(SECTN.EQ.6)  THEN
C          pump
C        PIPE1 - LEN
C        PIPE2 - DIA
```

```
C          PIPE3 - DP/DM
C          PIPE4 - IND
C          PIPE5 - CAP
          L=PIPE1
          DIA=PIPE2
          AREA=PIPE3
          PCAP=PIPE4
          PIND=PIPE5
         ELSEIF(SECTN.EQ.7)  THEN
C             manifold
C          PIPE1 - VOLMF
C          PIPE2 - KMAN
          VOLMF=PIPE1
          KMAN=PIPE2
          CMAN=DENS*VOLMF/KMAN
          L=VOLMF
          DIA=CMAN
         ENDIF
       RETURN
       END
       SUBROUTINE STSECT(J,ITYPE,POINT,LEN,DIA)
C        Computes plot coordinates for a straight section
       COMMON /PIPPXY/X,XH,XL,Y,YH,YL,XMIN,XMAX,YMIN,YMAX,SINA,COSA
       REAL LEN,POINT(8,200)
       INTEGER*2 ITYPE(200)
       J=J+1
       ITYPE(J)=1
       XH=X-0.5*SINA*DIA
       XL=X+0.5*SINA*DIA
       YH=Y+0.5*COSA*DIA
       YL=Y-0.5*COSA*DIA
       POINT(1,J)=XH
       POINT(2,J)=YH
       POINT(3,J)=XL
       POINT(4,J)=YL
       X=X+COSA*LEN
       XH=X-0.5*SINA*DIA
       XL=X+0.5*SINA*DIA
       Y=Y+SINA*LEN
       YH=Y+0.5*COSA*DIA
       YL=Y-0.5*COSA*DIA
       POINT(5,J)=XH
       POINT(6,J)=YH
       POINT(7,J)=XL
       POINT(8,J)=YL
       XMIN=AMIN1(X,XL,XH,XMIN)
       XMAX=AMAX1(X,XL,XH,XMAX)
       YMIN=AMIN1(Y,YL,YH,YMIN)
       YMAX=AMAX1(Y,YL,YH,YMAX)
       RETURN
       END
       SUBROUTINE TANKNO(MTANK,VOL,LFLOW,KTANK,DENS,A,CTANK,IUNIT)
C        Reads tank parameters
```

```
      REAL VOL(25),LFLOW(25),KTANK(25),DENS(25),A(25),CTANK(25)
      DATA GRAV/32.2/
      READ(IUNIT,*)MTANK
      IF(MTANK.GT.25)  THEN
       WRITE(*,*)' Number of tanks must be less than 25'
       STOP
      ENDIF
      IF(MTANK.LE.0)  MTANK=1
      DO 21 I=1,MTANK
      READ(IUNIT,*)VOL(I),LFLOW(I),KTANK(I),DENS(I)
      A(I)=SQRT(GRAV*KTANK(I)/DENS(I))
      CTANK(I)=DENS(I)*VOL(I)/KTANK(I)
   21 CONTINUE
      RETURN
      END
      SUBROUTINE TSSECT(J,ITYPE,POINT,LEN,DIA)
C       Computes plot coordinates for a tuned stub
      COMMON /PIPPXY/X,XH,XL,Y,YH,YL,XMIN,XMAX,YMIN,YMAX,SINA,COSA
      REAL LEN,POINT(8,200)
      INTEGER*2 ITYPE(200)
      J=J+1
      ITYPE(J)=1
      DIAM=SQRT((XH-XL)**2+(YH-YL)**2)
      XH=X-SINA*(LEN+0.5*DIAM)
      YH=Y+COSA*(LEN+0.5*DIAM)
      POINT(1,J)=XH
      POINT(2,J)=YH
      POINT(3,J)=XL
      POINT(4,J)=YL
      X=X+COSA*DIA
      XH=X-SINA*(LEN+0.5*DIAM)
      XL=XL+COSA*DIA
      Y=Y+SINA*DIA
      YH=Y+COSA*(LEN+0.5*DIAM)
      YL=YL+SINA*DIA
      POINT(5,J)=XH
      POINT(6,J)=YH
      POINT(7,J)=XL
      POINT(8,J)=YL
      XMIN=AMIN1(X,XL,XH,XMIN)
      XMAX=AMAX1(X,XL,XH,XMAX)
      YMIN=AMIN1(Y,YL,YH,YMIN)
      YMAX=AMAX1(Y,YL,YH,YMAX)
      RETURN
      END
      SUBROUTINE ZREAD(NAME,VALUE)
C       Reads input for input modification
      CHARACTER*1 NAME(8)
      CHARACTER*1 CARD(80),PLUS,MINUS,PERIOD,LE,E,NUMBER(10)
      CHARACTER*1 LEND(3),CEND(3),POUND,QUEST,BLK,COMMA
      CHARACTER*1 LTIT(5),CTIT(5)
      CHARACTER*80 DCARD
      EQUIVALENCE (CARD(1),DCARD)
```

```
      DATA PLUS/'+'/,MINUS/'-'/,PERIOD/'.'/,LE/'e'/,E/'E'/,BLK/' '/
      DATA NUMBER/'0','1','2','3','4','5','6','7','8','9'/,COMMA/','/
      DATA LEND/'e','n','d'/,CEND/'E','N','D'/,POUND/'#'/,QUEST/'?'/
      DATA LTIT/'t','i','t','l','e'/,CTIT/'T','I','T','L','E'/
    1 FORMAT(A)
      DO 21 I=1,8
       NAME(I)=BLK
   21 CONTINUE
      READ(*,1)DCARD
      IF(CARD(1).EQ.POUND)  THEN
       NAME(1)=POUND
       RETURN
      ENDIF
      IF(CARD(1).EQ.QUEST)  THEN
       NAME(1)=QUEST
       RETURN
      ENDIF
      DO 22 I=1,3
       IF(CARD(I).NE.LEND(I).AND.CARD(I).NE.CEND(I))  GO TO 23
       NAME(I)=CEND(I)
   22 CONTINUE
      RETURN
   23 CONTINUE
      DO 24 I=1,5
       IF(CARD(I).NE.LTIT(I).AND.CARD(I).NE.CTIT(I))  GO TO 25
       NAME(I)=CTIT(I)
   24 CONTINUE
      RETURN
   25 CONTINUE
      DO 26 I=1,8
       II=I
       IF(CARD(I).EQ.BLK.OR.CARD(I).EQ.COMMA)  GO TO 27
       NAME(I)=CARD(I)
   26 CONTINUE
   27 CONTINUE
      DO 28 I=II,80
       ID=I
       IF(CARD(I).NE.BLK.AND.CARD(I).NE.COMMA)  GO TO 29
   28 CONTINUE
      VALUE=0.0
      WRITE(*,*)'   No value given, ZERO assumed'
      RETURN
   29 CONTINUE
      SIGN=1.0
      IF(CARD(ID).EQ.MINUS)  THEN
       SIGN=-1.0
       ID=ID+1
      ELSEIF(CARD(ID).EQ.PLUS)  THEN
       ID=ID+1
      ENDIF
      WHOLE=0.0
      DO 32 I=ID,80
       II=I
```

```
      IF(CARD(I).EQ.PERIOD)  GO TO 33
      IF(CARD(I).EQ.PLUS)  GO TO 38
      IF(CARD(I).EQ.MINUS)  GO TO 38
      IF(CARD(I).EQ.E.OR.CARD(I).EQ.LE)  GO TO 37
      DO 30 J=1,10
       JJ=J-1
       IF(CARD(I).EQ.NUMBER(J))  GO TO 31
30 CONTINUE
      VALUE=SIGN*WHOLE
      IF(CARD(I).EQ.BLK)  RETURN
      WRITE(*,*)'  Input error, value set to ZERO'
      VALUE=0.0
      RETURN
31 CONTINUE
      WHOLE=WHOLE*10.0+JJ
32 CONTINUE
     VALUE=SIGN*WHOLE
     RETURN
33 CONTINUE
     ID=II+1
     FRACT=0.0
     ICOUNT=0
     DO 36 I=ID,80
      ICOUNT=ICOUNT+1
      II=I
      IF(CARD(I).EQ.PERIOD)  THEN
       WRITE(*,*)'  Input error, value set to ZERO'
       VALUE=0.0
       RETURN
      ENDIF
      IF(CARD(I).EQ.PLUS)  GO TO 38
      IF(CARD(I).EQ.MINUS)  GO TO 38
      IF(CARD(I).EQ.E.OR.CARD(I).EQ.LE)  GO TO 37
      DO 34 J=1,10
       JJ=J-1
       IF(CARD(I).EQ.NUMBER(J))  GO TO 35
34 CONTINUE
      VALUE=SIGN*(WHOLE+FRACT)
      IF(CARD(I).EQ.BLK)  RETURN
      WRITE(*,*)'  Input error, value set to ZERO'
      VALUE=0.0
      RETURN
35 CONTINUE
      FRACT=FRACT+JJ/10.0**ICOUNT
36 CONTINUE
     VALUE=SIGN*(WHOLE+FRACT)
     RETURN
37 CONTINUE
     II=II+1
38 CONTINUE
     VALUE=SIGN*(WHOLE+FRACT)
     SIGN=1.0
     IF(CARD(II).EQ.MINUS)  THEN
```

```
      SIGN=-1.0
      II=II+1
     ELSEIF(CARD(II).EQ.PLUS)  THEN
      II=II+1
     ENDIF
     WHOLE=0.0
     DO 41 I=II,80
      DO 39 J=1,10
       JJ=J-1
       IF(CARD(I).EQ.NUMBER(J))  GO TO 40
39   CONTINUE
      VALUE=VALUE*10.0**(SIGN*WHOLE)
      IF(CARD(I).EQ.BLK)  RETURN
      WRITE(*,*)'  Input error, value set to ZERO'
      VALUE=0.0
      RETURN
40   CONTINUE
      WHOLE=WHOLE*10.0+JJ
41   CONTINUE
     VALUE=VALUE*10.0**(SIGN*WHOLE)
     RETURN
     END
```