NASA Contractor Report 181694

WALL ADJUSTMENT STRATEGY SOFTWARE FOR USE WITH THE NASA LANGLEY 0.3-METER TRANSONIC CRYOGENIC TUNNEL ADAPTIVE WALL TEST SECTION

Stephen W. D. Wolf

Vigyan Research Associates, Inc.
Hampton, Virginia

NAS1-17919
NOVEMBER 1988

National Aeronautics and
Space Administration

Langley Research Center
Hampton, Virginia 23665

ABSTRACT

This Wall Adjustment Strategy (WAS) software provides successful on-line control of the 2-D flexible walled test section of the Langley 0.3-m Transonic Cryogenic Tunnel. This software package allows the level of operator intervention to be regulated as necessary for research and production type 2-D testing using an Adaptive Wall Test Section (AWTS). The software is designed to accept modification for future requirements, such as 3-D testing, with a minimum of complexity. The WAS software described here is a unique attempt to provide a *user friendly* package which could be used to control any flexible walled AWTS. Control system constraints influence the details of data transfer, not the data type. Consequently, this entire software package could be used in different control systems, if suitable interface software is available.

A complete overview of the software highlights the data flow paths, the modular architecture of the software and the various operating and analysis modes available. A detailed description of the software modules includes listings of the code. We provide a user's manual to explain task generation, operating environment, user options and what to expect at execution. A typical output listing is shown as part of one of the software tests described in the text. Necessary utility software for datafile management is only briefly described, since this software is system oriented.

The WAS software forms the major component of any AWTS control system. We intend this report to help those who need to know the details of controlling an AWTS with flexible walls and those involved with WAS software maintenance.

CONTENTS

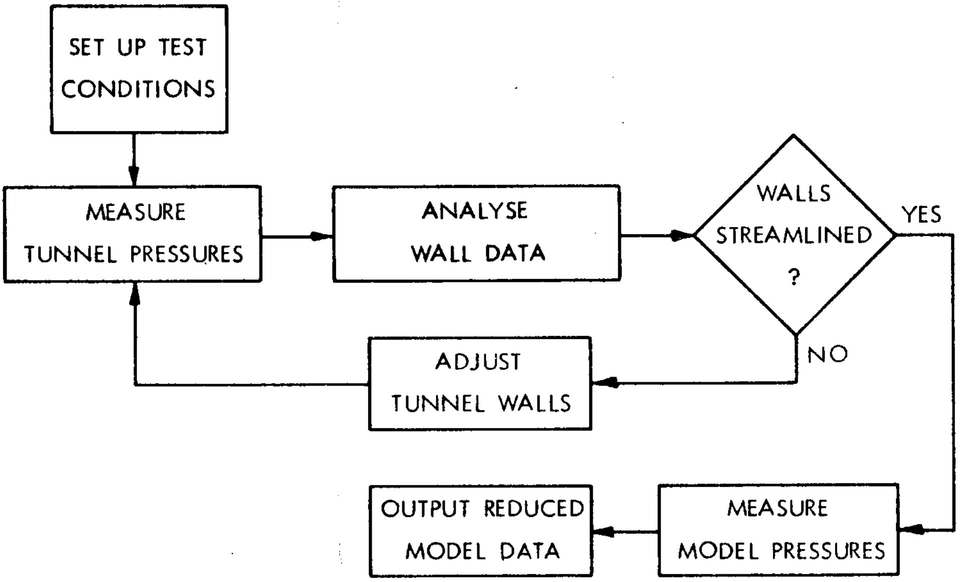
CONTENTS

APPENDICES

1. INTRODUCTION

The Wall Adjustment Strategy (WAS) software describe here is for on-line control of the Adaptive Wall Test Section[1] (AWTS) in the NASA Langley 0.3-meter Transonic Cryogenic Tunnel (TCT) for two-dimensional testing. We have based this code on control software[2] extensively proven with the Transonic Self-Streamlining Wind Tunnel (TSWT) at the University of Southampton, UK.[3,4] The WAS software fulfills the following requirements:

1) To determine the shapes of the test section walls, for each data point, by the necessary sequencing of events for a wall streamlining/adaptation cycle as shown below:



2) To provide various selectable analyses of the wall data including prediction of new wall shapes to minimize boundary interference.
3) To operate in various selectable modes to accommodate production testing and research needs.
4) To store raw and reduced tunnel data on disk as required.
5) To provide selection of initial wall contours independent of actual tunnel test conditions.
6) To allow critical computation factors to be easily assigned new values.

The integration of a flexible walled test section in a continuous flow cryogenic tunnel is unique in the 0.3-m TCT. This software was primarily a research tool to allow the tunnel operators to investigate the capabilities of this unique facility. However, as we have gained experience with the AWTS, we have made unique efforts to ensure that non-specialist operators can run the software for "routine" airfoil testing within a known test envelope. These demands together with system hardware and software constraints have produced the operator interface described in this work.

The software is designed to accept modification, such as for 3-D testing, with a minimum of complexity. We achieve this desirable situation by use of a modular architecture. The WAS software described here is a unique attempt to provide a *user friendly* package which could be used with any flexible walled AWTS. We have generalized the WAS software to assist those engineers who may wish to use this software to control other similar type AWTS. (The essentials of the AWTS hardware are outlined in reference 5.) For example, all the geometric data about the 0.3-m TCT AWTS is written into the code using informative variable and array names. Fortunately, control system constraints only influence the details of data transfer not the data type. Consequently, this entire software package could be used in different control systems, if suitable interface software is made available. The WAS software forms the major component of any AWTS control system. We do not describe the other system dependent components here. What we do describe though is the data interface that the WAS software has with these other components. We intend this report to assist those who need to know the details of controlling an AWTS with flexible walls, and also those involved with WAS software maintenance.

The WAS software is necessarily complex and we attempt to unravel this complexity in an overview found in section 2. The overview highlights the data flow paths, the modular architecture of the software and the various operating and analysis modes available. More information on the modes of analysis for different testing requirements is found in section 3. We briefly describe the associated computations in section 4, more information is available in the literature.

We give a detailed description of the WAS software in section 5, with code listings in Appendix A. Definitions, declarations and equivalences of the data commons are contained in source files actually included in the WAS software sources. We provide code listings of these included sources in Appendix B.

In section 6, we include a *User's Manual* for the WAS software. In sub-section 6.1, we explain how to generate the executable task/module, called FLXWAS, should changes to the WAS software be necessary. Sub-section 6.2 of the manual provides detailed information on the complex operating environment for FLXWAS. The correct environment must be setup if the task is to function successfully. In sub-section 6.3, there is an overview of the user options available with FLXWAS. We give examples of how these options allow a knowledgeable operator to change the wall adaptation (testing technique) to accommodate "non-routine" testing. We explain execution instructions in sub-section 6.4 which includes an explanation of what to expect from the WAS software at execution time. If errors are flagged during execution, the Appendix C contains abort procedures for error recovery.

We describe numerous software computation tests in section 7, with one test case shown to illustrate what the typical output from the WAS software looks like for one iteration. This test case is a good check of any software modifications. We describe the software necessary to support some of these tests in sub-section 7.2, with code listings in Appendix F.

In section 8, we briefly describe utility software necessary for management of the three datafiles used by the WAS software. These routines provide essential off-line back-up to the WAS software and are system orientated. We explain how to use these routines for various datafile manipulation and documentation procedures in Appendix D and E.

2. OVERVIEW

We have written the WAS software to operate with existing 0.3-m TCT real-time data acquisition, control and plotting tasks (see **Programmer's Refrence Manual for the 0.3-meter Transonic Cryogenic Tunnel Modcomp CPU-A Flexwall Control System**) on the Modcomp Classic 7863 mini-computer (called CPU-A). The Modcomp Classic runs under the MAX IV multi-tasking operating system. We have written the software in the FORTRAN IV language.

The successful operation of the WAS software requires a continual exchange of information between the wind tunnel and the wall adaptation control computer (CPU-A) via many data paths including the tunnel data acquisition computer (another Modcomp Classic mini-computer called CPU-B). We show the flexwall control system hardware on Figure 1. Meanwhile, the flexwall control system software has a data path chart for the task of controlling the flexible wall shapes as shown on Figure 2. We can breakdown this control task into its functional elements as shown on Figure 3. These functional elements are *Determine Wall Shapes; Move Walls; Acquire and Translate Tunnel and Wall Data; Monitor Wall Shapes.* The WAS software forms the most important functional element of the task, namely *Determine Wall Shapes.* The other functional elements are not described in this report.

We show the WAS software interface with the flexwall control system software and the CPU-A system on Figure 4. The WAS software communicates with the flexwall control system software via a data common called FLXCOM (a private shared region listed in Appendix B). For example, the WAS software outputs the destination wall shapes by loading array JPOSN (located in FLXCOM). Then, we initiate wall movements by use of control bits in the status words FLXISW and FSSTAT (both located in FLXCOM). In addition, we pass data between the two Modcomp mini-computers (CPU-A and CPU-B) by the MAXNET link which uses the Communications Control Word IACCW, the Communications Status Word IASTAT and the move status bit in FLXISW. Tunnel data from CPU-B is loaded into variables included in FLXCOM. Unfortunately, some exceptions to this rule exist. For example, the WAS software initiates the acquisition of the wall pressures by calls to the flexwall software subroutine FRQST. Also, the WAS software reads the digital constants (Analysis mode designator and computational factors) through a system data common called OAPCOM by using calls to a system subroutine DIGICO. Other system subroutines called are SETB, TSTB, RLTBZ4, DELAY, T2MCAL, TIMEOUT and TIME as shown on Figure 4.

In order to perform research and production type testing in the 0.3-m TCT, we have configured the WAS software with several analysis and operating modes (See sub-section 6.3). We have supplemented a normal mode of analysis for routine 2-D airfoil testing with three other modes of analysis. One mode is for system development purposes during wall adaptation research or tunnel check-out, when it is useful to have extra real-time data available. Another is for empty test section or straight wall streamlining used during test section calibration. Also, a mode exists to allow off-line re-analysis of raw wall and tunnel data to allow software checking (See section 3).

Of the several modes of operation, one is a normal or default mode with flexwall control task word options CONTINUE, NOWAIT and NOPAUSE selected. Then a mode exists where completion of a streamlining cycle is at the discretion of the operator using the option PAUSE. When testing at high subsonic Mach numbers, the test conditions may be significantly affected by each change of wall shapes. So a mode exists to allow the tunnel operator to check the test conditions before acquiring each set of wall pressures. We select this mode with the option WAIT. Finally, a mode exists to allow the tunnel user to investigate the effects of wall adaptation on the model pressures. So, in this mode, we acquire model data for each iteration of a streamlining cycle by selecting the option SINGLE for single iteration mode. We can select various combinations of the operating modes (See sub-section 6.3). On Figure 5, we show how these operating modes affect the overall sequence of events necessary for determining streamline wall shapes. Please note, we achieve event sequencing between CPU-A and CPU-B by pausing the executable module FLXWAS on CPU-A with calls to the system subroutine WAIT4.

The WAS software is written using a versatile modular architecture with the program segmented into a collection of ten logical modules. A review of the software modules is shown below:

WAS Software Modules

File Type & Name	Function
Main Program (FLXWAS)	i) Sequences streamlining cycle events for a given operating mode. ii) Loads AWTS parameters into private shared region WASCOM. iii) Performs system exit for various normal & error conditions.
Subroutine 1 (INITWAL)	Selects initial wall contours for a given set of test parameters (Mach number, Reynolds number and AoA) calling subroutines STWALL and WALCAL when necessary.
Subroutine 2 (STWALL)	Selects the wall data for "aerodynamically straight" wall contours for a given Reynolds number and Mach number, from the *Reference Table* datafile using calls to subroutines FILESORT and GETDATA.
Subroutine 3 (WALCAL)	Calculate streamline wall contours and the associated pressure distributions in the farfield of a thin airfoil, based on estimates of lift and drag coefficients, using calls to subroutines BLOCK, LIFT and WAKE.
Subroutine 4 (WALDAT)	i) Loads "aerodynamically straight" wall data into WASCOM. ii) Stores or reads raw wall and tunnel data in the raw data datafile when the appropriate analysis mode is selected.
Subroutine 5 (STAR)	Performs boundary layer calculations along each flexwall.
Subroutine 6 (WAS)	Predicts a new set of wall contours for streamlining and calculates the associated external (imaginary) velocity distributions.
Subroutine 7 (SUME)	Assesses the quality of wall streamlining using the wall pressure loading.
Subroutine 8 (OUT)	Sends an iteration summary to the lineprinter and loads reduced wall data in the library of wall data.
Subroutine 9 (ERROR)	Sends error information to the operator console and lineprinter.

We link these modules together to create the executable module/task FLXWAS (see sub-section 6.1). The subroutine modules communicate through the data commons in private shared regions WASCOM and FLXCOM. We include the definitions, declarations and equivalences of all the data commons in the WAS software using the source files FLXCOM, FLXTYP, FWPTYP, FWPEQU, WASCOM and OAPCM. We list each of these source files in Appendix B. In addition, we include the assignments for the *Reference Table* datafile in the WAS software using the source file FMASSIGN listed in Appendix D.1. Detailed information on each module is found in Section 5 with listings in Appendix A.

This structure allows the user to initially test modified subroutine modules independent of the main task, a useful debugging feature. In addition, we can easily include different analyses by simply replacing an existing module or including another module into the software structure.

3. MODES OF ANALYSIS

We have configured the WAS software to function in four analysis modes. We have assigned each mode a value of the analysis mode designator IANAL from 0 to 3 (IANAL is assigned a value at the CPU-A Digital Constants Panel).

3.1 Infinite Flow Streamlining

IANAL = 0 for "routine" infinite flow streamlining during 2-D airfoil tests. The WAS software then streamlines/adapts the walls according to the wall adjustment strategy of Judd, Goodyer and Wolf[5,6] outlined in sub-section 4.1.2. Successful wall streamlining will automatically end only when the measures of the wall streamlining quality are below fixed maxima (see sub-section 6.4). Alternatively, an error condition will end an unsuccessful attempt to streamline the walls. During each data point, we store in a datafile, a set of data for the last predicted wall shapes. So after a successful streamlining, only the streamlined wall shapes are stored. If the streamlining was unsuccessful then the next set of predicted wall shapes in the streamlining cycle is stored and this data set is available for retaking the data point. Consequently, with each data point a set of wall data is added to a library of wall data. This reduced data datafile forms a temporary wall library for subsequent use in selecting initial wall contours for future data points.

3.2 Empty Test Section Streamlining

For the test section calibration, the test section itself is empty to perform what we know as straight wall streamlining (IANAL = 1). This streamlining/adaptation attempts to generate a constant Mach number distribution along each flexwall using a technique described in sub-section 4.1.1. We can use these tests to create "aerodynamically straight" wall contours which are a reference for all future wall adaptations. In this analysis mode, the WAS software must have the PAUSE operating mode selected, since the operator selects the best "aerodynamically straight" contours for each data point. We then store these selected contours in a library of wall data on the reduced data datafile. Later we can select the best contours to transfer to a permanent wall library called the *Reference Table* datafile, following procedures outlined in Appendix D.

3.3 Re-analysis of Wall Data

For checking of new or old WAS software off-line (See section 7 for more details), we select a re-analysis mode by setting IANAL = 2. In this mode, the WAS software bypasses all communication with the wind tunnel and CPU-B. The WAS software performs a single dummy iteration in a simulated infinite flow streamlining cycle using raw wind tunnel and wall data stored in the raw data datafile. The operator selects the initial wall contours for the dummy iteration in the normal manner and selects the the set of raw data for re-analysis. Before activating the WAS software, the operator must load the appropriate raw data for re-analysis into the raw data datafile (See Appendix E) and initialize the *Reference Table* datafile as appropriate (See Appendix D).

3.4 Development Streamlining

For development testing where infinite flow streamlining is not routine, it is helpful to select this mode of analysis by setting IANAL = 3. A full printout of WAS data is then created for each wall streamlining/adaptation iteration. In addition, raw wall pressures for the last iteration of each data point (usually with the walls streamlined) are stored in the raw data datafile for subsequent re-analysis.

4. COMPUTATIONS

4.1 Wall Shape Predictions (Subroutine WAS)

4.1.1 Empty Test Section Streamlining

The WAS software calculates a local Mach number error, relative to the measured free stream Mach number, for each of the eighteen wall jacks (#1 to #18) which control the streamlining portion of the test section. We multiply these errors by an empirical factor called WMOVF, to compute wall movements for constant Mach number distributions. The value of the variable WMOVF used with the 0.3-m TCT is 1.0 (We assign a value to WMOVF at the CPU-A Digital Constants Panel).

4.1.2 Infinite Flow Streamlining

The WAS[5,6] computes, from the wall pressures alone, wall movements at jacks #1 to #18 per flexwall. These wall jacks control the streamlined portion of the test section. The strategy adjusts these computed wall movements for the aerodynamic interactions between the top and bottom walls using computational factors called Coupling Factors TWCPLF and BWCPLF (Usually each assigned an empirical value of 0.35 at the CPU-A Digital Constants Panel). In addition, the strategy minimizes a natural overshoot by applying other computational factors called Scaling Factors TWSF and BWSF (Usually each assigned an empirical value of 0.8 at the CPU-A Digital Constants Panel) to the computed wall movements.

4.1.3 General Information

In all streamlining modes, the WAS software treats the downstream wall jacks (#19, #20 and #21) on each wall as part of the variable diffuser. The WAS software does not use these jacks for test section streamlining. We control these three jacks per wall to give a smooth wall slope from the wall 'kink', located between jacks #18 and #19, back to the sliding joint at the downstream end of the flexwall plate. We relate the shape of the variable diffuser to the position of jack #18 by the following: jack #19 position is 3/4 of jack #18 position from geometrically straight ; jack #20 position is 3/8 of jack #18 position ; jack #21 position is 1/8 of jack #18 position. We have made no provision for the creation of a sonic throat in the variable diffuser, however we could include this provision if required.

The WAS software stores the predicted jack movement demands for all 21 jacks per flexwall in arrays TWMOV (for the top wall) and BWMOV (for the bottom wall). We have included both these arrays in the private shared region WASCOM. Positive movement is always up.

The WAS software normalizes the external local wall velocity increments (imagined to exist over the outside of the predicted wall shapes for the next iteration) with respect to the free stream velocity. The software stores these normalized velocity increments in arrays TWNVEL (for the top wall) and BWNVEL (for the bottom wall). We have included both arrays in the private shared region WASCOM.

NOTE: Due to power limitations, jack #1 on each wall has become a special case and the software moves the jack half the demanded movement of jack #2 until the jack reaches a movement limit, i.e. 1.9mm (0.075 inch) on the top wall and 1.65mm (0.065 inch) on the bottom wall. Hardware modifications will remove this special case in due course.

4.2 Wall Boundary Layers (Subroutine STAR)

We compute the boundary layer displacement thickness d* along each wall from measured wall pressures using the Von Karman momentum integral equation solved for a turbulent boundary layer by a numerical technique. We assume that the wall boundary layer starts from an arbitrary point, one large jack spacing (12.06cm (4.75 inches)) ahead of the flexwall anchor point, and behave as a flat plate boundary layer for the first 12.06cm (4.75 inches). We obtain the wall Mach number distributions from the measured wall pressures using isentropic flow relationships. No special procedure is included for shock/boundary layer interactions. Consequently, a warning message is given to the operator if sonic speeds have been measured at either wall. The software stores the d* values in arrays TWDS (for the top wall) and BWDS (for the bottom wall). Again, we have included both these arrays in the private shared region WASCOM.

The software computes a dd* value for each of the streamlining jacks. These dd* values are the local difference between a straight wall value of d* found from the *Reference Table* datafile for the current test Mach number and test Reynolds number, and the computed value of d* for the test. The software uses these dd* values to determine the aerodynamic contours of the flexwalls for each wall adaptation iteration.

4.3 Assessment of Wall Induced Interferences (Subroutine SUME)

We compute a distribution of induced velocities in the horizontal and vertical planes along the test section centerline and the model chordline. We calculate these induced velocities using potential flow theory with the Prandtl-Glauert factor for linearised compressible flows. We represent each flexwall with 18 panels of wall vorticity placed on the aerodynamic contours of each wall at the location of each streamlining jack. The local strength of the vorticity is proportional to the local pressure loading on the wall.[4] The potential flow field for assessment of the wall induced interferences consists of an undisturbed free stream in which we place two vortex sheets. The operator defines the position of the model in the test section (between the two vortex sheets) using the setup test parameter table WASPAR (located in the private shared region FLXCOM). The tunnel operator uses a flexwall control system task FLXOP to make the necessary assignments in WASPAR. In the WASPAR table, XLORIG is defined as the horizontal distance of the model quarter chord point from the model pivot (positive upstream), in inches. While YLORIG is defined as the vertical distance of the model quarter chord point from the model pivot (positive up), in inches. The walls are streamlined/adapted when a streamlining criterion is satisfied. In the 0.3-m TCT, the criterion for routine 2-D testing requires the following levels of residual interferences to exist in the test section:

1) The average of the modulus of the local Cp error between internal (real) and external (imaginary) flows along each wall < 0.01.

2) The modulus of the induced angle of attack at the model leading edge < 0.015 degree.

3) The modulus of the induced camber along the model chordline < 0.07 degree.

4) The modulus of the average induced Cp error along the model chordline < 0.007.

For empty test section streamlining, we use the magnitude of the standard deviation in the wall Mach number distribution as the streamlining criterion. During 0.3-m TCT calibration, we obtained values of Mach number standard deviation of less than 0.003 at low subsonic Mach numbers. The standard deviations increased at transonic speeds due to increased sensitivity of the tunnel flow to flexwall imperfections. These standard deviations are a measure of the quality of the test section instrumentation and aerodynamic performance and may provide a useful way of comparing facilities in the future.

5. SOFTWARE DESCRIPTION

We have written the WAS software so that each module is hopefully self-explanatory. A title block (which includes most of the descriptive information in this section), comments and meaningful variable names have all been included to this end. The description of each module highlights the data inputs and outputs in an attempt to allow easy and effective use of the software as a whole or part. We group data relative to the test section geometry after each title block, as required, to simplify the use of the software with different tunnels.

5.1 Program FLXWAS (Version 3.0) - WAS control task (See Appendix A.1 for a listing)

This is the main program which loads the AWTS parameters into memory and then sequences events during a streamlining cycle for the selected operating mode, as shown on Figure 6.

Date 06/15/88

Operating Environment: See sub-section 6.2 for full description.

Included Source Files:
OAPCM - System Common Definitions.
FLXCOM - Flexwall Control System Common Definitions.
WASCOM - WAS Common Definitions.
FLXTYP - Flexwall Common Type Declarations.

Logical Files Used:
CO - Input/output to console
LO - Output to lineprinter

Shared Regions:
OAPCOM (Inserted, no restrictions)
FLXCOM (Inserted, no restrictions)
WASCOM (Inserted, no restrictions)

User-Defined Subroutine Calls:
INITWAL
WALDAT
STAR
WAS
SUME
OUT
FRQST (Flexwall control system subroutine)

System Subroutine Calls:
DIGICO - Reads digital constants from data common OAPCM.
DELAY - Inserts a delay of in the task execution.
WAIT4 - Causes task to go into a pause state.
SETB - Sets selected bits in control words.
TSTB - Tests state of selected bits in control words.
RLTBZ4 - Wait for selected bit to change state.
TIMEOUT, TM2CAL and TIME - Read current date and time.

5.1 Program FLXWAS (Continued)

Inputs:

/OAPCOM/ NAME

Contents of array NAME become computational factors TWCPLF, BWCPLF, TWSF, BWSF and WMOVF, and the analysis mode designator IANAL (via variable ANAL).

/FLXCOM/ IBTSTNR, IBRUNNR, IBPTNR, XCHORD, XBMACH, XBPS, XBPT, XBTT, XBRHOS, XREYNO, XBAOA, FLXISW, FSSTAT, FLXOPT, MVSTAT, WASPAR, WPRS, JSTAT, JPOSA

IBTSTNR, IBRUNNR, IBPTNR, XCHORD, XBMACH, XBPS, XBPT, XBTT, XBRHOS, XREYNO, XBAOA are all wind tunnel test conditions.
FLXISW - Flexwall interrupt status word - Bit 7 indicates move status.
FSSTAT - Flexwall system status word - Flexwall hardware status.
FLXOPT - Flexwall task option word - Set by using task FLXOP.
MVSTAT - Move status word - Task FMVSTR information on flexwall move.
WASPAR - Wall alignment strategy setup test parameters - Set by using task FLXOP.[6]
WPRS - Array of wall pressures (PSIA).
JSTAT - Wall jack status table.
JPOSA - Array of averaged current jack positions (Inches).

Outputs:

/FLXCOM/ STRSTAT, FLXISW, FSSTAT, IACCW, IASTAT, JPOSN, JPOSA, WPRS

STRSTAT - Streamline status word.
FLXISW - Flexwall interrupt status word - Bit 5 initiates flexwall movement.
FSSTAT - Flexwall system status word - Bit 7 enables flexwall movement.
IACCW - Communications control word.
IASTAT - Communications status word.
JPOSN - Array of new (destination) jack positions (Inches).
JPOSA - Array of averaged current jack positions (Inches).
WPRS - Array of wall pressures (PSIA).

/WASCOM/ NOJACK, NOCPT, XJACK, WL, IANAL, FMACH, PSTATIC, PTOTAL, TTOTAL, DENSITY, CREY, ALPHA, CHORD
(See listing of WASCOM in Appendix B for variable descriptions)

Output Messages:

```
                    NASA LANGLEY RESEARCH CENTER
           TRANSONIC CRYOGENIC ADAPTIVE WALL TEST SECTION
                        FLXWAS VERSION 3.0

            ******  FLEXWALL CONTOURS INITIALIZED  ******
          ============  READY TO STREAMLINE  ===========

          FLXWAS> ***  WAIT FOR TEST CONDITION CHECK  ***

               ***  ACQUIRING FLEXWALL PRESSURES  ***

          FLXWAS> ***  SINGLE ITERATION COMPLETE  ***

          FLXWAS> *****  WAIT FOR ANALYSIS CHECK  *****
      TYPE IN ANY CHARACTER TO CONTINUE OR [WS] FOR WALLS S/LINED
```

5.1 Program FLXWAS (Continued)

Output Messages (Continued):

```
                 ***********************************
                 * STREAMLINING TERMINATED *
                 *     HARDWARE ERROR          *
                 ***********************************
                         JACK STATUS
                      #    TOP    BOTTOM
                      (Data tabulated below)
                      N     X          Y
     PRESS RETURN TO RECALL 'FLXOP' MENU FOR NEXT PT

                 ******************************
                 *   WALLS STREAMLINED    *
                 ******************************

                 ***********************************
                 * STREAMLINING TERMINATED *
                 *       SOFTWARE ERROR        *
                 ***********************************

                 ***********************************
                 * STREAMLINING TERMINATED *
                 *    JACK LIMIT REACHED       *
                 ***********************************
                         JACK STATUS
                      #    TOP    BOTTOM
                      (Data tabulated below)

        FLXWAS> *** RE-ANALYSIS COMPLETED ***
```

Processing:

1) Load fixed tunnel data into shared region WASCOM.
2) In re-analysis mode, jump to 13.
3) Initialize MAXNET link.
4) Select initial wall contours for next data point (Subroutine INITWAL).
5) Set chord for empty test section streamlining.
6) Move flexwalls to initial contours.
7) Wait for CPU-B Operator to initiate the streamlining cycle.
8) DO - Enter streamlining cycle.
9) In Wait mode, wait for test condition update.
10) Read Digital Constants Panel (Subroutine DIGICO).
11) Acquire data from the flexwall test section (Subroutine FRQST).
12) Re-locate tunnel data in WASCOM.
13) In re-analysis mode, load tunnel data from disk (Subroutine WALDAT).
14) Select "Straight Wall" reference contours for next iteration (Subroutine WALDAT).
15) Output raw data to disk if required (Subroutine WALDAT).
16) Compute wall boundary layers (Subroutine STAR).
17) Compute new wall contours for streamlining (Subroutine WAS).
18) Sum residual wall interferences (Subroutine SUME).
19) Check streamlining status.

(Continued on next page)

Processing (Continued):

20) If walls are streamlined, output reduced data information and jump to 30.
21) In Single Iteration mode, set data acquisition bit and then wait to continue.
22) Printout FLXCOM data on first pass only.
23) Dump core device to the lineprinter on the first pass only.
24) Output reduced data to lineprinter and disk (Subroutine OUT).
25) In Pause mode, wait for response from CPU-A operator.
26) If excessive iterations, abort streamlining, jump to 30.
27) In re-analysis mode, jump to 31.
28) Move flexwalls to new contours for next iteration.
29) ENDDO - Continue the streamlining cycle.
30) Exit conditions.
31) Exit the system.

Exception Handling:

Whenever an error condition is encountered (STRSTAT = -1) an appropriate message is displayed on the operator console and lineprinter, IASTAT is set appropriately, CPU-B is notified, and FLXWAS stops executing.

If the CPU-A operator enters a non-numeric response when a numeric answer is required, no error message is written. Instead, the task waits until a valid response is supplied.

5.2 Subroutine INITWAL (Version 1.9) - Flexwall initialization subroutine
(See Appendix A.2 for a listing)

This subroutine selects the appropriate initial wall contours for the next streamlining cycle and loads the jack positions into the private shared regions FLXCOM and WASCOM, and the associated external (imaginary) wall velocity increments into WASCOM. The subroutine selects initial wall contours with respect to operator defined setup test parameters in the WASPAR table in FLXCOM (Mach number, Reynolds number and angle of attack). The selection follows a sequence shown below until an appropriate set of wall contours is found:

i) If a fictitious angle of attack of 99.0 is entered in the set-up parameters table then select "aerodynamically straight" contours from the *Reference Table* datafile (using Subroutine STWALL).
ii) If a fictitious angle of attack of 88.0 is entered in the set-up parameters table then let the tunnel operator select a specific set of wall contours from the reduced data datafile.
iii) Look at the directory of the reduced data datafile and select a "good" set of wall contours with respect to the data in the set-up parameters table.
iv) If no "good" set of wall contours can be found, then calculate a set of wall contours based on rough estimates of model lift and drag coefficients as well as data in the set-up parameters table (using Subroutine WALCAL).

For more details about the selection of "aerodynamically straight" wall contours see sub-section 5.3 (Subroutine STWALL). A description of the software available to calculate initial wall contours, when necessary, can be found in sub-section 5.4 (Subroutine WALCAL). The associated computations are described in Appendix G.

The selection of "good" wall contours from the wall library requires some qualification. "Good" wall contours are defined as having the associated test parameters matching the operator defined set-up parameters within the following limits:

Above Mach 0.7, Mach number match better than .02
Angle of attack match better than 4.0 degrees

In addition, there are some special requirements. The software only considers those wall contours with associated Mach numbers equal or less than the set-up Mach number (plus a tolerance of 0.01) are considered. Also, the software only considers those wall contours with an associated angle of attack modulus less than the modulus of the set-up angle of attack (plus a tolerance of 0.2°). The need for small adjustments to the operator defined set-up parameters is to accommodate small deviations in actual test parameters from demanded values. The selection process starts by selecting the last set of wall contours stored in the library of wall data (Usually the streamlined wall contours for the last data point) and then attempts to find a better set by scanning the complete directory from the last wall data record to the first. Reynolds number has only a small influence on the wall contours but this is model size and shape dependent. Consequently, we keep Reynolds number information available to use in the selection process should it become necessary. The large tolerance used in the angle of attack match is to provide the software with the necessary flexibility to deal with any test program.

Date 04/03/87

<u>Included Source Files</u>:

FLXCOM - Flexwall Control System Common Definitions.
WASCOM - WAS Common Definitions.
FLXTYP - Flexwall Common Type Declarations.
FWPTYP - Flexwall "WASPAR" Type Declarations.
FWPEQU - Flexwall "WASPAR" Equivalences.

5.2 Subroutine INITWAL (Continued)

Logical Files Used:

CO - Input/output to console
LO - Output to lineprinter
12 - Reduced data datafile

Shared Regions:

FLXCOM
WASCOM

User-Defined Subroutine Calls:

WALCAL
STWALL
ERROR

Inputs:

/FLXCOM/ WASPAR

WASPAR - Table of operator defined set-up parameters (see Appendix B).

/WASCOM/ NOJACK,CHORD

NOJACK - Number of flexwall jacks.
CHORD - Model chord (Inches).

Reduced data datafile - KBUFF, LBUFF, TMACH, TREYNO, TALPHA, BUFF
(See datafile management assignments in listing of TABASS in Appendix E.)

Outputs:

/FLXCOM/ JPOSN, STRSTAT

JPOSN - Array of new (destination) jack positions (Inches).
STRSTAT - Streamline status word.

/WASCOM/ XORIG, YORIG, TOPY, BOTY, TWVEL, BWVEL

XORIG - Horizontal distance of model 1/4 chord point upstream of model pivot (Inches).
YORIG - Vertical distance of model 1/4 chord point above the model pivot (Inches).
TOPY - Array of top wall jack positions for the initial contour (Inches).
BOTY - Array of bottom wall jack positions for the initial contour (Inches).
TWVEL - Array of top wall external velocities for the initial contour (v/U0).
BWVEL - Array of bottom wall external velocities for the initial contour (v/U0).

Output Messages:

```
              INITWAL VER 1.9> SETUP PARAMETERS (5 values listed on next line)

****************************************************************************************
INITIAL WALL CONTOURS - RECORD ### MACH #.### RN ####E### AOA ###.##
****************************************************************************************

****************************************************************************************
              INITIAL WALL CONTOURS 'STRAIGHT' FOR NEXT POINT
****************************************************************************************

              INITWAL> STRAIGHT WALL DATA ERROR

              INITWAL> INPUT RECORD NO. FOR INITIAL CONTOURS (0-EXIT)
```

Processing:

1) Display setup parameters.
2) If "aerodynamically straight" wall contours required jump to 12.
3) Select initial wall contours from reduced data library and load from disc.
4) If no wall contours in library (NREC < 7) jump to 11.
5) Select data records with wall information (NREC > 6).
6) Choose previous wall contours until a better set is found where ISAME = 3.
7) Define tolerances for selection of wall contours: TMDIFF,TRDIFF,TADIFF.
8) If a very good set of wall contours are found (IGOOD = 3) then jump to 10.
9) If no good set of wall contours (IGOOD < 2) can be found then jump to 11.
10) Load good wall contour data (IGOOD > 1) into the private shared region WASCOM, then jump to 13.
11) Compute initial wall contours for current test parameters (Subroutine WALCAL) then jump to 13.
12) Select initial wall contours from the *Reference Table* datafile (Subroutine STWALL).
13) Load up new jack positions for initial wall contours in private shared region FLXCOM.
14) Return to main program.

Exception Handling:

Whenever an error condition is encountered (STRSTAT = -1) an appropriate message is displayed on the operator console and the lineprinter, subroutine ERROR is called to provide additional error information, control returns to the main program.

5.3 Subroutine STWALL (Version 1.5) - "Aerodynamically straight" wall data subroutine (See Appendix A.3 for a listing)

This subroutine acquires the appropriate reference wall data for the current test conditions, i.e. "aerodynamically straight" wall data from the *Reference Table* datafile. Wall contours and boundary layer displacement thicknesses (d*s) are found from data sets within the *Reference Table* datafile. We have configured the directory of this datafile with a 10 by 10 matrix based on Mach number and Reynolds number. The software finds four data sets (from the 100 in the datafile) which bracket the given Mach number and Reynolds number. The software linearly interpolates between these data sets to find the appropriate reference wall shapes and d*s for the given test conditions.

Date 12/22/86

Included Source Files:

FMASSIGN - *Reference Table* Datafile Management Assignments.

Logical Files Used:

LO - Output to lineprinter
10 - Reference table datafile

Shared Regions: None

User-Defined Subroutine Calls:

FILESORT - Sort array into ascending order (Listed with STWALL).
GETDATA - Get wall data from the reference table (Listed with STWALL).

Inputs:

Call statement - FMACH, REYNO

FMACH - Free stream Mach number.
REYNO - Unit Reynolds number per foot.

Reference Table datafile - IBUFF, JBUFF

IBUFF - Reference table directory record buffer (See Appendix D).
JBUFF - Wall data record buffer (See Appendix D).

Outputs:

Call statement - TWY, BWY, DST, DSB, IERROR

TWY - Array of top wall jack positions for "aero. straight" contour (Inches).
BWY - Array of bottom wall jack positions for "aero. straight" contour (Inches).
DST - Array of top wall d* values for "straight" contour (Inch).
DSB - Array of bottom wall d* values for "aero. straight" contour (Inch).
IERROR - Output error code.

Output Messages:

```
STWALL> VERSION 1.5 STRAIGHT WALL DATA

INTERPOLATION RECORDS = ### , ### , ### , ###

MACH NO. RANGE  ##.## - ##.##
REYNOLDS NO. PER FT   ###E## - ###E##

TOP WALL JACK POSITIONS

BOTTOM WALL JACK POSITIONS

TOP WALL D* CONTOUR

BOTTOM WALL D* CONTOUR
```

Processing:

1) Initialization of datafile pointers.
2) Sort Mach numbers in ascending order (Subroutine FILESORT).
3) Sort Reynolds number in ascending order (Subroutine FILESORT).
4) Determine which Mach numbers bracket desired Mach numbers.
5) Determine which Reynolds numbers bracket desired Reynolds number.
6) Get wall data (Subroutine GETDAT).
7) Interpolate data between four sets of data.
8) Load interpolated data into working arrays.
9) Return to calling subroutine.

Exception Handling:

If an error occurs during execution then control will return to the calling subroutine with one of the following output error codes in variable IERROR:

1 - Failure to bracket Mach no. on datafile
2 - Failure to bracket Reynolds no. on datafile
3 - Combination of above failures on datafile
4 - Missing data records on datafile

The no error condition is IERROR = 0

5.4 Subroutine WALCAL (Version 1.5) - Wall shape calculation subroutine
(See Appendix A.4 for a listing)

This subroutine calculates streamline contours and the associated pressure distributions in the farfield of a thin airfoil based on expected test conditions and rough estimates of model performance given by the tunnel operator. Linearised potential flow is assumed with compressibility effects taken into account by use of the Prandtl-Glauert factor. A full description of the computations in this subroutine is given in Appendix G.

Date 09/29/88

Included Source Files:

WASCOM - WAS Common Definitions.

Logical Files Used:

CO - Input/output to console
LO - Output to lineprinter

Shared Regions:

WASCOM

User-Defined Subroutine Calls:

STWALL
ERROR
BLOCK - Computes disturbance velocity due to blockage (Listed with WALCAL).
WAKE - Computes disturbance velocity due to wake (Listed with WALCAL).
LIFT - Computes disturbance velocity due to lift (Listed with WALCAL).

Inputs:

/WASCOM/ NOCPT, CHORD, XORIG, XJACK

NOCPT - Number of wall computing points.
CHORD - Model chord (Inches).
XORIG - Horizontal distance of model 1/4 chord upstream of model pivot (Inches).
XJACK - Array of jack X co-ordinates (Inches).

Call statement - SPMACH, SPREYN, SPAOA

SPMACH - Setup parameters Mach number.
SPREYN - Setup parameters Reynolds number.
SPAOA - Setup parameters angle of attack.

Console keyboard - EPS, CL, CD

EPS - Model's maximum section thickness (%chord).
CL - Estimated model lift coefficient.
CD - Estimated model drag coefficient.

Outputs:

/WASCOM/ TOPY, BOTY, TWVEL, BWVEL

TOPY - Array of top wall jack positions for computed shape (Inches).
BOTY - Array of bottom wall jack positions for computed shape (Inches).
TWVEL - Array of top wall external velocities over computed shape (v/U0).
BWVEL - Array of bottom wall external velocities over computed shape (v/U0).

5.4 Subroutine WALCAL (Continued)

Output Messages:

```
          WALCAL> ENTER ESTIMATED LIFT- AND DRAG COEFFICIENTS

          WALCAL> ENTER AIRFOIL MAXIMUM THICKNESS (%) 0-EXIT
                       THICKNESS NOT TO EXCEED 20%

          WALCAL> INPUT ERROR   THICKNESS IS ###.##%

          WALCAL> **** CHORD ERROR CHECK CPU-B RTP VALUE ****

WALCAL> AIRFOIL GEOMETRY: CHORD ##.## INCHES MAXIMUM THICKNESS##%

WALCAL> FLOW PARAMETERS: FREE STREAM MACH NUMBER ###.##
                         CHORD REYNOLDS NUMBER ####E##
                         LIFT COEFFICIENT ###.###
                         DRAG COEFFICIENT ###.###

       WALCAL> INITIAL WALL CONTOURS (Followed by table of jack positions
                                       and associated wall velocities (v/U0))
```

Processing:

1) Define functions.
2) Input section from operator console.
3) Load test parameters into work variables.
4) Define jack positions relative to model based reference system.
5) Initialize singularity strengths.
6) Compute wall displacement due to blockage effect.
7) Compute wall displacement due to lift effect.
8) Compute wall displacement due to wake blockage.
9) Determine wall contours due to lift and blockage.
10) Compute the disturbance velocity tangential to the walls.
11) Determine the "aero. straight" wall shapes for test conditions in the setup parameter table.
12) Determine total wall deflections.
13) Output data summary.
14) Interface data to private shared region WASCOM.
15) Limit Jack #1 and #22 movement.
16) Determine jack movements for the variable diffuser.
17) Return to subroutine INITWAL.

Exception Handling:

If an excessive model thickness (> 20%) is accidentally entered by the operator then the operator is given an opportunity to correctly enter the model thickness.

If a strange chord value is found, an error message is sent to the CPU-A console and the WAS software is halted.

5.5 Subroutine WALDAT (Version 2.3) - Wall data acquisition subroutine
(See Appendix A.5 for a listing)

This subroutine selects "aerodynamically straight" wall data for the current test conditions from the *Reference Table* datafile (See Appendix D). If IANAL = 3, the software stores raw wall and tunnel data in the raw data datafile. Under all analysis modes, each raw data set is re-located into working arrays located in the private shared region WASCOM. The current position of the wall jacks is checked against the expected position to warn of position transducer (LVDT) drift.

Date 06/15/88

Included Source Files:

FLXCOM - Flexwall Control System Common Definitions.
WASCOM - WAS Common Definitions.
FLXTYP - Flexwall Common Type Declarations.

Logical Files Used:

CO - Input/output to console
LO - Output to lineprinter
11 - Raw data datafile

Shared Regions:

FLXCOM
WASCOM

User-Defined Subroutine Calls:

STWALL
ERROR

Inputs:

/FLXCOM/ XBMACH, XBPS, XBPT, XBTT, XBRHOS, XREYNO, ALPHA, XCHORD, IBHOUR, IBMIN, IBTSTNR, IBRUNNR, IBPTNR, WPRS, JPOS
XBMACH - Test Mach number.
XBPS - Reference static pressure (PSIA).
XBPT - Reference total pressure (PSIA).
XBTT - Reference total temperature (K).
XBRHOS - Static density of working fluid (Slugs/ft^3).
XREYNO - Test chord Reynolds number (millions).
XBAOA - Test model angle of attack (degrees).
XCHORD - Model chord (Inches).
IBHOUR - Hour data acquired.
IBMIN - Minute data acquired.
IBTSTNR - Test number.
IBRUNNR - Run number.
IBPTNR - Data point number.
WPRS - Array of wall pressures (PSIA).
JPOS - Array of current wall jack positions (Inches).

/WASCOM/ NOJACK, IANAL, NOCPT, TOPY, BOTY
NOJACK - Number of wall jacks.
IANAL - Analysis mode designator.
NOCPT - Number of wall computing points.
TOPY - Array of demanded top wall jack Y co-ordinates (Inches).
BOTY - Array of demanded bottom wall jack Y co-ordinates (Inches).

Raw data datafile - KBUFF, LBUFF (See listing of TABASS in Appendix E.1.)

5.5 Subroutine WALDAT (Continued)

Outputs:

/WASCOM/ DSTAS, DSBAS, TWYAS, BWYAS, TOPWP, BOTWP

DSTAS - Array of top wall d* thicknesses on "aero. straight" contour (Inches).
DSBAS - Array of bottom wall d* thicknesses with "aero. straight" contour (Inches).
TWYAS - Array of top wall jack Y co-ordinates relative to "aero. straight" contour (Inches).
BWYAS - Array of bottom wall jack Y co-ordinates relative to "aero. straight" contour (Inches).
TOPWP - Array of top wall tapping pressures (PSIA).
BOTWP - Array of bottom wall tapping pressures (PSIA).

Raw data datafile - KBUFF, LBUFF, DB

KBUFF - Directory record buffer (See listing of TABASS in Appendix E).
LBUFF - Directory record buffer (See listing of TABASS in Appendix E).
DB - Raw data record buffer (See Appendix E).

Output Messages:

```
ENTER RECORD NO. FOR DATA RE-ANALYSIS

WALDAT> WARNING - USING LAST RECORD AVAILABLE ***

WALDAT> RAW DATA ERROR
```

FLXWAS RAW DATA FROM RECORD ### (Followed by table of raw data)

```
WALDAT> STRAIGHT WALL DATA ERROR

WALDAT> JACK ## POSITION ERROR
```

Processing:

1) Decide if re-analysis mode selected, if yes jump to 5.
2) Load raw data into array DB.
3) If IANAL = 0, jump to 6.
4) Send raw data in 'DB' to the raw data datafile.
5) Read loaded data from the raw data datafile as a check or re-analysis.
6) Select "aerodynamically straight" wall data record and acquire wall data.
7) Determine jack positions relative to "aero. straight" contours.
8) Load wall pressures into work arrays.
9) Check wall data for auto streamlining.
10) Return to main program.

Exception Handling:

Whenever an error condition is encountered (STRSTAT = -1) an appropriate message is displayed on the operator console and lineprinter, subroutine ERROR is called to provide additional error information, control returns to program FLXWAS.

A jack position error due to LVDT drift is displayed as a warning to the operator, subroutine ERROR is called to provide additional information. If the error is severe, then a hardware error will result when the walls are moved.

A warning is given if the last available record on the raw data datafile is being used. This datafile needs to be copied to tape and then initialized (See Appendix E).

5.6 Subroutine STAR (Version 1.3) - Flexwall boundary layer assessment subroutine (See Appendix A.6 for a listing)

This subroutine numerically solves the Von Karman momentum integral equation to compute values for the local wall boundary layer displacement thickness (d*). (See sub-section 4.2 for more details.) Local wall Mach numbers are calculated using isentropic flow relationships. A warning of sonic velocity at the flexwalls is sent to the CPU-A console and the lineprinter.

Date 12/23/86

Included Source Files:
WASCOM - WAS Common Definitions.

Logical Files Used:
CO - Output to console
LO - Output to lineprinter

Shared Regions:
WASCOM

User-Defined Subroutine Calls: None

Inputs:
/WASCOM/ NOCPT, XJACK, TOPWP, BOTWP, PTOTAL, DENSITY, CREY, CHORD, DSTAS, DSBAS

NOCPT - Number of wall computing points.
XJACK - Array of jack X co-ordinates (Inches).
TOPWP - Array of top wall tapping pressures (PSIA).
BOTWP - Array of bottom wall tapping pressures (PSIA).
PTOTAL - Reference total pressure (PSIA).
DENSITY - Working fluid density (Slugs/ft^3).
CREY - Chord Reynolds number.
CHORD - Model chord (Inches).
DSTAS - Array of top wall d* values for "aero. straight" contour (Inches).
DSBAS - Array of bottom wall d* values for "aero. straight" contour (Inches).

Outputs:
/WASCOM/ TWDS, BWDS, TWMACH, BWMACH
TWDS - Array of top wall d* values for current contour (Inches).
BWDS - Array of bottom wall d* values for current contour (Inches).
TWMACH - Array of Mach numbers along current top wall contour.
BWMACH - Array of Mach numbers along current bottom wall contour.

Output Messages:

```
BOUNDARY LAYER CALCULATIONS
TOP WALL
TAP NO.  DU/DX  MACH NO.  D*  DD*
  (Table of data)
BOTTOM WALL
  (Table of data)
```

```
FLXWAS> *** WARNING - SONIC VEL. AT TOP WALL JACK ##  ***

FLXWAS> *** WARNING - SONIC VEL. AT BOTTOM WALL JACK ##  ***
```

Processing:

1) Calculate d* at each wall jack location using sets of data for three jack locations.
2) Load top wall pressures.
3) Load bottom wall pressures.
4) Local Mach no. calculations.
5) Calculate local wall flow velocities.
6) Assume a turbulent b/l growth to the second measuring point on each wall.
7) Calculate the velocity gradient at measuring point 1.
8) Guess the size of d*.
9) Calculate components of the M.I. Eqn.
10) Calculate d* from dd*/dx for last measuring point on each wall.
11) Calculate d* from dd*/dx for second measuring point on each wall.
12) Calculate d* from dd*/dx for all other measuring points.
13) Check d* guess and iterate to a solution.
14) Store correct values of d* (Inches).
15) Calculate dd*.
16) Check for sonic flow on the flexwalls.
17) Return to main program.

Exception Handling:

A warning is given when sonic velocity is encountered at any of the wall jacks. The operator should then proceed with caution in the knowledge that wall streamlining is no longer "routine" and may not be possible at all.

5.7 Subroutine WAS (Version 1.5) - Wall adjustment strategy subroutine
(See Appendix A.7 for a listing)

The Wall Adjustment Strategy (WAS) predicts new wall contours for test section self-streamlining and computes new external (imaginary) velocities over these new contours[5] (See sub-section 4.1 for more details). The subroutine is configured for two modes of streamlining (See section 4.1): A) Infinite flow (Analysis mode designator IANAL = 0 or 3); B) Empty test section (IANAL = 1).

Date 12/22/86

Included Source Files:
WASCOM - WAS Common Definitions.

Logical Files Used:
CO - Output to console
LO - Output to lineprinter

Shared Regions:
WASCOM

User-Defined Subroutine Calls: None

Inputs:
/WASCOM/ NOCPT, FMACH, TWCPLF, BWCPLF, TWSF, BWSF, PSTATIC, TOPWP, BOTWP, TWVEL, BWVEL, XJACK, TWMACH, BWMACH, WMOVF, IANAL

NOCPT - Number of wall computing points.
FMACH - Free stream Mach number.
TWCPLF - Top wall coupling factor.
BWCPLF - Bottom wall coupling factor.
TWSF - Top wall scaling factor.
BWSF - Bottom wall scaling factor.
PSTATIC - Reference static pressure (PSIA).
TOPWP - Array of top wall tapping pressures (PSIA).
BOTWP - Array of bottom wall tapping pressures (PSIA).
TWVEL - Array of top wall external velocities for current contour (v/U0).
BWVEL - Array of bottom wall external velocities for current contour (v/U0).
XJACK - Array of jack X co-ordinates (Inches).
TWMACH - Array of top wall Mach numbers for current contour.
BWMACH - Array of bottom wall Mach numbers for current contour.
WMOVF - Wall movement factor for empty test section streamlining.
IANAL - Analysis mode designator.

Outputs:
/WASCOM/ TWVDIF, BWVDIF, TWVSQ, BWVSQ, TWNVEL, BWNVEL, TWMOV, BWMOV

TWVDIF - Array of top wall velocity differences (dv/U0).
BWVDIF - Array of bottom wall velocity differences (dv/U0).
TWVSQ - Array of top wall velocities squared ($v^2/U0^2$).
BWVSQ - Array of bottom wall velocities squared ($v^2/U0^2$).
TWNVEL - Array of top wall external velocities for the new (destination) contour (v/U0).
BWNVEL - Array of bot. wall external velocities for the new (destination) contour(v/U0).
TWMOV - Array of top wall jack movements for the new (destination) contour (Inches).
BWMOV - Array of bottom wall jack movements for the new (destination) contour (Inches).

5.7 Subroutine WAS (Continued)

Output Messages:

```
****** WAS V1.5 COMPUTING NOW !! ******

COUPLING FACTORS ##.## ##.## SCALING FACTORS ##.## ##.##

 WALL MOVE FACTOR (DY/DME)  ##.###
```

Processing:

1) DO - Compute the velocity imbalance/vorticity at each wall computing point and external velocities for the predicted wall contours for the next iteration.
2) Apply Prandtl-Glauert factor to measured top wall Cps.
3) Apply Prandtl-Glauert factor to measured bottom wall Cps.
4) Apply scaling factors to the external velocity calculations.
5) ENDDO - Apply coupling factors to the external velocities.
6) Freeze calculations for jack #1 and #22.
7) For empty test section streamlining jump to 22.
8) Make the first mid-jack co-ordinate at the wall anchor point to ensure a zero wall slope at this location.
9) Determine other mid-jack co-ords between wall computing points.
10) DO-Piecewise cubic curve fit to the wall vorticity using sets of four computing points (labeled 1,2,3,4).
11) ENDDO-Calculate coefficients for each piecewise cubic curve fit.
12) At each mid-jack point, integrate the vorticity along each wall to find the induced vertical velocities, assumed normal to the top and bottom walls, which must be canceled by changes in the free stream component caused by local adjustments of wall slope.
13) Initialize wall movement demand accumulators.
14) Set wall slopes at the wall anchor points equal to zero.
15) DO-Find the jack movements required for wall streamlining, by performing integrations of piecewise quadratic curves fitted to sets of three wall slopes.
16) Top wall - movement demand coefficients.
17) Bottom wall- movement demand coefficients.
18) Scale jack movement demands.
19) ENDDO-Couple jack movement demands.
20) Compute jack #1 and #22 movement.
21) Jump to 24.
22) Calculate jack movement demands for constant wall Mach number during empty test section streamlining.
23) Set imaginary (external) velocities to free stream.
24) Determine jack movement demands for the variable diffuser.
25) Return to main program.

Exception Handling: None

5.8 Subroutine SUME (Version 1.6) - Wall induced interferences assessment subroutine (See Appendix A.8 for a listing)

This subroutine sums estimates of the wall induced errors in the test section for each set of wall shapes used during a streamlining cycle. Wall induced interferences are calculated along the test section centerline and the model chord-line (See sub-section 4.3 for more details). We use these interferences as measures of streamlining quality for deciding when the walls are streamlined.

Date 02/18/87

Included Source Files:

FLXCOM - Flexwall Control System Common Definitions.
WASCOM - WAS Common Definitions.
FLXTYP - Flexwall Common Type Declarations.

Logical Files Used:

CO - Output to console
LO - Output to lineprinter

Shared Regions:

FLXCOM
WASCOM

User-Definded Subroutine Calls: None

Inputs:

/WASCOM/ ALPHA, CHORD, XORIG, YORIG, NOJACK, NOCPT, BWVEL, TWVEL, TWVSQ, BWVSQ, TWYAS, BWYAS, DSTAS, DSBAS, TWDS, BWDS, TWVDIF, BWVDIF, XJACK, WL, IANAL, BETA (See listing of shared region WASCOM in Appendix B for variable descriptions)

Outputs:

/FLXCOM/ STRSTAT
STRSTAT - Streamlining status word.

Output Messages:

```
              WALL CP ERROR
         TOP - ###.####   BOTTOM - ###.###  ##.## (XORIG) ##.## (YORIG)

         SUME> V1.6 RESIDUAL ERRORS
         XO       U/UFS      V/UFS
     (Table of values on tunnel centerline)
         MODEL ERRORS
     (Table of values on model chordline)

              EFFECT                  DELTA CL
         ALPHA ERROR = #.### DEGREES      ###.###
         INDUCED CAMBER = #.### DEGREES  ###.###
         CP ERROR (1/4 CHORD) = #.###
         CP ERROR (AVERAGE) = #.###      ###.###

         RESIDUAL INTERFERENCES =  ###.####  ###.####  ###.####
                                    A.O.A.   CAMBER    CP

         ITERATION ##  CP ERROR - TOP ##.###  BOT ##.###
         RESIDUALS - ###.#### , ###.#### , ###.####
```

Processing:

1) Set acceptable levels of residual interference.
2) Calculate average wall Cp errors (Big E).
3) Sum wall induced velocities on tunnel centerline.
4) Betaize the vertical ordinate of the vortex sheet.
5) Sum wall induced velocities along model chordline.
6) Betaize the vertical ordinate of the vortex sheet.
7) Integrate the induced camber over the model chord.
8) Assess the quality of wall streamlining and decide if the streamlining cycle can be terminated.
9) Output data summary to operator console.
10) Return to main program.

Exception Handling:

None

5.9 Subroutine OUT (Version 2.1) - Data manipulation and display subroutine (See Appendix A.9 for a listing)

This subroutine prints out a data summary on each wall adaptation iteration, on the lineprinter. The software also sorts out the reduced wall data and loads this data in the reduced data datafile (wall library) as required.

Date 12/31/86

Included Source Files:

FLXCOM - Flexwall Control System Common Definitions.
WASCOM - WAS Common Definitions.
FLXTYP - Flexwall Common Type Declarations.

Logical Files Used:

CO - Output to console
LO - Output to lineprinter
12 - Reduced data datafile

Shared Regions:

FLXCOM
WASCOM

User-Defined Subroutine Calls:

ERROR

Inputs:

/FLXCOM/ STRSTAT
STRSTAT - Streamlining status word.

/WASCOM/ CREY, CHORD, IANAL, NOJACK, FMACH, ALPHA, TOPY, BOTY, TWMOV, BWMOV, TWYAS, BWYAS, TWNVEL, BWNVEL, TWDS, BWDS
(See listing of shared region /WASCOM/ in Appendix B for variable descriptions.)

Outputs:

/FLXCOM/ JPOSN
JPOSN - Array of new (destination) jack positions (Inches).

/WASCOM/ TWVEL, BWVEL, TOPY, BOTY
TWVEL - Array of top wall external velocities for the new (destination) contour (v/U0).
BWVEL - Array of bot. wall external velocities for the new (destination) contour (v/U0).
TOPY - Array of top wall new (destination) jack positions (Inches).
BOTY - Array of bottom wall new (destination) jack positions (Inches).

Reduced data datafile - KBUFF, LBUFF, TMACH, TREYNO, TALPHA, TCHORD, DATA (See listing of TABASS in Appendix E for variable descriptions)

5.9 Subroutine OUT (Continued)

Output Messages:

```
DATA SUMMARY - EMPTY TEST SECTION STREAMLINING

DATA SUMMARY - RE-ANALYSIS OUTPUT

DATA SUMMARY - DEVELOPMENT STREAMLINING

DATA SUMMARY - INFINITE FLOW STREAMLINING

TOP WALL
JACK   FROM ST    NEXT VEL   MOVE   JACKS - NOW - TO SET
       (Table of reduced data)
BOTTOM WALL
       (Table of reduced data)

*** WARNING - USING LAST RECORD AVAILABLE ***

OUT> REDUCED DATA DISKFILE IS FULL !!!

REDUCED DATA STORED IN RECORD ###
```

Processing:

1) Decide what sort of printout is required.
2) Printout data summary and overwrite current jack positions with predicted positions (TOPY(*), BOTY(*)).
3) Load up TWVEL and BWVEL for next iteration.
4) If walls are streamlined (STRSTAT = 1) jump to 7.
5) Read config. info. from the reduced data datafile.
6) Write reduced data and config. info. to the reduced data datafile.
7) Return to main program.

Exception Handling:

A warning is given to the operator that the last available record on the reduced data datafile is being used. This datafile needs to be copied to tape and then initialized (See Appendix E). If no corrective action is taken the next data point will result in an error trap (STRSTAT = -1) causing an error message, a call to subroutine ERROR for additional output information and the return of control to the main program.

5.10 Subroutine ERROR (Version 1.2) - FLXWAS error message subroutine (See Appendix A.10 for a listing)

This subroutine generates error messages associated with the WAS software which are sent to the lineprinter and CPU-A console.

Date 12/23/86

Logical Files Used:

CO - Output to console
LO - Output to lineprinter

Shared Regions: None

User-Defined Subroutine Calls: None

Inputs:

Subroutine call - IERROR
IERROR - Input error code.

Outputs: None

Output Messages:

```
FLXWAS>  *** MACH NUMBER NOT BRACKETED ON DATAFILE ***

FLXWAS>** REYNOLDS NUMBER NOT BRACKETED ON DATAFILE **

FLXWAS>** TEST PARAMETERS NOT BRACKETED ON DATAFILE **

FLXWAS> *** MISSING DATA RECORD ON DATAFILE ***

FLXWAS> *** TOP WALL JACK POSITION ERROR ***

FLXWAS> *** BOTTOM WALL JACK POSITION ERROR ***

FLXWAS> *** DATAFILES NEED TO BE INITIALIZED ***
```

Processing: None

Exception Handling: None

6. SOFTWARE USER'S MANUAL

6.1 FLXWAS Task Generation

As mentioned in the introduction and overview, we use the WAS software on a Modcomp Classic mini-computer (CPU-A) as the executable module/task FLXWAS. This task is generated by first compiling the main program and the nine associated subroutines (listed in the overview), then linking together the ten object modules and any required modules from the system library of routines and functions. These two operations are performed using procedures unique to the Modcomp CPU-A system. We carry out the FORTRAN compiling operation using a procedure called XFTN. We simply activate this procedure by entering the command XFTN (Filename to be compiled), assuming the pointer JC is assigned to the memory partition where the procedure XFTN resides. The procedure expects the source of the filename to be residing on the User Source Library (USL) and the procedure deposits the compiled object in the User Library (UL). The WAS software includes the data common declarations, definitions and equivalences by accessing source files FLXCOM, FLXTYP, FWPTYP, FWPEQU, WASCOM and OAPCM on the USL. In addition, the WAS software includes the assignments for the *Reference Table* datafile by accessing the source file FMASSIGN on the USL. Hence, these source files must be on the USL for successful compilation of the WAS software modeules. Please note that if the user wishes to assign USL and UL to memory partitions other than the default condition, the user should make these assignments before activating XFTN.

Linking is performed by a special procedure P:FLXWAS and the generated task FLXWAS is stored on the logical file AAA. This logical file must be assigned to a Load Module (LM) memory partition name before activating P:FLXWAS. The procedure overwrites any previous version of the task with the same filename on the memory partition assigned to AAA.

****

Please note, anyone using procedures XFTN and P:FLXWAS should not assign USL, UL or AAA to any configuration controlled memory partitions without conforming to the software configuration control procedures on CPU-A.

****

6.2 Operating Environment

The WAS software requires the following operating environment before we can successfully activate the software:

1) Modcomp CPU-A system operational with both the system software OAP (Operational Acquisition Program) and the flexwall control system software activated with option AUTO selected in the flexwall task option word FLXOPT. Please note, in re-analysis mode (IANAL=2) the WAS software requires only task FLXOP to be active if the flexwall control system software aborts for any reason.

2) *Reference Table* datafile of "aerodynamically straight" wall data existing on memory partition ASW (See Appendix D for how to view this datafile).

3) Reduced data datafile existing on memory partition BSB (see Appendix E for how to view this datafile).

4) Set-up parameters table assigned values in array WASPAR on the private shared region FLXCOM using the SP option from the FLXOP Menu. (See equivalences in listing of FWPEQU in Appendix B.)

5) Computational factors and the analysis mode designator assigned appropriate values on the CPU-A Digital Constants Panel (DCP) as listed below:

   TWCPLF = Top wall coupling factor (Not used when IANAL = 1).
   BWCPLF = Bottom wall coupling factor (Not used when IANAL = 1).
   TWSF = Top wall scaling factor (Not used when IANAL = 1).
   BWSF = Bottom wall scaling factor (Not used when IANAL = 1).
   ANAL = Analysis mode designator IANAL.
   WMOVF = Wall movement factor (Only used when IANAL = 1).

6) If IANAL is **Not Equal** to 2, Modcomp CPU-B system operational with OAP and RTP software active.

7) If IANAL is **Not Equal** to 2, the operating mode for the WAS software (see sub-section 6.3) selected using task FLXOP or default options accepted for "routine" 2-D airfoil testing.

8) If IANAL is **Greater Than** 1, a raw data datafile existing on memory partition BSA (See Appendix E for how to view this datafile)

6.3 User Options

6.3.1 **Selection of analysis mode.** We make this selection by assigning a value to the analysis mode designator IANAL at the CPU-A Digital Constants Panel. The analysis modes are described in Section 3 and summarized as follows:

IANAL = 0 Infinite flow streamlining.
IANAL = 1 Empty test section streamlining.
IANAL = 2 Re-analysis of raw wall data.
IANAL = 3 Development infinite flow streamlining.

6.3.2 **Setting the setup parameters table.** This table contains important parameters for the selection of initial wall contours. We use task FLXOP to input values into this table under the SP option from the FLXOP Menu. The software stores this inputted data in array WASPAR located in the private shared region FLXCOM. The operator must select One of the following instruction sets for each data point:

i) Routine initial wall contours - Set parameters SPMACH, SPREYN and SPAOA to the test conditions for the next data point. (Note that at execution time, the WAS software may request additional information from the CPU-A operator, i.e. model lift and drag coefficients, if no suitable set of wall data can be found in the reduced data datafile.)

ii) Straight wall contours - Set test parameters to Mach no. and Reynolds no. for the next data point and input a fictitious SPAOA of 99.0.

iii) Next wall contours in a streamlining cycle (halted for whatever reason) - Set test parameters to the current test parameters.

iv) Particular wall contours in the reduced data datafile - Set test parameters to Mach no. and Reynolds no. of the next data point and input a fictitious SPAOA of 88.0. (Note that at execution time, the WAS software will request from the CPU-A operator, the record number of the set of wall data which the software will use to set the initial wall contours.)

6.3.3 Selection of operating mode. We make this selection using task FLXOP under the Select Options Menu. The menu gives the following options:

SINGLE/CONTINUE
WAIT/NOWAIT
PAUSE/NOPAUSE

We use these different modes to achieve the desired level of operator involvement, i.e. from casual observer to expert. The default mode selection is for routine infinite flow streamlining with select options of CONTINUE : NOWAIT : NOPAUSE. For routine empty test section streamlining the select options become CONTINUE : NOWAIT : PAUSE (See section 2).

6.3.3.1 Examples of the different operating modes are as follows:

a) **Manual termination of wall streamlining** (Operating mode: PAUSE) (Use this option to take fixed wall data by declaring walls streamlined after one iteration)

i) Select PAUSE operating mode using the FLXOP Select Options Menu.
ii) Observe that the WAS software will now pause after each wall adaptation iteration and await a response from the CPU-A console, either to continue the streamlining cycle or finish the streamlining cycle by declaring the walls streamlined.
iii) At each pause the operator can assess the brief iteration report on the CPU-A console and/or the more detailed printout.

b) **Model data acquisition during a streamlining cycle** (Operating Mode: SINGLE) (Used to assess performance and effect of test section wall adaptation on model data)

i) Select SINGLE mode using the FLXOP Select Options Menu.
ii) Observe that Modcomp CPU-B will now take model data at the completion of each iteration of the streamlining cycle regardless of whether the walls are streamlined or not. (In CONTINUE mode, the model data is only acquired when the walls are declared streamlined.)

c) **Check test conditions before acquiring wall pressures** (Operating Mode: WAIT) (Used when test conditions become sensitive to wall adjustments, i.e. Mach nos.> 0.75 say)

i) Select WAIT mode using the FLXOP Select Options Menu.
ii) Observe a "Check test conditions" message will now be sent to the CPU-B operator before the WAS software acquires each set of wall pressures during a streamlining cycle. The WAS software is then put in a pause state.
iii) When the test conditions are correct, the CPU-B operator gives a response to continue streamlining on the CPU-B terminal and the WAS software is re-activated to acquire another set of wall pressures.

6.4 Execution of the WAS software

With the required operating environment established (as described in sub-section 6.2), we execute the WAS software using the IC option from the FLXOP Select Options Menu. We use the FLXOP Menu as the operator interface to the WAS software. The IC option executes the module/task FLXWAS, previously established from the Load Module partition LM. This execution triggers a sequence of events shown on Figure 5 for analysis modes IANAL = 0 or 1 and on Figure 6 for IANAL = 1. In re-analysis mode (IANAL = 2), a sequence of events take place similar to that for a single iteration of a routine data point with no acquisition of wall or tunnel pressures. If any errors occur during execution consult Appendix C for remedial actions.

During all streamlining cycles, the first event is the selection of initial wall contours. A message "READY TO STREAMLINE" will then appear on the CPU-A console. The streamlining cycle is then resumed by the CPU-B operator requesting the AWTS to "CONFIGURE WALLS" via CPU-A. Then the important events in the streamlining cycle (as shown on Figures 5 and 6) are prompted by messages to the CPU-A console as follows:

```
i) ACQUIRING FLEXWALL PRESSURES
ii) WAS COMPUTING NOW
iii) FMVSTR> MOVE IN PROGRESS
```

Between prompts ii) and iii) a brief report is given on the quality of wall streamlining. For infinite flow streamlining (IANAL = 0 or 3) the report is as follows:

```
*****************************************************************
  ITERATION ##  CP ERROR - TOP ##.#### BOT ## .####
    RESIDUALS - ###.#### , ###.#### , ###.####
```

The residuals are induced angle of attack (degrees), induced camber (degrees), and average Cp error along the model chord, respectively. More details on the Cp errors and the residuals can be found in sub-section 4.3. We have configured the software to declare the *Walls Streamlined* if the modulus of the residual interferences reduce below certain levels as shown below:

Wall Cp Error	< 0.01
Induced AoA	< 0.015°
Induced Camber	< 0.07°
Cp Error	< 0.007

For empty test section streamlining (IANAL=1), the report consists of standard deviations in the wall Mach numbers. Values of 0.003 are the target, but the decision to accept the current values or press on remains with the CPU-A operator, because the PAUSE operating mode is used when IANAL=1.

6.4 Execution of WAS software (Continued)

A streamlining cycle is complete when the message "WALLS STREAMLINED" appears on CPU-A console, i.e. Streamlining Status Word STRSTAT = 1. During normal infinite flow streamlining cycle, the following printout appears on the lineprinter:

i) Setup parameters and associated initial wall contours.
ii) Raw data and raw data datafile record number if stored.
iii) Straight wall data associated with current test conditions.
iv) Iteration heading.
v) Wall boundary layer calculations together with wall Mach nos.
vi) Coupling and scaling factors used in WAS calculations.
vii) Wall Cp Errors.
viii) Residual interferences.
ix) Data summary of WAS calculations.
x) Repeat of output ii) to ix) for each iteration.
xi) Summary of wall pressures and position in decimal and hex when walls are streamlined (First data point of each day only).

We show a typical example of the lineprinter output for just one iteration, on Figure 7. We obtained this output by running the WAS software in re-analysis mode.

In re-analysis mode, the execution of the WAS software causes a message "ENTER RECORD NO. FOR DATA RE-ANALYSIS" to appear on the CPU-A console. When the operator enters the desired record number, the software performs just one dummy iteration and stops. We can carry out further re-analysis of raw data by simply performing another execution of the WAS software. We describe software checks in section 7.

7. SOFTWARE COMPUTATION TESTS

7.1 Technique Overview

During development of the WAS software, we have developed several techniques for testing the WAS computations. These techniques rotate around the acquisition of dummy sets of raw wall and tunnel data sets read by WAS software in re-analysis mode (IANAL = 2) only. The software reads these data sets as if the data were actual measurements from the AWTS. Each set of raw data is loaded into a record in the raw data datafile with the following format:

XBMACH = Test Mach number.
XBPS = Reference static pressure - PTC (PSIA).
XBPT = Reference total pressure - PTINF (PSIA).
XBTT = Reference total temperature (K).
XBRHOS = Flow density (Slug/ft^3).
REYNO = Test chord Reynolds number.
XBAOA = Model incidence (degrees).
CHORD = Model chord (inches).
IBHOUR = Hour data acquired.
IBMIN = Minute data acquired.
IBTSTNR = Test number.
IBRUNNR = Run number.
IBPTNR = Data point number.
WPRS = Array of wall tapping pressures (PSIA).
JPOS = Array of wall jack positions (Inches).

Either the WAS software (operating in any mode other than re-analysis) or the user can generate these data sets on the raw data datafile. For the computational tests described here, we typed in each set of raw data using a utility task DFINPUT (see section 8). During this operation, the software stores the new set of raw data in the next available record on the raw data datafile. When the user exits task DFINPUT, the software prints a hardcopy of the new data set with the associated record number.

For all the computational tests, we used geometrically straight/flat wall shapes for the initial contours in the re-analysis, i.e. SPAOA = 99.0 in the set-up parameter table with the *Reference Table* datafile initialized using task SWINIT (see Appendix D). Consequently, the external (imaginary) wall velocities (stored in arrays TWVEL and BWVEL) are always zero together with the jack positions (stored in arrays TOPY and BOTY). XLORIG and YLORIG should also be zero in the set-up parameter table, which we modify using the SP option in the FLXOP Menu.

7.2 Software Test Techniques

7.2.1 Raw data transfer check

We can check the transfer of data from the raw data datafile into memory by viewing to the WAS software lineprinter output titled 'FLXWAS RAW DATA FROM RECORD ###'. This output has the same format as shown in sub-section 7.1. and should match the output of task DFVIEW for the same record in the raw data datafile.

7.2.2 Boundary layer check

We check the flexwall boundary layer calculations by loading constant pressures into the array WPRS equal to the reference static pressure XBPS. With this set of raw data, the boundary layer calculations should generate a flat plate boundary layer growth that is well known.

7.2.3 Wall induced interference checks

We can check the wall induced interferences by loading different sets of pressure data as follows:

i) Wall pressures set to the reference static pressure generates zero interference.
ii) The same pressures along each wall produce only blockage interference.
iii) Wall pressure distributions from a potential flow simulation of a straight walled AWTS with a non-lifting model installed (found using program SWFLOW) allows checks of interference symmetry and magnitudes.
iv) The transposing of top and bottom wall pressures allows checks of independence of top and bottom wall contributions to the wall induced interferences.

7.2.4 Checks of wall shape predictions

We have checked the wall adjustment calculations for infinite flow streamlining. Again we have used different sets of wall pressures as the raw data for each re-analysis as follows:

i) We check the predicted direction of movement by observing jack movement demands away from the tunnel centerline with a set of wall pressures lower than the reference static pressure XBPS.
ii) We check for any effects of sign changes in the raw data by transposing sets of dummy wall data from the top to the bottom wall and visa versa.
iii) We use wall pressures from a potential flow simulation (see sub-section 7.3) to perform a numerical check of the basic WAS theory[6]. This check is made with compressible flow corrections made insignificant. We treat each flexwall separately with wall coupling factors set to zero. In addition, we do not scale the wall movements and the wall scaling factors are set to unity. We check that the predicted wall shapes (shown on Figure 7) should closely match an exact infinite flow streamline shape for the same test conditions (from program TSFLOW). The set of raw data used in this check corresponds to theoretical wall data (from program SWFLOW) where the model representation has a normalized vortex strength (circulation) of 0.5 and a free stream Mach number of 0.3 . For ease of use this raw data set is Run 999 on the FLXWAS software test datafile tape.
iv) Using the same set of wall data as in iii) above, we check the effect of the coupling and scaling factors by varying the magnitude of these factors and observing the changes to the predicted wall shapes.
v) We use any set of dummy wall data which causes the WAS software to predict a wall movement at Jack #18 to check the movement demands for the variable diffuser jacks #19, #20 and #21.

7.3 Support Software (TSFLOW and SWFLOW)

These two programs provide dummy wall data for numerical checks of the WAS software. These numerical checks are described in the previous sub-section:

i) Program TSFLOW computes the shape and flow velocities along two selected infinite flow streamlines in an incompressible flow field. We constrain these streamlines to pass through the two flexible wall anchor points at the entrance to the 0.3-m TCT adaptive wall test section. We assume the model's center of lift is located at the center of the test section turntable. The model representation consists simply of a co-incident doublet and vortex. In addition, there is an option to include a source (of normalized strength 0.25) placed on the furthest downstream point on the boundary of the doublet, to simulate the model's wake. This option involves the deactivation of statement lines 42, 56 and 63 coupled with the activation of comment line 41 (See listing in Appendix F.1). The doublet has a fixed radius of 1 inch, while the user decides the normalized vortex strength at run time. This software computes streamline vertical displacements and normalized velocity increments at the wall jack positions along the top and bottom flexwalls (ceiling and floor respectively). A sample of the output from TSFLOW is shown in Appendix F.2.

ii) Program SWFLOW computes Cps along potential flow streamlines constrained to be straight or flat in the presence of a model and its images in an incompressible flow field. This flow field represents the test section flow where the flexwalls are set to "Aerodynamically Straight" contours. The model representation and location are identical to those used in program TSFLOW. Again there is an option for a source to be included to simulate a model wake. This option involves the de-activation of statement lines 46, 64 and 72 and the activation of comment line 45 (See listing in Appendix F.3). This software computes local Cps and the local stream function, Ψ, along each straight flexwall as shown in Appendix F.4.

Note: Both TSFLOW and SWFLOW do not require any special procedures for task building. They are both stand alone programs. Available Modcomp procedures XFTN (Compiling task) and P:TB (Task builder) were used to generate executable tasks TSFLOW and SWFLOW on the logical file AAA. (See sub-section 6.1 for more more details about task generation on the Modcomp.)

8. UTILITY SOFTWARE OVERVIEW

During the development of the WAS software, we have found it necessary to evolve several utility routines to support the three datafiles used by the WAS software. These routines provide management of the *Reference Table* datafile of "aerodynamically straight" wall data and the raw data and reduced data datafiles. Documentation on the management procedures for these datafiles can be found in Appendices D and E, respectively.

The utility software for the *Reference Table* datafile is as follows:

Task SWINIT — Initializes the *Reference Table* datafile with geometrically flat wall contours.

Task SWINI — Initializes the *Reference Table* with linear divergent wall contours based on experimental findings with the 0.3-m TCT and allows the tunnel centerline to be effectively rotated by a desired angle.

Task SWVIEW — Tabulates the contents of the *Reference Table* directory and any selected data records.

Task SWDUMP — Generates a tape back-up of the *Reference Table* datafile.

Task SWLOAD — Loads the *Reference Table* datafile from a tape back-up.

Task SWMOD — Allows data records in the *Reference Table* datafile to be overwritten by records selected from the wall library.

Task SWFLIST — Sends a listing of the entire contents of the *Reference Table* to the lineprinter.

The utility software for the raw data and reduced data datafiles (wall library) is as follows:

Task DFINIT — Initializes the raw and reduced data datafiles.

Task DFVIEW — Tabulates the directories of either the raw or reduced data datafiles and lists any selected records.

Task DFDUMP — Generates a tape back-up of the raw data and reduced data datafiles.

Task DFLOAD — Loads the raw data and reduced data datafiles from the tape back-up.

Task DFMOD — Allows selected data records from a tape back-up to be added to the existing reduced data datafile.

Task DFKILL — Removes the latest record in the reduced data datafile.

Task DFINPUT — Allows the data set in any record on the current raw data datafile to be modified with keyboard inputs.

Task DFSHAPI — Allows the data set in any record on the current reduced data datafile to be modified or created with keyboard inputs.

8. UTILITY SOFTWARE OVERVIEW (Continued)

The utility software tasks, cataloged on the previous page, are all executable modules residing on a Load Module (LM). For task generation on the Modcomp, each task has its own task build procedure. These procedures look for the appropriate compiled object (created using the procedure XFTN) on the User Library (UL) and then store the executable module/task on the logical file AAA. It is necessary for this logical file AAA to be assigned to a Load Module (LM) memory partition before activating any of the procedures.

The task building procedures have the following names:

P:SWINI	Generates task SWINIT
P:INISW	Generates task SWINI
P:SWVIEW	Generates task SWVIEW
P:SWDUMP	Generates task SWDUMP
P:SWLOAD	Generates task SWLOAD
P:SWMOD	Generates task SWMOD
P:SWFLIST	Generates task SWFLIST
P:DFINIT	Generates task DFINIT
P:DFVIEW	Generates task DFVIEW
P:DFDUMP	Generates task DFDUMP
P:DFLOAD	Generates task DFLOAD
P:DFMOD	Generates task DFMOD
P:DFKILL	Generates task DFKILL
P:DFPUT	Generates task DFINPUT
P:DFSHA	Generates task DFSHAPI

9. REFERENCES

1. Wolf, S. W. D.: **Evaluation of a Flexible Wall Testing Technique to Minimize Wall Interferences in the NASA Langley 0.3-m Transonic Cryogenic Tunnel.** AIAA Paper No. 88-0140. Presented at the AIAA 26th Aerospace Sciences Meeting, Reno, Nevada, January 11-14, 1988, 11 pp, A88-22101.

2. Wolf, S. W. D.: **Control Software for Two-Dimensional Airfoil Tests using a Self-Streamlining Flexible Walled Transonic Test Section - Semiannual Progress Report to Feb. 1982.** NASA CR-165941, August 1982, 92 pp, N82-30314#.

3. Goodyer, M. J.; and Wolf, S. W. D.: **Development of a Self-Streamlining Flexible Walled Transonic Test Section.** AIAA Paper No. 80-0440-CP. Presented at the AIAA 11th Aerodynamic Testing Conference, Colorado Springs, Colorado, March 18-20, 1980. In: Technical Papers, CP801 (A80-26929), pp 325-335. Also AIAA Journal, Vol. 20, no. 2, February 1982, pp. 227-234, A80-26964#.

4. Wolf, S. W. D.: **The Design and Operational Development of Self-Streamlining Two-Dimensional Flexible Walled Test Sections.** Ph.D. Thesis, University of Southampton. NASA CR-172328, March 1984, 281 pp, N84-22534#.

5. Wolf, S. W. D.; and Goodyer, M. J.: **Predictive Wall Adjustment Strategy for Two-Dimensional Flexible Walled Adaptive Wind Tunnels - A Detailed Description of the First One-Step Method.** NASA CR-181635, January 1988, 28 pp.

6. Judd, M.; Goodyer, M. J.; and Wolf, S. W. D.: **Application of the Computer for on-Site Definition and Control of Wind Tunnel Shape for Minimum Boundary Interference.** Presented at AGARD Specialists' Meeting on "Numerical Methods and Windtunnel Testing," Rhode-St-Genese, Belgium, June 23-24, 1976. In: AGARD CP-210 (N77-11969#), October 1976, Paper No. 6, 14 pp, N77-11975#.

ACKNOWLEGEMENT

This work was done while the author held a National Research Council - NASA Langley Research Center Research Associateship.

The author is most grateful for the support given by Dr. Robert Kilgore and Mr. Edward Ray during this work.

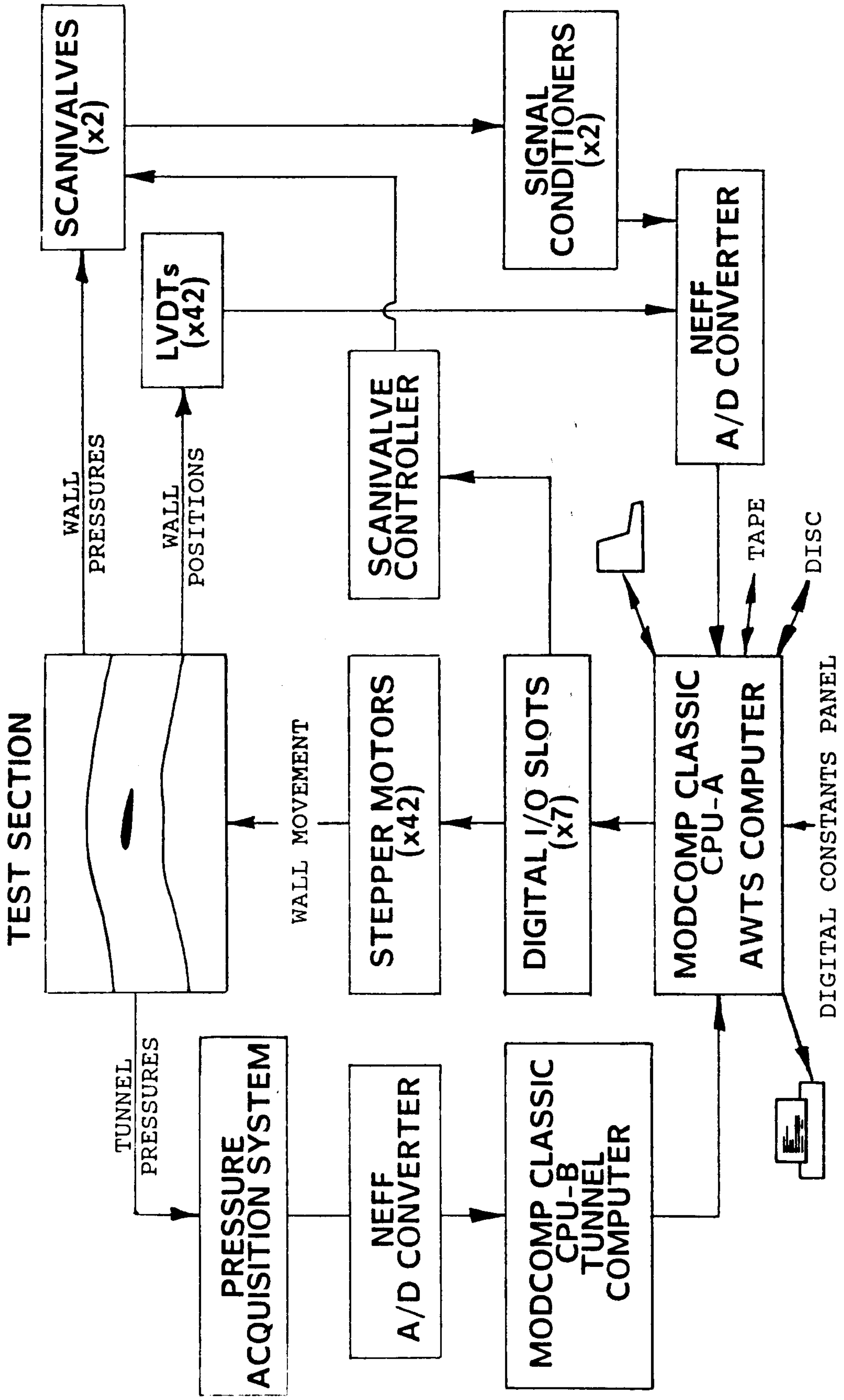


FIG. 1 Flexwall Control System Hardware

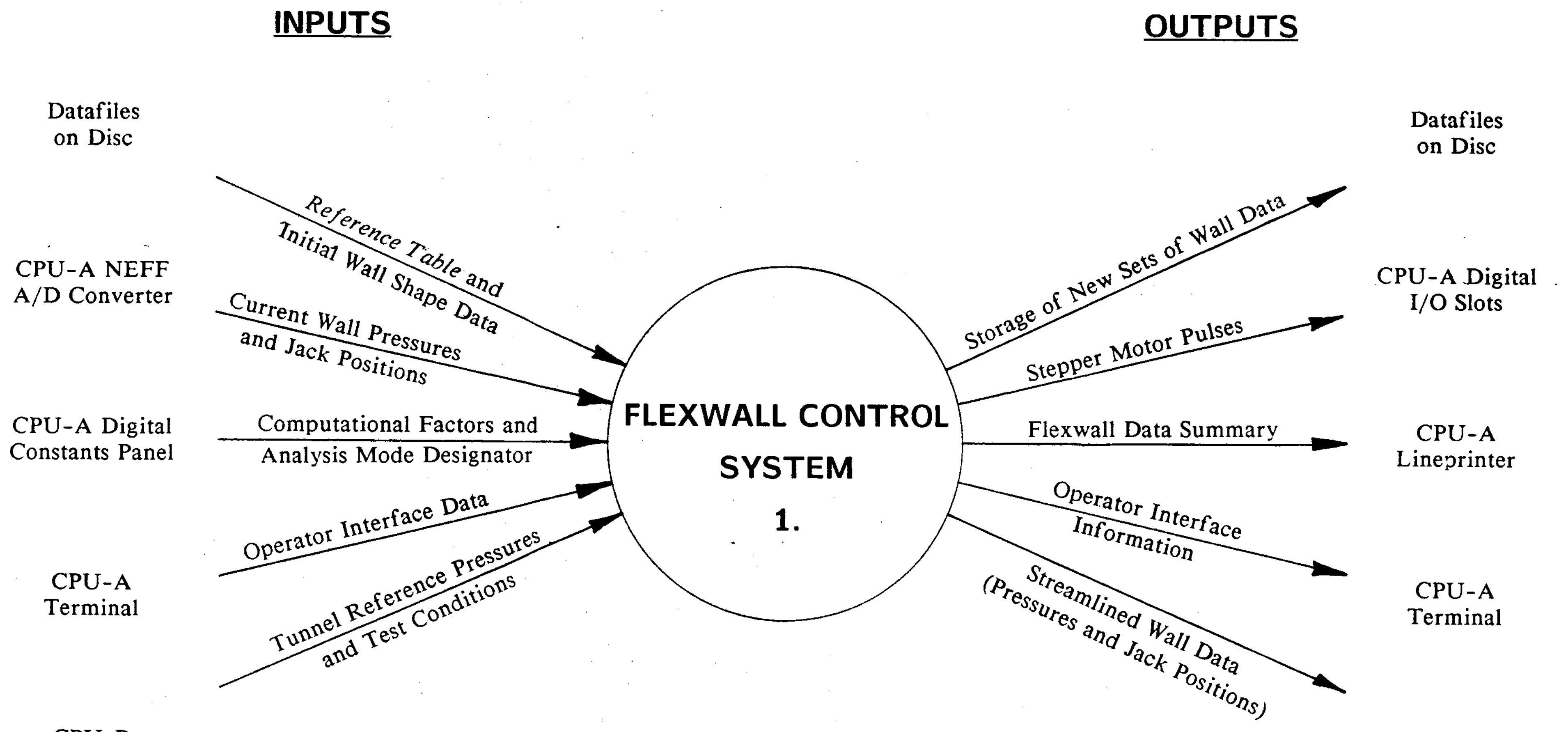

INPUTS
OUTPUTS
Datafiles on Disc
CPU-A NEFF A/D Converter
CPU-A Digital Constants Panel
CPU-A Terminal
CPU-B
Reference Table and Initial Wall Shape Data
Current Wall Pressures and Jack Positions
Computational Factors and Analysis Mode Designator
Operator Interface Data
Tunnel Reference Pressures and Test Conditions
FLEXWALL CONTROL SYSTEM
1.
Storage of New Sets of Wall Data
Stepper Motor Pulses
Flexwall Data Summary
Operator Interface Information
Streamlined Wall Data (Pressures and Jack Positions)
Datafiles on Disc
CPU-A Digital I/O Slots
CPU-A Lineprinter
CPU-A Terminal
CPU-B

FIG. 2 Data Path Chart for the Flexwall Control System

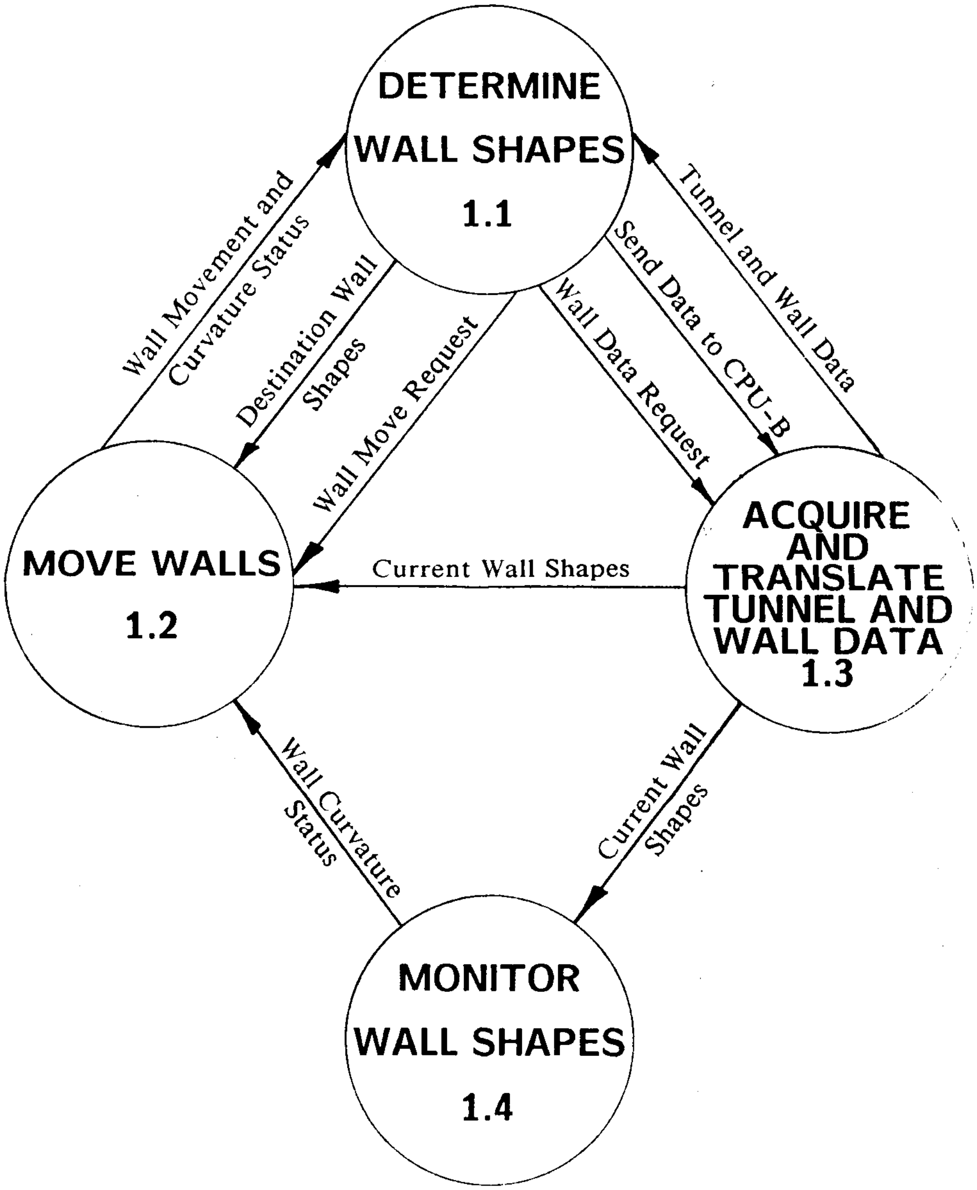


FIG. 3 Functional Elements of the Flexwall Control System

	DATA TYPE	TRANSFER MEDIUM	DATA FLOW	TRANSFER METHOD
Flexwall Control Software Interface	**Wall Movement Request**	FLXCOM	OUT	FSSTAT Bit 7 - Enable Move FLXISW Bit 5 - Initiate Move (Both bits set using system subroutine SETB)
	Wall Movement Status	FLXCOM	IN	MVSTAT - Move Status Word FSSTAT Bit 7 - Move Complete (Pause for bit change using system subroutine RLTBZ4)
	Wall Curvature Status	FLXCOM	IN	MVSTAT - Move Status Word JSTAT - Current Jack Position Status Table
	Destination Jack Positions	FLXCOM	OUT	JPOSN - New (Destination) Jack Positions
	Wall Data Request	Subroutine Call	OUT	CALL FRQST (Flexwall system subroutine)
	Send Data to CPU-B Over the MAXNET Link	FLXCOM	OUT	IACCW - Communications Control Word IASTAT - Communications Status Word FLXISW Bit 7 - Move Status
	Wall Data (Pressures & Jack Positions)	FLXCOM	IN	JPOSA and JPOS - Current Jack Positions WPRS - Wall Pressures
	Tunnel Data (Reference Pressures & Test Conditions)	FLXCOM	IN	XB++++, IB++++, XCHORD - Tunnel Data
CPU-A System Interface	**Digital Constants** (Analysis Mode Designator and Computational Factors)	OAPCOM	IN	NAMES - Names Table for DCP (Get values using system subroutine DIGICO)
	CPU-A Terminal Inputs (Setup Parameters and Operating Mode Designators)	FLXCOM	IN	WASPAR - WAS Setup Parameters FLXOPT - Flexwall Task Option Word (Bits checked using system subroutine TSTB)
	Time Delay	System Calls	IN	CALL DELAY (System subroutine)
	Current Time and Date	System Calls	IN	CALL T2MCAL, TIMEOUT and TIME (System subroutines)

FIG. 4 Data Interface with the Flexwall Control System and the CPU-A System

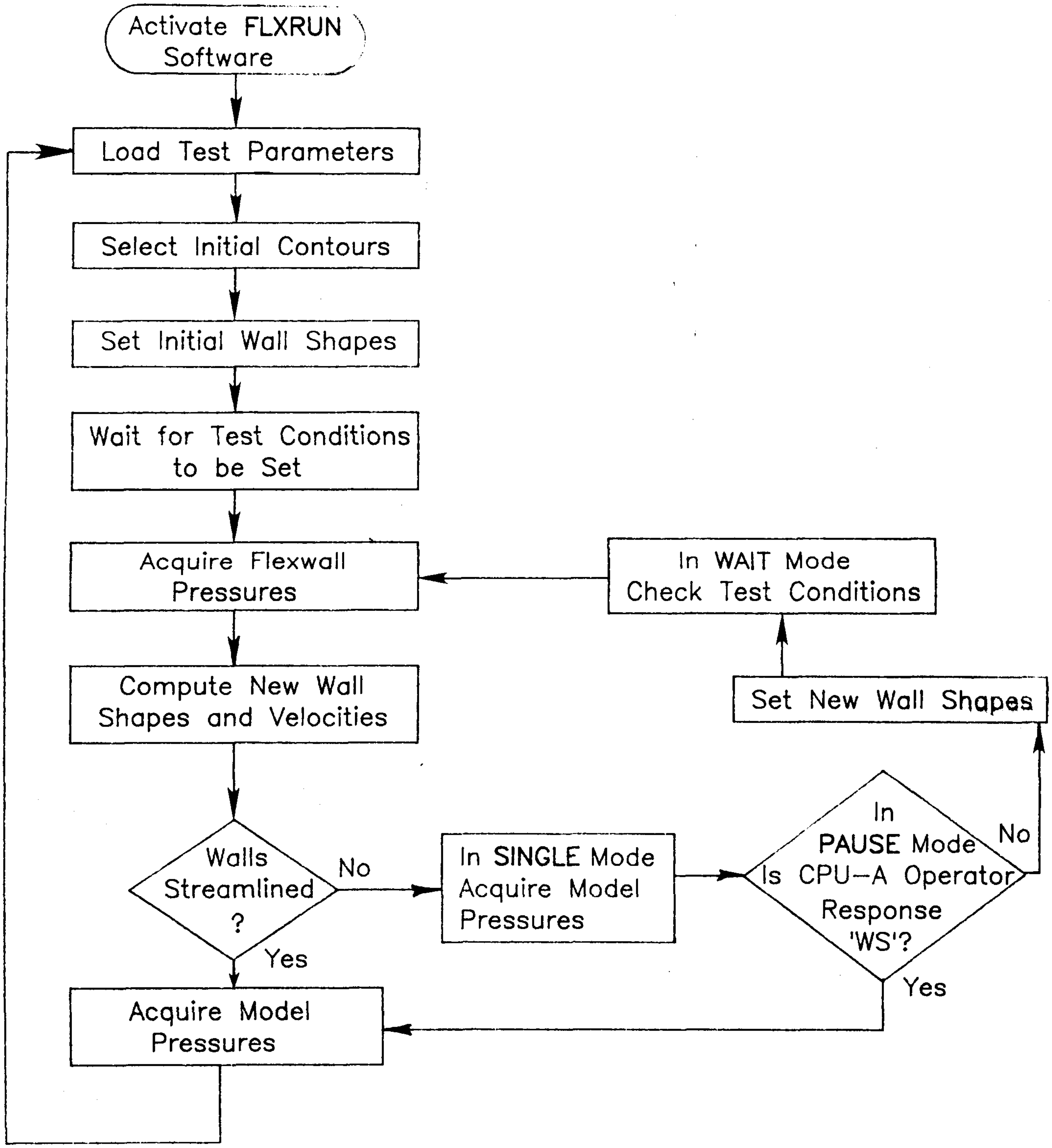

Activate FLXRUN Software
Load Test Parameters
Select Initial Contours
Set Initial Wall Shapes
Wait for Test Conditions to be Set
Acquire Flexwall Pressures
In WAIT Mode Check Test Conditions
Compute New Wall Shapes and Velocities
Set New Wall Shapes
Walls Streamlined ?
No
Yes
In SINGLE Mode Acquire Model Pressures
In PAUSE Mode Is CPU–A Operator Response 'WS'?
No
Yes
Acquire Model Pressures

FIG. 5 Sequence of Events during Wall Adaptation/Streamlining

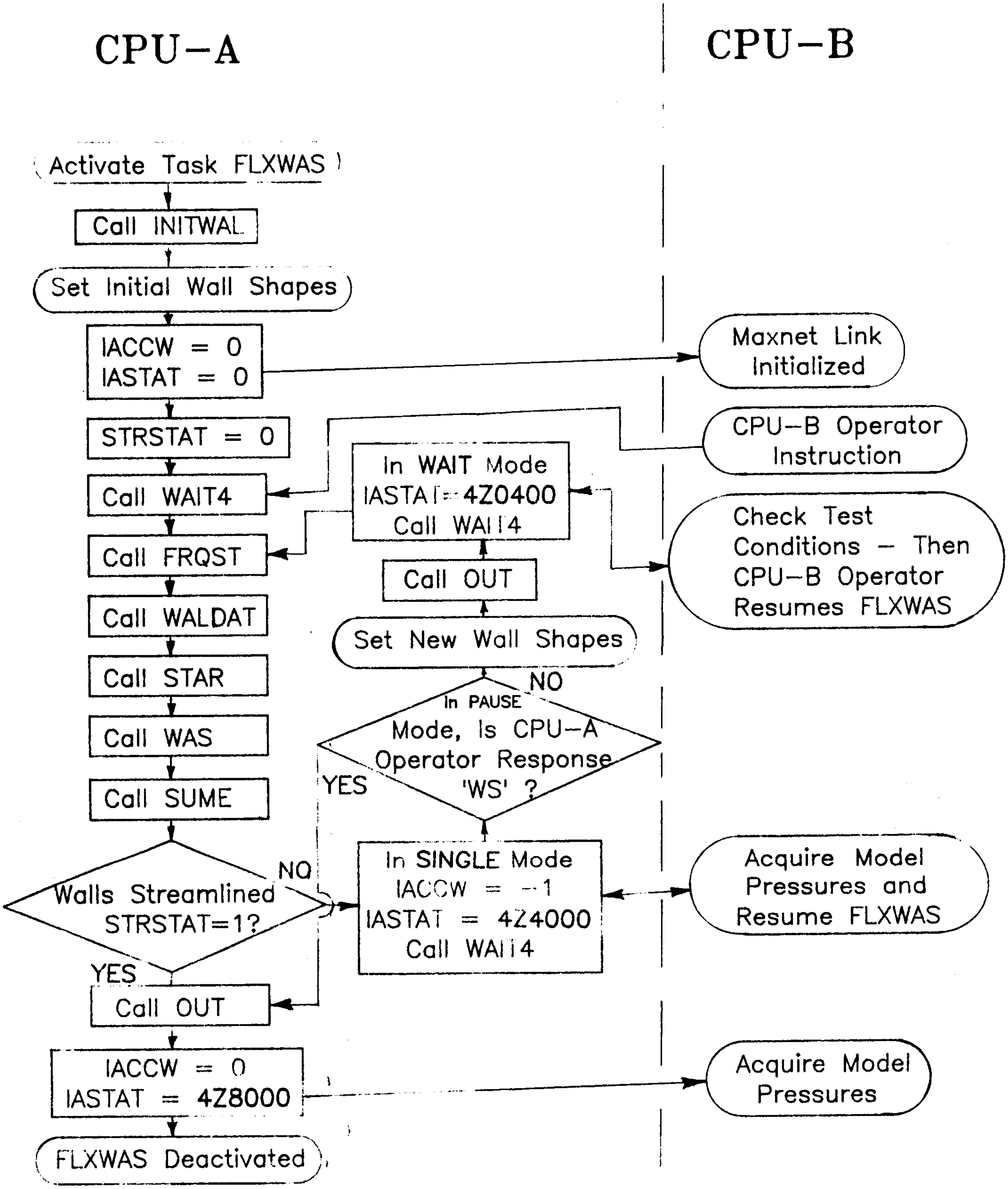
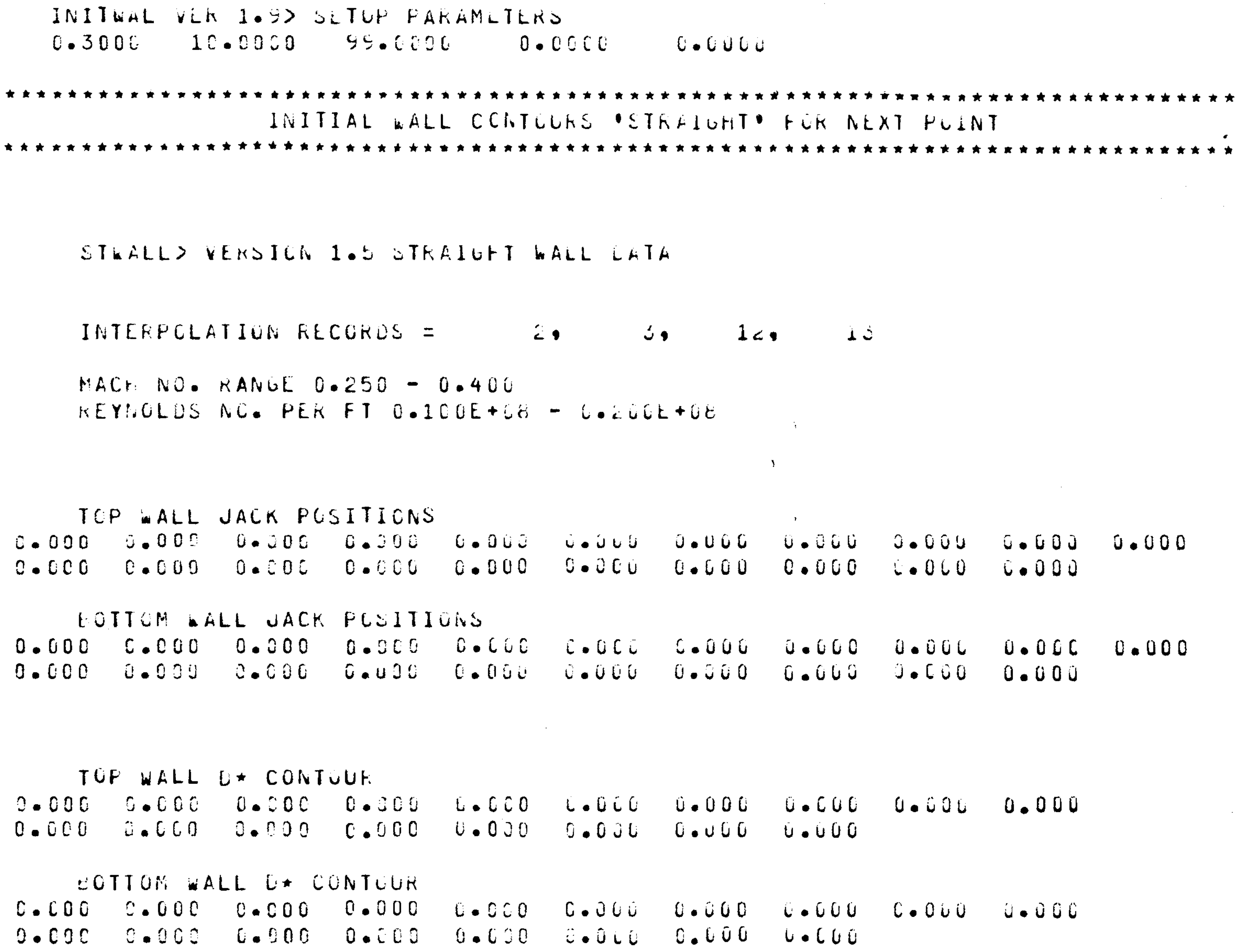


FIG. 6 WAS Software Flow Chart for Wall Adaptation/Streamlining

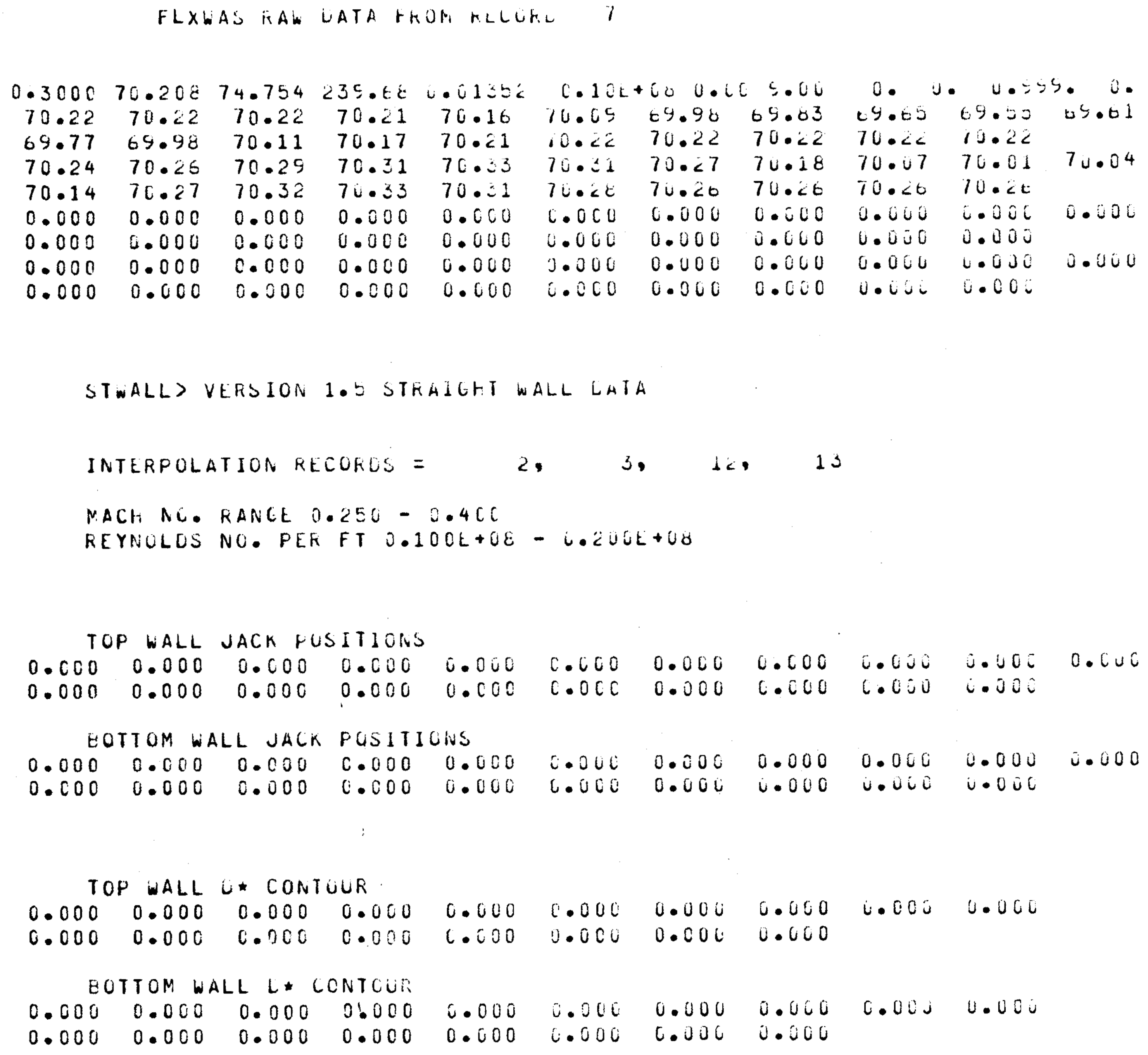
ORIGINAL PAGE IS
OF POOR QUALITY



```
   INITWAL VER 1.9> SETUP PARAMETERS
   0.3000    10.0000    99.0000     0.0000     0.0000

*******************************************************************************************
                         INITIAL WALL CONTOURS 'STRAIGHT' FOR NEXT POINT
*******************************************************************************************


     STWALL> VERSION 1.5 STRAIGHT WALL DATA


     INTERPOLATION RECORDS =        2,      3,     12,     13

     MACH NO. RANGE 0.250 - 0.400
     REYNOLDS NO. PER FT 0.100E+08 - 0.200E+08


     TOP WALL JACK POSITIONS
 0.000   0.000   0.000   0.000   0.000   0.000   0.000   0.000   0.000   0.000   0.000
 0.000   0.000   0.000   0.000   0.000   0.000   0.000   0.000   0.000   0.000

     BOTTOM WALL JACK POSITIONS
 0.000   0.000   0.000   0.000   0.000   0.000   0.000   0.000   0.000   0.000   0.000
 0.000   0.000   0.000   0.000   0.000   0.000   0.000   0.000   0.000   0.000


     TOP WALL D* CONTOUR
 0.000   0.000   0.000   0.000   0.000   0.000   0.000   0.000   0.000   0.000
 0.000   0.000   0.000   0.000   0.000   0.000   0.000   0.000

     BOTTOM WALL D* CONTOUR
 0.000   0.000   0.000   0.000   0.000   0.000   0.000   0.000   0.000   0.000
 0.000   0.000   0.000   0.000   0.000   0.000   0.000   0.000
```

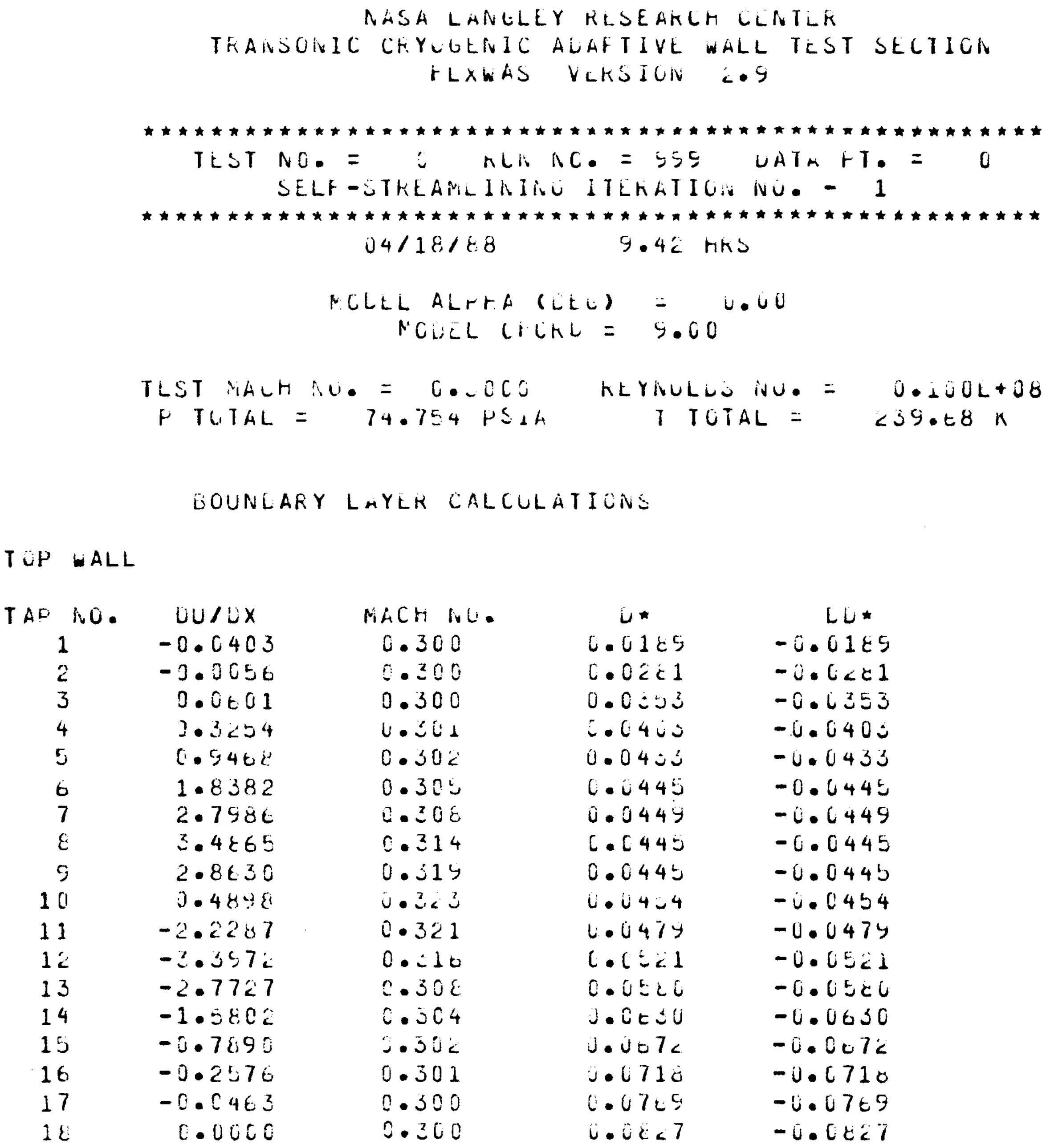
FIG. 7 WAS Software Lineprinter Output for One Iteration of a Streamlining Cycle

ORIGINAL PAGE IS
OF POOR QUALITY



```
        FLXWAS RAW DATA FROM RECORD     7


0.3000 70.208 74.754 239.68 0.01352  0.10E+06 0.00 9.00   0.   0.   0.999.  0.
 70.22  70.22  70.22  70.21  70.16  70.09  69.98  69.83  69.65  69.55  69.61
 69.77  69.98  70.11  70.17  70.21  70.22  70.22  70.22  70.22  70.22
 70.24  70.26  70.29  70.31  70.33  70.31  70.27  70.18  70.07  70.01  70.04
 70.14  70.27  70.32  70.33  70.31  70.28  70.26  70.26  70.26  70.26
 0.000  0.000  0.000  0.000  0.000  0.000  0.000  0.000  0.000  0.000  0.000
 0.000  0.000  0.000  0.000  0.000  0.000  0.000  0.000  0.000  0.000
 0.000  0.000  0.000  0.000  0.000  0.000  0.000  0.000  0.000  0.000  0.000
 0.000  0.000  0.000  0.000  0.000  0.000  0.000  0.000  0.000  0.000


     STWALL> VERSION 1.5 STRAIGHT WALL DATA

     INTERPOLATION RECORDS =       2,     3,    12,    13

     MACH NO. RANGE 0.250 - 0.400
     REYNOLDS NO. PER FT 0.100E+08 - 0.200E+08


     TOP WALL JACK POSITIONS
 0.000  0.000  0.000  0.000  0.000  0.000  0.000  0.000  0.000  0.000  0.000
 0.000  0.000  0.000  0.000  0.000  0.000  0.000  0.000  0.000  0.000

     BOTTOM WALL JACK POSITIONS
 0.000  0.000  0.000  0.000  0.000  0.000  0.000  0.000  0.000  0.000  0.000
 0.000  0.000  0.000  0.000  0.000  0.000  0.000  0.000  0.000  0.000


     TOP WALL Y* CONTOUR
 0.000  0.000  0.000  0.000  0.000  0.000  0.000  0.000  0.000  0.000
 0.000  0.000  0.000  0.000  0.000  0.000  0.000  0.000

     BOTTOM WALL Y* CONTOUR
 0.000  0.000  0.000  0.000  0.000  0.000  0.000  0.000  0.000  0.000
 0.000  0.000  0.000  0.000  0.000  0.000  0.000  0.000
```

FIG. 7 WAS Software Lineprinter Output for One Iteration of a Streamlining Cycle
(Continued)

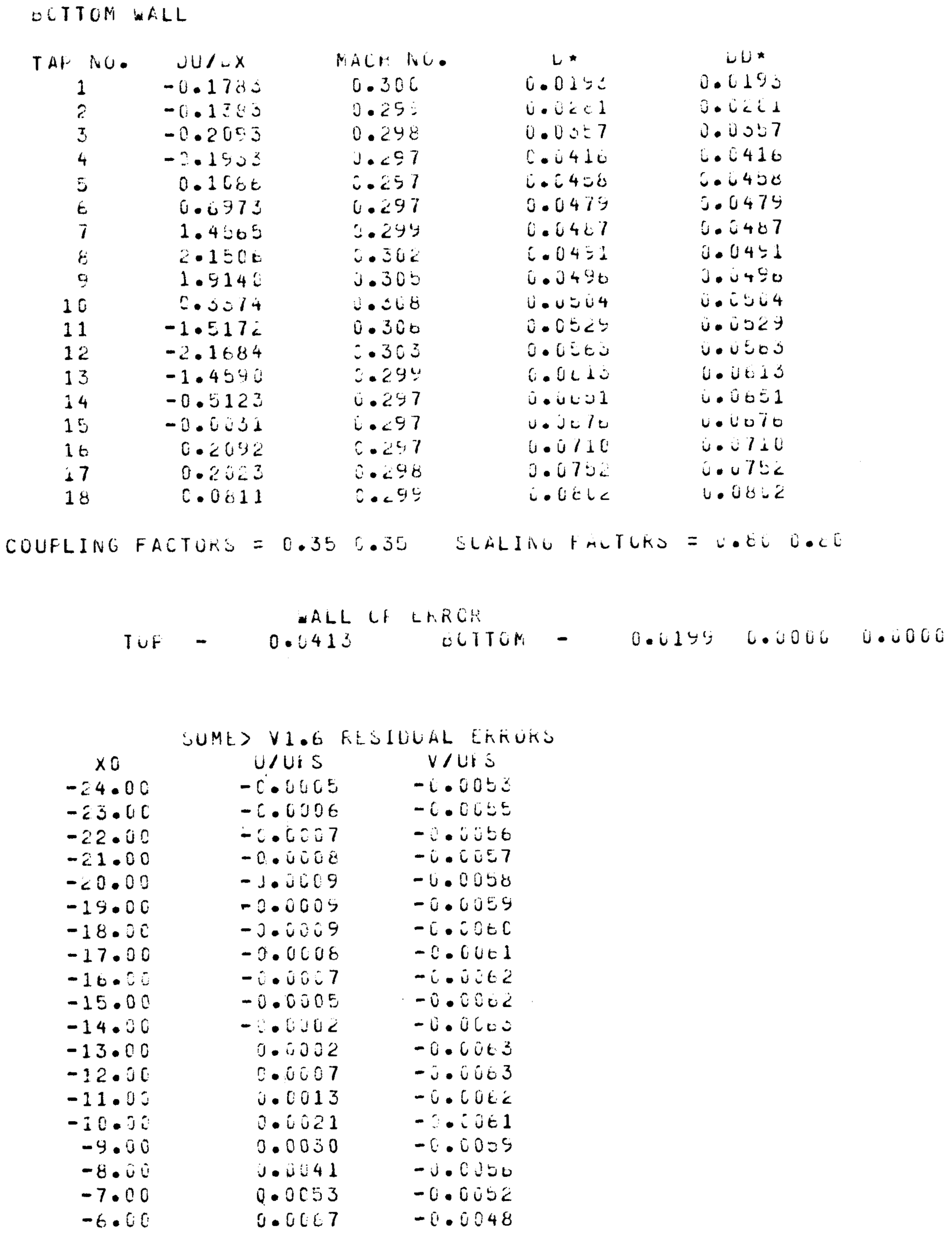


```
                          NASA LANGLEY RESEARCH CENTER
               TRANSONIC CRYOGENIC ADAPTIVE WALL TEST SECTION
                             FLXWAS   VERSION  2.9

          ***************************************************
             TEST NO. =    0    RUN NO. = 559    DATA PT. =    0
                   SELF-STREAMLINING ITERATION NO. -  1
          ***************************************************
                       04/18/88          9.42 HRS

                      MODEL ALPHA (DEG)  =    0.00
                         MODEL CHORD =  9.00

          TEST MACH NO. =  0.3000      REYNOLDS NO. =   0.100E+08
           P TOTAL =    74.754 PSIA        T TOTAL =     239.68 K


             BOUNDARY LAYER CALCULATIONS

 TOP WALL

 TAP NO.     DU/DX         MACH NO.         D*           DD*
    1       -0.0403         0.300         0.0189       -0.0189
    2       -0.0056         0.300         0.0281       -0.0281
    3        0.0601         0.300         0.0353       -0.0353
    4        0.3254         0.301         0.0403       -0.0403
    5        0.9468         0.302         0.0433       -0.0433
    6        1.8382         0.305         0.0445       -0.0445
    7        2.7986         0.308         0.0449       -0.0449
    8        3.4865         0.314         0.0445       -0.0445
    9        2.8630         0.319         0.0445       -0.0445
   10        0.4898         0.323         0.0454       -0.0454
   11       -2.2287         0.321         0.0479       -0.0479
   12       -3.3972         0.316         0.0521       -0.0521
   13       -2.7727         0.308         0.0580       -0.0580
   14       -1.5802         0.304         0.0630       -0.0630
   15       -0.7890         0.302         0.0672       -0.0672
   16       -0.2576         0.301         0.0718       -0.0718
   17       -0.0463         0.300         0.0769       -0.0769
   18        0.0000         0.300         0.0827       -0.0827
```

FIG. 7 WAS Software Lineprinter Output for One Iteration of a Streamlining Cycle (Continued)

ORIGINAL PAGE IS
OF POOR QUALITY

ORIGINAL PAGE IS
OF POOR QUALITY



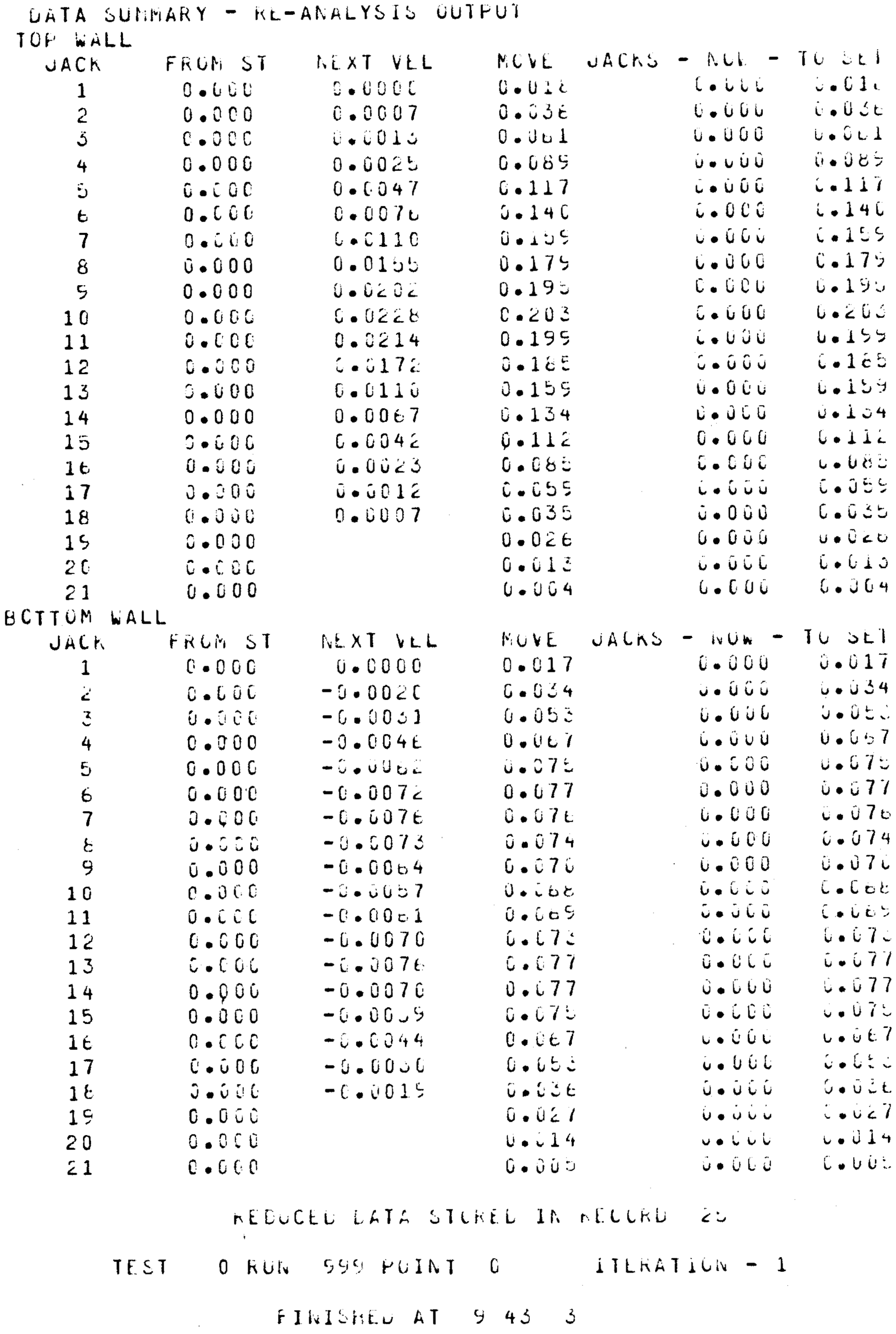
```
BOTTOM WALL

TAP NO.     DU/DX        MACH NO.        D*          DD*
   1       -0.1783        0.300        0.0193       0.0193
   2       -0.1383        0.299        0.0281       0.0281
   3       -0.2093        0.298        0.0357       0.0357
   4       -0.1953        0.297        0.0416       0.0416
   5        0.1066        0.297        0.0458       0.0458
   6        0.6973        0.297        0.0479       0.0479
   7        1.4565        0.299        0.0487       0.0487
   8        2.1506        0.302        0.0491       0.0491
   9        1.9140        0.305        0.0496       0.0496
  10        0.3374        0.308        0.0504       0.0504
  11       -1.5172        0.306        0.0529       0.0529
  12       -2.1684        0.303        0.0563       0.0563
  13       -1.4590        0.299        0.0613       0.0613
  14       -0.5123        0.297        0.0651       0.0651
  15       -0.0031        0.297        0.0676       0.0676
  16        0.2092        0.297        0.0710       0.0710
  17        0.2023        0.298        0.0752       0.0752
  18        0.0811        0.299        0.0802       0.0802

COUPLING FACTORS = 0.35 0.35   SCALING FACTORS = 0.80 0.80


                    WALL CP ERROR
      TOP   -     0.0413       BOTTOM   -     0.0199  0.0000  0.0000


              SUMEX V1.6 RESIDUAL ERRORS
    XO           U/UFS         V/UFS
  -24.00       -0.0005       -0.0053
  -23.00       -0.0006       -0.0055
  -22.00       -0.0007       -0.0056
  -21.00       -0.0008       -0.0057
  -20.00       -0.0009       -0.0058
  -19.00       -0.0009       -0.0059
  -18.00       -0.0009       -0.0060
  -17.00       -0.0008       -0.0061
  -16.00       -0.0007       -0.0062
  -15.00       -0.0005       -0.0062
  -14.00       -0.0002       -0.0063
  -13.00        0.0002       -0.0063
  -12.00        0.0007       -0.0063
  -11.00        0.0013       -0.0062
  -10.00        0.0021       -0.0061
   -9.00        0.0030       -0.0059
   -8.00        0.0041       -0.0056
   -7.00        Q.0053       -0.0052
   -6.00        0.0067       -0.0048
```

FIG. 7 WAS Software Lineprinter Output for One Iteration of a Streamlining Cycle (Continued)

```
 -5.00        0.0081     -0.0042
 -4.00        0.0096     -0.0035
 -3.00        0.0108     -0.0027
 -2.00        0.0119     -0.0019
 -1.00        0.0125     -0.0010
  0.00        0.0128      0.0000
  1.00        0.0126      0.0010
  2.00        0.0115      0.0019
  3.00        0.0108      0.0026
  4.00        0.0096      0.0035
  5.00        0.0081      0.0042
  6.00        0.0067      0.0048
  7.00        0.0053      0.0052
  8.00        0.0041      0.0056
  9.00        0.0030      0.0059
 10.00        0.0020      0.0061
 11.00        0.0013      0.0062
 12.00        0.0007      0.0063
 13.00        0.0002      0.0063
 14.00       -0.0002      0.0063
 15.00       -0.0005      0.0062
 16.00       -0.0007      0.0062
 17.00       -0.0008      0.0061
 18.00       -0.0009      0.0060
 19.00       -0.0009      0.0059
 20.00       -0.0009      0.0058
 21.00       -0.0008      0.0057
 22.00       -0.0007      0.0056
 23.00       -0.0006      0.0055
 24.00       -0.0005      0.0053

 MODEL ERRORS                                  CP

-2.2500       0.0116     -0.0021      -0.0234
-1.1250       0.0125     -0.0011      -0.0251
 0.0000       0.0128      0.0000      -0.0257
 1.1250       0.0125      0.0011      -0.0251
 2.2500       0.0117      0.0021      -0.0234
 3.3750       0.0104      0.0031      -0.0209
 4.5000       0.0089      0.0039      -0.0178
 5.6250       0.0072      0.0046      -0.0145
 6.7500       0.0057      0.0051      -0.0113

     EFFECT                                  DELTA CL

 ALPHA ERROR =  -0.1233 DEGREES             -0.0131

 INDUCED CAMBER =  -0.4254 DEGREES           0.0563

 CP ERROR (1/4 CHORD) = -0.0257
 CP ERROR ( AVERAGE ) = -0.0208              0.0257
```

FIG. 7 WAS Software Lineprinter Output for One Iteration of a Streamlining Cycle (Continued)

ORIGINAL PAGE IS
OF POOR QUALITY



```
 DATA SUMMARY - RE-ANALYSIS OUTPUT
TOP WALL
  JACK      FROM ST    NEXT VEL      MOVE  JACKS - NOW - TO SET
    1       0.000      0.0000       0.018            0.000   0.018
    2       0.000      0.0007       0.036            0.000   0.036
    3       0.000      0.0013       0.061            0.000   0.061
    4       0.000      0.0025       0.089            0.000   0.089
    5       0.000      0.0047       0.117            0.000   0.117
    6       0.000      0.0076       0.140            0.000   0.140
    7       0.000      0.0110       0.159            0.000   0.159
    8       0.000      0.0155       0.179            0.000   0.179
    9       0.000      0.0202       0.193            0.000   0.193
   10       0.000      0.0228       0.203            0.000   0.203
   11       0.000      0.0214       0.199            0.000   0.199
   12       0.000      0.0172       0.165            0.000   0.165
   13       0.000      0.0110       0.159            0.000   0.159
   14       0.000      0.0067       0.134            0.000   0.134
   15       0.000      0.0042       0.112            0.000   0.112
   16       0.000      0.0023       0.085            0.000   0.085
   17       0.000      0.0012       0.059            0.000   0.059
   18       0.000      0.0007       0.035            0.000   0.035
   19       0.000                   0.026            0.000   0.026
   20       0.000                   0.013            0.000   0.013
   21       0.000                   0.004            0.000   0.004
BOTTOM WALL
  JACK      FROM ST    NEXT VEL      MOVE  JACKS - NOW - TO SET
    1       0.000      0.0000       0.017            0.000   0.017
    2       0.000     -0.0020       0.034            0.000   0.034
    3       0.000     -0.0031       0.053            0.000   0.053
    4       0.000     -0.0046       0.067            0.000   0.067
    5       0.000     -0.0062       0.075            0.000   0.075
    6       0.000     -0.0072       0.077            0.000   0.077
    7       0.000     -0.0076       0.076            0.000   0.076
    8       0.000     -0.0073       0.074            0.000   0.074
    9       0.000     -0.0064       0.070            0.000   0.070
   10       0.000     -0.0057       0.068            0.000   0.068
   11       0.000     -0.0061       0.069            0.000   0.069
   12       0.000     -0.0070       0.073            0.000   0.073
   13       0.000     -0.0076       0.077            0.000   0.077
   14       0.000     -0.0070       0.077            0.000   0.077
   15       0.000     -0.0059       0.075            0.000   0.075
   16       0.000     -0.0044       0.067            0.000   0.067
   17       0.000     -0.0030       0.053            0.000   0.053
   18       0.000     -0.0019       0.036            0.000   0.036
   19       0.000                   0.027            0.000   0.027
   20       0.000                   0.014            0.000   0.014
   21       0.000                   0.005            0.000   0.005

                 REDUCED DATA STORED IN RECORD  25

        TEST    0 RUN   999 POINT   0        ITERATION - 1

                   FINISHED AT  9 43  3
```

FIG. 7 WAS Software Lineprinter Output for One Iteration of a Streamlining Cycle (Concluded)

APPENDIX A

WAS SOFTWARE LISTINGS

APPENDIX A.1 - LISTING OF PROGRAM FLXWAS

```
      PROGRAM FLXWAS
C*****************************************************************
C
C
C     TITLE:          WALL ADJUSTMENT STRATEGY CONTROL TASK
C
C     AUTHOR:         STEPHEN W.D. WOLF , NATIONAL RESEARCH COUNCIL
C
C     SYSTEM:         CRYO-A
C
C
C     PURPOSE:        SEQUENCE STREAMLINING CYCLE EVENTS FOR THE
C                     SET OPERATING MODE AND ANALYSIS MODE. ALSO
C                     EXIT THE TASK FROM THE SYSTEM FOR WALLS
C                     STREAMLINED AND VARIOUS ERROR CONDITIONS
C
C
C     REVISION HISTORY:
C
C     VERSION        DATE         DESCRIPTION
C     =======        ====         ========================================
C       1.1        06/13/85       INITIAL VERSION DEVELOPED BY STEPHEN WOLF
C                                 AT THE UNIVERSITY OF SOUTHAMPTON (UK) AND
C                                 ORIGINALLY INSTALLED ON THE CRYO-A SYSTEM
C                                 AT LARC DURING FALL 1982.
C       1.2        08/29/85       AUTO WALL DATA SELECTION AND STORAGE OF
C                                 RAW AND REDUCED DATA ON DISK INCLUDED FOR
C                                 EACH STREAMLINING CYCLE
C       1.3        02/12/86       INTEGRATED WITH FLXRUN
C       1.4        02/14/86       INTEGRATED WITH "FMVSTR"
C       1.5        03/18/86       *SWDW* SCANIVALVE SWITCHING REMOVED
C       1.6        03/27/86       *SWDW* INTRODUCE SOFTWARE AND JACK LIMIT
C                                        ERROR EXIT CONDITIONS
C       1.7        03/28/86       *SWDW* MAKE ALLOWANCES FOR CPU-B ABORT
C       1.8        04/02/86       *SWDW* DEVELOPMENT VERSION OF FLXWAS
C       1.9        04/11/86       *SWDW* ALLOW WALL MOVEMENT  UP TO
C                                        TEN ITERATIONS
C       2.0        04/17/86       *SWDW* INITIALISE IACCW
C       2.1        04/23/86       *SWDW* CHANGE 'WAS' ANALYSIS HOLD
C       2.2        04/25/86       *SWDW* OUTPUT JSTAT WHEN HARDWARE ERROR
C       2.3        05/02/86       *SWDW* REPOSITION CALL TO "OUT" AND
C                                        ADJUST CORE DEVICE OUTPUT
C       2.4        05/15/86       *SWDW* REMOVE FIXED MODEL CHORD
C       2.5        05/23/86       *SWDW* ENSURE CHORD READ ON FIRST PASS
C                                        AND MODIFY OUT CALLS
C       2.6        10/23/86       *SWDW* OUTPUT CORE DEVICE ON FIRST PASS
C                                        ONLY AND PRINT JACK STATUS ON CONSOLE
C       2.7        12/31/86       *SWDW* CLEAN UP RE-ANALYSIS MODE OPERATIONS
C       2.8        05/06/87       *SWDW* JACK STATUS ENHANCEMENT
C
C       2.9        01/26/87       *EJW* ADIGICO CHANGED TO DIGICO
C                                       AOAP CALL NAMES AS ALTERED FOR USE
C                                       ON CDC DISK DRIVE.
C
C       3.0        6/15/88        CHANGED INCLUDE OF OAPCM FROM USL TO
C                                 PSL. ERIC PLUS MADE MINOR CHANGES TO
C                                 OUTPUT MESSAGES.
C
```

```
C
C
C
C***********************************************************************
C
C     COMPILER OPTIONS:             $23
C
C     LINKER OPTIONS:               ATTR OAPCOM,REL,IMAP,NOR,IPR
C                                   COM /OAPCOM,5376/OAPCOM
C                                   ATTR FLXCOM,REL,OMAP,NOR,IPR
C                                   COM /FLXCOM,1024/ FLXCOM
C                                   ATTR WASCOM,REL,OMAP,NOR,IPR
C                                   COM /WASCOM,1131/ WASCOM
C
C     TASK OVERLAYER OPTIONS:       PAGESHARE #03
C
C
C     EXECUTION INSTRUCTIONS:
C                   THIS TASK IS ESTABLISHED BY TASK FLXINI,THEN
C                   ACTIVATED BY TASK FLXCTL
C
C
C     LOGICAL FILES USED:
C                   CO - INPUT/OUTPUT TO CONSOLE
C                   LO - OUTPUT TO THE LINEPRINTER
C
C     SHARED REGIONS:
C         AOAPC     (Inserted, no restrictions
C         FLXCOM  (Inserted, no restrictions)
C         WASCOM    (Inserted, no restrictions)
C
C
C     User-Defined Subroutine Calls:
C         INITWAL
C         WALDAT
C         STAR
C         WAS
C         SUME
C         OUT
C         FRQST(Flexwall control system subroutine)
C
C
C     System Subroutine Calls:
C         DIGICO - Reads digital constants from data common OAPCM.
C         DELAY  - Inserts a delay of in the task execution.
C         WAIT4  - Causes task to go into a pause state.
C         SETB   - Sets selected bits in control words.
C         TSTB   - Tests state of selected bits in control words.
C         RLTBZ4 - Wait for selected bit to change state.
C         TIMEOUT, TM2CAL and TIME - Read current date and time.
C
C     Inputs:
C         /OAPCOM/ NAME
C     Contents of array NAME become computational factors TWCPLF, BWCPLF,
C     TWSF, BWSF and WMOVF, and the analysis mode designator IANAL (via
C     variable ANAL).
C
C
```

```
C           /FLXCOM/ IBTSTNR, IBRUNNR, IBPTNR, XCHORD, XBMACH, XBPS, XBPT,
C       XBTT, XBRHOS, XREYNO, XBAOA, FLXISW, FSSTAT, FLXOPT, MVSTAT, WASPAR,
C       WPRS, JSTAT, JPOSA
C
C       IBTSTNR, IBRUNNR, IBPTNR, XCHORD, XBMACH, XBPS, XBPT, XBTT, XBRHOS,
C       XREYNO, XBAOA are all wind tunnel test conditions.
C       FLXISW - Flexwall interrupt status word - Bit 7 indicates move
C                status.
C       FSSTAT - Flexwall system status word - Flexwall hardware status.
C       FLXOPT - Flexwall task option word - Set by using task FLXOP.
C       MVSTAT - Move status word - Task FMVSTR information on flexwall move.
C       WASPAR - Wall alignment strategy setup test parameters - Set by using
C                task FLXOP.
C       WPRS - Array of wall pressures (PSIA).
C       JSTAT - Wall jack status table.
C       JPOSA - Array of averaged current jack positions (Inches).
C
C       Output:
C           /FLXCOM/ STRSTAT, FLXISW, FSSTAT, IACCW, IASTAT, JPOSN, JPOSA,
C       WPRS
C
C       STRSTAT - Streamline status word.
C       FLXISW - Flexwall interrupt status word - Bit 5 initiates flexwall
C                movement.
C       FSSTAT - Flexwall system status word - Bit 7 enables flexwall
C                movement.
C       IACCW -  Communications control word.
C       IASTAT - Communications status word.
C       JPOSN - Array of new (destination) jack positions (Inches).
C       JPOSA - Array of averaged current jack positions (Inches).
C       WPRS - Array of wall pressures (PSIA).
C
C           /WASCOM/ NOJACK, NOCPT, XJACK, WL, IANAL, FMACH, PSTATIC,
C       PTOTAL, TTOTAL, DENSITY, CREY, ALPHA, CHORD
C
C       Output Messages:
C                         NASA LANGLEY RESEARCH CENTER
C                  TRANSONIC CRYOGENIC ADAPTIVE WALL TEST SECTION
C                              FLXWAS VERSION 3.0
C
C             ******   FLEXWALL CONTOURS INITIALIZED   ******
C           ============  READY TO STREAMLINE   ===========
C
C           FLXWAS>  ***   WAIT FOR TEST CONDITION CHECK   ***
C
C                ***   ACQUIRING FLEXWALL PRESSURES   ***
C
C           FLXWAS> ***   SINGLE ITERATION COMPLETE   ***
C
C           FLXWAS>  *****   WAIT FOR ANALYSIS CHECK   *****
C       TYPE IN ANY CHARACTER TO CONTINUE OR [WS] FOR WALLS S/LINED
C
C           **********************************
C           *     STREAMLINING TERMINATED    *
C           *         HARDWARE ERROR         *
C           **********************************
C                       JACK STATUS
C                  #     TOP     BOTTOM
```

```
C               (Data tabulated below)
C                N      X          Y
C       PRESS RETURN TO RECALL 'FLXOP' MENU FOR NEXT PT
C
C           ***************************
C           *    WALLS STREAMLINED    *
C           ***************************
C
C           ***********************************
C           *      STREAMLINING TERMINATED    *
C           *          SOFTWARE ERROR         *
C           ***********************************
C
C           ***********************************
C           *     STREAMLINING TERMINATED     *
C           *       JACK LIMIT REACHED        *
C           ***********************************
C                    JACK STATUS
C                #     TOP     BOTTOM
C                (Data tabulated below)
C
C       FLXWAS>  ***  RE-ANALYSIS COMPLETED  ***
C
C       Processing:
C           1) Load fixed tunnel data into shared region WASCOM.
C           2) In re-analysis mode, jump to 13.
C           3) Initialize MAXNET link.
C           4) Select initial wall contours for next data point (Subroutine
C              INITWAL).
C           5) Set chord for empty test section streamlining.
C           6) Move flexwalls to initial contours.
C           7) Wait for CPU-B Operator to initiate the streamlining cycle.
C           8) DO - Enter streamlining cycle.
C           9) In Wait mode, wait for test condition update.
C           10) Read Digital Constants Panel (Subroutine DIGICO).
C           11) Acquire data from the flexwall test section (Subroutine
C               FRQST).
C           12) Re-locate tunnel data in WASCOM.
C           13) In re-analysis mode, load tunnel data from disk (Subroutine
C               WALDAT).
C           14) Select ''Straight Wall'' reference contours for next
C               iteration (Subroutine WALDAT).
C           15) Output raw data to disk if required (Subroutine WALDAT).
C           16) Compute wall boundary layers (Subroutine STAR).
C           17) Compute new wall contours for streamlining (Subroutine WAS).
C           18) Sum residual wall interferences (Subroutine SUME).
C           19) Check streamlining status.
C           20) If walls are streamlined, output reduced data information and
C               jump to 30.
C           21) In Single Iteration mode, set data acquisition bit and the
C               wait to continue.
C           22) Printout FLXCOM data on first pass only.
C           23) Dump core device to the lineprinter on the first pass only.
C           24) Output reduced data to lineprinter and disk (Subroutine OUT).
C           25) In Pause mode, wait for response from CPU-A operator.
C           26) If excessive iterations, abort streamlining, jump to 30.
C           27) In re-analysis mode, jump to 31.
C           28) Move flexwalls to new contours for next iteration.
```

```
C         29) ENDDO - Continue the streamlining cycle.
C         30) Exit conditions.
C         31) Exit the system.
C
C
C     Exception Handling:
C
C         Whenever an error condition is encountered (STRSTAT = -1) an
C     appropriate message is displayed on the operator console and
C     lineprinter, IASTAT is set appropriately, CPU-B is notified, and
C     FLXWAS stops executing.
C
C         If the CPU-A operator enters a non-numeric response when a
C     numeric answer is required, no error message is written.  Instead,
C     the task waits until a valid response is supplied.
C
C************************************************************************
C
C
C
C     ***  DECLARATIONS  ***
C
      INTEGER B1,B2,B3,B4,B5,B6,B7,B8
      INTEGER CO,PRIOR,ARRAY
      INTEGER*4 NAME
      LOGICAL*2 TSTB
      REAL*4 JKPOSN(42), JKPOS(42)
      INCLUDE USL/FLXTYP,NOLIST
      DIMENSION DB(23),IDAT(8),IDATE(8),ITIME(3),NAME(2,6),ISTART(3)
      DIMENSION ARRAY(256)
      DIMENSION IFIN(3),JNO(19)
C
C     ***   COMMON   ***
C
      INCLUDE PSL/OAPCM,NOLIST
      INCLUDE USL/FLXCOM,NOLIST
      INCLUDE USL/WASCOM,NOLIST
C
C     *** EQUIVALENCES ***
C
      EQUIVALENCE (JPOSN,JKPOSN), (JPOS,JKPOS)
      EQUIVALENCE (ARRAY(1),IASTAT)
C
C     ****  INITIALIZATION  ****
C
      DATA CO/@CO/,LO/@LO/
      DATA B1,B2,B3,B4,B5,B6,B7,B8/1,2,3,4,5,6,7,8/
      DATA NAME/@TWCPLF,0,@BWCPLF,0,@TWSF,0,@BWSF,0,@IANAL,0,@WMOVF,0/
      DATA JNO/2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20/
      DATA VER/3.0/,PRIOR/255/
C************************************************************************
C
C              0.3-M TCT TEST SECTION GEOMETRY
C
C************************************************************************
C
C     STREAMWISE LOCATION OF WALL COMPUTING POINTS RELATIVE
C     TO THE FLEXWALL ANCHOR POINT (INCHES)
```

```
C
      DATA DB/-4.75,0.,4.75,10.5,15.5,19.5,22.5,24.5,26.,27.5,29.,30.5
     +,32.,33.5,35.5,37.5,39.5,42.5,46.5,51.5,56.5,61.5,67./
C
C     TOTAL NO. OF WALL JACKS
C
      NOJACK=21
C
C     NO. OF WALL COMPUTING POINTS
C
      NOCPT=22
C***********************************************************************
      DO 5 J=1,23
      XJACK(J)=DB(J)
5     CONTINUE
      DO 15 J = 1,NOJACK
      K = J + 2
      WL(J) = (XJACK(K+1)-XJACK(K-1))/2
15    CONTINUE
      STRSTAT = 0
      CALL DIGICO (NAME(1,5),ANAL,IERROR)
      IANAL = ANAL
      IF (IANAL.EQ.2) GO TO 50
C***********************************************************************
C
C     INITIALIZE MAXNET LINK
C
      IACCW = 0
      IASTAT = 0
      CALL SETB(FLXISW,B7)
C***********************************************************************
35    FORMAT(1H1,///23X," NASA LANGLEY RESEARCH CENTER"/14X,
     + " TRANSONIC CRYOGENIC ADAPTIVE WALL TEST SECTION")
C************************************************************
      CALL DELAY (0,100,.FALSE.,.FALSE.,IERR,.TRUE.)
      WRITE(CO,35)
45    FORMAT(28X,"FLXWAS  VERSION ",F4.1/)
      WRITE(CO,45) VER
C
C     SELECT INITIAL WALL CONTOURS FOR NEXT DATA POINT
C
      CALL FRQST(4Z2000,AQBUSY)
      CHORD = XCHORD
C
C     SET CHORD FOR EMPTY TEST SECTION STREAMLINING
C
50    IF(IANAL.EQ.1.OR.IANAL.EQ.2)CHORD = 12.
C      CALL INITWAL(JPOSN,TOPY,BOTY,TWVEL,BWVEL,STRSTAT)
      CALL INITWAL
      IF(STRSTAT.EQ.-1) GO TO 270
      IF (IANAL.EQ.2) GO TO 80
C
C     MOVE FLEXWALLS TO INITIAL CONTOURS
C
      MVSTAT = -1
      CALL SETB(FSSTAT,B7)
      CALL SETB(FLXISW,B5)
      CALL RLTBZ4(FSSTAT,B7)
      IF (MVSTAT.EQ.4.OR.MVSTAT.EQ.5) GO TO 280
      IF (MVSTAT.NE.1) GO TO 250
      ITERATION = 0
      WRITE(CO,55)
55    FORMAT(//12X,"   ******   FLEXWALL CONTOURS INITIALIZED   ******"
     + /13X,"  ===========   READY TO STREAMLINE   ===========")
C
C     WAIT FOR CPU-B OPERATOR TO INITIATE THE STREAMLINING CYCLE
C
      CALL WAIT4
C************************************************************************
C
C                         ENTER STREAMLINING CYCLE
C
C************************************************************************
      IF (ITERATION.EQ.0) CALL TIME(ISTART)
      IF (ITERATION.EQ.0) GO TO 70
C
C     IN WAIT MODE, WAIT FOR TEST CONDITION UPDATE
C
60    IF(.NOT.TSTB(FLXOPT,B5)) GO TO 80
      IASTAT=4Z0400
      CALL SETB(FLXISW,B7)
      WRITE(CO,65)
65    FORMAT(" FLXWAS>    ***  WAIT FOR TEST CONDITION CHECK  ***")
      CALL WAIT4
70    CONTINUE
C
C       READ DIGITAL CONSTANTS PANEL
C
80    CALL DIGICO(NAME(1,1),TWCPLF,IERROR)
      CALL DIGICO(NAME(1,2),BWCPLF,IERROR)
      CALL DIGICO(NAME(1,3),TWSF,IERROR)
      CALL DIGICO(NAME(1,4),BWSF,IERROR)
      CALL DIGICO(NAME(1,5),ANAL,IERROR)
      CALL DIGICO(NAME(1,6),WMOVF,IERROR)
      IANAL = ANAL
      IF (IANAL.EQ.2) GO TO 90
C
C             IANAL CODE TABLE
C
C     0 = STREAMLINING ANALYSIS WITH REDUCED PRINT-OUT
C     1 = STRAIGHT WALL ANALYSIS
C     2 = RE-ANALYSIS
C     3 = STREAMLINING ANALYSIS WITH FULL PRINT-OUT
C
C *****************************************************
C *                                                   *
C *   ACQUIRE DATA FROM THE FLEXWALL TEST SECTION     *
C *                                                   *
C *****************************************************
      WRITE(CO,85)
85    FORMAT(6(/),16X"   ***  ACQUIRING FLEXWALL PRESSURES  ***",6(/))
      CALL FRQST(4Z7800,AQBUSY)
C****************************************************
C
C     RE-LOCATE TUNNEL DATA IN WASCOM
C
```

```
      FMACH = XBMACH
      PSTATIC = XBPS
      PTOTAL = XBPT
      TTOTAL = XBTT
      DENSITY = XBRHOS
      CREY = XREYNO*1E06
      ALPHA = XBAOA
      IF (IANAL .EQ. 1) ALPHA = 0.0
      CHORD = XCHORD
C****************************************************
      ITERATION = ITERATION + 1
90    CALL T2MCAL(IDAT,IERR)
      CALL TIMOUT(IDAT,IDATE)
      CALL TIME(ITIME)
C
C     SELECT "STRAIGHT WALL" REFERENCE CONTOURS FOR CURRENT ITERATION
C              AND OUTPUT RAW DATA TO DISK IF REQUIRED
C
C     CALL WALDAT(TWY,BWY,DSTAS,BSBAS,TOPWP,BOTWP,STRSTAT)
      CALL WALDAT
C
C      PRINT HEADER INFORMATION
C
      IF(STRSTAT.EQ.-1) GO TO 270
      WRITE(LO,35)
      WRITE(LO,45) VER
125   FORMAT(11X,53("*")
     +,/12X,"  TEST NO. =",I4,2X," RUN NO. =",I4,2X," DATA PT. =",I4)
135   FORMAT (18X," SELF-STREAMLINING ITERATION NO. - ",I2)
145   FORMAT(11X,"*****************************************************"
     +/24X,4A2,6X,I2,'.',I2,' HRS')
155   FORMAT(/21X," MODEL ALPHA (DEG)  =  ",F5.2/25X," MODEL CHORD = ",
     +F5.2)
165   FORMAT(/10X," TEST MACH NO. = ",F7.4,2X,"  REYNOLDS NO. = ",E11.3)
175   FORMAT(11X," P TOTAL = ",F8.3," PSIA",5X," T TOTAL = ",F9.2," K")
      WRITE(LO,125) IBTSTNR,IBRUNNR,IBPTNR
      WRITE(LO,135) ITERATION
      WRITE(LO,145)IDATE(1),IDATE(2),IDATE(3),IDATE(4),ITIME(1),ITIME(2)
      WRITE(LO,155) ALPHA,CHORD
      WRITE(LO,165) FMACH,CREY
      WRITE(LO,175) PTOTAL,TTOTAL
C
C     COMPUTE WALL BOUNDARY LAYERS
C
C      CALL STAR(DST,DSB)
      CALL STAR
C
C     COMPUTE NEW WALL CONTOURS FOR STREAMLINING
C
C      CALL WAS(JPOSN)
      CALL WAS
C
C     SUM RESIDUAL WALL INTERFERENCES
C
C      CALL SUME(STRSTAT)
      CALL SUME
      IF (IANAL.EQ.2) GO TO 180
C
```

```
C     CHECK STREAMLINING STATUS
C
      IF (STRSTAT.EQ.-1) GO TO 270
      IF (STRSTAT.NE.1)  GO TO 220
C
C     OUTPUT REDUCED DATA TO LINEPRINTER AND DISK
C             IF WALLS ARE STREAMLINED
C
C      CALL OUT (STRSTAT)
180   CALL OUT
      CALL TIME(IFIN)
      WRITE(LO,195)IBTSTNR,IBRUNNR,IBPTNR,ITERATION
195   FORMAT(/9X," TEST ",I3," RUN",I5," POINT",I3,5X," ITERATION -",I2)
      IF (IANAL.NE.2) WRITE(LO,205) ISTART(1),ISTART(2),ISTART(3)
205   FORMAT(//10X,' S/LINING INITIATED AT ',3(I2,' '))
      WRITE(LO,215) IFIN(1),IFIN(2),IFIN(3)
215   FORMAT(/13X,'        FINISHED AT ',3(I2,' '))
      IF (IANAL.EQ.2) GO TO 290
      GO TO 221
220   END FILE LO
C
C     IN SINGLE ITERATION MODE, SET DATA ACQUISITION BIT
C               AND THEN WAIT TO BE CONTINUED
C
      IF(.NOT.TSTB(FLXOPT,B4)) GO TO 231
C
C     PRINTOUT FLXCOM DATA ON FIRST PASS ONLY
C
221   IF (IPASS .EQ. 1) GO TO 229
      WRITE(LO,35)
      WRITE(LO,45) VER
      WRITE(LO,125) IBTSTNR,IBRUNNR,IBPTNR
      WRITE(LO,145)IDATE(1),IDATE(2),IDATE(3),IDATE(4),ITIME(1),ITIME(2)
222   FORMAT(/,27X," TOP WALL DATA"/
     +19X," JACK",6X," JPOSA",7X," WPRS"/)
      WRITE(LO,222)
      WRITE(LO,223)(I,JPOSA(I,1),WPRS(I,1),I=1,21)
223   FORMAT(19X,I4,2F13.5)
      I = 22
      WRITE(LO,224)I,WPRS(I,1)
224   FORMAT(19X,I4,13X,F13.5)
225   FORMAT(/////,25X," BOTTOM WALL DATA"/
     +19X," JACK",6X," JPOSA",6X," WPRS"/)
      WRITE(LO,225)
      WRITE(LO,223)(I,JPOSA(I,2),WPRS(I,2),I=1,21)
      I = 22
      WRITE(LO,224)I,WPRS(I,2)
C
C     DUMP CORE DEVICE TO THE PRINTER
C
      WRITE(LO,227)
227   FORMAT(1H1//15X,"                   CORE DEVICE IMAGE"/)
      WRITE(LO,228) ARRAY
228   FORMAT(16Z5)
      IPASS = 1
229   IF (STRSTAT.EQ.1) GO TO 260
      WRITE(CO,230)
230   FORMAT(" FLXWAS>          *** SINGLE ITERATION COMPLETE ***")
```

```
      END FILE LO
      IACCW = -1
      IASTAT = 4Z4000
      CALL SETB(FLXISW,B7)
      CALL WAIT4
231   CONTINUE
C
C     IN PAUSE MODE, WAIT FOR RESPONSE FROM CPU-A OPERATOR
C
      IF(.NOT.TSTB(FLXOPT,B6)) GO TO 240
      WRITE(CO,235)
235   FORMAT(5X," FLXWAS>   *****  PAUSE FOR ANALYSIS CHECK  *****"/
     +/"  TYPE IN ANY CHARACTER TO CONTINUE OR [WS] FOR WALLS S/LINED")
      READ(CO,236) ANS
236   FORMAT(A2)
      IF (ANS.EQ.'WS') STRSTAT=1
      IF (ANS.EQ.'WS') GO TO 180
240   CONTINUE
C
C     OUTPUT REDUCED DATA TO LINEPRINTER AND DISK
C
C      CALL OUT (STRSTAT)
      CALL OUT
      IF (STRSTAT.EQ.-1) GO TO 270
      WRITE(LO,195)IBTSTNR,IBRUNNR,IBPTNR,ITERATION
      END FILE LO
C
C     IF EXCESSIVE ITERATIONS, ABORT STREAMLINING
C
      IF(ITERATION.GE.6) GO TO 270
C
C     MOVE FLEXWALLS TO NEW CONTOURS FOR NEXT ITERATION
C
      MVSTAT = -1
      CALL SETB(FSSTAT,B7)
      CALL SETB(FLXISW,B5)
      CALL RLTBZ4(FSSTAT,B7)
      IF (MVSTAT.EQ.4.OR.MVSTAT.EQ.5) GO TO 280
      IF (MVSTAT.NE.1) GO TO 250
C*********************************************************************
C
C     CONTINUE THE STREAMLINING CYCLE
C
      GO TO 60
C*********************************************************************
C*************************************
C*                                   *
C*        EXIT CONDITIONS            *
C*                                   *
C*************************************
250   WRITE(LO,254)
      WRITE(CO,254)
254   FORMAT(/////20X,27("*")/20X,"* STREAMLINING TERMINATED *"/20X
     +,"*     HARDWARE ERROR      *"/20X
     +,27("*"))
      WRITE (LO,255)
255   FORMAT(/20X,"          JACK STATUS"/
     +        ,22X,"#",6X,"TOP",3X,"BOTTOM")
```

```
      WRITE (CO,256)
256   FORMAT(/27X,"     JACK STATUS"/
     +        ,22X,"#",6X,"TOP",3X,"BOTTOM",4X,"#",6X,"TOP",3X,"BOTTOM")
      WRITE (LO,257) (I,JSTAT(I+1,1),JSTAT(I+1,2),I=1,21)
      WRITE (CO,258) (I,JSTAT(I+1,1),JSTAT(I+1,2),JNO(I),JSTAT(I+2,1),
     +    JSTAT(I+2,2),I=1,19,2)
      ILAST = 21
      WRITE (CO,257) ILAST,JSTAT(22,1),JSTAT(22,2)
257   FORMAT(21X,I2,2X,Z6,2X,Z6)
258   FORMAT(21X,I2,2X,Z6,2X,Z6,5X,I2,2X,Z6,2X,Z6)
C     WRITE(CO,259)
259   FORMAT(/15X," PRESS RETURN TO RECALL 'FLXOP' MENU")
      IASTAT = 4Z2000
      GO TO 290
260   WRITE(LO,265)
      WRITE(CO,265)
265   FORMAT(/20X,27("*")/20X,"*    WALLS STREAMLINED    *"/20X
     +,27("*"))
      WRITE(CO,266)
266   FORMAT(/10X,"PRESS RETURN TO RECALL 'FLXOP' MENU FOR NEXT PT.")
      IACCW = 0
      IASTAT = 4Z8000
      GO TO 290
270   WRITE(LO,275)
      WRITE(CO,275)
275   FORMAT(/////20X,27("*")/20X,"* STREAMLINING TERMINATED *"/20X
     +,"*     SOFTWARE ERROR      *"/20X
     +,27("*"))
      WRITE(CO,259)
      IASTAT = 4Z0800
      GO TO 290
280   WRITE(LO,285)
      WRITE(CO,285)
285   FORMAT(/////20X,27("*")/20X,"* STREAMLINING TERMINATED *"/20X
     +,"*   JACK LIMIT REACHED    *"/20X
     +,27("*"))
      WRITE (LO,255)
      WRITE (CO,256)
      WRITE (LO,257) (I,JSTAT(I+1,1),JSTAT(I+1,2),I=1,21)
      WRITE (CO,258) (I,JSTAT(I+1,1),JSTAT(I+1,2),JNO(I),JSTAT(I+2,1)
     +        ,JSTAT(I+2,2),I=1,19,2)
C     WRITE(CO,259)
      IASTAT = 4Z1000
290   IF (IANAL.NE.2) CALL SETB(FLXISW,B7)
      IF (IANAL.EQ.2) WRITE(CO,295)
295   FORMAT(//" FLXWAS>    ***  RE-ANALYSIS COMPLETED  ***")
      END FILE LO
      STOP
      END
```

APPENDIX A.2 - LISTING OF SUBROUTINE INITWAL

```
C     SUBROUTINE INITWAL(JPOSN,TOPY,BOTY,TWVEL,BWVEL,STRSTAT)
      SUBROUTINE INITWAL
C*******************************************************************
C
C
C     TITLE:          FLEXWALL INITIALIZATION SUBROUTINE
C
C     AUTHOR:         STEPHEN W.D. WOLF , NATIONAL RESEARCH COUNCIL
C
C     SYSTEM:         CRYO-A
C
C
C     PURPOSE:        SELECT APPROPRIATE INITIAL WALL CONTOURS FOR NEXT
C                     STREAMLINING CYCLE AND LOAD INTO SHARED REGION
C
C
C     REVISION HISTORY:
C
C     VERSION       DATE        DESCRIPTION
C     =======       ====        ==================================
C       1.1       02/11/86      INITIAL VERSION FROM WALDAT VERSION 1.3
C       1.2       02/25/86      SIMULATE THE READING OF SETUP PARAMETERS
C                               FROM FLXCOM
C       1.3       04/01/86      DEVELOPMENT VERSION
C       1.4       04/25/86      PRINT OUT SETUP PARAMETERS
C       1.5       06/25/86      REVISE ERROR MESSAGES AND MODIFY
C                               WALL RECORD SELECTION TOLERANCES
C       1.6       09/15/86      REVISE WALL RECORD SELECTION TECHNIQUE
C                               DEFAULT TO PREVIOUS WALL CONTOURS
C       1.7       09/17/86      OPTION TO CALCULATE WALL SHAPE INCLUDED
C       1.8       12/22/86      CORRECTION TO WALL LIBRARY SCAN
C       1.9       04/03/87      REDUCE TOLERANCES ON WALL LIBRARY SCAN
C                               AND ALLOW MANUAL RECORD SELECTION
*******************************************************************
C
C      COMPILER OPTIONS:              $23
C
C      LOGICAL FILES USED:
C                   CO - INPUT/OUTPUT TO CONSOLE
C                   LO - OUTPUT TO LINEPRINTER
C                   12 - REDUCED DATA DATAFILE
C
C      Shared Regions:
C                   FLXCOM
C                   WASCOM
C
C      User-Defined Subroutine Calls:
C                   WALCAL
C                   STWALL
C                   ERROR
C
C      Inputs:
C              /FLXCOM/ WASPAR
C      WASPAR - Table of operator defined set-up parameters
C
C              /WASCOM/ NOJACK,CHORD
C      NOJACK - Number of flexwall jacks.
C      CHORD - Model chord (Inches).
```

```
C
C       Reduced data datafile - KBUFF, LBUFF, TMACH, TREYNO, TALPHA, BUFF
C
C
C       Outputs:
C               /FLXCOM/ JPOSN, STRSTAT
C       JPOSN - Array of new (destination) jack positions (Inches).
C       STRSTAT - Streamline status word.
C
C               /WASCOM/ XORIG, YORIG, TOPY, BOTY, TWVEL, BWVEL
C       XORIG - Horizontal distance of model 1/4 chord point upstream of
C               model pivot (Inches).
C       YORIG - Vertical distance of model 1/4 chord point above the model
C               pivot (Inches).
C       TOPY - Array of top wall jack positions for the initial contour
C              (Inches).
C       BOTY - Array of bottom wall jack positions for the initial contour
C              (Inches).
C       TWVEL - Array of top wall external velocities for the initial
C               contour (v/U0).
C       BWVEL - Array of bottom wall external velocities for the initial
C               contour (v/U0).
C
C       Output Messages:
C
C                     INITWAL VER 1.9>  SETUP PARAMETERS
C
C*********************************************************************
C   INITIAL WALL CONTOURS - RECORD ### MACH #.### RN ####E### AOA ###.##
C*********************************************************************
C
C*********************************************************************
C          INITIAL WALL CONTOURS 'STRAIGHT' FOR NEXT POINT
C*********************************************************************
C
C          INITWAL> STRAIGHT WALL DATA ERROR
C
C          INITWAL> INPUT RECORD NO. FOR INITIAL CONTOURS (0-EXIT)
C
C       Processing:
C
C               1) Display setup parameters
C               2) If ''aerodynamically straight'' wall contours required
C                  jump to 12.
C               3) Select initial wall contours from reduced data library
C                  and load from disc.
C               4) If no wall contours in library (NREC < 7) jump to 11.
C               5) Select data records with wall information (NREC > 6).
C               6) Choose previous wall contours until a better set is found
C                  where ISAME = 3.
C               7) Define tolerances for selection of wall contours:
C                  TMDIFF,TRDIFF,TADIFF.
C               8) If a  very good set of wall contours are found (IGOOD =
C                  3) then jump to 10.
C               9) If no good set of wall contours (IGOOD < 2) can be found
C                  then jump to 11.
C               10) Load good wall contour data (IGOOD > 1) into the private
C                   shared region WASCOM, then jump to 13.
```

```
C               11) Compute initial wall contours for current test
C                   parameters (Subroutine WALCAL) then jump to 13.
C               12) Select initial wall contours from the datafile
C                   (Subroutine STWALL).
C               13) Load up new jack positions for initial wall contours in
C                   private shared region FLXCOM.
C               14) Return to main program.
C
C        Exception Handling:
C
C               Whenever an error condition is encountered (STRSTAT = -1) an
C        appropriate message is displayed on the operator console and the
C        lineprinter, subroutine ERROR is called to provide additional
C        error information, control returns to the main program.
C
C*********************************************************************
      INTEGER CO
      INTEGER*4 IREC
      INCLUDE USL/FWPTYP,LIST
      INCLUDE USL/FLXTYP,NOLIST
      DIMENSION DB(128),TWY(21),BWY(21),KBUFF(256),LBUFF(256),BUFF(128)
      DIMENSION ITEST(126),ITRUN(126),TMACH(126),TREYNO(126),TALPHA(126)
      DIMENSION ITDATA(126),ITER(126)
C
C     ***     COMMON     ***
C
      INCLUDE USL/FLXCOM,NOLIST
      INCLUDE USL/WASCOM,NOLIST
C
C     ***  EQUIVALENCES  ***
C
      EQUIVALENCE (KBUFF(1),NREC)
      EQUIVALENCE (KBUFF(2),ITEST(1))
      EQUIVALENCE (KBUFF(128),ITRUN(1))
      EQUIVALENCE (LBUFF(1),ITDATA(1))
      EQUIVALENCE (LBUFF(128),ITER(1))
      INCLUDE USL/FWPEQU,LIST
      DEFINE FILE 12(126,256,U,IREC)
      DATA CO/@CO/,LO/@LO/
      DATA VER/1.9/
      NJ1 = 2*NOJACK
      WRITE(LO,10)VER,SPMACH,SPREYN,SPAOA,XLORIG,YLORIG
      WRITE(CO,10)VER,SPMACH,SPREYN,SPAOA,XLORIG,YLORIG
10    FORMAT(1H1/"    INITWAL VER ",F3.1,"> SETUP PARAMETERS"/5F10.4)
      XORIG=XLORIG
      YORIG=YLORIG
      UREYNO = SPREYN * 12E06 / CHORD
      IF (SPAOA.EQ.99) GO TO 80
C
C                        SELECT INITIAL WALL CONTOURS
C                FROM REDUCED DATA DATAFILE AND LOAD FROM DISK
C
40    CONTINUE
      IERROR = 0
      IREDUCE = 12
      READ(IREDUCE'1) KBUFF
      READ(IREDUCE'2) LBUFF
      READ(IREDUCE'3) TMACH
      READ(IREDUCE'4) TREYNO
      READ(IREDUCE'5) TALPHA
C
C     IF MANUAL RECORD SELECTION REQUESTED, THEN JUMP
C
      IF (SPAOA.EQ.88) GO TO 46
      IF(NREC.LT.7) GO TO 76
C
C     SELECT DATA RECORD WITH WALL INFORMATION
C
      DO 45 I = 7,NREC
      J = NREC - I + 7
      ISAME = 0
      TMD = TMACH(J)-SPMACH
      ATMD = ABS(TMD)
      IF(SPAOA.GT.0.0)TAD = SPAOA+0.2-TALPHA(J)
      IF(SPAOA.LE.0.0)TAD = -SPAOA+0.2+TALPHA(J)
      ATAD = ABS(TAD)
      TRD = ABS(TREYNO(J)-UREYNO)
C
C     CHOOSE PREVIOUS WALL CONTOURS UNTIL A BETTER SET IS FOUND
C
      IF (I.EQ.7) GO TO 42
C
C     ASSESS GOODNESS FACTORS FOR EACH RECORD
C
C     ***  IGNORE RECORDS WITH MACH ERROR >0.02 WHEN SPMACH >0.7   ***
C
      IF(TMD.GT.0.02 .AND. SPMACH.GT.0.7) GO TO 45
      IF(ATMD.LE.TMDIFF) ISAME = 1
      IF(TRD.LE.TRDIFF) ISAME = ISAME + 1
      IF(ATAD.GT.TADIFF) GO TO 45
C
C     SCREEN ONLY FOR ALPHA IN DIRECTION OF 0 DEGREES
C
      IF(TAD.LT.0.0) GO TO 45
      ISAME = ISAME + 1
      IF(ISAME.NE.3) GO TO 45
C
C     DEFINE TOLERANCES FOR WALL CONTOURS SELECTION
C
42    IRN = J
      IGOOD = 0
      TMDIFF = ATMD
      IF (TMDIFF.LE.0.008) IGOOD = 1
      IF (TMDIFF.LE.0.008) TMDIFF = 0.008
      TRDIFF = TRD
      IF (TRDIFF.LE.50E06) IGOOD = IGOOD + 1
      IF (TRDIFF.LE.50E06) TRDIFF = 50E06
      TADIFF = ATAD
      IF (TAD.LT.0.0) TADIFF = 4.0
      IF (TADIFF.LE.0.8) IGOOD = IGOOD +1
      IF (TADIFF.LE.0.8) TADIFF = 0.8
      WRITE(LO,43)J,IGOOD,TMDIFF,TRDIFF,TADIFF
43    FORMAT(2I6,F8.3,E10.3,F8.3)
      IF (IGOOD.EQ.3) GO TO 50
45    CONTINUE
      IF (IGOOD.LT.2) GO TO 76
      GO TO 50
46    WRITE(CO,47)
47    FORMAT(" INITWAL> INPUT RECORD NO. FOR INITIAL CONTOURS (0-EXIT)")
      READ(CO,*)IRN
      IF (IRN.EQ.0) STOP
50    J = IRN
      WRITE(CO,55)J,TMACH(J),TREYNO(J),TALPHA(J)
55    FORMAT(/80("*")/2X
     +," INITIAL WALL CONTOURS - RECORD ",I3,",MACH ",F5.3,
     +" ;RN ",E8.3," ;AOA ",F6.2/80("*"))
      WRITE(LO,55)J,TMACH(J),TREYNO(J),TALPHA(J)
      READ(IREDUCE'J) BUFF
      IF(ABS(TMACH(J)-BUFF(1)).GT.0.001) IERROR = 1
      DO 65 J = 1,NOJACK
      TOPY(J) = BUFF(J+3)
      BOTY(J) = BUFF(J+NOJACK+3)
65    CONTINUE
      DO 75 J = 1,NOCPT
      TWVEL(J) = BUFF(J+NJ1+3)
      BWVEL(J) = BUFF(J+NJ1+NOCPT+3)
75    CONTINUE
      GO TO 100
C
C     COMPUTE INITIAL WALL CONTOURS FOR CURRENT TEST CONDITIONS
C
76    CALL WALCAL(SPMACH,SPREYN,SPAOA)
      GO TO 100
C
C     IF REQUESTED SELECT INITIAL WALL CONTOURS FROM REFERENCE TABLE
C
80    WRITE(CO,85)
      WRITE(LO,85)
85    FORMAT(/80("*")/17X
     +," INITIAL WALL CONTOURS 'STRAIGHT' FOR NEXT POINT"/80("*"))
      CALL STWALL(SPMACH,UREYNO,TOPY,BOTY,DSTAS,DSBAS,IERROR)
      IF(IERROR.EQ.0) GO TO 90
      WRITE(CO,86)
      WRITE(LO,86)
86    FORMAT(/" INITWAL> STRAIGHT WALL DATA ERROR")
      CALL ERROR(IERROR)
      STRSTAT = -1
      GO TO 110
90    DO 95 K = 1,NOCPT
      TWVEL(K) = 0.0
      BWVEL(K) = 0.0
95    CONTINUE
C
C       LOAD UP NEW JACK POSITIONS FOR INITIAL WALL CONTOURS
C
100   DO 105 J = 1,NOJACK
      JPOSN(J,1) = TOPY(J)
      JPOSN(J,2) = BOTY(J)
105   CONTINUE
110   RETURN
      END
```

APPENDIX A.3 - LISTING OF SUBROUTINE STWALL

```
      SUBROUTINE STWALL(FMACH,REYNO,TWY,BWY,DST,DSB,IERROR)
C*****************************************************************
C
C
C     TITLE:        "STRAIGHT WALL" DATA ACQUISITION SUBROUTINE
C
C     AUTHORS:      RAYMOND E. MINECK , NASA
C                   STEPHEN W.D. WOLF , NATIONAL RESEARCH COUNCIL
C
C     SYSTEM:       CRYO-A
C
C     PURPOSE:      SELECT THE WALL DATA FOR 'AERODYNAMICALLY
C                   STRAIGHT' WALL CONTOURS (CONSTANT MACH NO.)
C                   FOR A GIVEN REYNOLDS NO. AND MACH NO., FROM
C                   THE REFERENCE TABLE ON DISK.
C
C
C     REVISION HISTORY:
C
C     VERSION      DATA      DESCRIPTION
C     =======      ====      ===================================
C       1.1      06/26/85    MODIFIED VERSION 0.0 TO SUPPORT THE
C                            WALL ADJUSTMENT STRATEGY SOFTWARE
C       1.2      04/11/86    *SWDW* CHANGE LP TO LO
C       1.3      04/17/86    *SWDW* ALLOW FOR TEST CONDITIONS
C                            BEYOND THE REFERENCE TABLE BOUNDARIES
C       1.4      05/15/86    *SWDW* INTERPOLATE WALL CONTOURS
C       1.5      12/22/86    *SWDW* REVISE ERROR MESSAGES
C
C
C*****************************************************************
C
C     COMPILER OPTIONS:              $23
C
C     LOGICAL FILES USED:
C                   LO - OUTPUT TO LINEPRINTER
C                   10 - REFERENCE TABLE OUTPUT
C
C     CALLING SEQUENCE:
C                CALL STWALL(FMACH,REYNO,TWY,BWY,DST,DSB,IERROR)
C             WHERE -
C                    FMACH - CURRENT MACH NO.
C                    REYNO - CURRENT UNIT REYNOLDS NO. PER FOOT
C                    TWY   - TOP WALL JACK POSITIONS
C                    BWY   - BOTTOM WALL JACK POSITIONS
C                    DST   - TOP WALL D* VALUES
C                    DSB   - BOTTOM WALL D* VALUES
C                   IERROR - OUTPUT ERROR CODE :
C                                   0 - NO ERROR
C                                   1 - FAILURE TO BRACKET MACH NO.
C                                   2 - FAILURE TO BRACKET REYNOLDS NO.
C                                   3 - COMBINATION OF ABOVE FAILURES
C                                   4 - MISSING DATA RECORDS
C
C     Shared Regions:  None
C
C
C     User-Defined Subroutine Calls:
```

```
C         FILESORT - Sort array into ascending order (Listed with STWALL).
C         GETDATA - Get wall data from the reference table (Listed with
C         STWALL).
C
C
C     Inputs:
C
C         Call statement - FMACH, REYNO
C     FMACH - Free stream Mach number.
C     REYNO - Unit Reynolds number per foot.
C
C         Reference table datafile - IBUFF, JBUFF
C     IBUFF - Reference table directory record buffer
C     JBUFF - Wall data record buffer
C
C     Outputs:
C
C         Call statement - TWY, BWY, DST, DSB, IERROR
C     TWY - Array of top wall jack positions for ''aero. straight''
C           contour (Inches).
C     BWY - Array of bottom wall jack positions for ''aero. straight''
C           contour (Inches).
C     DST - Array of top wall d* values for ''straight'' contour (Inch).
C     DSB - Array of bottom wall d* values for ''aero. straight'' contour
C           (Inch).
C     ERROR - Output error code.
C
C
C     Output Messages:
C
C
C     STWALL> VERSION 1.5 STRAIGHT WALL DATA
C
C         INTERPOLATION RECORDS = ### , ### , ### , ###
C
C         MACH NO. RANGE  ##.## - ##.##
C         REYNOLDS NO. PER FT   ###E## - ###E##
C
C         TOP WALL JACK POSITIONS
C
C         BOTTOM WALL JACK POSITIONS
C
C         TOP WALL D* CONTOUR
C
C         BOTTOM WALL D* CONTOUR
C
C
C     Processing:
C              1) Initialization of datafile pointers.
C              2) Sort Mach numbers in ascending order (Subroutine
C                 FILESORT).
C              3) Sort Reynolds number in ascending order (Subroutine
C                 FILESORT).
C              4) Determine which Mach numbers bracket desired Mach
C                 numbers.
C              5) Determine which Reynolds numbers bracket desired Reynolds
C                 number.
C              6) Get wall data (Subroutine GETDAT).
```

```
C              7) Interpolate data between four sets of data.
C              8) Load interpolated data into working arrays.
C              9) Return to calling subroutine.
C
C
C     Exception Handling:
C
C           If an error occurs during execution then control will return to
C     the calling subroutine with one of the following output error codes
C     in variable IERROR:
C                   1 - Failure to bracket Mach no. on datafile
C                   2 - Failure to bracket Reynolds no. on datafile
C                   3 - Combination of above failures on datafile
C                   4 - Missing data records on datafile
C
C     The no error condition is IERROR = 0
C
C
C***************************************************************
      DIMENSION IM(10),IR(10),TWY(21),BWY(21),DST(18),DSB(18)
      DIMENSION TWY1(21),BWY1(21),DST1(18),DSB1(18)
      DIMENSION TWY2(21),BWY2(21),DST2(18),DSB2(18)
      DIMENSION TWY3(21),BWY3(21),DST3(18),DSB3(18)
      DIMENSION TWY4(21),BWY4(21),DST4(18),DSB4(18)
      INCLUDE USL/FMASSIGN,NOLIST
      DATA VER/1.5/
      DO 5 J = 1,10
      IM(J) = J
      IR(J) = J
5     CONTINUE
C
C      INITIALIZATION OF DISK FILE POINTERS
C
      IREC=1
      READ(10'IREC)IBUFF
C
C       SORT MACH NUMBERS IN ASCENDING ORDER
C
      CALL FILESORT(NTMACH,TMACH,IM)
C
C       SORT REYNOLDS NUMBERS IN ASCENDING ORDER
C
      CALL FILESORT(NTREYNO,TREYNO,IR)
C
C      DETERMINE WHICH MACH NOS. BRACKET DESIRED MACH NO.
C
10    IERROR=0
      NTM1=NTMACH-1
      DO 105 I=1,NTM1
      IF (FMACH.LT.TMACH(1)) GO TO 15
      IF (FMACH.GE.TMACH(10)) GO TO 16
      IF(TMACH(I).LE.FMACH .AND. TMACH(I+1).GT.FMACH) GO TO 20
105   CONTINUE
      IERROR=IERROR+1
      GO TO 25
15    IMH = 1
      GO TO 25
16    IMH = 10
```

```
      GO TO 25
20    IMH = I+1
25    IML=IMH-1
      IF (IML.LE.0) IML = 1
C
C      DETERMINE WHICH REY.NOS. BRACKET DESIRED REY. NO.
C
      NTR1=NTREYNO-1
      DO 115 I=1,NTR1
      IF (REYNO.LT.TREYNO(1)) GO TO 26
      IF (REYNO.GE.TREYNO(10)) GO TO 27
      IF(TREYNO(I).LE.REYNO .AND. TREYNO(I+1).GT.REYNO) GO TO 30
115   CONTINUE
      IERROR=IERROR+2
      GO TO 35
26    IRH=1
      GO TO 35
27    IRH=10
      GO TO 35
30    IRH = I+1
35    IF(IERROR.GT.0)GO TO 220
      IRL=IRH-1
      IF (IRL.LE.0) IRL = 1
C
C       GET WALL DATA
C
      IMPNT = IM(IML)
      IRPNT = IR(IRL)
      IRN1 = INDEX(IMPNT,IRPNT)
      IF(IRN1.EQ.0) IERROR =  4
      CALL GETDATA(IRN1,TWY1,BWY1,DST1,DSB1)
      IMPNT = IM(IMH)
      IRPNT = IR(IRL)
      IRN2 = INDEX(IMPNT,IRPNT)
      IF(IRN2.EQ.0) IERROR =  4
      CALL GETDATA(IRN2,TWY2,BWY2,DST2,DSB2)
      IMPNT = IM(IML)
      IRPNT = IR(IRH)
      IRN3 = INDEX(IMPNT,IRPNT)
      IF(IRN3.EQ.0) IERROR =  4
      CALL GETDATA(IRN3,TWY3,BWY3,DST3,DSB3)
      IMPNT = IM(IMH)
      IRPNT = IR(IRH)
      IRN4 = INDEX(IMPNT,IRPNT)
      IF(IRN4.EQ.0) IERROR =  4
      CALL GETDATA(IRN4,TWY4,BWY4,DST4,DSB4)
      IF(IERROR.NE.0) GO TO 220
C
C     INTERPOLATE DATA BETWEEN FOUR SETS OF DATA
C
      FACTOR = (FMACH-TMACH(IML))/(TMACH(IMH)-TMACH(IML))
      DO 40 J = 1,21
      TWY1(J)=(TWY2(J)-TWY1(J))*FACTOR +TWY1(J)
      BWY1(J)=(BWY2(J)-BWY1(J))*FACTOR +BWY1(J)
40    CONTINUE
      DO 45 J = 1,18
      DST1(J)=(DST2(J)-DST1(J))*FACTOR +DST1(J)
      DSB1(J)=(DSB2(J)-DSB1(J))*FACTOR +DSB1(J)
```

```
45     CONTINUE
       DO 60 J = 1,21
       TWY3(J)=(TWY4(J)-TWY3(J))*FACTOR +TWY3(J)
       BWY3(J)=(BWY4(J)-BWY3(J))*FACTOR +BWY3(J)
60     CONTINUE
       DO 65 J = 1,18
       DST3(J)=(DST4(J)-DST3(J))*FACTOR +DST3(J)
       DSB3(J)=(DSB4(J)-DSB3(J))*FACTOR +DSB3(J)
65     CONTINUE
       FACTOR = (REYNO-TREYNO(IRL))/(TREYNO(IRH)-TREYNO(IRL))
       DO 80 J = 1,21
       TWYAS(J)=(TWY3(J)-TWY1(J))*FACTOR +TWY1(J)
       BWYAS(J)=(BWY3(J)-BWY1(J))*FACTOR +BWY1(J)
80     CONTINUE
       DO 85 J = 1,18
       DSTAS(J)=(DST3(J)-DST1(J))*FACTOR +DST1(J)
       DSBAS(J)=(DSB3(J)-DSB1(J))*FACTOR +DSB1(J)
85     CONTINUE
C
C      LOAD INTERPOLATED DATA INTO WORKING ARRAYS
C
       DO 125 I = 1,21
       TWY(I) = TWYAS(I)
       BWY(I) = BWYAS(I)
125    CONTINUE
       DO 135 I = 1,18
       DST(I) = DSTAS(I)
       DSB(I) = DSBAS(I)
135    CONTINUE
       WRITE(LO,145) VER
145    FORMAT(///5X,' STWALL> VERSION ',F3.1,' STRAIGHT WALL DATA  '/)
       WRITE(LO,155) IRN1,IRN2,IRN3,IRN4
155    FORMAT(/5X," INTERPOLATION RECORDS = ",I6,3(",",I6))
       WRITE(LO,165)TMACH(IML),TMACH(IMH),TREYNO(IRL),TREYNO(IRH)
165    FORMAT(/5X,' MACH NO. RANGE ',F5.3,' - ',F5.3/
      +5X,' REYNOLDS NO. PER FT ',E9.3,' - ',E9.3)
       WRITE(LO,175) (TWYAS(I),I=1,11)
175    FORMAT(///5X" TOP WALL JACK POSITIONS"/,11F7.3)
       WRITE(LO,185) (TWYAS(I),I=12,21)
185    FORMAT(10F7.3)
       WRITE(LO,195) (BWYAS(I),I=1,11)
195    FORMAT(/5X" BOTTOM WALL JACK POSITIONS"/,11F7.3)
       WRITE(LO,185) (BWYAS(I),I=12,21)
       WRITE(LO,205) (DSTAS(I),I=1,10)
205    FORMAT(///5X" TOP WALL D* CONTOUR"/,10F7.3)
       WRITE(LO,185) (DSTAS(I),I=11,18)
       WRITE(LO,215) (DSBAS(I),I=1,10)
215    FORMAT(/5X" BOTTOM WALL D* CONTOUR"/,11F7.3)
       WRITE(LO,185) (DSBAS(I),I=11,18)
220    RETURN
       END
C
       SUBROUTINE FILESORT(N,X,IX)
C
C        SORT ARRAY X INTO ASCENDING ORDER
C
       DIMENSION IX(10),X(10)
       NM1=N-1
      DO 105 I=1,NM1
      IP1=I+1
      DO 115 J=IP1,N
      IF(X(J).GT.X(I))GO TO 115
      HOLD=X(I)
      IHOLD=IX(I)
      X(I)=X(J)
      IX(I)=IX(J)
      IX(J)=IHOLD
      X(J)=HOLD
115   CONTINUE
105   CONTINUE
      RETURN
      END
C
      SUBROUTINE GETDATA(IREC,TWY,BWY,DTW,DBW)
C
C     GET WALL DATA FROM THE REFERENCE TABLE
C
      DIMENSION TY(21),BY(21),DT(18),DB(18),JBUFF(256)
      DIMENSION TWY(21),BWY(21),DTW(18),DBW(18)
      EQUIVALENCE (JBUFF(7),TY(1))
      EQUIVALENCE (JBUFF(49),BY(1))
      EQUIVALENCE (JBUFF(179),DT(1))
      EQUIVALENCE (JBUFF(215),DB(1))
      READ(10'IREC) JBUFF
      DO 5 J = 1,21
      TWY(J)=TY(J)
      BWY(J)=BY(J)
5     CONTINUE
      DO 10 J = 1,18
      DTW(J)=DT(J)
      DBW(J)=DB(J)
10    CONTINUE
      RETURN
      END
```

APPENDIX A.4 - LISTING OF SUBROUTINE WALCAL

```
      SUBROUTINE WALCAL (SPMACH,SPREYN,SPAOA)
C***********************************************************************
C
C     TITLE:         WALL SHAPE CALCULATION SUBROUTINE
C
C     AUTHORS:       RAINER REBSTOCK , NATIONAL RESEARCH COUNCIL
C                    STEPHEN W.D. WOLF , NATIONAL RESEARCH COUNCIL
C
C     SYSTEM:        CRYO-A
C
C     PURPOSE:
C THIS PROGRAM CALCULATES STREAMLINE CONTOURS AND THEIR PRESSURE
C DISTRIBUTIONS IN THE FARFIELD OF A THIN AIRFOIL. LINEAR POTENTIAL
C FLOW IS ASSUMED WITH COMPRESSIBILITY TAKEN INTO ACCOUNT BY THE USE
C OF THE PRANDTL-GLAUERT FACTOR.
C CALCULATIONS ARE BASED ON A SIMPLIFIED SINGULARITY DISTRIBUTION
C OF THE AIRFOIL. THE SOLID BLOCKAGE IS SIMULATED BY A RANKINE
C BODY OF SAME LENGTH AND MAXIMUM THICKNESS (REPRESENTED BY A
C SOURCE AND A SINK OF EQUAL STRENGTH). THE LIFT IS
C REPRESENTED BY A SINGLE VORTEX THE STRENGTH OF WHICH IS GIVEN
C BY THE ESTIMATED MODEL LIFT COEFFICIENT.
C THE WAKE BLOCKAGE IS ACCOUNTED FOR BY A SOURCE THE STRENGTH OF
C WHICH IS GIVEN BY THE ESTIMATED WAKE DRAG COEFFICIENT.
C THE PROGRAM IS USED HERE TO PREDICT THE WALL DEFLECTION AND
C PRESSURE DISTRIBUTION (UPPER AND LOWER WALL) OF THE 0.3-M TEST
C SECTION. VALUES ARE GIVEN AT THE JACK LOCATIONS.
C THE WALL BOUNDARY LAYER DISPLACEMENT THICKNESS IS TAKEN INTO
C ACCOUNT BY ADDING IN THE ''AERO. STRAIGHT'' WALL SHAPES.  THE
C COMPUTED WALL DEFLECTIONS THEREFORE INDICATE THE JACK MOVEMENT
C WITH RESPECT TO THE GEOMETRICALLY STAIGHT WALLS.
C THE AIRFOIL CENTER OF ROTATION IS AT X=30.75 INCHES
C ON THE TUNNEL CENTER LINE,WHERE STATION X=0 IS AT THE UPSTREAM END
C OF THE FLEXIBLE WALL. THE MODEL MID-CHORD POSTION IS DETERMINED AT
C RUN TIME.
C
C     REVISION HISTORY:
C
C     VERSION       DATE       DESCRIPTION
C     =======       ====       ======================================
C       1.1       09/16/86     INITIAL VERSION BASED ON PROGRAM PST2D
C       1.2       09/17/86     INTRODUCE MOVE LIMITS TO JACK #1 AND #22
C       1.3       12/22/86     *SWDW* INITIALIZE SINGULARITY STRENGTHS
C       1.4       04/01/87     *SWDW* EXIT CONDITION INCLUDED
C       1.5       09/29/88     *SWDW* REFERENCE WALL MOVEMENT DEMANDS
C                                     TO ''AERO. STRAIGHT'' SHAPES
C
C***********************************************************************
C
C     COMPILER OPTIONS:            $23
C
C     LOGICAL FILES USED:
C                    CO - INPUT/OUTPUT TO CONSOLE
C                    LO - OUTPUT TO LINEPRINTER
C
C     CALLING SEQUENCE:
C                    CALL WALCAL (SPMACH,SPREYN,SPAOA)
C             WHERE:
C                    SPMACH - SETUP PARMETERS MACH NUMBER
```

```
C                     SPREYN - SETUP PARAMETERS REYNOLD NUMBER
C                     SPAOA - SETUP PARAMETERS ANGLE OF ATTACK
C
C     Shared Regions:
C         WASCOM
C
C
C     User-Defined Subroutine Calls:
C1.5      STWALL
C1.5      ERROR
C         BLOCK - Computes disturbance velocity due to blockage (Listed
C                 with WALCAL).
C         WAKE - Computes disturbance velocity due to wake (Listed with
C                WALCAL).
C         LIFT - Computes disturbance velocity due to lift (Listed with
C                WALCAL).
C
C     Inputs:
C         /WASCOM/ NOCPT, CHORD, XORIG, XJACK
C     NOCPT - Number of wall computing points.
C     CHORD - Model chord (Inches).
C     XORIG - Horizontal distance of model 1/4 chord upstream of model
C             pivot (Inches).
C     XJACK - Array of jack X co-ordinates (Inches).
C
C         Call statement - SPMACH, SPREYN, SPAOA
C     SPMACH - Setup parameters Mach number.
C     SPREYN - Setup parameters Reynolds number.
C     SPAOA - Setup parameters angle of attack.
C
C         Console keyboard - EPS, CL, CD
C     EPS - Model's maximum section thickness (%chord).
C     CL - Estimated model lift coefficient.
C     CD - Estimated model drag coefficient.
C
C
C     Outputs:
C         /WASCOM/  TOPY, BOTY, TWVEL, BWVEL
C     TOPY - Array of top wall jack positions for computed shape (Inches).
C     BOTY - Array of bottom wall jack positions for computed shape
C            (Inches).
C     TWVEL - Array of top wall external velocities over computed shape
C             (v/U0),
C     BWVEL - Array of bottom wall external velocities over computed shape
C             (v/U0).
C
C     Output Messages:
C
C         WALCAL> ENTER ESTIMATED LIFT- AND DRAG COEFFICIENTS
C
C         WALCAL> ENTER AIRFOIL MAXIMUM THICKNESS (%) 0-EXIT
C              THICKNESS NOT TO EXCEED 20%
C
C         WALCAL> INPUT ERROR   THICKNESS IS ###.##%
C
C         WALCAL> **** CHORD ERROR CHECK CPU-B RTP VALUE ****
C
C    WALCAL> AIRFOIL GEOMETRY: CHORD ##.## INCHES MAXIMUM THICKNESS##%
```

```
C
C    WALCAL> FLOW PARAMETERS: FREE STREAM MACH NUMBER ###.##
C                   CHORD REYNOLDS NUMBER ####E##
C                   LIFT COEFFICIENT ###.###
C                   DRAG COEFFICIENT ###.###
C
C       WALCAL> INITIAL WALL CONTOURS
C
C
C     Processing:
C              1) Define functions.
C              2) Input section from operator console.
C              3) Load test parameters into work variables.
C              4) Define jack positions relative to model based reference
C                 system.
C              5) Initialize singularity strengths.
C              6) Compute wall displacement due to blockage effect.
C              7) Compute wall displacement due to lift effect.
C              8) Compute wall displacement due to wake blockage.
C              9) Determine wall contours due to lift and blockage.
C              10) Compute the disturbance velocity tangential to the
C                  walls.
C1.5           11) Determine the ''aero. straight'' wall shapes for test
C1.5                conditions in the setup parameter table.
C              12) Determine total wall deflections.
C              13) Output data summary.
C              14) Interface data to private shared region WASCOM.
C              15) Limit Jack #1 and #22 movement.
C              16) Determine jack movements for the variable diffuser.
C              17) Return to subroutine INITWAL.
C
C
C     Exception Handling:
C
C         If an excessive model thickness (> 20%) is accidentally entered
C     by the operator then the operator is given an opportunity to
C     correctly enter the model thickness.
C
C         If a strange chord value is found, an error message is sent to
C     the CPU-A console and the WAS software is halted.
C
C*********************************************************************
C
C1.5
      DIMENSION X(21),YB(21),YL(21),YUW(21),YLW(21),TWY(21),BWY(21)
     +    UUW(21),ULW(21),YWB(21)
      REAL MF
      INTEGER CO
      INCLUDE USL/WASCOM,NOLIST
      DATA CO/@CO/,LO/@LO/
C
C
C STREAM FUNCTIONS
C SOLID BLOCKAGE
      PSIB(X1,Y,Q,D)=Q*(ATAN((X1-D)/Y)-ATAN((X1+D)/Y))
C LIFT
      PSIL(X1,Y,G)=0.5*G*ALOG(X1*X1+Y*Y)
C WAKE BLOCKAGE
```

```
      PSIW(X1,Y,Q)=-Q*ATAN(X1/Y)
C***********************************************************************
C
C     0.3-M TEST SECTION GEOMETRY
C
C***********************************************************************
C
C     TUNNEL HALF HEIGHT
C1.5
      THALFH=6.5
C
C     STREAMWISE LOCATION OF MODEL PIVOT (TURNTABLE CENTER)
C     RELATIVE TO THE FLEXWALL ANCHOR POINT (INCHES)
C1.5
      TURNTC=30.75
C
C     NO. OF STREAMLINING JACKS
C
      NJACK = NOCPT - 4
C
C FORMATS
25    FORMAT(' WALCAL> ENTER ESTIMATED LIFT- AND DRAG COEFFICIENT',/)
40    FORMAT(' WALCAL> ENTER AIRFOIL MAXIMUM THICKNESS (%) 0-EXIT'/
     + 8X,' THICKNESS NOT TO EXCEED 20%',/)
50    FORMAT(//,' WALCAL> INPUT ERROR',5X,'THICKNESS IS',F7.2,'%',/)
60    FORMAT(/,' WALCAL> **** CHORD ERROR CHECK CPU-B RTP VALUE ***')
70    FORMAT(' WALCAL> AIRFOIL GEOMETRY:  CHORD ',F5.2,' INCHES ',
     + 3X,'MAXIMUM THICKNESS',F6.2,'%',/)
80    FORMAT(' WALCAL> FLOW PARAMETERS:   FREE STREAM MACH NUMBER',F6.2,
     + /,27X,' CHORD REYNOLDS NUMBER ',E10.2,
     + /,27X,' LIFT COEFFICIENT',F8.4,
     + /,27X,' DRAG COEFFICIENT',F8.4,///)
90    FORMAT(///' WALCAL>  INITIAL WALL CONTOURS'/)
100   FORMAT(/3X,' JACK #',' TOP WALL',9X,' BOTTOM WALL')
110   FORMAT(A3)
120   FORMAT(I6,2F8.3,5X,2F8.3)
C
C INPUT SECTION
C READ IN MAXIMUM THICKNESS T/C (%).
15    IF(IPASS.NE.1) WRITE(CO,40)
      IF(IPASS.NE.1) READ(CO,*) EPS
      IF(EPS.EQ.0.0) STOP
      IF (EPS.GT.20.) WRITE(CO,50) EPS
      IF (EPS.GT.20.) GO TO 15
      IPASS = 1
C
C     LOAD TEST PARAMETERS
C
      C = CHORD
      MF = SPMACH
      RE = SPREYN*1E06
C READ ESTIMATED LIFT AND DRAG COEFFICIENT
      WRITE(CO,25)
      READ(CO,*) CL,CD
C DEFINE
C JACK POSITIONS RELATIVE TO MODEL BASED REFERENCE SYSTEM
C1.5
      XMODEL = TURNTC-XORIG+(C/4)
      DO 1 I=1,NJACK
       X(I)=XJACK(I+2)-XMODEL
       YB(I)=0.
       YL(I)=0.
       YWB(I)=0.
       YUW(I)=0.
       YLW(I)=0.
       DEL(I)=0.
       UUW(I)=0.
       ULW(I)=0.
1     CONTINUE
C
C INITIALIZE SINGULARITY STRENGTHS
C
      QST = 0.0
      QWST = 0.0
      GAMST = 0.0
C UPSTREAM END OF UPPER TEST SECTION WALL
      X0=-XMODEL
C1.5
      Y0=THALFH
C COMPRESSIBILITY FACTOR BET
      BET=SQRT(1-MF*MF)
      PI=6*ARSIN(0.5)
C
      IF(EPS.EQ.0.) GOTO 1000
C
C WALL DISPLACEMENT DUE TO BLOCKAGE EFFECT
C EQUIVALENT RANKINE BODY HAS STAGNATION POINTS (+XC,0) AND (-XC,0)
C AND MAXIMUM THICKNESS AT (0,+YT) AND (0,-YT)
      XC=0.5*C
      YT=0.5*EPS*C/100.
C SINGULARITY REPRESENTATION IS A SOURCE AT X=-A AND A SINK AT
C X=+A BOTH OF EQUAL STRENGTH QST (NORMALIZED WITH 2*PI*UINF)
C CALCULATION OF 'QST' AND 'A' FROM INPUT VALUES
C VERIFY
      IF(YT/XC.GE.PI/SQRT(27.)) WRITE(CO,60)
      IF(YT/XC.GE.PI/SQRT(27.)) STOP
C CALCULATE
      ARG=YT*SQRT(27.)/(PI*XC)
      DEL1=ARCOS(-ARG)
      A=2*XC*COS(DEL1/3.)/SQRT(3.)
      QST=(XC*XC-A*A)/(2.*A)
C WALL DEFLECTION DUE TO BLOCKAGE EFFECT IS STORED
C IN ARRAY 'YB' (UPPER WALL).
      Y0I=BET*Y0
      C1=Y0I+PSIB(X0,Y0I,QST,A)
      YN=Y0I
      DO 2 I=1,NJACK
C
C FIXPOINT ITERATION
51    YA=YN
      YN=C1-PSIB(X(I),YA,QST,A)
      IF(ABS(YN-YA).GE.0.001) GOTO 51
C
      YB(I)=YN-Y0I
2     CONTINUE
C
```

```
C WALL DISPLACEMENT DUE TO LIFT EFFECT
C THE VORTEX IS LOCATED AT THE CENTER OF LIFT,ASSUMED TO BE THE
C 1/4 CHORD POINT. ITS STRENGTH 'GAM' IS RELATED TO THE LIFT
C COEFFICIENT CL BY THE KUTTA-JOUKOWSKI THEOREM. USE NORMALIZED
C STRENGTH GAMST=GAM/(2*PI*UINF).
C
1000  IF(CL.EQ.0.) GOTO 2000
      GAMST=C*CL/(4.*PI)
C WALL DEFLECTION DUE TO LIFT EFFECT IS STORED IN
C ARRAY 'YL' (UPPER WALL).
      Y0I=BET*Y0
      C1=Y0I+PSIL(X0+C/4.,Y0I,BET*GAMST)
      YN=Y0I
      DO 3 I=1,NJACK
C
C FIXPOINT ITERATION
52    YA=YN
      YN=C1-PSIL(X(I)+C/4.,YA,BET*GAMST)
      IF(ABS(YN-YA).GE.0.001) GOTO 52
C
      YL(I)=YN-Y0I
3     CONTINUE
C
C
C WALL DISPLACEMENT DUE TO WAKE BLOCKAGE
C THE SOURCE REPRESENTING THE WAKE IS ASSUMED TO BE AT X2=C/2
C (TRAILING EDGE).USE NORMALIZED STRENGTH QWST=QW/(2*PI*UINF)
C
2000  IF(CD.EQ.0) GOTO 3000
      QWST=C*CD/(4.*PI)
C WALL DEFLECTIONS DUE TO WAKE BLOCKAGE ARE STORED IN ARRAY 'YWB'
C (UPPER WALL)
      Y0I=BET*Y0
      C1=Y0I+PSIW(X0-C/2.,Y0I,QWST)
      YN=Y0I
      DO 31 I=1,NJACK
C
C FIXPOINT ITERATION
53    YA=YN
      YN=C1-PSIW(X(I)-C/2.,YA,QWST)
      IF(ABS(YN-YA).GE.0.001) GOTO 53
C
      YWB(I)=YN-Y0I
31    CONTINUE
C
C
C WALL CONTOUR DUE TO LIFT AND BLOCKAGE
3000  DO 21 I=1,NJACK
       YUW(I)=Y0+YB(I)+YWB(I)+YL(I)
       YLW(I)=-Y0-YB(I)-YWB(I)+YL(I)
21    CONTINUE
C
C DISTURBANCE VELOCITY TANGENTIAL TO THE WALLS
C UPPER WALL
      DO 22 I=1,NJACK
       CALL BLOCK(X(I),YUW(I),A,QST,BET,UB,VB)
       CALL WAKE(X(I),YUW(I),X2,QWST,BET,UW,VW)
       CALL LIFT(X(I),YUW(I),C,GAMST,BET,UL,VL)
       UD=UB+UW+UL
       VD=VB+VW+VL
       UUW(I)=SQRT((1+UD)*(1+UD)+VD*VD)-1.
22    CONTINUE
C
C LOWER WALL
      DO 23 I=1,NJACK
       CALL BLOCK(X(I),YLW(I),A,QST,BET,UB,VB)
       CALL WAKE(X(I),YLW(I),X2,QWST,BET,UW,VW)
       CALL LIFT(X(I),YLW(I),C,GAMST,BET,UL,VL)
       UD=UB+UW+UL
       VD=VB+VW+VL
       ULW(I)=SQRT((1+UD)*(1+UD)+VD*VD)-1.
23    CONTINUE
C
C1.5
C1.5 DETERMINE THE ''AERO. STRAIGHT'' WALL SHAPES FOR
C1.5 TEST CONDITIONS IN THE SETUP PARAMETERS TABLE
C1.5
      CALL STWALL(MF,RE,TWY,BWY,DSTAS,DSBAS,IERROR)
C1.5
      IF(IERROR.NE.0) CALL ERROR
C1.5
      IF(IERROR.NE.0) STOP
C1.5
C1.5
C
C TOTAL WALL DEFLECTIONS
      DO 4 I=1,NJACK
C1.5
       YUW(I)=YB(I)+YWB(I)+YL(I)+TWY(I)
C1.5
       YLW(I)=-YB(I)-YWB(I)+YL(I)+BWY(I)
4     CONTINUE
C
C OUTPUT SECTION
      WRITE(LO,90)
      WRITE(LO,70) C,EPS
      WRITE(LO,80) MF,RE,CL,CD
      WRITE(LO,100)
C
C     INTERFACE DATA TO WASCOM FORMAT
C
      TWVEL(1) = 0.0
      TWVEL(2) = 0.0
      BWVEL(1) = 0.0
      BWVEL(2) = 0.0
      DO 7 N = 1,NJACK
      TWVEL(N+2) = UUW(N)
      BWVEL(N+2) = ULW(N)
      TOPY(N)=YUW(N)
      BOTY(N)=YLW(N)
      WRITE(LO,120)N,TOPY(N),TWVEL(N+2),BOTY(N),BWVEL(N+2)
7     CONTINUE
      TWVEL(NJACK+3) = TWVEL(NJACK)
      TWVEL(NJACK+4) = TWVEL(NJACK)
      BWVEL(NJACK+3) = BWVEL(NJACK)
      BWVEL(NJACK+4) = BWVEL(NJACK)
```

```
C?
C?    0.3-M TCT HARDWARE REQUIREMENT
C?    LIMIT JACK #1 AND #22 MOVEMENT
C?
      IF (TOPY(1).GT.0.075) TOPY(1)=0.075
      IF (TOPY(1).LT.-0.075) TOPY(1)=-0.075
      IF (BOTY(1).GT.0.065) BOTY(1)=0.065
      IF (BOTY(1).LT.-0.065) BOTY(1)=-0.065
C
C        0.3-M TCT HARDWARE REQUIREMENT
C     DETERMINE JACK MOVEMENTS FOR THE VARIABLE DIFFUSER
C     JACKS #19, #20, AND #21 ON BOTH WALLS
C
      FRACT = 6
      DO 8 J=19,21
      TOPY(J) = TOPY(18)*(FRACT/8.)
      BOTY(J) = BOTY(18)*(FRACT/8.)
      FRACT = FRACT + (J-22)
8     CONTINUE
      RETURN
      END
C
C
C
      SUBROUTINE BLOCK(X,Y,A,Q,BET,UB,VB)
C THIS SUBROUTINE CALCULATES THE BLOCKAGE INDUCED DISTURBANCE
C VELOCITIES UB AND VB AT THE POINT (X,Y) RELATIVE TO THE MODEL
C BASED REFERENCE SYSTEM
C
      Z1=(X+A)*(X+A)+BET*Y*BET*Y
      Z2=(X-A)*(X-A)+BET*Y*BET*Y
      UB=Q*((X+A)/Z1-(X-A)/Z2)/BET
      VB=BET*Q*Y*(1./Z1-1./Z2)
      RETURN
      END
C
C
      SUBROUTINE LIFT(X,Y,C,G,BET,UL,VL)
C THIS SUBROUTINE CALCULATES THE LIFT INDUCED DISTURBANCE VELO
C CITIES UL AND VL AT THE POINT (X,Y) RELATIVE TO THE MODEL BASED
C REFERENCE SYSTEM
C
      Z=(X+C/4.)*(X+C/4.)+BET*Y*BET*Y
      UL=G*BET*Y/Z
      VL=-BET*G*(X+C/4.)/Z
      RETURN
      END
C
C
      SUBROUTINE WAKE(X,Y,X2,QWST,BET,UW,VW)
C THIS SUBROUTINE CALCULATES THE WAKE INDUCED DISTURBANCE VELOCITIES
C UW AND VW AT THE POINT (X,Y) RELATIVE TO THE MODEL BASED REFERENCE
C SYSTEM
      Z=(X-X2)*(X-X2)+BET*Y*BET*Y
      UW=QWST*(X-X2)/(BET*Z)
      VW=BET*QWST*Y/Z
      RETURN
      END
```

APPENDIX A.5 - LISTING OF SUBROUTINE WALDAT

```
C     SUBROUTINE WALDAT(TWY,BWY,DSTAS,DSBAS,TOPWP,BOTWP,STRSTAT)
      SUBROUTINE WALDAT
C******************************************************************
C
C
C     TITLE:        WALL DATA ACQUISITION AND SORT SUBROUTINE
C
C     AUTHOR:       STEPHEN W.D. WOLF , NATIONAL RESEARCH COUNCIL
C
C     SYSTEM:       CRYO-A
C
C
C     PURPOSE:      SELECT APPROPRIATE 'AERO. STRAIGHT' WALL
C                   CONTOUR DATA SETS AND LOAD INTO SHARED REGION
C
C
C     REVISION HISTORY:
C
C     VERSION      DATE      DESCRIPTION
C     =======      ====      ===================================
C       1.1      05/23/85    INITIAL VERSION WITH FIXED 'STRAIGHT WALL'
C                            DATA AND TSWT DATA REDUCTION.
C       1.2      06/07/85    'STRAIGHT WALL' DATA SELECTED FROM REFERENCE
C                            TABLE ON DISK.
C       1.3      09/20/85    START WALL CONTOURS SELECTED FROM DISK FILES
C       1.4      02/25/86    INITIALIZATION SECTION REMOVED
C       1.5      03/28/86    INCLUDED TWYAS AND BWYAS CALCS
C       1.6      04/02/86    FREEZE WALL POSITION CHECK AND
C                            ALLOCATE ONE RAW DATA RECORD PER DATA PT.
C       1.7      04/10/86    UPDATE REYNOLDS NO TO REAL VALUE
C       1.8      04/25/86    THAW WALL POSITION CHECK + STORE RAW DATA
C                            DIRECT TO TAPE
C       1.9      07/21/86    REVISED ERROR MESSAGES PLUS OPTION TO
C                            DISPENCE WITH RAW DATA STORAGE
C       2.0      12/22/86    REMOVE REAL-TIME DATA STORAGE ON TAPE
C       2.1      12/23/86    RE-ANALYSIS MODE CHANGES
C       2.2      12/31/86    IMPROVED I/O OPERATIONS
C       2.3      06/15/88    ALPHA ERROR CORRECTED IN RAW DATA
C******************************************************************
C
C     COMPILER OPTIONS:            $23
C
C     LOGICAL FILES USED:
C                   CO - INPUT/OUTPUT TO CONSOLE
C                   LO - OUTPUT TO LINEPRINTER
C                   11 - RAW DATA DATAFILE
C
C     Shared Regions:
C         FLXCOM
C         WASCOM
C
C     User-Defined Subroutine Calls:
C         STWALL
C         ERROR
C
C     Inputs:
C             /FLXCOM/ XBMACH, XBPS, XBPT, XBTT, XBRHOS, XREYNO, ALPHA,
C     XCHORD, IBHOUR, IBMIN, IBTSTNR, IBRUNNR, IBPTNR, WPRS, JPOS
```

```
C     XBMACH - Test Mach number.
C     XBPS - Reference static pressure (PSIA).
C     XBPT - Reference total pressure (PSIA).
C     XBTT - Reference total temperature (K).
C     XBRHOS - Static density of working fluid (Slugs/cub. ft).
C     XREYNO - Test chord Reynolds number (millions).
C     XBAOA - Test model angle of attack (degrees).
C     XCHORD - Model chord (Inches).
C     IBHOUR - Hour data acquired.
C     IBMIN - Minute data acquired.
C     IBTSTNR - Test number.
C     IBRUNNR - Run number.
C     IBPTNR - Data point number.
C     WPRS - Array of wall pressures (PSIA).
C     JPOS - Array of current wall jack positions (Inches).
C
C         /WASCOM/ NOJACK, IANAL, NOCPT, TOPY, BOTY
C     NOJACK - Number of wall jacks.
C     IANAL - Analysis mode designator.
C     NOCPT - Number of wall computing points.
C     TOPY - Array of demanded top wall jack Y co-ordinates (Inches).
C     BOTY - Array of demanded bottom wall jack Y co-ordinates (Inches).
C
C         Raw data datafile - KBUFF, LBUFF
C
C     Outputs:
C         /WASCOM/ DSTAS, DSBAS, TWYAS, BWYAS, TOPWP, BOTWP
C     DSTAS - Array of top wall d* thicknesses on ''aero. straight''
C             contour (Inches).
C     DSBAS - Array of bottom wall d* thicknesses with ''aero. straight''
C             contour (Inches).
C     TWYAS - Array of top wall jack Y co-ordinates relative to ''aero.
C             straight'' contour (Inches).
C     BWYAS - Array of bottom wall jack Y co-ordinates relative to ''aero.
C             straight'' contour (Inches).
C     TOPWP - Array of top wall tapping pressures (PSIA).
C     BOTWP - Array of bottom wall tapping pressures (PSIA).
C
C         Raw data datafile - KBUFF, LBUFF, DB
C     KBUFF - Directory record buffer.
C     LBUFF - Directory record buffer .
C     DB - Raw data record buffer.
C
C
C     Output Messages:
C              ENTER RECORD NO. FOR DATA RE-ANALYSIS
C
C              WALDAT> WARNING - USING LAST RECORD AVAILABLE  ***
C
C              WALDAT> RAW DATA ERROR
C
C    FLXWAS RAW DATA FROM RECORD ### (Followed by table of raw data)
C
C              WALDAT> STRAIGHT WALL DATA ERROR
C
C              WALDAT> JACK ## POSITION ERROR
C
C
```

```
C     Processing:
C               1) Decide if re-analysis mode selected, if yes jump to 5.
C               2) Load raw data into array DB.
C               3) If IANAL = 0, jump to 6.
C               4) Send raw data in 'DB' to the raw data datafile.
C               5) Read loaded data from the raw data datafile as a check or
C                  re-analysis.
C               6) Select ''aerodynamically straight'' wall data record and
C                  acquire wall data.
C               7) Determine jack positions relative to ''aero. straight''
C                  contours.
C               8) Load wall pressures into work arrays.
C               9) Check wall data for auto streamlining.
C              10) Return to main program.
C
C
C     Exception Handling:
C
C         Whenever an error condition is encountered (STRSTAT = -1) an
C     appropriate message is displayed on the operator console and
C     lineprinter, subroutine ERROR is called to provide additional error
C     information, control returns to program FLXWAS.
C
C         A jack position error due to LVDT drift is displayed as a warning
C     to the operator, subroutine ERROR is called to provide additional
C     information.  If the error is severe, then a hardware error will
C     result when the walls are moved.
C
C         A warning is given if the last available record on the raw data
C     datafile is being used.  This datafile needs to be copied to tape and
C     then initialized.
C
C**********************************************************************
      INTEGER CO
      INTEGER*4 IREC
      INCLUDE USL/FLXTYP,NOLIST
      DIMENSION DB(128),TWY(21),BWY(21),KBUFF(256),LBUFF(256),BUFF(128)
      DIMENSION ITEST(126),ITRUN(126),TMACH(126),TREYNO(126),TALOHA(126)
      DIMENSION ITDATA(126),ITER(126)
C
C     ***       COMMON      ***
C
      INCLUDE USL/FLXCOM,NOLIST
      INCLUDE USL/WASCOM,NOLIST
C
C     ***   EQUIVALENCES   ***
C
      EQUIVALENCE (KBUFF(1),NREC)
      EQUIVALENCE (KBUFF(2),ITEST(1))
      EQUIVALENCE (KBUFF(128),ITRUN(1))
      EQUIVALENCE (LBUFF(1),ITDATA(1))
      EQUIVALENCE (LBUFF(128),ITER(1))
      DEFINE FILE 11(126,256,U,IREC)
      DATA CO/@CO/,LO/@LO/
      IFLIM = 126
      NJ1 = 2*NOJACK
      NJ2 = 3*NOJACK
C
```

```
C     DECIDE IF RE-ANALYSIS MODE SELECTED
C
      IF (IANAL.NE.2) GO TO 10
      WRITE(CO,5)
5     FORMAT(/" ENTER RECORD NO. FOR DATA RE-ANALYSIS")
      READ(CO,*) NREC
      GO TO 20
C
C*********************************************************************
C
C           LOAD RAW DATA INTO DATA BUFFER
C
10    DB(1) = XBMACH
      DB(2) = XBPS
      DB(3) = XBPT
      DB(4) = XBTT
      DB(5) = XBRHOS
      DB(6) = XREYNO*1E06
      DB(7) = XBAOA
      DB(8) = XCHORD
      DB(9) = IBHOUR
      DB(10) = IBMIN
      DB(11) = IBTSTNR
      DB(12) = IBRUNNR
      DB(13) = IBPTNR
      DO 15 J = 1,NOJACK
      DB(13+J) = WPRS(J,1)
      DB(13+NOJACK+J) = WPRS(J,2)
      DB(13+NJ1+J) = JPOS(J,1)
      DB(13+NJ2+J) = JPOS(J,2)
15    CONTINUE
      IF(IANAL.EQ.0.) GO TO 24
C
C*********************************************************************
C
C             SEND RAW DATA IN DATA BUFFER TO DISK
C
      IERROR = 0
      READ(11'1)KBUFF
      READ(11'2)LBUFF
      IF (ITERATION.EQ.1) NREC = NREC + 1
      IF(NREC.EQ.IFLIM) WRITE(CO,16)
      IF(NREC.EQ.IFLIM) WRITE(LO,16)
16    FORMAT(" WALDAT> WARNING - USING LAST RECORD AVAILABLE ***")
      IF(NREC.GT.IFLIM.OR.NREC.LT.2) IERROR = 999
      IF(IERROR.EQ.0) GO TO 18
      WRITE(CO,17)
      WRITE(LO,17)
17    FORMAT(/" WALDAT> RAW DATA ERROR")
      CALL ERROR(IERROR)
      STRSTAT = -1
18    ITEST(NREC) = IBTSTNR
      ITRUN(NREC) = IBRUNNR
      ITDATA(NREC) = IBPTNR
      ITER(NREC) = ITERATION
      WRITE(11'1)KBUFF
      WRITE(11'2)LBUFF
      IREC = NREC
      WRITE(11'IREC) DB
C********************************************************************
C
C         READ LOADED RAW DATA FROM DISK AS CHECK OR REANALYSIS
C
      READ(11'1) KBUFF
      READ(11'2) LBUFF
20    IREC = NREC
      READ(11'IREC) DB
24    IF(ABS(NREC).GT.IFLIM) NREC = 0
      WRITE(LO,25) NREC
25    FORMAT(/10X," FLXWAS RAW DATA FROM RECORD ",I3//)
      WRITE(LO,35)(DB(J),J=1,13)
35    FORMAT(F7.4,2F7.3,F7.2,F8.5,E10.2,2F5.2,F5.0,F4.0,F4.0,F4.0,F4.0)
      WRITE(LO,45)(DB(J+13),J=1,21)
      WRITE(LO,45)(DB(J+13+NOJACK),J=1,21)
      WRITE(LO,55)(DB(J+13+NJ1),J=1,21)
      WRITE(LO,55)(DB(J+13+NJ2),J=1,21)
45    FORMAT(11F7.2)
55    FORMAT(11F7.3)
      IF (IANAL.NE.2) GO TO 60
C
C     LOAD TUNNEL DATA FROM DISK
C
      FMACH = DB(1)
      PSTATIC = DB(2)
      PTOTAL = DB(3)
      TTOTAL = DB(4)
      DENSITY = DB(5)
      CREY = DB(6)
      XREYNO = CREY * 1E-06
      ALPHA = DB(7)
      CHORD = DB(8)
      IBHOUR = DB(9)
      IBMIN = DB(10)
      IBTSTNR = DB(11)
      IBRUNNR = DB(12)
      IBPTNR = DB(13)
      DO 56 J = 1,NOJACK
      WPRS(J,1) = DB(13+J)
      WPRS(J,2) = DB(13+NOJACK+J)
      JPOS(J,1) = DB(13+NJ1+J)
      JPOS(J,2) = DB(13+NJ2+J)
56    CONTINUE
      ITERATION = 1
C********************************************************************
C
C       SELECT 'STRAIGHT WALL' DATA RECORD AND ACQUIRE WALL DATA
C
60    UREYNO = XREYNO*12E06/CHORD
      CALL STWALL(FMACH,UREYNO,TWY,BWY,DSTAS,DSBAS,IERROR)
      IF(IERROR.EQ.0) GO TO 70
      WRITE(CO,65)
      WRITE(LO,65)
65    FORMAT(/" WALDAT> STRAIGHT WALL DATA ERROR")
      CALL ERROR(IERROR)
      STRSTAT=-1
      GO TO 130
```

```
70     CONTINUE
C
C       DETERMINE WALL POSITIONS FROM "STRAIGHT" (INCHES)
C
       DO 75 J = 1,NOJACK
       TWYAS(J) = TOPY(J) - TWY(J)
       BWYAS(J) = BOTY(J) - BWY(J)
75     CONTINUE
C
C              LOAD WALL PRESSURES INTO WORK ARRAYS
C
       NMPT1 = NOCPT - 1
       NMPT2 = NOCPT - 2
       DO 85 N = 1,2
       TOPWP(N) = PSTATIC
       BOTWP(N) = PSTATIC
85     CONTINUE
       DO 95 N = 3,NMPT2
       NN = N-2
       TOPWP(N) = WPRS(NN,1)
       BOTWP(N) = WPRS(NN,2)
95     CONTINUE
       DO 105 N = NMPT1,NOCPT
       TOPWP(N) = TOPWP(NMPT2)
       BOTWP(N) = BOTWP(NMPT2)
105    CONTINUE
C
C       CHECK WALL DATA FOR AUTO STREAMLINING
C
       DO 125 J = 1,NOJACK
       IF(ABS(JPOS(J,1)-TOPY(J)).GT.0.010) GO TO 110
       IF(ABS(JPOS(J,2)-BOTY(J)).GT.0.010) GO TO 115
       GO TO 125
110    IERROR = 5
       GO TO 120
115    IERROR = 6
120    WRITE(CO,121) J
       WRITE(LO,121) J
121    FORMAT(//," WALDAT> JACK",I3," POSITION ERROR")
       CALL ERROR(IERROR)
C?     STRSTAT = -1
125    CONTINUE
130    RETURN
       END
```

APPENDIX A.6 - LISTING OF SUBROUTINE STAR

PRECEDING PAGE BLANK NOT FILMED

PAGE 98 INTENTIONALLY BLANK

```
C     SUBROUTINE STAR(DST,DSB)
      SUBROUTINE STAR
C*****************************************************************
C
C
C     TITLE:          FLEXWALL BOUNDARY LAYER ASSESSMENT SUBROUTINE
C
C     AUTHOR:         STEPHEN W.D. WOLF , NATIONAL RESEARCH COUNCIL
C
C     SYSTEM:         CRYO-A
C
C     PURPOSE:        CALCULATE THE BOUNDARY LAYER DISPLACEMENT
C                     THICKNESS AND THE MACH NO. DISTRIBUTION
C                     ALONG EACH FLEXWALL
C
C
C     REVISION HISTORY:
C
C     VERSION      DATE       DESCRIPTION
C     =======      ====       ===================================
C       1.1      07/29/85     INITIAL VERSION USING THE VON KARMAN
C                             MOMENTUM INTEGRAL EQUATION.
C       1.2      12/22/86     CHANGE LP TO LO
C       1.3      12/23/86     REVISED WARNING MESSAGES
C*******************************************************************
C
C     COMPILER OPTIONS:             $23
C
C     LOGICAL FILES USED:
C                    CO - OUTPUT TO CONSOLE
C                    LO - OUTPUT TO LINEPRINTER
C
C     Shared Regions:
C         WASCOM
C
C     User-Defined Subroutine Calls:    None
C
C     Inputs:
C         /WASCOM/ NOCPT, XJACK, TOPWP, BOTWP, PTOTAL, DENSITY, CREY,
C     CHORD, DSTAS, DSBAS
C
C     NOCPT - Number of wall computing points.
C     XJACK - Array of jack X co-ordinates (Inches).
C     TOPWP - Array of top wall tapping pressures (PSIA).
C     BOTWP - Array of bottom wall tapping pressures (PSIA).
C     PTOTAL - Reference total pressure (PSIA).
C     DENSITY - Working fluid density (Slugs/ft**3).
C     CREY - Chord Reynolds number.
C     CHORD - Model chord (Inches).
C     DSTAS - Array of top wall d* values for ''aero. straight'' contour
C            (Inches).
C     DSBAS - Array of bottom wall d* values for ''aero. straight'' contour
C             (Inches).
C
C     Outputs:
C         /WASCOM/  TWDS, BWDS, TWMACH, BWMACH
C     TWDS - Array of top wall d* values for current contour (Inches).
C     BWDS - Array of bottom wall d* values for current contour (Inches).
```

```
C     TWMACH - Array of Mach numbers along current top wall contour.
C     BWMACH - Array of Mach numbers along current bottom wall contour.
C
C
C     Output Messages:
C              BOUNDARY LAYER CALCULATIONS
C              TOP WALL
C              TAP NO.  DU/DX  MACH NO.  D*  DD*
C                 (Table of data)
C              BOTTOM WALL
C                 (Table of data)
C
C    FLXWAS> *** WARNING - SONIC VEL. AT TOP WALL JACK ##   ***
C
C    FLXWAS> *** WARNING - SONIC VEL. AT BOTTOM WALL JACK ##   ***
C
C
C     Processing:
C              1) Calculate d* at each wall jack location using sets of
C                 data for three jack locations.
C              2) Load top wall pressures.
C              3) Load bottom wall pressures.
C              4) Local Mach no. calculations.
C              5) Calculate local wall flow velocities.
C              6) Assume a turbulent b/l growth to the second measuring
C                 point on each wall.
C              7) Calculate the velocity gradient at measuring point 1.
C              8) Guess the size of d*.
C              9) Calculate components of the M.I. Eqn.
C              10) Calculate d* from dd*/dx for last measuring point on
C                   each wall.
C              11) Calculate d* from dd*/dx for second measuring point on
C                   each wall.
C              12) Calculate d* from dd*/dx for all other measuring points.
C              13) Check d* guess and iterate to a solution.
C              14) Store correct values of d* (Inches).
C              15) Calculate dd*.
C              16) Check for sonic flow on the flexwalls.
C              17) Return to main program.
C
C
C     Exception Handling:
C
C        A warning is given when sonic velocity is encountered at any of
C     the wall jacks.  The operator should then proceed with caution in the
C     knowledge that wall streamlining is no longer ''routine'' and may not
C     be possible at all.
C
C
C*********************************************************************
      INTEGER CO
      INCLUDE USL/WASCOM,NOLIST
      DATA CO/@CO/,LO/@LO/
5     FORMAT(//13X," BOUNDARY LAYER CALCULATIONS"//"   TOP WALL"/)
      WRITE(LO,5)
15    FORMAT(2X," TAP NO.",2X," DU/DX",5X," MACH NO.",6X," D*",
     + 9X," DD*")
      WRITE (LO,15)
      NJ2 = NOCPT * 2
C
C     CALCULATE D* AT EACH WALL MEASURING POINT USING SETS OF THREE
C                MEASURING POINTS (LABELLED 0,1,2)
C
      DO 105 N = 2,NJ2
      NN = N
      IF(N.GT.NOCPT) NN = N-NOCPT
      IF(NN.GT.(NOCPT-2)) GO TO 105
25    FORMAT(//"   BOTTOM WALL"/)
      IF(N.EQ.(NOCPT+1)) WRITE (LO,25)
      IF(N.EQ.(NOCPT+1)) WRITE (LO,15)
      IF(NN.EQ.1) GO TO 105
      X1 = XJACK(NN) - XJACK(NN-1)
      X2 = XJACK(NN+1) - XJACK(NN-1)
      IF(N.GT.NOCPT) GO TO 10
C
C     LOAD TOP WALL PRESSURES
C
      WP1 = TOPWP(NN)
      WP2 = TOPWP(NN+1)
      WP3 = TOPWP(NN-1)
      GO TO 20
C
C     LOAD BOTTOM WALL PRESSURES
C
10    WP1 = BOTWP(NN)
      WP2 = BOTWP(NN+1)
      WP3 = BOTWP(NN-1)
C
C     LOCAL MACH NO. CALCULATIONS
C
20    P1 = 0.28571*ALOG(PTOTAL/WP1)
      P2 = 5.0*(EXP(P1)-1)
      WMACH = SQRT(P2)
C
C     CALCULATE LOCAL WALL FLOW VELOCITES
C
      Q1 = 0.7*WP1*WMACH*WMACH
      P1 = 0.28571*ALOG(PTOTAL/WP2)
      P2 = 5.0*(EXP(P1)-1)
      WM1 = SQRT(P2)
      Q2 = 0.7*WP2*WM1*WM1
      P1 = 0.28571*ALOG(PTOTAL/WP3)
      P2 = 5.0*(EXP(P1)-1)
      WM2 = SQRT(P2)
      Q3 = 0.7*WP3*WM2*WM2
      U1 = SQRT(291.508*Q1/DENSITY)
      U2 = SQRT(291.508*Q2/DENSITY)
      U3 = SQRT(291.508*Q3/DENSITY)
      IF (NN.EQ.2) GO TO 30
      GO TO 40
C
C     ASSUME A TURBULENT B/L GROWTH TO THE SECOND MEASURING
C                      POINT ON EACH WALL
C
30    R1 = CREY/(CHORD/12)
      VISCOSITY = U1*DENSITY*32.18E6/R1
      REYNO = (R1*X1)/12
      P1 = 0.142857*ALOG(REYNO)
      P2 = EXP(P1)
      P3 = 0.00127 * X1
      FPDELTA = P3/P2
      GO TO 105
C
C     CALCULATE THE VELOCITY GRADIENT AT MEASURING POINT 1
C
40    Y1 = U1 - U3
      Y2 = U2 - U3
      A1 = (Y2-((X2*Y1)/X1))/((X2*X2)-(X1*X2))
      B2 = (Y1-(A1*(X1*X1)))/X1
      VELGRAD = (2*A1*X1) + B2
C
C     GUESS THE SIZE OF D*
C
      IDINC = 0
      ITRIP = 1
50    IDINC = IDINC+1
      IF(ITRIP.EQ.1)DGUESS = (25 + IDINC*100)/1E6
      IF(ITRIP.EQ.2)DGUESS = DGUESS - 25E-6
C
C     CALCULATE COMPONENTS OF THE M.I. EQN
C
      P1 = (U1*DGUESS*DENSITY*32.18E6)/VISCOSITY
      P2 = 0.25 * ALOG(P1)
      P3 = EXP(P2)
      P4 = 0.0128/P3
      IF (NN.EQ.NOCPT) GO TO 60
      RCDEL = P4-(3.4*DGUESS*VELGRAD*12/U1)
      IF (NN.EQ.3) GO TO 70
      GO TO 80
C
C     CALCULATE D* FROM DD*/DX FOR LAST MEASURING POINT ON EACH WALL
C
60    RCDEL = P4-(3.4*DGUESS*OLDGRAD*12/U1)
      GO TO 80
C
C     CALCULATE D* FROM DD*/DX FOR SECOND MEASURING POINT ON EACH WALL
C
70    DELTA = (RCDEL*X1/12) + FPDELTA
      GO TO 90
C
C     CALCULATE D* FROM DD*/DX FOR ALL OTHER MEASURING POINTS
C
80    DELTA = OLDELTA+(0.5*(RCDEL+OLDRCD)*X1/12)
90    IF (ITRIP.EQ.2.AND.DELTA.GT.DGUESS) GO TO 100
C
C     CHECK D* GUESS AND ITERATE TO A SOLUTION
C
      IF (DELTA.LT.DGUESS) ITRIP = 2
      GO TO 50
100   DGUESS = DGUESS + 25E-6
C
C     STORE CORRECT VALUES OF D* (INCHES)
C
      NNJ = NN - 2
      IF(N.LE.NOCPT) TWDS(NNJ) = 16.8 * DGUESS
      IF(N.GT.NOCPT) BWDS(NNJ) = 16.8 * DGUESS
C
C     CALCULATE DD*
C
      IF (N.LE.NOCPT) DDSTAR = DSTAS(NNJ) - TWDS(NNJ)
      IF (N.GT.NOCPT) DDSTAR = BWDS(NNJ) - DSBAS(NNJ)
35    FORMAT(5X,I2,F12.4,F11.3,1X,2F12.4)
45    FORMAT(5X,I2,F12.4,"  ***",F6.3," ***",F9.4,F12.4)
      IF(N.GT.NOCPT) GO TO 120
      IF(WMACH.LE..99) WRITE(LO,35) NNJ,VELGRAD,WMACH,TWDS(NNJ),DDSTAR
      IF(WMACH.GT..99) WRITE(LO,45) NNJ,VELGRAD,WMACH,TWDS(NNJ),DDSTAR
      IF(WMACH.GT..99)  WRITE(CO,55) NNJ
55    FORMAT(" FLXWAS> WARNING - ***  SONIC VEL. AT TOP WALL JACK"
     +      ,I4," ***")
      TWMACH(NNJ)=WMACH
      GO TO 110
120   IF(WMACH.LE..99) WRITE(LO,35) NNJ,VELGRAD,WMACH,BWDS(NNJ),DDSTAR
      IF(WMACH.GT..99) WRITE(LO,45) NNJ,VELGRAD,WMACH,BWDS(NNJ),DDSTAR
      IF(WMACH.GT..99)  WRITE(CO,65) NNJ
65    FORMAT(2X," FLXWAS> WARNING - *** SONIC VEL. AT BOTTOM WALL JACK"
     +     ,I4," ***")
      BWMACH(NNJ)=WMACH
110   OLDELTA = DELTA
      OLDRCD = RCDEL
      IF(NN.EQ.(NOCPT-1)) OLDGRAD=VELGRAD
105   CONTINUE
      RETURN
      END
```

APPENDIX A.7 - LISTING OF SUBROUTINE WAS

```
C     SUBROUTINE WAS(JPOSN)
      SUBROUTINE WAS
C****************************************************************
C
C
C     TITLE:           WALL ADJUSTMENT STRATEGY SUBROUTINE
C
C     AUTHORS:         MICHAEL JUDD , UNIVERSITY OF SOUTHAMPTON
C                      STEPHEN W.D. WOLF , NATIONAL RESEARCH COUNCIL
C
C     SYSTEM:          CRYO-A
C
C     PURPOSE:         PREDICTS A NEW SET OF FLEXWALL CONTOURS AND
C                      ASSOCIATED EXTERNAL (IMAGINARY) VELOCITIES
C
C
C     REVISION HISTORY:
C
C     VERSION      DATE       DESCRIPTION
C     =======      ====       ========================
C       1.1      08/13/85     STRATEGY A ( VERSION USED IN TSWT
C                             UNIVERSITY OF SOUTHAMPTON , UK ) AND
C                             STRATEGY B ( EMPTY TEST SECTION
C                             STREAMLINING ) COMBINED.
C       1.2      09/26/85     BETA CORRECTION TO JACK MOVEMENTS
C                             REMOVED
C       1.3      02/25/86     CHANGED LP TO LO
C       1.4      04/30/86     REVISED COMMENTS
C       1.5      12/22/86     ALLOW JACKS #1 AND #22 TO FLOAT
C                             UNDER CONTROL + CHANGES TO DIFFUSER JACKS
C*******************************************************************
C
C     COMPILER OPTIONS:             $23
C
C     LOGICAL FILES USED:
C                    CO - OUTPUT TO CONSOLE
C                    LO - OUTPUT TO LINEPRINTER
C
C     Shared Regions:
C         WASCOM
C
C     User-Defined Subroutine Calls:  None
C
C     Inputs:
C         /WASCOM/ NOCPT, FMACH, TWCPLF, BWCPLF, TWSF, BWSF, PSTATIC,
C     TOPWP, BOTWP, TWVEL, BWVEL, XJACK, TWMACH, BWMACH, WMOVF, IANAL
C
C     NOCPT - Number of wall computing points.
C     FMACH - Free stream Mach number.
C     TWCPLF - Top wall coupling factor.
C     BWCPLF - Bottom wall coupling factor.
C     TWSF - Top wall scaling factor.
C     BWSF - Bottom wall scaling factor.
C     PSTATIC - Reference static pressure (PSIA).
C     TOPWP - Array of top wall tapping pressures (PSIA).
C     BOTWP - Array of bottom wall tapping pressures (PSIA).
C     TWVEL - Array of top wall external velocities for current contour
C             (v/U0).
```

```
C     BWVEL - Array of bottom wall external velocities for current contour
C           (v/U0).
C     XJACK - Array of jack X co-ordinates (Inches).
C     TWMACH - Array of top wall Mach numbers for current contour.
C     BWMACH - Array of bottom wall Mach numbers for current contour.
C     WMOVF - Wall movement factor for empty test section streamlining.
C     IANAL - Analysis mode designator.
C
C     Outputs:
C         /WASCOM/ TWVDIF, BWVDIF, TWVSQ, BWVSQ, TWNVEL, BWNVEL, TWMOV,
C     BWMOV
C
C     TWVDIF - Array of top wall velocity differences (dv/U0).
C     BWVDIF - Array of bottom wall velocity differences (dv/U0).
C     TWVSQ - Array of top wall velocities squared (v**2/U0**2).
C     BWVSQ - Array of bottom wall velocities squared (v**2/U0**2).
C     TWNVEL - Array of top wall external velocities for the new
C              (destination) contour (v/U0).
C     BWNVEL - Array of bot. wall external velocities for the new
C              (destination) contour(v/U0).
C     TWMOV - Array of top wall jack movements for the new (destination)
C             contour (Inches).
C     BWMOV - Array of bottom wall jack movements for the new (destination)
C             contour (Inches).
C
C
C     Output Messages:
C
C         ****** WAS V1.5 COMPUTING NOW !! ******
C
C         COUPLING FACTORS  ##.##  ##.##  SCALING FACTORS  ##.##  ##.##
C
C           WALL MOVE FACTOR (DY/DME)    ##.###
C
C
C     Processing:
C              1) DO - Compute the velocity imbalance/vorticity at each
C                 wall computing point and external velocities for the
C                 predicted wall contours for the next iteration.
C              2) Apply Prandtl-Glauert factor to measured top wall Cps.
C              3) Apply Prandtl-Glauert factor to measured bottom wall Cps.
C              4) Apply scaling factors to the external velocity
C                 calculations.
C              5) ENDDO - Apply coupling factors to the external
C                 velocities.
C              6) Freeze calculations for jack #1 and #22.
C              7) For empty test section streamlining jump to 22.
C              8) Make the first mid-jack co-ordinate at the wall anchor
C                 point to ensure a zero wall slope at this location.
C              9) Determine other mid-jack co-ords between wall computing
C                 points.
C              10) DO-Piecewise cubic curve fit to the wall vorticity using
C                  sets of four computing points (labeled 1,2,3,4).
C              11) ENDDO-Calculate coefficients for each piecewise cubic
C                  curve fit.
C              12) At each mid-jack point, integrate the vorticity along
C                  each wall to find  the induced vertical velocities,
C                  assumed normal to the top and bottom walls, which must
```

```
C                    be canceled by changes in the free stream component
C                    caused by local adjustments of wall slope.
C                13) Initialize wall movement demand accumulators.
C                14) Set wall slopes at the wall anchor points equal to zero.
C                15) DO-Find the jack movements required for wall
C                    streamlining, by performing integrations of piecewise
C                    quadratic curves fitted to sets of three wall slopes.
C                16) Top wall - movement demand coefficients.
C                17) Bottom wall- movement demand coefficients.
C                18) Scale jack movement demands.
C                19) ENDDO-Couple jack movement demands.
C                20) Compute jack #1 and #22 movement.
C                21) Jump to 24.
C                22) Calculate jack movement demands for constant wall Mach
C                    number during empty test section streamlining.
C                23) Set imaginary (external) velocities to free stream.
C                24) Determine jack movement demands for the variable
C                    diffuser.
C                25) Return to main program.
C
C
C     Exception Handling:  None
C
C
C**********************************************************************
      REAL N
      INTEGER CO
      DIMENSION X(4),VEL(4),CUBCOE(30,4),XMIDJ(30),TSLOPE(30),BSLOPE(30)
      INCLUDE USL/WASCOM,NOLIST
      DATA CO/@CO/,LO/@LO/
105   FORMAT(/9X," ****** WAS V1.5 COMPUTING NOW !! ******")
      WRITE(CO,105)
      NCPT1 = NOCPT-2
      NCPT2 = NOCPT-3
      NCPT3 = NOCPT-4
135   FORMAT(/" COUPLING FACTORS =",2F5.2,"   SCALING FACTORS =",2F5.2)
      IF(IANAL.NE.1) WRITE(LO,135) TWCPLF,BWCPLF,TWSF,BWSF
C
C      COMPUTE THE VELOCITY IMBALANCE/WALL VORTICITY AT EACH WALL
      COMPUTING POINT AND EXTERNAL VELOCITIES FOR THE NEXT
C                    PREDICTED WALL CONTOURS
C
      BETA = SQRT(1-(FMACH*FMACH))
      Q1 = 0.7 * PSTATIC * FMACH * FMACH
10    DO 5 I = 1,NOCPT
      TCPC = (TOPWP(I)-PSTATIC)/Q1
C
C      APPLY PRANDTL-GLAUERT FACTOR TO MEASURED TOP WALL CPS
C
      TCPI = BETA*TCPC
      TVRATIO = SQRT(1-TCPI)
      TVEL = TVRATIO-1
      TWVDIF(I) = TVEL-TWVEL(I)
      TWVSQ(I) = (TVEL+1)*(TVEL+1)
      BCPC = (BOTWP(I)-PSTATIC)/Q1
C
C      APPLY PRANDTL-GLAUERT FACTOR TO MEASURED BOTTOM WALL CPS
C
```

```
      BCPI = BETA*BCPC
      BVRATIO = SQRT(1-BCPI)
      BVEL = BVRATIO-1
      BWVDIF(I) = BWVEL(I)-BVEL
      BWVSQ(I) = (BVEL+1)*(BVEL+1)
C
C      APPLY SCALING FACTORS TO THE EXTERNAL VELOCITY CALCULATIONS
C
      TWNVEL(I) = TWVEL(I)+(TWSF*TWVDIF(I)/2)
      BWNVEL(I) = BWVEL(I)-(BWSF*BWVDIF(I)/2)
C
C      APPLY COUPLING FACTORS TO THE EXTERNAL VELOCITIES
C
      TNVEL = TWNVEL(I)
      TWNVEL(I) = TWNVEL(I)+(BWCPLF*(BWNVEL(I)-BVEL))
      BWNVEL(I) = BWNVEL(I)+(TWCPLF*(TNVEL-TVEL))
5     CONTINUE
C?
C?    0.3-M TCT HARDWARE REQUIREMENTS
C?    FREEZE CALCULATIONS FOR JACK #1 AND #22
C?
      TWVDIF(3) = 0.0
      BWVDIF(3) = 0.0
      TWNVEL(3) = 0.0
      BWNVEL(3) = 0.0
      IF(IANAL.EQ.1) GO TO 80
C
C      MAKE THE FIRST MID-JACK CO-ORD AT THE WALL ANCHOR POINT
C           TO ENSURE A ZERO WALL SLOPE AT THIS LOCATION
C
      XMIDJ(1) = XJACK(1)
C
C      DETERMINE OTHER MID-JACK CO-ORDS BETWEEN WALL COMPUTING POINTS
C
      DO 15 I = 2,NCPT1
      XMIDJ(I) = (XJACK(I)+XJACK(I+1))/2
15    CONTINUE
      DO 25 NN = 1,2
C
C      PIECEWISE CUBIC CURVE FIT TO THE WALL VORTICITY USING
C        SETS OF FOUR COMPUTING POINTS (LABELLED 1,2,3,4)
C
      DO 95 IL = 1,NCPT2
      I = IL - 1
40    DO 35 J = 1,4
      X(J) = XJACK(I+J)
      IF (NN.EQ.1) GO TO 50
      VEL(J) = BWVDIF(I+J)
      GO TO 35
50    VEL(J) = TWVDIF(I+J)
35    CONTINUE
      V0 = (VEL(3)-VEL(2))/(X(3)-X(2))
      V1 = VEL(2)-V0*X(2)
      DIST1 = 1/(X(4)-X(1))
      V2 = (VEL(4)-V0*X(4)-V1)/((X(4)-X(2))*(X(3)-X(4)))
      V3 = (VEL(1)-V0*X(1)-V1)/((X(1)-X(2))*(X(3)-X(1)))
      V4 = DIST1*(V2-V3)
      V5 = V3-V4*X(1)
      DIST2 = X(2) + X(3)
C
C      CALCULATE COEFFICIENTS FOR EACH PIECEWISE CUBIC CURVE FIT
C
      CUBCOE(IL,1) = V1-X(2)*X(3)*V5
      CUBCOE(IL,2) = V0+V5*DIST2-V4*X(2)*X(3)
      CUBCOE(IL,3) = V4*DIST2-V5
      CUBCOE(IL,4) = -V4
95    CONTINUE
C
C      AT EACH MID-JACK POINT,INTEGRATE THE VORTICITY ALONG EACH WALL
C      TO FIND THE INDUCED VERTICAL VELOCITIES,ASSUMED NORMAL TO THE
C      TOP AND BOTTOM WALLS,WHICH MUST BE CANCELLED BY CHANGES IN THE
C      FREE STREAM COMPONENT CAUSED BY LOCAL ADJUSTMENTS OF WALL SLOPE
C
      DO 45 J = 2,NCPT1
      X0 = XMIDJ(J)
      X0SQ = X0*X0
      X0CUB = X0SQ*X0
      VELSUM = 0.0
      DO 55 I = 1,NCPT2
      X1 =XJACK(I+1)
      COEFF0 = CUBCOE(I,1)
      COEFF1 = CUBCOE(I,2)
      COEFF2 = CUBCOE(I,3)
      COEFF3 = CUBCOE(I,4)
      X2 = XJACK(I+2)
      X2SQ = X2 * X2
      X1SQ = X1 * X1
      SUM0 = COEFF0+COEFF1*X0+COEFF2*(X0SQ)+COEFF3*(X0CUB)
      X3 = ABS(X2-X0)/ABS(X1-X0)
      X4 = ALOG(X3)
      SUM1 = (COEFF1+COEFF2*X0+COEFF3*X0SQ)*(X2-X1)
      SUM2 = (COEFF2+COEFF3*X0)*((X2SQ)-(X1SQ))/2
      SUM3 = COEFF3*((X2SQ*X2)-(X1SQ*X1))/3
      VELSUM = VELSUM+SUM0*X4+SUM1+SUM2+SUM3
55    CONTINUE
      IF (NN.EQ.2) GO TO 60
      TSLOPE(J) = VELSUM/6.28319
      GO TO 45
60    BSLOPE(J) = VELSUM/6.28319
45    CONTINUE
25    CONTINUE
C
C      INITIALISE WALL MOVEMENT DEMAND ACCUMULATORS
C
      TMOV = 0.0
      BMOV = 0.0
C
C      SET WALL SLOPES AT THE WALL ANCHOR POINTS EQUAL TO ZERO
C
      TSLOPE(1) = 0.0
      BSLOPE(1) = 0.0
C
C        FIND THE JACK MOVEMENTS REQUIRED FOR WALL STREAMLINING,
C      BY PERFORMING INTEGRATIONS OF PIECEWISE QUADRATIC CURVES
C                FITTED TO SETS OF THREE WALL SLOPES
C
```

```
      DO 65 I = 1,NCPT3
      I1 = I+1
      I2 = I+2
      TSGRAD = (TSLOPE(I2)-TSLOPE(I1))/(XMIDJ(I2)-XMIDJ(I1))
      BSGRAD = (BSLOPE(I2)-BSLOPE(I1))/(XMIDJ(I2)-XMIDJ(I1))
      XJ1SQ = XJACK(I1)*XJACK(I1)
      XJ2SQ = XJACK(I2)*XJACK(I2)
      XJ1CUB = XJ1SQ * XJACK(I1)
      XJ2CUB = XJ2SQ * XJACK(I2)
      X1 = XMIDJ(I)-XMIDJ(I1)
      X2 = XMIDJ(I2)-XMIDJ(I)
      P1 = (TSGRAD-(TSLOPE(I)-TSLOPE(I1))/X1)/X2
      P2 = TSGRAD - P1*XMIDJ(I2)
      X3 = XJACK(I2)-XJACK(I1)
C
C      TOP WALL - MOVEMENT DEMAND CUBIC COEFFICIENTS
C
      A = P1/3
      B = (P2-P1*XMIDJ(I1))/2
      C = TSLOPE(I1) - P2*XMIDJ(I1)
      TMOV = TMOV+(A*(XJ2CUB-XJ1CUB))+(B*(XJ2SQ-XJ1SQ))+(C*X3)
      P1 = (BSGRAD-(BSLOPE(I)-BSLOPE(I1))/X1)/X2
      P2 = BSGRAD - P1*XMIDJ(I2)
C
C      BOTTOM WALL - MOVEMENT DEMAND CUBIC COEFFICIENTS
C
      A = P1/3
      B = (P2-P1*XMIDJ(I1))/2
      C = BSLOPE(I1) - P2*XMIDJ(I1)
      BMOV = BMOV+(A*(XJ2CUB-XJ1CUB))+(B*(XJ2SQ-XJ1SQ))+(C*X3)
C
C      SCALE JACK MOVEMENT DEMANDS
C
      STMOV = TWSF * TMOV
      SBMOV = BWSF * BMOV
C
C      COUPLE JACK MOVEMENT DEMANDS
C
      TWMOV(I) = STMOV+(BWCPLF*SBMOV)
      BWMOV(I) = SBMOV+(TWCPLF*STMOV)
65    CONTINUE
C?
C?    0.3-M TCT HARDWARE REQUIREMENT
C?    COMPUTE JACK #1 AND #22 MOVEMENT
C?
      TWMOV(1) = TWMOV(2)/2
      BWMOV(1) = BWMOV(2)/2
      TW1 = TOPY(1)+TWMOV(1)
      BW1 = BOTY(1)+BWMOV(1)
      IF (TW1.GT.0.075) TWMOV(1) = 0.075-TOPY(1)
      IF (BW1.GT.0.065) BWMOV(1) = 0.065-BOTY(1)
      IF (TW1.LT.-.075) TWMOV(1) = -0.075-TOPY(1)
      IF (BW1.LT.-.065) BWMOV(1) = -0.065-BOTY(1)
      GO TO 70
C
C      CALCULATE JACK MOVEMENT DEMANDS FOR CONSTANT WALL MACH NUMBER
C                DURING EMPTY TEST SECTION STREAMLINING
C
```

```
80    SUM1 = 0.0
      SUM2 = 0.0
      SUM3 = 0.0
      SUM4 = 0.0
      DO 75 J = 1,NCPT3
      NM = J+2
      TMERR = TWMACH(J)-FMACH
      SUM1 = SUM1+TMERR
      SUM2 = SUM2+(TMERR*TMERR)
      TWMOV(J) = TMERR*WMOVF
      BMERR = FMACH-BWMACH(J)
      SUM3 = SUM3+BMERR
      SUM4 = SUM4+(BMERR*BMERR)
      BWMOV(J) = BMERR*WMOVF
75    CONTINUE
C
C     SET EXTERNAL (IMAGIANARY) VELOCITIES TO FREE STREAM
C
      DO 145 J = 1,NOCPT
      TWNVEL(J) = 0.0
      BWNVEL(J) = 0.0
145   CONTINUE
      TSD = SQRT((SUM2-((SUM1*SUM1)/NCPT3))/(NCPT3-1))
      BSD = SQRT((SUM4-((SUM3*SUM3)/NCPT3))/(NCPT3-1))
115   FORMAT(/9X,"  WALL MACH NO. STANDARD DEVIATIONS"/10X," TOP =",
     + F8.4,2X,"  BOTTOM   = ",F8.4)
      WRITE(CO,115)TSD,BSD
      WRITE(LO,115)TSD,BSD
125   FORMAT(/10X," WALL MOVE FACTOR (DY/DME) = ",F5.3)
      WRITE(LO,125)WMOVF
C
C     0.3-M TCT HARDWARE REQUIREMENT
C     DETERMINE MOVEMENT DEMANDS FOR THE VARIABLE DIFFUSER
C     JACKS #19, #20, AND #21 ON BOTH WALLS
C
70    FRACT = 6
      DO 85 J = 19,21
C     TWMOV(J) = TWMOV(18)-((TWMOV(18)*(XJACK(J+2)-55.8))/14.9)
      TWMOV(J) = TWMOV(18) * (FRACT/8.)
C     BWMOV(J) = BWMOV(18)-((BWMOV(18)*(XJACK(J+2)-55.8))/14.9)
      BWMOV(J) = BWMOV(18) * (FRACT/8.)
      FRACT = FRACT + (J-22)
85    CONTINUE
      RETURN
      END
```

APPENDIX A.8 - LISTING OF SUBROUTINE SUME

```
C     SUBROUTINE SUME (STRSTAT)
      SUBROUTINE SUME
C*********************************************************************
C
C
C     TITLE:        WALL INDUCED INTERFERENCES ASSESSMENT SUBROUTINE
C
C     AUTHOR:       STEPHEN W.D. WOLF , NATIONAL RESEARCH COUNCIL
C
C     SYSTEM:       CRYO-A
C
C
C     PURPOSE:      CALCULATE WALL INDUCED INTERFERENCES ALONG THE
C                   TEST SECTION CENTERLINE AND THE MODEL CHORD-
C                   LINE, USING THE FLEXWALL PRESSURES. THEN
C                   ASSESS THE QUALITY OF WALL STREAMLINING.
C
C
C     REVISION HISTORY:
C     VERSION      DATE      DESCRIPTION
C     =======      ====      =====================================
C       1.1      05/23/85    INITIAL VERSION USING SIMPLE LINEARISED
C                            THEORY.
C       1.2      02/21/86    CHANGE LP TO LO
C       1.3      04/01/86    REMOVE STREAMLINE MESSAGE
C       1.4      04/04/86    PRINT OUT XORIG AND YORIG
C       1.5      12/22/86    IMPROVE OUTPUT TO CONSOLE AND
C                            MAKE RESIDUAL LIMITS MORE VISIBLE
C       1.6      02/18/87    INTERFERENCES BETAIZED
C
C**********************************************************************
C
C     COMPILER OPTIONS:             $23
C
C     LOGICAL FILES USED:
C                   CO - OUTPUT TO CONSOLE
C                   LO - OUTPUT TO LINEPRINTER
C
C     Shared Regions:
C         FLXCOM
C         WASCOM
C
C     User-Definded Subroutine Calls:  None
C
C     Inputs:
C         /WASCOM/ ALPHA, CHORD, XORIG, YORIG, NOJACK, NOCPT, BWVEL, TWVEL,
C     TWVSQ, BWVSQ, TWYAS, BWYAS, DSTAS, DSBAS, TWDS, BWDS, TWVDIF, BWVDIF,
C     XJACK, WL, IANAL, BETA
C
C     Outputs:
C         /FLXCOM/ STRSTAT
C     STRSTAT - Streamlining status word.
C
C     Output Messages:
C              WALL CP ERROR
C         TOP - ###.####   BOTTOM - ###.###  ##.## (XORIG) ##.## (YORIG)
C
C         SUME> V1.6 RESIDUAL ERRORS
```

```
C         XO        U/UFS      V/UFS
C       (Table of values on tunnel centerline)
C         MODEL ERRORS
C       (Table of values on model chordline)
C
C              EFFECT         DELTA CL
C         ALPHA ERROR = #.### DEGREES     ###.###
C         INDUCED CAMBER = #.### DEGREES  ###.###
C         CP ERROR (1/4 CHORD) = #.###
C         CP ERROR (AVERAGE) = #.###      ###.###
C
C         RESIDUAL INTERFERENCES =  ###.####  ###.####  ###.####
C                                   A.O.A.    CAMBER    CP
C
C         ITERATION ##  CP ERROR - TOP ##.###  BOT ##.###
C         RESIDUALS - ###.#### , ###.#### , ###.####
C
C     Processing:
C              1) Set acceptable levels of residual interference.
C              2) Calculate average wall Cp errors (Big E).
C              3) Sum wall induced velocities on tunnel centerline.
C              4) Betaize the vertical ordinate of the vortex sheet.
C              5) Sum wall induced velocities along model chordline.
C              6) Betaize the vertical ordinate of the vortex sheet.
C              7) Integrate the induced camber over the model chord.
C              8) Assess the quality of wall streamlining and decide if the
C                  streamlining cycle can be terminated.
C              9) Output data summary to operator console.
C              10) Return to main program.
C
C     Exception Handling:     None
C
C*********************************************************************
      INTEGER CO
      INCLUDE USL/FLXTYP,NOLIST
      DIMENSION UIND(49),VIND(49),CAMC(9)
      INCLUDE USL/FLXCOM,NOLIST
      INCLUDE USL/WASCOM,NOLIST
      DATA CO/@CO/,LO/@LO/
      DATA CAMC/1.,.75,.5,.25,0.,-.25,-.5,-.75,-1./
      TOPI = 6.283185
C*********************************************************************
C
C              0.3-M TCT TEST SECTION GEOMETRY
C
C*********************************************************************
C
C     STREAMWISE LOCATION OF MODEL PIVOT (TURNTABLE CENTER)
C     RELATIVE TO THE FLEXWALL ANCHOR POINT (INCHES)
      TURNTC = 30.75
C
C     TUNNEL HALF HEIGHT (INCHES)
      THALFH=6.5
C****************************************************************
C
C       SET ACCEPTABLE LEVELS OF RESIDUAL WALL INTERFERENCES
C
C****************************************************************
```

```
      WCPE = 0.01
      RIAOA = 0.015
      RIAVA = 0.07
      RICPE = 0.007
C****************************************************************
      AN1 = ALPHA * .01745
      AX = (CHORD*COS(AN1))/8.
      AY = (CHORD*SIN(AN1))/8.
      OR = TURNTC - (XORIG*COS(AN1)) + (YORIG*SIN(AN1))
      CPSUM = 0.0
      NJ1 = NOJACK-3
      YPOS = 0.0
      WDATUM = THALFH
      TSUM = 0.
      BSUM = 0.
      NMPT1 = NOCPT-2
C
C     CALCULATE AVERAGE WALL CP ERRORS (BIG E)
C
      DO 5 I = 3,NMPT1
      V1 = BWVEL(I)+1
      V2 = TWVEL(I)+1
      BWVEL(I) = V1 * V1
      TWVEL(I) = V2 * V2
      TSUM = TSUM + ABS(TWVSQ(I)-TWVEL(I))
      BSUM = BSUM + ABS(BWVSQ(I)-BWVEL(I))
5     CONTINUE
      CET = TSUM/NJ1
      CEB = BSUM/NJ1
105   FORMAT(//17X,"    WALL CP ERROR"/8X," TOP  -   ",F8.4,
     +        5X," BOTTOM  -  ",3F8.4)
      WRITE(LO,105) CET,CEB,XORIG,YORIG
      IF(IANAL.EQ.0)GO TO 10
115   FORMAT(///12X," SUME> V1.6 RESIDUAL ERRORS")
      WRITE(LO,115)
125   FORMAT(6X," X0",8X," U/UFS",6X," V/UFS")
      WRITE(LO,125)
      DO 15 I=1,49
      X1 = OR + ((I-25)*1.0)
      SUMU = 0.0
      SUMV = 0.0
C
C     SUM WALL INDUCED VELOCITIES ON TUNNEL CENTERLINE
C
      DO 25 J = 1,NJ1
      X2 = X1 - (XJACK(J+2))
      Y1 = YPOS+WDATUM+TWYAS(J)+(DSTAS(J)-TWDS(J))
      Y2 = YPOS-WDATUM-BWYAS(J)+(BWDS(J)-DSBAS(J))
      P1 = X2 * X2
C
C     BETAIZE THE VERTICAL ORDINATE OF THE VORTEX SHEET
C
      BY1 = BETA * Y1
      BY2 = BETA * Y2
      P2 = P1 + (BY1*BY1)
      P3 = P1 + (BY2*BY2)
      R1 = X2/P2
      R2 = Y1/P2
      R3 = Y2/P3
      U1 = (TWVDIF(J+2)*R2)+(BWVDIF(J+2)*R3)
      U1 = (U1*WL(J))/TOPI
      V1 = (TWVDIF(J+2)+BWVDIF(J+2))*R1
      V1 = (V1*WL(J))/TOPI
      SUMU = SUMU + U1
      SUMV = SUMV + V1
25    CONTINUE
      UIND(I) = SUMU
      VIND(I) = SUMV
      X3 = (I-25) * 1.0
135   FORMAT(F11.2,1X,2F12.4)
      WRITE(LO,135) X3,UIND(I),VIND(I)
15    CONTINUE
145   FORMAT(/5X," MODEL ERRORS",25X," CP"/)
      WRITE(LO,145)
10    SUMODD = 0.0
      SUMEVEN = 0.0
      SUMEND = 0.0
      DO 35 K = 1,9
      X1 = OR - ((3-K)*AX)
      YPOS = -(YORIG*COS(AN1))-(XORIG*SIN(AN1))-((3-K)*AY)
      SUMU = 0.0
      SUMV = 0.0
C
C     SUM WALL INDUCED VELOCITIES ALONG MODEL CHORDLINE
C
      DO 45 J = 1,NJ1
      X2 = X1 - (XJACK(J+2))
      Y1 = YPOS+WDATUM+TWYAS(J)+(DSTAS(J)-TWDS(J))
      Y2 = YPOS-WDATUM-BWYAS(J)+(BWDS(J)-DSBAS(J))
      P1 = X2 * X2
C
C     BETAIZE THE VERTICAL ORDINATE OF THE VORTEX SHEET
C
      BY1 = BETA * Y1
      BY2 = BETA * Y2
      P2 = P1 + (BY1*BY1)
      P3 = P1 + (BY2*BY2)
      R1 = X2/P2
      R2 = Y1/P2
      R3 = Y2/P3
      U1 = (TWVDIF(J+2)*R2)+(BWVDIF(J+2)*R3)
      U1 = (U1*WL(J))/TOPI
      V1 = (TWVDIF(J+2)+BWVDIF(J+2))*R1
      V1 = (V1*WL(J))/TOPI
      SUMU = SUMU + U1
      SUMV = SUMV + V1
45    CONTINUE
      ANGIND = ATAN(SUMV/(1+SUMU))
      CPIND = 1.0 - ((1+SUMU)*(1+SUMU))
      IF (K.EQ.3) CPERR = CPIND
      CPSUM = CPSUM + CPIND
      X1 = X1 - OR
155   FORMAT(4F12.4)
      IF(IANAL.GT.0)WRITE(LO,155) X1,SUMU,SUMV,CPIND
      IF (K.EQ.1) ANGLE = ANGIND
      IF (K.EQ.9) ANGTE = ANGIND
```

```
C
C     INTEGRATE THE INDUCED CAMBER OVER THE MODEL CHORD
C
      CAMCL = ANGIND * (1-CAMC(K))
      IF (K.EQ.1.OR.K.EQ.9) GO TO 20
      INTEGER = K/2
      INTEGER = INTEGER*2
      IF (INTEGER.EQ.K) GO TO 30
      SUMODD = SUMODD + CAMCL
      GO TO 35
30    SUMEVEN = SUMEVEN + CAMCL
      GO TO 35
20    SUMEND = SUMEND + CAMCL
35    CONTINUE
      AVCPE = CPSUM/9.0
      P1 = SUMEND+(2*SUMODD)+(4*SUMEVEN)
      P2 = (AX/3)*P1
      TCAMCL = 2*P2
      AVANG = (ANGLE-ANGTE) * 59.29578
165   FORMAT(/10X," EFFECT",25X," DELTA CL")
      IF(IANAL.GT.0)WRITE(LO,165)
      ANGCL = TOPI * ANGLE
      ANGLE = ANGLE * 59.29578
      IF(IANAL.EQ.0) GO TO 40
175   FORMAT(/5X," ALPHA ERROR = ",F8.4," DEGREES",5X,F8.4)
      WRITE(LO,175) ANGLE,ANGCL
185   FORMAT(/5X," INDUCED CAMBER = ",F8.4," DEGREES",2X,F8.4)
      WRITE(LO,185) AVANG,TCAMCL
195   FORMAT(/5X," CP ERROR (1/4 CHORD) =",F8.4)
      WRITE(LO,195) CPERR
      VELCL=-CPERR
205   FORMAT(5X," CP ERROR ( AVERAGE ) =",F8.4,5X,F8.4///)
      WRITE (LO,205) AVCPE,VELCL
215   FORMAT(/"    RESIDUAL INTERFERENCES = ",3F8.4/30X,
     +" A.O.A.  CAMBER    CP ")
40    IF(IANAL.EQ.0) WRITE(LO,215) ANGLE,AVANG,AVCPE
      IF(IANAL.EQ.1) GO TO 50
C
C     ASSESS THE QUALITY OF WALL STREAMLINING AND DECIDE IF THE
C          STREAMLINING CYCLE CAN BE TERMINATED
C
      ISC=0
      IF(CET.LE.WCPE)ISC=ISC+1
      IF(CEB.LE.WCPE)ISC=ISC+1
      IF(ABS(ANGLE).LE.RIAOA)ISC=ISC+1
      IF(ABS(AVANG).LE.RIAVA)ISC=ISC+1
      IF(ABS(AVCPE).LE.RICPE)ISC=ISC+1
      IF(ISC.EQ.5) STRSTAT=1
C
C     OUTPUT DATA SUMMARY TO OPERATOR CONSOLE
C
      WRITE(CO,225) ITERATION,CET,CEB
225   FORMAT(55("*")/5X," ITERATION ",I2," CP ERROR - TOP ",F7.4,
     +" , BOT ",F7.4)
      WRITE(CO,235) ANGLE,AVANG,AVCPE
235   FORMAT(7X," RESIDUALS - ",F8.4,2(" ,",F8.4))
50    RETURN
      END
```

APPENDIX A.9 - LISTING OF SUBROUTINE OUT

```
C     SUBROUTINE OUT(STRSTAT)
      SUBROUTINE OUT
C********************************************************************
C
C
C     TITLE:            DATA MANIPULATION AND DISPLAY SUBROUTINE
C
C     AUTHOR:           STEPHEN W.D. WOLF , NATIONAL RESEARCH COUNCIL
C
C     SYSTEM:           CRYO-A
C
C
C     PURPOSE:          SORT WALL DATA AND OUTPUT SAME TO THE LINEPRINTER
C                       AND DISK.
C
C
C     REVISION HISTORY:
C     VERSION      DATE      DESCRIPTION
C     =======      ====      ========================================
C       1.1      05/23/85    INITIAL VERSION CAPABLE OF ONLY
C                            WRITING DATA TO LINEPRINTER
C       1.2      06/17/85    INCLUDE CALL TO SWFILE TO STORE
C                            NEW "STRAIGHT WALL" DATA IN THE
C                            REFERENCE TABLE ON DISK.
C       1.3      06/27/85    REDUCED WALL DATA STORAGE ON DISK
C       1.4      02/21/86    CHANGE LP TO LO
C       1.5      03/31/86    SET LOGICAL NAME OF REDUCED DATA FILE TO 12
C       1.6      05/16/86    SEND REDUCED DATA DIRECT TO TAPE
C       1.7      05/21/86    READ COMPLETE REDUCED DATAFILE DIRECTORY
C       1.8      06/25/86    REVISED ERROR MESSAGES
C       1.9      10/17/86    REMOVE REAL-TIME DATA STORAGE ON TAPE
C       2.0      12/22/86    ENSURE B/LAYER DATA IS STORED ON DISK
C       2.1      12/31/86    IMPROVED I/O OPERATIONS
C
C*********************************************************************
C
C     COMPILER OPTIONS:            $23
C
C     LOGICAL FILES USED:
C                       CO - OUTPUT TO CONSOLE
C                       LO - OUTPUT TO LINEPRINTER
C                       12 - REDUCED DATA DATAFILE
C
C     Shared Regions:
C         FLXCOM
C         WASCOM
C
C     User-Defined Subroutine Calls:
C         ERROR
C
C
C     Inputs:
C         /FLXCOM/ STRSTAT
C     STRSTAT - Streamlining status word.
C
C         /WASCOM/ CREY, CHORD, IANAL, NOJACK, FMACH, ALPHA, TOPY, BOTY,
C     TWMOV, BWMOV, TWYAS, BWYAS, TWNVEL, BWNVEL, TWDS, BWDS
C
```

```
C
C     Outputs:
C         /FLXCOM/ JPOSN
C     JPOSN - Array of new (destination) jack positions (Inches).
C
C         /WASCOM/ TWVEL, BWVEL, TOPY, BOTY
C     TWVEL - Array of top wall external velocities for the new
C             (destination) contour (v/U0).
C     BWVEL - Array of bot. wall external velocities for the new
C             (destination) contour (v/U0).
C     TOPY - Array of top wall new (destination) jack positions (Inches).
C     BOTY - Array of bottom wall new (destination) jack positions
C            (Inches).
C
C         Reduced data datafile - KBUFF, LBUFF, TMACH, TREYNO, TALPHA,
C     TCHORD, DATA (See listing of file TABASS for variable descriptions)
C
C
C     Output Messages:
C
C         DATA SUMMARY - EMPTY TEST SECTION STREAMLINING
C
C         DATA SUMMARY - RE-ANALYSIS OUTPUT
C
C         DATA SUMMARY - DEVELOPMENT STREAMLINING
C
C         DATA SUMMARY - INFINITE FLOW STREAMLINING
C
C         TOP WALL
C         JACK   FROM ST     NEXT VEL   MOVE   JACKS - NOW - TO SET
C              (Table of reduced data)
C         BOTTOM WALL
C              (Table of reduced data)
C
C         *** WARNING - USING LAST RECORD AVAILABLE ***
C
C         OUT>  REDUCED DATA DISKFILE IS FULL !!!
C
C         REDUCED DATA STORED IN RECORD  ###
C
C
C     Processing:
C              1) Decide what sort of printout is required.
C              2) Printout data summary and overwrite current jack
C                 positions with predicted positions (TOPY(*), BOTY(*)).
C              3) Load up TWVEL and BWVEL for next iteration.
C              4) If walls are streamlined (STRSTAT = 1) jump to 7.
C              5) Read oonfig. info. from the reduced data datafile.
C              6) Write reduced data and config. info. to the reduced data
C                 datafile.
C              7) Return to main program.
C
C
C     Exception Handling:
C
C         A warning is given to the operator that the last available record
C    on the reduced data datafile is being used.  The operator needs to
C    copy this datafile to tape and then initialize the datafile.   If no
```

```
C     corrective action is taken the next data point will result in an
C     error trap (STRSTAT = -1) causing an error message, a call to
C     subroutine ERROR for additional output information and the return of
C     control to the main program.
C
C
C***********************************************************************
      INTEGER CO
      INTEGER*4 IREC
      INCLUDE USL/FLXTYP,NOLIST
      DIMENSION TWY(21),BWY(21),TMACH(126),TREYNO(126),TALPHA(126)
      DIMENSION TCHORD(126),KBUFF(256),LBUFF(256),ITEST(126),ITRUN(126)
      DIMENSION ITDATA(126),ITER(126),DATA(128)
C
C     ***     COMMON     ***
C
      INCLUDE USL/FLXCOM,NOLIST
      INCLUDE USL/WASCOM,NOLIST
C
C     ***   EQUIVALENCE   ***
C
      EQUIVALENCE (KBUFF(1),NREC)
      EQUIVALENCE (KBUFF(2),ITEST(1))
      EQUIVALENCE (KBUFF(128),ITRUN(1))
      EQUIVALENCE (LBUFF(1),ITDATA(1))
      EQUIVALENCE (LBUFF(128),ITER(1))
      DEFINE FILE 12(126,256,U,IREC)
      DATA CO/@CO/,LO/@LO/
      IFLIM = 126
      UREYNO = CREY * 12 / CHORD
C
C          DECIDE WHAT SORT OF PRINTOUT IS REQUIRED
C
      IF(IANAL.EQ.0) WRITE(LO,205)
      IF(IANAL.EQ.1) WRITE(LO,155)
      IF(IANAL.EQ.2) WRITE(LO,165)
      IF(IANAL.EQ.3) WRITE(LO,175)
155   FORMAT(1H1,"    DATA SUMMARY - EMPTY TEST SECTION STREAMLINING")
165   FORMAT(1H1,"    DATA SUMMARY - RE-ANALYSIS OUTPUT")
175   FORMAT(1H1,"    DATA SUMMARY - DEVELOPMENT STREAMLINING")
205   FORMAT(1H1,"    DATA SUMMARY - INFINITE FLOW STREAMLINING")
105   FORMAT(3X," TOP WALL")
      WRITE(LO,105)
115   FORMAT(5X," JACK",3X," FROM ST",2X," NEXT VEL",3X," MOVE",
     + "  JACKS - NOW - TO SET")
      WRITE(LO,115)
      NJ1 = NOJACK-3
      NJ2 = NOJACK + NOJACK
      DATA(1) = FMACH
      DATA(2) = CREY
      DATA(3) = ALPHA
C
C      PRINTOUT DATA SUMMARY AND OVERWRITE CURRENT JACK POSITIONS
C             WITH PREDICTED POSITIONS ( TOPY(*) , BOTY(*) )
C
10    DO 5 J = 1,NOJACK
      NM = J + 2
      TWY(J)=TOPY(J)+TWMOV(J)
      JPOSN(J,1) = TWY(J)
      DATA(J+3) = TWY(J)
      WP = TWYAS(J)
125   FORMAT(7X,I2,F11.3,F11.4,F10.3,5X,2F8.3)
      IF(J.LE.NJ1)WRITE(LO,125)J,WP,TWNVEL(NM),TWMOV(J),TOPY(J),TWY(J)
145   FORMAT(7X,I2,F11.3,11X,F10.3,5X,2F8.3)
      IF (J.GT.NJ1) WRITE(LO,145) J,WP,TWMOV(J),TOPY(J),TWY(J)
      TOPY(J) = TWY(J)
      IF (IANAL.EQ.1) TWY(J) = TWY(J) - TWMOV(J)
5     CONTINUE
135   FORMAT(2X," BOTTOM WALL")
      WRITE(LO,135)
      WRITE(LO,115)
20    DO 15 J = 1,NOJACK
      NM = J + 2
      BWY(J)=BOTY(J)+BWMOV(J)
      JPOSN(J,2) = BWY(J)
      DATA(J+NOJACK+3) = BWY(J)
      WP = BWYAS(J)
      IF(J.LE.NJ1)WRITE(LO,125)J,WP,BWNVEL(NM),BWMOV(J),BOTY(J),BWY(J)
      IF (J.GT.NJ1) WRITE(LO,145) J,WP,BWMOV(J),BOTY(J),BWY(J)
      BOTY(J) = BWY(J)
      IF (IANAL.EQ.1) BWY(J) = BWY(J)-BWMOV(J)
15    CONTINUE
C
C      LOAD UP TWVEL & BWVEL FOR NEXT ITERATION
C
      DO 25 I = 1,NOCPT
      TWVEL(I) = TWNVEL(I)
      DATA(I+NJ2+3) = TWNVEL(I)
      BWVEL(I) = BWNVEL(I)
      DATA(I+NOCPT+NJ2+3) = BWNVEL(I)
25    CONTINUE
      NFILL = NJ2+3+NOCPT+NOCPT
      DO 30 I = 1,NJ1
      DATA(NFILL+I) = TWDS(I)
      DATA(NFILL+NJ1+I) = BWDS(I)
30    CONTINUE
      IF(STRSTAT.EQ.1) GO TO 40
C
C       READ REDUCED DATA CONFIG INFO FROM DISK
C
      READ(12'1) KBUFF
      READ(12'2) LBUFF
      READ(12'3) TMACH
      READ(12'4) TREYNO
      READ(12'5) TALPHA
      READ(12'6) TCHORD
      IF(ITERATION.EQ.1)NREC = NREC + 1
      IF(NREC.EQ.IFLIM) WRITE(CO,215)
      IF(NREC.EQ.IFLIM) WRITE(LO,215)
215   FORMAT(///10X," *** WARNING - USING LAST RECORD AVAILABLE ***")
      IF(NREC.LE.IFLIM)GO TO 230
      WRITE(LO,225)
225   FORMAT(//" OUT> REDUCED DATA DISKFILE IS FULL !!! ")
      IERROR = 999
      CALL ERROR(IERROR)
      STRSTAT = -1
      GO TO 40
 230  ITEST(NREC) = IBTSTNR
      ITRUN(NREC) = IBRUNNR
      ITDATA(NREC) = IBPTNR
      ITER(NREC) = ITERATION+1
      TMACH(NREC) = FMACH
      TREYNO(NREC) = UREYNO
      TALPHA(NREC) = ALPHA
      TCHORD(NREC) = CHORD
C
C       WRITE REDUCED DATA AND CONFIG INFO TO DISK
C
      WRITE(12'1) KBUFF
      WRITE(12'2) LBUFF
      WRITE(12'3) TMACH
      WRITE(12'4) TREYNO
      WRITE(12'5) TALPHA
      WRITE(12'6) TCHORD
      IREC = NREC
      WRITE(12'IREC) DATA
      WRITE(LO,235) NREC
235   FORMAT(/16X,"  REDUCED DATA STORED IN RECORD ",I3)
40    RETURN
      END
```

APPENDIX A.10 - LISTING OF SUBROUTINE ERROR

```
      SUBROUTINE ERROR(IERROR)
C*********************************************************************
C
C
C     TITLE:        FLXWAS ERROR MESSAGE SUBROUTINE
C
C     AUTHOR:       STEPHEN W.D. WOLF , NATIONAL RESEARCH COUNCIL
C
C     SYSTEM:       CRYO-A
C
C
C     PURPOSE:      OUTPUT ERROR MESSAGES TO THE LINEPRINTER
C                   AND CONSOLE
C
C
C     REVISION HISTORY:
C
C     VERSION     DATE       DESCRIPTION
C     =======     ====       ===========================================
C      1.1       06/25/86   INITIAL VERSION
C      1.2       12/23/86   REVISED ERROR MESSAGES
C
C*********************************************************************
C
C     COMPILER OPTIONS:               $23
C
C     LOGICAL FILES USED:
C                   CO - OUTPUT TO CONSOLE
C                   LO - OUTPUT TO LINEPRINTER
C
C     CALLING SEQUENCE:
C                   CALL ERROR(IERROR)
C                                  WHERE IERROR = INPUT ERROR CODE:
C                                      0 - NO ERROR
C                                      1 - FAILURE TO BRACKET MACH NO.
C                                      2 - FAILURE TO BRACKET REY. NO.
C                                      3 _ COMBINATION OF ABOVE ERRORS
C                                      4 - MISSING DATA RECORD
C                                      5 - JACK ERROR ON TOP WALL
C                                      6 - JACK ERROR ON BOTTOM WALL
C                                    999 - DATAFILE FULL
C
C     Shared Regions:   None
C
C     User-Defined Subroutine Calls:  None
C
C     Inputs:
C         Subroutine call - IERROR (Input error code)
C
C     Outputs:   None
C
C     Output Messages:
C
C         FLXWAS>  *** MACH NUMBER NOT BRACKETED ON DATAFILE ***
C
C         FLXWAS>** REYNOLDS NUMBER NOT BRACKETED ON DATAFILE **
C
C         FLXWAS>** TEST PARAMETERS NOT BRACKETED ON DATAFILE **
```

```
C
C          FLXWAS> *** MISSING DATA RECORD ON DATAFILE ***
C
C          FLXWAS> *** TOP WALL JACK POSITION ERROR ***
C
C          FLXWAS> *** BOTTOM WALL JACK POSITION ERROR ***
C
C          FLXWAS> *** DATAFILES NEED TO BE INITIALIZED ***
C
C      Processing:  None
C
C      Exception Handling:  None
C
C
C*********************************************************************
      INTEGER CO
      DATA CO/@CO/,LO/@LO/
      IF (IERROR.EQ.1) WRITE(LO,10)
      IF (IERROR.EQ.1) WRITE(CO,10)
10    FORMAT(" FLXWAS> ***  MACH NUMBER NOT BRACKETED ON DATAFILE ***")
      IF (IERROR.EQ.2) WRITE(LO,20)
      IF (IERROR.EQ.2) WRITE(CO,20)
20    FORMAT(" FLXWAS> ** REYNOLDS NUMBER NOT BRACKETED ON DATAFILE **")
      IF (IERROR.EQ.3) WRITE(LO,30)
      IF (IERROR.EQ.3) WRITE(CO,30)
30    FORMAT(" FLXWAS> ** TEST PARAMETERS NOT BRACKETED ON DATAFILE **")
      IF (IERROR.EQ.4) WRITE(LO,40)
      IF (IERROR.EQ.4) WRITE(CO,40)
40    FORMAT(" FLXWAS> ***  MISSING DATA RECORD ON DATAFILE ***")
      IF (IERROR.EQ.5) WRITE(LO,50)
      IF (IERROR.EQ.5) WRITE(CO,50)
50    FORMAT(" FLXWAS> ***  TOP WALL JACK POSITION ERROR ***")
      IF (IERROR.EQ.6) WRITE(LO,60)
      IF (IERROR.EQ.6) WRITE(CO,60)
60    FORMAT(" FLXWAS> ***  BOTTOM WALL JACK POSITION ERROR ***")
      IF (IERROR.EQ.999) WRITE(LO,70)
      IF (IERROR.EQ.999) WRITE(CO,70)
70    FORMAT(" FLXWAS> ***  DATAFILES NEED TO BE INITIALISED ***")
      RETURN
      END
```

APPENDIX B

LISTING OF INCLUDED SOURCE FILES - FLXCOM, FLXTYP, FWPTYP, FWPEQU, WASCOM and OAPCM

APPENDIX B.1 - LISTING OF FLXCOM (Flexwall Control System Common Definitions)

```
C***********************************************************************
C*                                                   *    FLXCOM       *
C*                                                   *    02/24/86     *
C*    FLEXWALL COMMON DEFINITIONS                    *                 *
C*                                                   *                 *
C*                                                   *                 *
C***********************************************************************
C
      COMMON  /FLXCOM/
C
C=======================================================================
C                       FLEXWALL TASK CONTROL SECTION
C=======================================================================
C
     *  FLXISW,
C                      FLEXWALL INTERRUPT STATUS WORD.  A BIT SET (=1)
C                      INDICATES AN INTERRUPT IS ACTIVE.  A BIT
C                      RESET (=0) INDICATES AN INTERRUPT IS INACTIVE.
C                      EACH BIT REPRESENTS AN INTERRUPT AS FOLLOWS:
C                        BIT  1    SYSTEM ERROR
C                             2    MICRO FAILURE     (FIFAIL)
C                             3    HARD LIMIT DETECTED BY MICRO  (FIHARD)
C                             4    TERMINATE WALL MOVE  (FSTAT,FLXOP)
C                             5    INITIATE WALL MOVE   (FLXOP)
C                             6    RESUME WALL MOVE     (FLXOP)
C                             7    REPORT MOVE STATUS  (FLXWAS,DASFLX)
C                             8    SOFT LIMIT DETECTED BY MICRO  (FISOFT)
C                             9    INITIATE LIMIT RECOVERY   (FLXOP)
C                            10    INITIALIZE WALL CONFIGURATION (FLXOP)
C                            11    CONFIGURE WALLS         (DASFLX)
C                         12-16    RESERVED
     *  FLXICW,
C                      FLEXWALL INTERRUPT CONDITION WORD.  EACH BIT
C                      INDICATES THE ENABLED/DISABLED CONDITION OF THE
C                      OF THE INTERRUPT CORRESPONDING TO BITS OF
C                      THE FLEXWALL INTERRUPT STATUS WORD [FLXISW].
C                          0 = DISABLED
C                          1 = ENABLED
     *  FSSTAT,
C                      FLEXWALL SYSTEM STATUS WORD.
C                        BIT       MEANING 0/1
C                          1       COMPUTER/MANUAL CONTROL MODE (FSTAT)
C                          2       NORMAL/OVERRIDE KEY-SWITCH POSITION
C                                   (FSTAT)
C                          3       NOHARD/HARD LIMIT EXISTS  (FSTAT)
C                          4       NOSOFT/SOFT LIMIT EXISTS  (FSTAT)
C                          5       JACK MOTORS DISABLED/ENABLED  (FSTAT)
C                          6       SYSTEM NOHOLD/HOLD BY OPERATOR
C                          7       WALL MOVE DISABLED/ENABLED
C                                   (FLXCTL,FMVSTR)
C                       8-16       RESERVED
     *  FLXXSW,
C                      RESERVED--NOT USED--OBSOLETE
     *  FLXOPT,
C                      FLEXWALL TASK OPTION WORD.
C                        BIT    MEANING 0/1
C                         1     NODUMP/DUMP MOVE PARAMETERS
C                         2     DISABLE/ENABLE LIMIT CHECKS
```

```
C                         3     AUTOMATIC/OPERATOR WALL MOVE
C                         4     CONTINUOUS/SINGLE ITERATION MODE
C                         5     CONTINUOUS MODE --
C                                 NOWAIT/WAIT FOR TEST CONDITIONS UPDATE
C                         6     CONTINUOUS MODE --
C                                 NOPAUSE/PAUSE 'WAS' ANALYSIS
C                      7-16     RESERVED
     *  PLTMODE,
C                      FLEXWALL PLOT TASK OPERATIOAL MODE CONTROL WORD
C                         -1=   STOP
C                          0=   REPLOT WHEN POINT NO. CHANGES
C                        N>0=   REPLOT AFTER N SECONDS
     *  PLTTYPE(2),
C                      PLOT TASK PLOT TYPE FOR GRAPHS 1 AND 2
C                          0=   NO GRAPH
C                          1=   WALL POSITIONS
C                          2=   WALL PRESSURES
C                          3=   WALL MACH NUMBERS
C                          4=   WALL POSITIONS (BOTTOM WALL SIGN REV.)
C                          5=   DELTA MACH NUMBER
     *  PLTPOS(2),
C                       PLOT TASK GRAPH POSITION FOR GRAPHS 1 AND 2
C                          1=   FULL SCREEN
C                          2=   TOP HALF SCREEN
C                          3=   BOTTOM HALF SCREEN
     *  PLTYMAX(2),
C                       PLOT TASK MAX. Y VALUE PLOTTED FOR GRAPHS 1 AND 2
     *  PLTYMIN(2),
C                       PLOT TASK MIN. Y VALUE PLOTTED FOR GRAPHS 1 AND 2
     *  PLTTEXT,
C                       PLOT TASK TEXT CONTROL WORD
C                          (SAME DEFINITION AS PLTTYPE)
     *  CTL001(47),
C                      RESERVED LOCATIONS.
     *  MVSTAT,
C                      MOVE STATUS WORD  (FMVSTR):
C                         0 = MOVE INITIATED OR IN PROGRESS
C                         1 = MOVE SUCCESSFULLY COMPLETED
C                         2 = MOVE FAILURE -- SYSTEM IN MANUAL MODE
C                         3 = MOVE FAILURE -- JACK MOTORS DISABLED
C                         4 = MOVE FAILURE -- TARGET HARD LIMITS
C                         5 = MOVE FAILURE -- CURRENT HARD LIMITS
C                         6 = MOVE FAILURE -- JACK MOVE ERROR(S)
C                         7 = MOVE TERMINATED
     *  STRSTAT,
C                      STREAMLINE STATUS WORD  (FLXWAS):
C                        -1 = ERROR--WALL STREAMLINING ABORTED
C                         0 = WALLS ARE NOT STREAMLINED
C                         1 = WALLS ARE STREAMLINED
     *  MCSTAT,
C                      MICROPROCESSOR STATUS:
C                          THIS VALUE IS READ FROM IOIS SLOT 7 (#6).
C                         FOR BITS 1-8, THE ACTUAL VALUE RECEIVED IS
C                         THE ONE'S COMPLEMENT OF THE VALUES SHOWN
C                         BELOW.  ALSO, X = 1 INDICATES THE UPPER WALL,
C                                       X = 2 INDICATES THE LOWER WALL
C
C
```

```
C                     BIT(S)   VALUE     MEANING
C                      1-8      00       NORMAL
C                              01-42     JACK OUT OF LIMITS
C                               5X       +FULL-SCALE CALIBRATION ERROR
C                               6X       ZERO-SCALE CALIBRATION ERROR
C                               7X       -FULL-SCALE CALIBRATION ERROR
C                               8X       TIMEOUT (A-TO-D ERROR)
C                        9       1       ABSOLUTE HIGH LIMIT
C                       10       1       ABSOLUTE LOW  LIMIT
C                       11       1       RELATIVE HIGH LIMIT
C                       12       1       RELATIVE LOW  LIMIT
C                       13       1       HARD LIMIT
C                       14       1       SOFT LIMIT
C                       15       0       COMPUTER CONTROL MODE
C                                1       MANUAL CONTROL MODE
C                       16       0       NORMAL OPERATION MODE
C                                1       OVERRIDE OPERATION MODE
     *  AQSTAT,
C                     ACQUISITION STATUS:
C                         0 = OPERATIONAL
     *  CALSTAT,
C                     CALIBRATION STATUS:
C                        -1 = NOT PERFORMED
C                         0 = FAILURE
C                         1 = SUCCESS
     *  AQBUSY,
C                     ACQUISITION TASK BUSY STATUS:
C                       0   = NOT BUSY
C                       1-N = REQUEST IN PROGRESS
     *  AQRQST,
C                     DATA ACQUISITION REQUEST CODE.  A BIT
C                     SET (=1) INDICATES THE TYPE OF DATA
C                     REQUESTED.  ONCE THE DATA IS OBTAINED AND
C                     PLACED IN THIS COMMON REGION (FLXCOM),
C                     THE RESPECTIVE BIT IS CLEARED BY THE
C                     ACQUISITION TASK.
C                     BIT 1  =  CURRENT JACK POSITIONS
C                         2  =  FLEXWALL PRESSURES
C                         3  =  RUN (TUNNEL) DATA FROM DAS (CPU B)
C                         4  =  AVERAGE CURRENT JACK POSITIONS
C                      5-16  =  RESERVED
     *  AQUSER(2),
C                     SIX-CHARACTER CAN-CODE NAME OF THE TASK
C                     CURRENTLY REQUESTING DATA.
     *  LMBUSY,
C                     LIMIT MONITOR TASK BUSY STATUS:
C                       0   = NOT BUSY
C                       1-N = REQUEST IN PROGRESS
     *  LMRQST,
C                     LIMIT REQUEST CODE.  THE VALUE OF BITS 2-16 OF
C                     THIS CODE INDICATES THE LIMITS BEING REQUESTED:
C                         0 = NONE
C                         1 = CURRENT JACK POSITION LIMITS
C                         2 = DESTINATION JACK POSITION LIMITS
C                      IF BIT 1 IS SET (=1), THEN A LIMIT REPORT WILL
C                      BE WRITTEN TO THE LINE PRINTER.
     *  LMUSER(2),
C                     SIX-CHARACTER CAN-CODE NAME OF THE TASK
```

```
C                      CURRENTLY REQUESTING LIMITS.
     *  LIMSTAT,
C                      LIMIT STATUS WORD. A BIT SET (=1) INDICATES
C                      THE SPECIFIED CONDITION IS TRUE.  A BIT IS
C                      RESET (=0) WHEN THE CONDITION CLEARS.
C                      BIT  1 = HARD LIMIT AT CURRENT JACK POSITIONS
C                               DETECTED BY MICRO.
C                           2 = HARD LIMIT AT CURRENT JACK POSITIONS
C                               DETECTED BY 'FLXLIM'.
C                           3 = SOFT LIMIT AT CURRENT JACK POSITIONS
C                               DETECTED BY MICRO.
C                           4 = SOFT LIMIT AT CURRENT JACK POSITIONS
C                               DETECTED BY 'FLXLIM'.
C                         5-8 = RESERVED
C                           9 = HARD LIMIT AT DESTINATION JACK POSITIONS
C                               DETECTED BY 'FLXLIM'.
C                          10 = SOFT LIMIT AT DESTINATION JACK POSITIONS
C                               DETECTED BY 'FLXLIM'.
C                       11-16 = RESERVED.
     *  JSTAT(23,2),
C                      CURRENT JACK POSITION STATUS TABLE.  EACH
C                      WORD OF THIS TABLE CORRESPONDS TO A WALL
C                      JACK STATION OR WALL ENDPOINT AS FOLLOWS:
C                      JSTAT( 1,1)              = FIXED END (UPPER)
C                      JSTAT( 2,1)-JSTAT(22,1)  = JACKS 1-21
C                      JSTAT(23,1)              = FLOATING END (UPPER)
C                      JSTAT( 1,2)              = FIXED END (LOWER)
C                      JSTAT( 2,2)-JSTAT(22,2)  = JACKS 22-42
C                      JSTAT(23,2)              = FLOATING END (LOWER)
C
C                      BITS IN A WORD ARE SET (=1) TO INDICATE THE
C                      FOLLOWING CONDITION(S):
C                      BIT  1 = HARD RELATIVE UPPER LIMIT
C                           2 = HARD ABSOLUTE UPPER LIMIT
C                           3 = SOFT RELATIVE UPPER LIMIT
C                           4 = SOFT ABSOLUTE UPPER LIMIT
C                         5-8 = RESERVED
C                           9 = HARD RELATIVE LOWER LIMIT
C                          10 = HARD ABSOLUTE LOWER LIMIT
C                          11 = SOFT RELATIVE LOWER LIMIT
C                          12 = SOFT ABSOLUTE LOWER LIMIT
C                       13-14 = RESERVED
C                          15 = JACK DIRECTION ERROR
C                          16 = JACK MOVE ERROR
     *  JNSTAT(23,2),
C                      NEW JACK POSITION STATUS TABLE.  THIS TABLE
C                      IS USED THE SAME WAY AS [JSTAT], EXCEPT
C                      FOR DESTINATION JACK POSITIONS.
     *  JPOS(21,2),
C                      CURRENT JACK POSITIONS (INCHES).
     *  JPOSN(21,2),
C                      NEW (DESTINATION) JACK POSITIONS (INCHES).
     *  JPOSL(21,2),
C                      LAST (PREVIOUS) JACK POSITIONS (INCHES).
     *  CTL999(4)
C                      RESERVED LOCATIONS.
C
C=========================================================================
C                  C D A  COMMUNICATIONS COREDEVICE IMAGE
C========================================================================
C
     *  IASTAT,
C                     COMMUNICATIONS STATUS WORD.
     *  IACCW,
C                     COMMUNICATIONS CONTROL WORD.
     *  IAMODE,
C                     COMMUNICATIONS MODE.
     *  IARSV
C                     COMMUNICTION RESERVED WORD.
     *  JPOSA(21,2),
C                     AVERAGED CURRENT JACK POSITIONS (INCHES).
     *  WPRS(22,2),
C                     WALL PRESSURES (PSIA).
     *  IARSV1(80),
C                     RESERVED LOCATIONS.
C
C========================================================================
C                 C D B  COMMUNICATIONS COREDEVICE IMAGE
C========================================================================
C
     *  IBCCW,
C                     COMMUNICATIONS CONTROL WORD.
C                         #8000 - CONFIGURE WALLS
C                         #4000 - CLEAR STATUS WORD
C                         #2000 - TERMINATE DATA POINT
     *  IBTSTNR,
C                     TEST NUMBER (CPU B)
     *  IBRUNNR,
C                     RUN NUMBER (CPU B)
     *  IBPTNR,
C                     POINT NUMBER (CPU B)
     *  IBHOUR,
C                     HOUR DATA ACQUIRED.
     *  IBMIN,
C                     MINUTE DATA ACQUIRED.
     *  XBSEC,
C                     SECOND DATA ACQUIRED
     *  XBPT,
C                     TOTAL PRESSURE (PSIA)
     *  XBTT
C                     TOTAL TEMPERATURE (KELVIN)
     *  XBPS,
C                     STATIC PRESSURE (PSIA)
     *  XBAOA,
C                     ANGLE OF ATTACK (DEGREES)
     *  XBMACH,
C                     FREE STREAM MACH NUMBER
     *  XBHT,
C                     TOTAL ENTHALPY (JOULES/MOLE)
     *  XBENTPY,
C                     ENTROPY (JOULES/MOLE-KELVIN)
     *  XBTS,
C                     STATIC TEMPERATURE (KELVIN)
     *  XBRHOS,
C                     STATIC DENSITY (SLUGS/FT**3)
     *  XBPBR,
C                             BAROCELL REFERENCE PRESSURE.
     *   XREYNO,
C                             REYNOLDS NUMBER BASED ON CHORD
     *   XCHORD,
C                             MODEL CHORD (INCHES)
     *   IBRSV(192),
C                             RESERVED LOCATIONS.
     *   WASPAR(16)
C                             WALL ADJUSTMENT STRATEGY PARAMETERS
C                             (SEE "FWPEQU'' FOR EQUIVALENCES).
C
C
C        END OF COMMON DEFINITION
C
C
```

APPENDIX B.2 - LISTING OF FLXTYP (Flexwall Common Type Declarations)

PRECEDING PAGE BLANK NOT FILMED

PAGE 136 INTENTIONALLY BLANK

```
C***********************************************************************
C*                                                      *    FLXTYP    *
C*                                                      *    02/26/86  *
C*    FLEXWALL COMMON TVPE DECLARATIONS                 *              *
C*                                                      *              *
C*                                                      *              *
C***********************************************************************
C
      INTEGER*2 FLXISW,FLXICW,FSSTAT
      INTEGER*2 FLXXSW,FLXOPT
      INTEGER*2 PLTMODE,PLTTYPE,PLTPOS
      REAL*4    PLTYMAX,PLTYMIN
      INTEGER*2 PLTTEXT
      INTEGER*2 MVSTAT,STRSTAT,MCSTAT,AQSTAT,CALSTAT
      INTEGER*2 CTL001,CTL999
      INTEGER*2 AQBUSY,AQRQST,AQUSER
      INTEGER*2 LMBUSY,LMRQST,LMUSER
      INTEGER*2 LIMSTAT,JSTAT,JNSTAT
C
      REAL*4    JPOS,JPOSN,JPOSL,JPOSI
C
      INTEGER*2 IASTAT,IACCW,IAMODE,IARSV
      INTEGER*2 IARSV1
C
      REAL*4    JPOSA,WPRS
C
      INTEGER*2 IBCCW
      INTEGER*2 IBTSTNR,IBRUNNR,IBPTNR
      INTEGER*2 IBHOUR,IBMIN
      REAL*4    XBSEC
      INTEGER*2 IBRSV
C
      REAL*4    XBPT,XBTT,XBPS,XBAOA,XBMACH,XBHT,XBENTPY
      REAL*4    XBTS,XBRHOS,XBPBR,XREYNO,XCHORD
      REAL*4    WASPAR
```

APPENDIX B.3 - LISTING OF FWPTYP (Flexwall "WASPAR" Type Declarations)

```
C***************************************************************************
C                                                      *    FWPTYP      *
C                                                      *   02/26/86     *
C        FLEXWALL "WASPAR" TYPE DECLARATIONS           *                *
C                                                      *                *
C                                                      *                *
C***************************************************************************
C
C
C
      REAL*4     SPMACH,SPREYN,SPAOA
      REAL*4     XLORIG,YLORIG
C
C
C
C***************************************************************************
```

APPENDIX B.4 - LISTING OF FWPEQU (Flexwall "WASPAR" Equivalences)

```
C***********************************************************************
C*                                                     *    FWPEQU     *
C*                                                     *    02/26/86   *
C*      FLEXWALL "WASPAR" EQUIVALENCES                 *               *
C*                                                     *               *
C*                                                     *               *
C***********************************************************************
C
      EQUIVALENCE (WASPAR(1),SPMACH),
C                                      MACH NUMBER.
     1            (WASPAR(2),SPREYN),
C                                      CHORD REYNOLDS NUMBER IN MILLIONS.
     2            (WASPAR(3),SPAOA ),
C                                      ANGLE OF ATTACK.
     3            (WASPAR(4),XLORIG),
C                                      MODEL 1/4 CHORD POSITION UPSTREAM
C                                      OF THE TURNTABLE CENTER.
     4            (WASPAR(5),YLORIG)
C                                      MODEL 1/4 CHORD POSITION ABOVE THE
C                                      TURNTABLE CENTER.
C
C***********************************************************************
```

APPENDIX B.5 - LISTING OF WASCOM (WAS Common Definitions)

```
C*********************************************************************
C*                                                    *   05/04/87   *
C*                                                    *              *
C*      WALL ADJUSTMENT STRATEGY COMMON DEFINITION    *    WASCOM    *
C*                                                    *              *
C*                                                    *              *
C*********************************************************************
C
C
      COMMON /WASCOM/
C
C
C     NO. OF FLEXWALL JACKS
     +    NOJACK,
C
C     NO. OF COMPUTING POINTS REQUIRED,USUALLY NO. OF S/LINING JACKS + 4
     +    NOCPT,
C
C     STREAMWISE LOCATION OF THE WALL COMPUTING POINTS
C     ON EACH FLEXWALL RELATIVE TO THE FLEXWALL
C     ANCHOR POINT (INCHES)
     +    XJACK(23),
C
C     WALL SEGMENT LENGTH PER JACK (INCHES)
     +    WL(21),
C
C     TOP WALL JACKS Y CO-ORDINATES (INCHES)
     +    TOPY(21),
C
C     BOTTOM WALL JACKS Y CO-ORDINATES (INCHES)
     +    BOTY(21),
C
C     TOP WALL IMAGINARY (EXTERNAL) VELOCITIES (V/U0)
     +    TWVEL(22),
C
C     BOTTOM WALL IMAGINARY (EXTERNAL) VELOCITIES (V/U0)
     +    BWVEL(22),
C
C     TOP WALL CENTERLINE STATIC PRESSURES (PSIA)
     +    TOPWP(22),
C
C     BOTTOM WALL CENTERLINE STATIC PRESSURES (PSIA)
     +    BOTWP(22),
C
C     D* THICKNESS ALONG THE TOP WALL (INCHES)
     +    TWDS(18),
C
C     D* THICKNESS ALONG THE BOTTOM WALL (INCHES)
     +    BWDS(18),
C
C     WAS COMPUTED NEW TOP WALL EXTERNAL VELOCITY DISTRIBUTION (V/U0)
     +    TWNVEL(22),
C
C     WAS COMPUTED NEW BOTTOM WALL EXTERNAL VELOCITY DISTRIBUTION (V/U0)
     +    BWNVEL(22),
C
C     WAS COMPUTED TOP WALL MOVEMENT DEMANDS (INCHES)
     +    TWMOV(21),
C
C     WAS COMPUTED BOTTOM WALL MOVEMENT DEMANDS (INCHES)
     +    BWMOV(21),
C
C     TOP WALL MEASURED VELOCITIES SQUARED
     +    TWVSQ(22),
C
C     BOTTOM WALL MEASURED VELOCITIES SQUARED
     +    BWVSQ(22),
C
C     VELOCITY DIFFERENCES ACROSS THE TOP WALL
     +    TWVDIF(22),
C
C     VELOCITY DIFFERENCES ACROSS THE BOTTOM WALL
     +    BWVDIF(22),
C
C     MACH NUMBER DISTRIBUTION ALONG THE TOP WALL
     +    TWMACH(21),
C
C     MACH NUMBER DISTRIBUTION ALONG THE BOTTOM WALL
     +    BWMACH(21),
C
C*****************************************************************
C                      TEST PARAMETER COMMON
C*****************************************************************
C
C     AIRFOIL AOA (DEGREE)
     +    ALPHA,
C
C     AIRFOIL CHORD (INCHES)
     +    CHORD,
C
C     STREAMWISE LOCATION OF MODEL 1/4 CHORD UPSTREAM OF MODEL
C     PIVOT (TURNTABLE CENTER) IN INCHES
     +    XORIG,
C
C     LOCATION OF MODEL 1/4 CHORD ABOVE MODEL PIVOT
C     (TURNTABLE CENTER) IN INCHES
     +    YORIG,
C
C     FREE STREAM MACH NUMBER
     +    FMACH,
C
C     CHORD REYNOLDS NUMBER
     +    CREY,
C
C     REFERENCE TOTAL PRESSURE (PSIA)
     +    PTOTAL,
C
C     REFERENCE STATIC PRESSURE (PSIA)
     +    PSTATIC,
C
C     REFERENCE TOTAL TEMPERATURE (KELVIN)
     +    TTOTAL,
C
C     WORKING FLUID DENSITY (SLUG/FT**3)
     +    DENSITY,
C
```

```
C*****************************************************************
C                 WALL ADJUSTMENT STRATEGY FACTORS
C*****************************************************************
C
C     TOP WALL MOVEMENT COUPLING FACTOR
     +    TWCPLF,
C
C     BOTTOM WALL MOVEMENT COUPLING FACTOR
     +    BWCPLF,
C
C     TOP WALL MOVEMENT SCALING FACTOR
     +    TWSF,
C
C     BOTTOM WALL MOVEMENT SCALING FACTOR
     +    BWSF,
C
C     EMPTY TEST SECTION STREAMLINING MOVEMENT FACTOR
     +    WMOVF,
C
C*****************************************************************
C                 FLEXWALL AERO-STRAIGHT COMMON
C*****************************************************************
C
C     TOP WALL JACK Y CO-ORDINATES RELATIVE TO AERO. STRAIGHT (INCHES)
     +    TWYAS(21),
C
C     BOTTOM WALL JACK Y CO-ORDS RELATIVE TO AER0. STRAIGHT (INCHES)
     +    BWYAS(21),
C
C     TOP WALL D* THICKNESS WITH AERO. STRAIGHT CONTOURS (INCHES)
     +    DSTAS(18),
C
C     BOTTOM WALL D* THICKNESS WITH AERO. STRAIGHT CONTOURS (INCHES)
     +    DSBAS(18),
C*****************************************************************
C
C     PRANDTL-GLAUERT FACTOR
     +    BETA,
C
C     Q COMPRESSIBILITY CORRECTION FACTOR
     +    QCOR,
C
C     PASS NO.
     +    PASS,
C
C     STREAMLINING CYCLE ITERATION NO.
     +    ITERATION,
C
C     ANALYSIS SPECIFIER
     +    IANAL,
C
C     SPARE ARRAY
     +    FLXSPR(21,2)
C
C
C     END OF COMMON DEFINITION
C
C
```

APPENDIX B.6 - LISTING OF OAPCM (System Common Definitions)

```
C***********************************************************************
C*                   O A P CONFIGURATION CONTROL
C*
C*               FOR THE TASK OAP OF THE CRYOA SYSTEM
C*
C***********************************************************************
C*
C*   FUNCTIONAL DESCRIPTION:
C*
C*         THIS IS THE FORTRAN LANGUAGE VERSION OF OAP COMMON.  REFER TO
C*    TECHNICAL DOCUMENTATION FOR A DETAILED BREAKDOWN OF THE DATA
C*    STRUCTURES WITHIN THIS COMMON.(THE CORRESPONDING ASSEMBLY VERSION
C*    IS CALLED AOAPR).
C*
C***********************************************************************
C*
C*   CALLING SEQUENCE
C*
C*          OAP CREATES OAPCOM COMMON AS A PRIVATE SHARED
C*       REGION, AND PROVIDES INITIAL VALUES AND SETS
C*       FLAGS TO INITIAL CONDITIONS.
C*
C***********************************************************************
C*
C*   REVISION  HISTORY:
C* MOD. NO.   DATE     BY                DESCRIPTION
C                            RESPONSIBLE PROGRAMMER: R.L.KRIEGER
C                                                    R.T.ROTH
C                             LATEST REVISION LEVEL: 0.0
C* --------  08-12-85  CPK   ADDED STANDARD OAP TEMPLATE TO COMMON
C*
C***********************************************************************
C            WARNING !!!!  VARIABLE ISCAN CAN BE MISTAKEN FOR ISCAN THE
C                          LIBRARY FUNCTION.
C
      INTEGER*2 SBUFF,SRATE,IWRTSN,IANSCN,PRPONT,DAUFLG,STATFG,IBYRD,
     * IBYWT,LIMPER,ALOBTS,GRP,POS,CRTCTL,IDGSCN,IBDAVG,FORMAT,PERCAL,
     * CLLIM,AVCKWK,PONTNO,CHANCT,IPSWD,IMODE,LMPBTS,NAMES,
     * VLVBTS,MXPT,DELY,SCNFLG,NOFREC,NOOFSV,NOPRSS,REQPRS,SPRCOM,TD1,
     * TD2,TD3,PAVG,PFDLY,NOSETS,PSDLY,PADLY,SNSOR,TOLPER,ESPFLG,ATMPER,
     * RECLEN,RECODE,TESTNO,BATCH,RUN,SCAN,FRAME,ID,DATE,TKINMN,TSYSGN,
     * MNDAY,YEAR,NOANL,NODG,NOSCV,HIPORT,IBDFRM,IABORT,PCALTP,SCANRT,
     * RECTYP,ESPLEN,NUMDM,SPARE,BADFLG,PORTNO,SYSID,IANRD1,
     * IANRD2,ISCAN,ACHAN,DCHAN,FULLCM,INTBTS,CRTTBL,LINE25,LIMTBL,
     * PNAME,PUNIT,GDT,CURPRT,PVCKWK,PNLINT,IPRSS1,IPRSS2
C
      INTEGER*4 AVGBUF,ESPAVG,SCNBUF
C
      REAL*6 DC
C
C
C     SOME COMMONLY USED POINTERS AND STATUS INDICATORS
C
C
      COMMON/OAPCOM/ SBUFF,SRATE,IWRTSN,IANSCN(2),PRPONT(2),DAUFLG,
     * STATFG,IBYRD,IBYWT,LIMPER,ALOBTS,GRP,POS,CRTCTL,IDGSCN(2),IBDAVG,
     * FORMAT,PERCAL,CLLIM,AVCKWK,PONTNO,CHANCT,IPSWD(2),IMODE,LMPBTS,
C
```

```
C      THE NAME TABLE FOR THE DIGITAL CONSTANTS PANEL
C
     * NAMES(7,18),
C
C      SCANNIVALVE TABLE
C
     * VLVBTS,MXPT,DELY,SCNFLG,CURPRT,PVCKWK,NOFREC,NOOFSV,
C
C
C      ESP PRESSURE SYSTEM TABLE
C
     * NOPRSS,REQPRS,SPRCOM,TD1,TD2,TD3,PAVG,PFDLY,NOSETS,PSDLY,
     * PADLY,SNSOR,TOLPER,ESPFLG,ATMPER,DC(12),
C
C      TAPE HEADER RECORD
C
     * RECLEN,RECODE,TESTNO,BATCH,RUN,SCAN,FRAME,ID,DATE,TKINMN,TSYSGN,
     * MNDAY,YEAR,NOANL,NODG,NOSCV,HIPORT,IBDFRM,IABORT,PCALTP,
     * SCANRT,RECTYP,ESPLEN,NUMDM,SPARE(3),BADFLG,PORTNO,SYSID,
C
C      READ SCAN TABLES
C
     * IANRD1(196),IPRSS1(512),
     * IANRD2(196),IPRSS2(512),
C
C      WRITE SCAN TABLES
C
     * ISCAN(216),
C
C      ANALOG CHANNEL ATTRIBUTE TABLE
C              ACHAN(1,I) = ADDRESS/ZERO IF NOT SCANNED
C              ACHAN(2,I) = ATTRIBUTE
C              ACHAN(3,I) = DATAMETRICS ADDRESS
C
     * ACHAN(3,128),
C
C      DIGITAL CHANNEL ATTRIBUTE TABLE
C               DCHAN(1,I) = ADDRESS - ZERO IF NOT SCANNED
C               DCHAN(2,1) = ATTRIBUTE WORD
C
     * DCHAN(2,16),
C
C
C      PANEL INTERRUPT FLAG
C
     *PNLINT,
C
C      SPARE COMMON
C
     * FULLCM(84),
C
C      INHIBIT BITS FOR MODACS COMMON ALARM INTERRUPTS
C
     * INTBTS,
C
C
C      CRT DISPLAY TABLE
C      CRTTBL(1,I) = DATA ADDRESS/CHANNEL NUMBER
```

```
C      CRTTBL(2,I) = CRT ATTRIBUTE WORD
C      CRTTBL(3,I) = DATA ATTRIBUTE WORD
C      CRTTBL(4,I)-(7,I) = CAN CODE PARAMETER NAME
C      CRTTBL(8,I)-(10,I) = CAN CODE UNITS NAME
C
     *CRTTBL(10,24),
C
C      REST OF DISPLAY TABLE FOR DISPLAYS
C      LINE25(1) = DATA ADDRESS/CHANNEL NUMBER
C      LINE25(2) = BLINK STATUS
C      LINE25(3) = FORMAT CODE/CHANGE FLAG
C      LINE25(4)-(6) = CAN CODE NAME
C      LINE25(7)-(10) = CAN CODE UNITS
C
C      5 OTHER DISPLAY UNITS
C
     *LINE25(60),
C
C      LIMIT CHECK TABLE
C      LIMTBL(1) = DATA ADDRESS OR CHANNEL NUMBER
C      LIMTBL(2) = ATTRIBUTE WORD
C      LIMTBL(3)-(4) = UPPER LIMIT
C      LIMTBL(5)-(6) = LOWER LIMIT
C
     *LIMTBL(6,30),
C
C      REST OF TABLE IS FOR NAMED PARAMETER LIMITS
C      PNAM(1,I) - (4,I) = CAN CODE NAME FOR PARRAMETER
C
     *PNAME(4,30),
C
C      TABLE FOR LIMIT UNITS
C
C      PUNIT(1,I) - (3,I) = UNIT ASSIGNED TO LIMIT
C
     *PUNIT(3,30),
C
C      GAIN DATA TABLE
C
     *GDT(36),
C
C      AVERAGE RECORD BUFFER
C
     * AVGBUF(144),
C
C      ESP AVERAGE DATA BUFFER
C
     * ESPAVG(512),
C
C      SCANNIVALVE AVERAGE DATA BUFFER
C
     *  SCNBUF(480)
C
C
C      END OF COMMON DEFINITION
C
C
```

APPENDIX C

ABORT PROCEDURES

Error Conditions and Causes

1) Software Error - Wall adjustment strategy has failed to converge in six iterations.
- Datafile(s) not initialized correctly.

2) Hardware Error - Jack movement and/or direction failure.

3) Jack Limit Reached - Wall curvature hard limit in current or target wall shapes.

4) CPU-B operator has aborted data point.
(e.g. messages "DASFLX> "TERMINATE MOVE" RECEIVED FROM CPU-B
!ABORT (EOJ FLX 7553*) /FLXWAS /FLXWAS" appears)

5) Link Abort - CPU-B System RTP has gone down.
(e.g. message "ABORT (LAF LKT 9C8C*) /FLXACQ /FLXACQ" appears)

6) Scanivalve Sync Error - Floor and/or ceiling scanivalve problem.
(e.g. message "ABORT (FER AL2 7EC6) /FLXWAS /FLXWAS" appears)

7) FMVSTR> Move Terminated - Jack movement and/or direction failure.

8) MT3 Offline - Failure to read or write to the tape drive.

9) Parity Error - Loss of memory integrity or disc memory partition.
(e.g. message "[Error Location] OFFLINE PAR" appears)

Remedial Actions

[Note - **Auto Mode** is used for wall streamlining, **Manual Mode** is used for flexwall exercising]

1. **Software Error** (Auto Mode only):

a) Look at the lineprinter and/or console output for the current data point.

b) Locate the error message(s) and interpret as follows:

i) "SOFTWARE ERROR" message only - Check that six iterations have been performed, if not then review output. Observe the variation of the wall Cp errors for each iteration. If they are reducing then re-start or continue the streamlining cycle (see sub-section 6.3). If these values are not reducing then check test condition stability and/or selection of initial wall contours, before continuing.

ii) "OUT> REDUCED DATA DISKFILE IS FULL !!" message appears with "FLXWAS> DATAFILES NEED TO BE INITIALISED" message - Make a tape backup of the reduced data datafile then initialise the same datafile and modify the datafile to include at least the data record for the last set of contours run in the test section, using procedures in Appendix E.

(Continued on next page)

1. **Software Error** (Continued):

iii) "INITWAL> STRAIGHT WALL DATA ERROR" message with one of the following messages:

"FLXWAS> FAILURE TO BRACKET MACH NO. ON DATAFILE" - Check Mach number in setup parameters table is within *Reference Table* limits.

"FLXWAS> FAILURE TO BRACKET REYNOLDS NO. ON DATAFILE" - Check Reynolds number in setup parameters table is within *Reference Table* limits.

"FLXWAS> TEST PARAMETERS NOT BRACKETED ON DATAFILE" - Check integrity of *Reference Table* directory using procedures in Appendix D.

"FLXWAS> MISSING RECORD ON DATAFILE" - Check integrity of *Reference Table* using viewing procedures in Appendix D.

iv) "WALDAT> STRAIGHT WALL DATA ERROR" message with same associated messages as for situation iii) above - Carry out the same actions except examine the test Mach number and Reynolds number in the raw data block on the lineprinter output. If all is well, check integrity of the *Reference Table* using the viewing procedures in Appendix D

v) "WALDAT> RAW DATA ERROR" message followed by "FLXWAS> DATAFILES NEED TO BE INITIALISED" message - Check the integrity of the raw data datafile using viewing procedures in Appendix E. If there is no directory, initialize the datafile. If 126 records have been used, make a tape backup of the datafile then initialize the datafile using procedures described in Appendix E.

2. **Hardware Error** (Auto Mode only):

a) View the jack status table on the CPU-A console and the lineprinter. The status code is as follows:

0 - Jack operational
1 - Jack movement failure after three attempts to move jammed jack
2 - Jack direction failure
3 - Combination of above failures
- Curvature limit reached (See documentation on task CRVLIM[7])

b) If there are **no** hard limit messages, try to move the walls again by selecting the IC option from the FLXOP Menu[7], repeat up to <u>three</u> times only.

c) If b) fails, attempt to return to the wall contours from the previous data point. First match the set-up parameters table with the test conditions for the previous data point, using the SP option from the FLXOP Menu. Then use the IC option to move the flexwalls. Note it may be useful to display wall movement information during this recovery operation using the DUMP operating option. The operator selects this option using the SO option from the FLXOP Menu.

d) If a jack movement failure occurs again then try manual control of the wall shapes. Change to manual mode by using the SO option from the FLXOP Menu, then selecting 'MO' for manual operation from Select Options Menu.

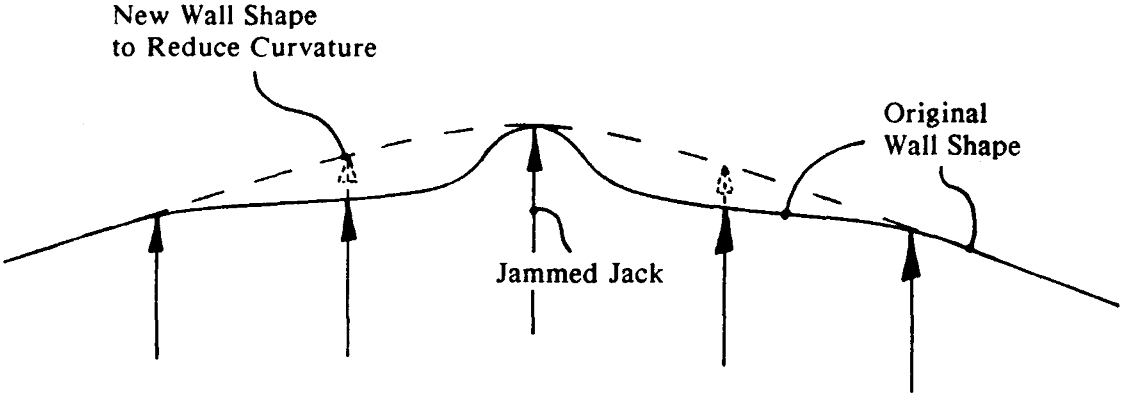
(Continued on next page)

e) Attempt to shake the jack free using the following procedure:

i) Select 'SM' from FLXOP Menu.

ii) Select 'RM' from Select Move Menu.

iii) Instruct jacks adjacent to the jammed jack to move to relieve the local wall curvature by viewing the current jack positions, as shown below. (Limit the movement per attempt to 0.03 inch to try to avoid further complications, i.e. more jacks jammed.)



iv) Move jacks using 'IM' option from FLXOP Menu.

v) Repeat steps i) to iv) to move the jammed jack to minimize wall curvature, i.e. move the jack towards its neighbour jack positions.

vi) Repeat steps i) to v) if the jack remains jammed.

vii) Abort testing and inspect the jack mechanisms.

f) If problems are encountered again whilst moving to the next required wall shapes consider the following questions:

i) Is the attempted wall move from a curved shape to flat?

ii) Are the target contours beyond the capabilities of the test section?

iii) Is there a problem with the jack mechanisms?

iv) Have the LVDT signals drifted?

g) If the answer to either question i) or ii) above is YES then use short Sine wave wall shapes to recover. First select the SM option from FLXOP Menu then 'SI' from the Select Move Meun. Then select a peak size initially roughly equal to the current maximum wall deflection near the model (jacks #9 and #10) and select the appropriate wall as prompted. Initiate wall movement using the IM option from the FLXOP Menu. If successful, reduce the peak size for subsequent moves in small increments until the wall is moving freely. If this procedure is unsuccessful, abort testing and inspect the jack mechanisms.

3. Jack Limit Reached

3.1 Auto Mode :

a) Select 'SO' from the FLXOP Menu.

b) Type in 'MO' then press [RETURN] key twice.

c) Activate the limit recovery task by selecting 'LR' from the FLXOP Menu.

d) If step c) is successful (ie message " FMVLR> MOVE COMPLETE will appear on the CPU-A console), select 'SO' from the FLXOP Menu, then type in 'MA' and press [RETURN] key twice. Exit procedure.

e) Find out where the limit has occurred by using the SR option from the FLXOP Menu and select 'CL' for a current location report, then look at the lineprinter output.

f) If the jack limit has occurred where a wall deflection of about 1 inch is indicated, abort testing and check that the corresponding position transducer (LVDT) has not gone open circuit.

g) If not, disable limits using the blind DL option from the Select Options Menu.

h) Reset limit status with CNTL A /FLXLIM/MOD #4F 0 then CNTL A //R.

i) Try the procedure for recovery from a Hardware Error starting at step (d).

j) Enable limits using the blind EL option from the Select Options Menu.

3.2 Manual Mode :

a) Determine where the limit has occurred by using the SR option from the FLXOP Menu and select 'CL' for a current location report, then look at the lineprinter output.

b) If the jack limit has occurred where the wall deflection is about 1 inch, check that the corresponding position transducer (LVDT) on the test section is not open circuit.

c) Disable limits using the blind DL option from the Select Options Menu.

d) Reset limit status with CNTL A /FLXLIM/MOD #4F 0 then CNTL A //R.

e) Try the procedure for recovery from a Hardware Error starting at step (e).

f) Enable limits using the blind EL option from the Select Options Menu.

4. **Terminate Move** (Auto Mode only):

a) Check status of test schedule with CPU-B operator.

b) Prepare for the next data point (See sub-section 6.2) or perform the end of the days testing procedures (See 0.3-m TCT Adaptive Wall Test Section Operator's Manual).

5. **Link Abort** (Auto Mode only):

a) Wait until the CPU-B RTP software is running again.

b) Re-established the flexwall control system (FLXRUN) software using Floor and Ceiling Datametrix calibrations recalled from disc. (See 0.3-m TCT Adaptive Wall Test Section Operator's Manual for more details.)

c) Re-set XLORIG and YLORIG using the SP option from the FLXOP Menu.

d) Prepare for the next data point (see sub-section 6.2).

6. **Scanivalve Sync Error** (Auto Mode only):

a) Ask CPU-B operator to abort the data point.

b) Check the operation of the CPU-A scanivalves in an 'Off-line' mode and ensure correct operation before continuing the data point.

c) Prepare for the next data point (see sub-section 6.2).

7. **FMVSTR> Move Terminated** (Manual Mode only):

a) Try remedial action for a Hardware Error starting at step (c).

b) If the jammed jack refuses to free up then inspect jack mechanisms.

8. **MT3 Offline** :

a) Check correct tape is loaded on MT3.

b) Ensure clean tape is not file protected.

c) Check MT3 select switch in cabinet below MT3 is in the CPU-A position.

9. Parity Error :

a) Write down the message from the CPU-A console.

b) Determine if the error has occured within memory partitions ASW, BSA or BSB, i.e. message "ASW OFFLINE PAR" appears. If so, one of the WAS software datafiles is lost and remedial action must be taken before continuing testing. If not, proceed with caution whilst informing the computer manager of this occurrence.

c) Re-assign the WAS software datafile(s) to other free memory partitions by typing in CNTL A /FLXWAS/ASS [Logical filename] [New memory partition], CNTL A //R.

d) Initialize the lost datafile (see Appendix D or E) and continue testing. This procedure requires the use of utility software which used to accesses the lost datafile. The affected tasks must now be given new datafile assignments before execution. We do this by first establishing the task by typing in the following: CNTL A /[Taskname]/EST, i.e. /DFVIEW/EST for task DFVIEW. Then any logical file assignments to the lost memory partition are changed by typing in the following: CNTL A /[Taskname]/ASS [Logical filename] [New memory partition name], i.e. /DFVIEW/ASS 12 BSC to reassign logical file 12 to BSC. Ensure that the assignment changes are consistent from task to task. To execute these established tasks type in CNTL A /[Taskname]/E, i.e. /DFVIEW/E for task DFVIEW.

e) Change disc packs when possible, the disc cannot be trusted after this sort of failure has occurred.

APPENDIX D

REFERENCE TABLE DOCUMENTATION

Introduction

We reference all movements of the flexible floor and ceiling of the AWTS to "Aerodynamically Straight" (AS) contours. We define the AS contours as wall shapes which produce a constant streamwise Mach number distribution along each flexible wall, with the test section empty. The resulting wall contours diverge to make an allowance for the boundary layer growth on all four walls of the test section. The AS contours are dependent on test Mach number and to a lesser extent test Reynolds number. The contours may be found experimentally or by calculation.

The WAS software described in this report uses a *Reference Table* datafile to look-up the appropriate AS contours for the actual test conditions in the wind tunnel. This *Reference Table* datafile resides on memory partition ASW with data assignments as listed in the source file FMASSIGN in Appendix D.1.

A suite of utility software exists to support the use of the *Reference Table* datafile. All the associated procedures require only the Modcomp CPU-A to be operational and the location of the *Reference Table* datafile defaults to memory partition ASW.

Listing the *Reference Table* Datafile

i) Type in the following command CNTL A /SWFLIST/E.
ii) The *Reference Table* directory will appear on the lineprinter.
iii) Respond with "Y" for a complete listing of the datafile's 100 data records to appear on the lineprinter, or "N" to exit.
iv) Type in the command CNTL A //R.

Viewing *Reference Table* Data Records

i) Type in the command CNTL A /SWVIEW/E.
ii) The *Reference Table* directory will appear on the screen, respond with a record number to be viewed or zero to exit.
iii) After exiting the task, type in the command CNTL A //R.

(Continued on next page)

Initialization of the *Reference Table*

i) AS contours geometrically flat

Type in the following commands:

```
CNTL A /SWINIT/E
CNTL A /R
```

(After message "INITIALIZATION COMPLETE" appears on CPU-A console.)

```
CNTL A //R
```

ii) AS contours with linear divergence and tunnel centerline rotation

i) Type in the following commands: CNTL A /SWINI/E CNTL A /R.
ii) When message "INPUT TUNNEL CENTERLINE ROTATION (+VE DOWN) - DEG." appears, respond by normally typing in a zero on the CPU-A console.
iii) When a message "INITIALIZATION COMPLETE" appears, type in CNTL A //R.

NOTE: If the amount of linear divergence for any AS contour needs to be changed then the source of task SWINI must be modified.

iii) AS contours found experimentally

i) Load the reduced data datafile (wall library) with AS wall contours found experimentally (See Appendix D).
ii) Type in the following command: CNTL A /SWMOD/E.
iii) Follow prompts and input the appropriate record number from the reduced data datafile and the associated record number in the *Reference Table* for each record transfer required.
iv) Exit program by responding with a zero.
v) Then type in the command CNTL A //R.

NOTE: Only the records in the *Reference Table* involved in a data transfer are modified by this procedure.

Generating a Tape Backup of the *Reference Table* Datafile

i) Type in the command CNTL A /SWDUMP/E.
ii) Load a clean tape on MT3 and respond with CNTL A /R.
iii) When message "DATA TRANSFER COMPLETE" appears, remove tape from MT3 and label "CPU-A REFERENCE TABLE" with the current date.
iv) Type in the command CNTL A //R.

Loading the *Reference Table* Datafile from a Tape Backup

i) Type in the command CNTL A /SWLOAD/E.
ii) Load required tape labelled "CPU-A REFERENCE TABLE" on MT3 and respond with CNTL A /R.
iii) When message "DATA TRANSFER COMPLETE" appears, then remove tape from MT3.
iv) Type in the command CNTL A //R.

APPENDIX D.1 - LISTING OF FMASSIGN (*Reference Table* Datafile Assignments)

```
C**********************************************************************
C*                                                *     FMASSIGN      *
C*                                                *     10/23/86      *
C*        DISKFILE MANAGEMENT ASSIGNMENTS         *                   *
C*                                                *                   *
C*                                                *                   *
C**********************************************************************
C
C   REFERENCE TABLE 'STRAIGHT WALL' DATA FILE FORMAT :
C           =======================
C                RECORD 1
C                    WORDS  TYPE         NAME         USE
C                      1     I*2        NTMACH  # OF MACH NOS. IN TABLE
C                      2     I*2        NTREYNO # OF REY. NOS. IN TABLE
C                     3-22   R*4        TMACH(*) MACH NOS.
C                    23-42   R*4        TREYNO(*) REYNOLDS NOS.
C                    43-142  I*2        INDEX(*,*) POINTERS TO DISK RECS.
C
C                RECORD 2 TO 101 -
C                    WORDS  TYPE         NAME         USE
C                     1-2    R*4        XMACH   RECORD MACH NO.
C                     3-4    R*4        XREYNO  RECORD REY. NO.
C                     5-6    R*4        XMPARA  MODEL PARAMETER
C                     7-48   R*4        TWYAS(*) TOP WALL Y CO-ORDS.
C                    49-90   R*4        BWYAS(*) BOTTOM WALL Y CO-ORDS.
C                    91-134  R*4        TWVEL(*) TOP WALL VELOCITIES
C                   135-178  R*4        BWVEL(*) BOTTOM WALL VELOCITIES
C                   179-214  R*4        DSTAS(*) TOP WALL D* CONTOUR
C                   215-250  R*4        DSBAS(*) BOTTOM WALL D* CONTOUR
C
C
C**********************************************************************
      INTEGER*2 CO,IBUFF,JBUFF,NTMACH,NTREYNO,INDEX,ITIME,IDATE
      INTEGER*4 IREC
      REAL*4 XMACH,XREYNO,XMPARA
C
      DIMENSION IBUFF(256),JBUFF(256),TMACH(10),TREYNO(10),INDEX(10,10)
      DIMENSION ITIME(3),IDATE(3),DSTAS(18),DSBAS(18)
      DIMENSION TWYAS(21),BWYAS(21),TWVEL(22),BWVEL(22)
C
      EQUIVALENCE (IBUFF(1),NTMACH),            (JBUFF(1),XMACH)
      EQUIVALENCE (IBUFF(2),NTREYNO),           (JBUFF(3),XREYNO)
      EQUIVALENCE                               (JBUFF(5),XMPARA)
      EQUIVALENCE (IBUFF(3),TMACH(1)),          (JBUFF(7),TWYAS(1))
      EQUIVALENCE (IBUFF(23),TREYNO(1)),        (JBUFF(49),BWYAS(1))
      EQUIVALENCE (IBUFF(43),INDEX(1,1)),       (JBUFF(91),TWVEL(1))
      EQUIVALENCE                               (JBUFF(135),BWVEL(1))
      EQUIVALENCE                               (JBUFF(179),DSTAS(1))
      EQUIVALENCE                               (JBUFF(215),DSBAS(1))
C
      DEFINE FILE 10(101,256,U,IREC)
      DATA CO/@CO/,LO/@LO/
C
C
C**********************************************************************
```

APPENDIX E

WALL LIBRARY DOCUMENTATION

Viewing the Current Raw Data Datafile or the Current Reduced Data Datafile

i) Type in CNTL A /DFVIEW/E.
ii) If the raw data datafile is to be viewed respond with 'RAW', if the reduced data datafile is to be viewed respond with 'RED.'
iii) If specific data records are to be viewed type in the record numbers, one at a time.
iv) Type in a '0' and observe the message "DFVIEW EXITED."
v) Type in CNTL A //R to exit the procedure, then see the printout on the lineprinter of the directory and selected records of the selected datafile.

Initializing Both the Raw Data and Reduced Data Datafiles

i) Type in CNTL A /DFINIT/E followed by CNTL A /R.
ii) When the message "INITIALIZATION COMPLETE" appears type in CNTL A //R.

Removing the Latest Record from the Reduced Data Datafile

i) If a bad set of wall data is acquired, immediately type in CNTL A /DFKILL/E.
ii) Wait for the message "LAST RECORD REMOVED."
iii) Type in CNTL A //R to exit the procedure.

Building a Modified Reduced Data Datafile

i) Ensure the current reduced data datafile requires modification by viewing the datafile (See the procedure for viewing a datafile).
ii) Load a backup tape with required data records onto the tape drive MT3.
iii) Transfer selected data records from the backup tape by typing in the following:
a) CNTL A /DFMOD/E
b) CNTL A /R
c) Record number for transfer
iv) When message "DATAFILE MODS COMPLETE" appears another record may be transfered one at a time.
v) Type in '0' to exit and observe the message "DFMOD EXITED."
vi) Check the contents of the current reduced data datafile by repeating step (i).
vii) Remove the backup tape from MT3.
viii) If transfer of other data records from another backup tape is necessary then repeat steps (ii) to (vii).

Removing Data Records from the Current Reduced Data Datafile

i) Generate a backup tape of the current reduced data datafile (See the procedure for making a tape backup).
ii) Initialize the reduced data datafile (See the procedure for initializing datafiles).
iii) Using the backup tape generated in step (i) follow the procedure for building a modified reduced data datafile to transfer the entire backup tape except for the data records to be removed.

(Continued on next page)

Making a Tape Backup for Both Raw Data and Reduced Data Datafiles

i) Ensure the raw data and reduced data datafiles contain the information required on the backup tape by viewing the datafiles. Do this by typing in CNTL A /DFVIEW/E then 'RED' or 'RAW' followed by '0'. Check the printout on the lineprinter.
ii) Load a clean tape onto tape drive MT3.
iii) Transfer data from disk to tape by typing in CNTL A /DFDUMP/E then CNTL A /R.
iv) When the message "DATA TRANSFER COMPLETE" appears exit the procedure by typing in CNTL A //R.
v) Remove the new backup tape from MT3 and label *Datafile Tape* with the current date, then store the tape in an area marked "FLXWAS Datafile Tapes."

Loading a Tape Backup into Both the Raw Data and Reduced Data Datafiles

i) Load a backup tape with the required data records onto tape drive MT3.
ii) Type in CNTL A /DFLOAD/E followed by CNTL A /R.
iii) When the message "DATA TRANSFER COMPLETE" appears exit the procedure by typing in CNTL A //R.
iv) Remove the backup tape from MT3.

Modifying a Selected Record of the Raw Data Datafile

i) Ensure the current raw data datafile is the correct one for modification by viewing the datafile (See the procedure for viewing a datafile).
ii) Type in CNTL A /DFINPUT/E.
iii) Type in the record number for input data.
iv) Type in one item of the input data with its location in the data record.
v) Observe after each item is inputed a display of the current contents of the record for data input is displayed on the CPU-A console. Check for the correct position of the input data in the record. If wrong, re-input the data item in the correct location.
vi) Repeat steps (iv) and (v) for all the input data.
vii) When data input is complete, store the modified record by typing in 0,0.
viii) When the message "DFINPUT EXITED"appears type in CNTL A //R to exit the procedure.
ix) View a printout of the data in the modified record on the lineprinter to check for errors.

Modifying Selected Record of the Reduced Data Datafile

i) Ensure the current reduced data datafile is the correct one for modification by viewing the datafile (See the procedure for viewing a datafile).
ii) Type in CNTL A /DFSHAPI/E.
iii) Type in the record number for input data.
iv) Type in one item of the input data its location in the data record.
v) Observe after each item is inputed a display of the current contents of the record for data input is displayed on the CPU-A console. Check for the correct position of the input data in the record. If wrong, re-input the data item in the correct location.
vi) Repeat steps (iv) and (v) for all the input data.
vii) When data input is complete store the modified record by typing in 0,0.
viii) When the message "DFINPUT EXITED" appears type in CNTL A //R to exit the procedure.
ix) View a printout of the data in the modified record on the lineprinter to check for errors.

APPENDIX E.1 - LISTING OF TABASS
(Raw Data and Reduced Data Datafiles Assignments)

```
C***********************************************************************
C*                                                *      TABASS         *
C*                                                *     07/03/86        *
C*       DISKFILE MANAGEMENT ASSIGNMENTS          *                     *
C*           FOR FLXWAS DATA RECORDS              *                     *
C*                                                *                     *
C*                                                *                     *
C***********************************************************************
C
C
C           REDUCED DATA FILE FORMAT :
C           ========================
C                RECORD 1
C                    WORDS  TYPE          NAME       CONTENTS
C                      1     I*2         NREC  #-NO OF RECORDS FILLED
C                     2-127  I*2         ITEST(#)    TEST NOS.
C                   128-253  I*2         ITRUN(#)     RUN NOS.
C
C                RECORD 2
C                    WORDS  TYPE          NAME       CONTENTS
C                     1-127  I*2         ITDATA(#) DATA PT. NOS.
C                   128-253  I*2         ITER(#)   ITERATION NOS.
C
C                RECORD 3
C                    WORDS  TYPE          NAME       CONTENTS
C                     1-252  R*4         TMACH(#)    MACH NOS.
C
C                RECORD 4
C                    WORDS  TYPE          NAME       CONTENTS
C                     1-252  R*4         TREYNO(#) REYNOLDS NOS.
C
C                RECORD 5
C                    WORDS  TYPE          NAME       CONTENTS
C                     1-252  R*4         TALPHA(#) MODEL AOA (DEG.)
C
C                RECORD 6
C                    WORDS  TYPE          NAME       CONTENTS
C                     1-252  R*4         TCHORD(#) MODEL CHORD (INCHES)
C
C                RECORD 7 TO 126
C                    WORDS  TYPE          NAME       CONTENTS
C                     1-2    R*4         XMACH    RECORD MACH NO.
C                     3-4    R*4         XREYNO   RECORD REYNOLDS NO.
C                     5-6    R*4         XMPARA   RECORD MODEL PARAMETER
C                     7-48   R*4         TWYAS(*) TOP WALL Y CO-ORDS.
C                    49-90   R*4         BWYAS(*) BOTTOM WALL Y CO-ORDS.
C                    91-134  R*4         TWVEL(*) TOP WALL VELOCITIES
C                   135-178  R*4         BWVEL(*) BOTTOM WALL VELOCITIES
C                   179-214  R*4         DSTAS(*) TOP WALL D* CONTOUR
C                   215-250  R*4         DSBAS(*) TOP WALL D* CONTOUR
C
C
C***********************************************************************
      INTEGER CO
      INTEGER*4 IREC
C
      DIMENSION KBUFF(256),LBUFF(256),JBUFF(256),MBUFF(256),NBUFF(256)
      DIMENSION ITEST(128),ITRUN(128),ITDATA(128),ITER(128)
      DIMENSION TMACH(128),TREYNO(128),TALPHA(128),TCHORD(128)
      DIMENSION ITIME(3),IDATE(3),DSTAS(18),DSBAS(18)
      DIMENSION TWYAS(21),BWYAS(21),TWVEL(22),BWVEL(22)
      DIMENSION ITES(128),ITR(128),ITD(128),ITERN(128)
      DIMENSION TM(128),TR(128),TA(128),TC(128)
C
      EQUIVALENCE (KBUFF(1),NREC),            (JBUFF(1),XMACH)
      EQUIVALENCE (KBUFF(2),ITEST(1)),        (JBUFF(3),XREYNO)
      EQUIVALENCE (KBUFF(128),ITRUN(1)),      (JBUFF(5),XMPARA)
      EQUIVALENCE (LBUFF(1),ITDATA(1)),       (JBUFF(7),TWYAS(1))
      EQUIVALENCE (LBUFF(128),ITER(1)),       (JBUFF(49),BWYAS(1))
      EQUIVALENCE (MBUFF(2),ITES(1)),         (JBUFF(91),TWVEL(1))
      EQUIVALENCE  (MBUFF(128),ITR(1)),       (JBUFF(135),BWVEL(1))
      EQUIVALENCE  (NBUFF(1),ITD(1)),         (JBUFF(179),DSTAS(1))
      EQUIVALENCE  (NBUFF(128),ITERN(1)),     (JBUFF(215),DSBAS(1))
C
      DEFINE FILE 11(126,256,U,IREC)
      DEFINE FILE 12(126,256,U,IREC)
      DATA CO/@CO/,LO/@LO/
```

APPENDIX F

SUPPORT SOFTWARE - TSFLOW & SWFLOW

APPENDIX F.1 - LISTING OF PROGRAM TSFLOW

```
      PROGRAM TSFLOW
C
C       EXACT POTENTIAL FLOW CALCULATIONS FOR FLOW ROUND A
C                  LIFTING CYLINDER WITHOUT A WAKE
C
C                       S.WOLF       05/27/86
C
C
      INTEGER CO
      DIMENSION A(43),B(43),C(43),XJACK(19)
      DATA LP/@LP/,CO/@CO/
      DATA XJACK/0.,4.75,10.5,15.5,19.5,22.5,24.5,26.,27.5,29.
     +,30.5,32.,33.5,35.5,37.5,39.5,42.5,46.5,51.5/
      SEMIH = 6.5
      WRITE(CO,165)
165   FORMAT(//10X," EXACT STREAMLINE CALCS"//" INPUT CIRCULATION")
      READ(CO,*) CIR
      PI = 3.14159
      WRITE(LP,105) CIR
105   FORMAT(1H1///,9X,"TOP WALL EXACT STREAMLINE"/
     +13X,"LIFTING CYLINDER "//10X," CIRCULATION = ",F7.4)
      Y1=SEMIH
70    WRITE(LP,115)
115   FORMAT(//"    X              Y                  VEL")
C
C      FOLLOW THAT STREAMLINE !!!
C
      DO 5 J = 1,19
      X1=XJACK(J)-30.75
50    F1=(1-(1/((X1**2)+(Y1**2))))*Y1
      F2=CIR*(ALOG((X1**2+Y1**2)**0.5))/(2*PI)
      IF(X1.GT.1)GO TO 10
      IF(X1.EQ.1)GO TO 20
      IF(Y1.GT.0)F4=(PI+ATAN(Y1/(X1-1)))/(8*PI)
      IF(Y1.LT.0)F4= -(PI-ATAN(Y1/(X1-1)))/(8*PI)
      GO TO 30
20    IF(Y1.GT.0)F4= 0.0625
      IF(Y1.LT.0)F4= -0.0625
      GO TO 30
10    F4=(ATAN(Y1/(X1-1)))/(8*PI)
C30    F3=F1+F2+F4
30    F3=F1+F2
      IF(J.EQ.1) STREAM = F3
      Y2=Y1*((STREAM/F3)**0.5)
      IF(ABS(Y1-Y2).LT.0.00001) GO TO 40
      Y1=Y2
      GO TO 50
C
C      COMPUTE STREAMLINE CPS
C
40    F3=(X1**2)+(Y1**2)
      A1=((X1-1)**2)+(Y1**2)
      F4=((X1**2)-(Y1**2))/(F3**2)
      A2=(X1-1)/(8*PI*A1)
      F5=CIR*Y1/(2*PI*F3)
      A2=0.
      F6=1-F4+F5+A2
      F7=CIR*X1/(2*PI*F3)
      A3=Y1/(8*PI*A1)
      F8=2*X1*Y1
      G1=F3**2
      F9=F8/G1
      A3=0.
      G2=A3-F7-F9
      G3=1-(F6**2)-(G2**2)
      VEL=(SQRT(1-G3))-1
      WMOVE = Y2-SEMIH
      IF(Y2.LT.0)WMOVE = Y2+SEMIH
      WRITE(LP,125)X1,WMOVE,VEL
125   FORMAT(F8.3,F14.4,F20.4)
      C2=SQRT(1-G3)
      A(J)=X1
      B(J)=Y2
      C(J)=C2
5     CONTINUE
C
C      PERFORM A CIRCULATION INTEGRAL
C
      S1=(B(2)-B(1))/SQRT(1+(B(2)-B(1))**2)
      S2=(B(18)-B(19))/SQRT(1+(B(18)-B(19))**2)
      P1=B(2)*S1*C(2)
      P3=B(18)*S2*C(18)
      S2=(C(18)+C(2))*0.5
      S3=0.
      DO 15 I = 2,18
      S3=S3+(C(I)*(XJACK(I+1)-XJACK(I))/2)
15    CONTINUE
      P2=S2+S3
      PART1 = P1 + P3
      GAMMA = PART1+P2
      IF(Y1.GT.0)GAMMAT=GAMMA
      WRITE(LP,135)PART1,P2,GAMMA
135   FORMAT(//,5X," CIRCULATION INTEGRAL"//12X," PART 1 = "
     +,F7.3/12X," PART 2 = ",F7.3//10X," TOTAL GAMMA = ",F7.3)
      IF(Y1.LT.0) GO TO 80
      WRITE(LP,145) CIR
145   FORMAT(1H1///,8X,"BOTTOM WALL EXACT STREAMLINE"/
     +13X,"LIFTING CYLINDER "//10X," CIRCULATION = ",F7.4)
      Y1= -SEMIH
      GO TO 70
80    CIRTOT=GAMMAT-GAMMA
      WRITE(LP,155) CIRTOT
155   FORMAT(///10X," CIRCULATION ESTIMATE = ",F7.4)
      END FILE LP
      STOP
      END
```

APPENDIX F.2 - SAMPLE OUTPUT OF TSFLOW

```
                 TOP WALL EXACT STREAMLINE
                      LIFTING CYLINDER

                  CIRCULATION =   0.5000

        X               Y                   VEL
     -30.750          0.0000              -0.0004
     -26.000          0.0152              -0.0005
     -20.250          0.0389              -0.0006
     -15.250          0.0680              -0.0006
     -11.250          0.1022               0.0003
      -8.250          0.1392               0.0029
      -6.250          0.1715               0.0073
      -4.750          0.1995               0.0130
      -3.250          0.2282               0.0209
      -1.750          0.2520               0.0291
      -0.250          0.2633               0.0335
       1.250          0.2575               0.0312
       2.750          0.2371               0.0238
       4.750          0.1996               0.0130
       6.750          0.1628               0.0059
       8.750          0.1321               0.0022
      11.750          0.0971               0.0001
      15.750          0.0646              -0.0006
      20.750          0.0365              -0.0006

    CIRCULATION INTEGRAL

       PART 1 =    0.283
       PART 2 =   24.520

     TOTAL GAMMA =   24.802
```

BOTTOM WALL EXACT STREAMLINE
LIFTING CYLINDER

CIRCULATION = 0.5000

X	Y	VEL
-30.750	0.0000	-0.0014
-26.000	0.0102	-0.0019
-20.250	0.0233	-0.0029
-15.250	0.0340	-0.0044
-11.250	0.0386	-0.0060
-8.250	0.0351	-0.0069
-6.250	0.0262	-0.0059
-4.750	0.0147	-0.0033
-3.250	0.0000	0.0016
-1.750	-0.0141	0.0077
-0.250	-0.0214	0.0112
1.250	-0.0177	0.0093
2.750	-0.0050	0.0036
4.750	0.0147	-0.0033
6.750	0.0290	-0.0064
8.750	0.0364	-0.0068
11.750	0.0384	-0.0058
15.750	0.0331	-0.0042
20.750	0.0222	-0.0028

CIRCULATION INTEGRAL

PART 1 = -0.136
PART 2 = 24.311

TOTAL GAMMA = 24.175

CIRCULATION ESTIMATE = 0.6270

APPENDIX F.3 - LISTING OF PROGRAM SWFLOW

PRECEDING PAGE BLANK NOT FILMED

PAGE 174 INTENTIONALLY BLANK

```
      PROGRAM SWFLOW
C
C      EXACT POTENTIAL FLOW CALCULATIONS FOR FLOW ROUND A
C     LIFTING CYLINDER WITHOUT A WAKE IN A STRAIGHT WALL TEST SECTION
C                    S.WOLF        08/28/85
C
      INTEGER CO
      DIMENSION A(43),B(43),C(43),XJACK(19)
      DATA LP/@LP/,CO/@CO/
      DATA XJACK/0.,4.75,10.5,15.5,19.5,22.5,24.5,26.,27.5,29.
     +,30.5,32.,33.5,35.5,37.5,39.5,42.5,46.5,51.5/
      SEMIH = 6.5
      IWALL=1
      WRITE(CO,165)
165   FORMAT(//10X," EXACT STREAMLINE CALCS"//" INPUT CIRCULATION")
      READ(CO,*) CIR
      PI = 3.14159
      WRITE(LP,105) CIR
105   FORMAT(1H1///,8X,"STRAIGHT TOP WALL STREAMLINE"/
     +13X,"LIFTING CYLINDER "//10X," CIRCULATION = ",F7.4)
      Y1=SEMIH
70    WRITE(LP,115)
115   FORMAT(//"    X              Y                  CP           PHI")
C
C      FOLLOW THAT STREAMLINE !!!
C
      DO 5 J = 1,19
      F3=0.
      IMAGE=1
      X1=XJACK(J)-30.75
50    F1=(1-(1/((X1**2)+(Y1**2))))*Y1
      F2=CIR*(ALOG((X1**2+Y1**2)**0.5))/(2*PI)
      IF(IMAGE.EQ.0)F2= -F2
      IF(X1.GT.1)GO TO 10
      IF(X1.EQ.1)GO TO 20
      IF(Y1.GT.0)F4=(PI+ATAN(Y1/(X1-1)))/(8*PI)
      IF(Y1.LT.0)F4= -(PI-ATAN(Y1/(X1-1)))/(8*PI)
      GO TO 30
20    IF(Y1.GT.0)F4= 0.0625
      IF(Y1.LT.0)F4= -0.0625
      GO TO 30
10    F4=(ATAN(Y1/(X1-1)))/(8*PI)
C30    F3=F1+F2+F4+F3
30    F3=F1+F2+F3
      Y1= -Y1
      IF(IMAGE.EQ.0) GO TO 40
      IMAGE=0
      GO TO 50
C
C      COMPUTE STREAMLINE CPS
C
40    IMAGE=2
      STREAM = F3
      F6=0.
      G2=0.
100   F3=(X1**2)+(Y1**2)
      A1=((X1-1)**2)+(Y1**2)
      F4=((X1**2)-(Y1**2))/(F3**2)
      A2=(X1-1)/(8*PI*A1)
      F5=CIR*Y1/(2*PI*F3)
      IF(IMAGE.EQ.0)F5= -F5
      A2=0.
      F6=F6-F4+F5+A2
      F7=CIR*X1/(2*PI*F3)
      IF(IMAGE.EQ.0)F7= -F7
      A3=Y1/(8*PI*A1)
      F8=2*X1*Y1
      G1=F3**2
      F9=F8/G1
      A3=0.
      G2=A3-F7-F9+G2
      Y1 = -Y1
      IF(IMAGE.EQ.0) GO TO 90
      IMAGE=0
      GO TO 100
90    F6=F6+1.
      G3=1-(F6**2)-(G2**2)
      WRITE(LP,125)X1,Y1,G3,STREAM
125   FORMAT(F8.3,F14.4,F20.4,F15.4)
      C2=SQRT(1-G3)
      A(J)=X1
      B(J)=Y1
      C(J)=C2
5     CONTINUE
C
C      PERFORM A CIRCULATION INTEGRAL
C
      S1=(B(2)-B(1))/SQRT(1+(B(2)-B(1))**2)
      S2=(B(18)-B(19))/SQRT(1+(B(18)-B(19))**2)
      P1=B(2)*S1*C(2)
      P3=B(18)*S2*C(18)
      S2=(C(18)+C(2))*0.5
      S3=0.
      DO 15 I = 2,18
      S3=S3+(C(I)*(XJACK(I+1)-XJACK(I))/2)
15    CONTINUE
      P2=S2+S3
      PART1 = P1 + P3
      GAMMA = PART1+P2
      IF(IWALL.EQ.1)GAMMAT=GAMMA
      WRITE(LP,135)PART1,P2,GAMMA
135   FORMAT(//,5X," CIRCULATION INTEGRAL"//12X," PART 1 = "
     +,F7.3/12X," PART 2 = ",F7.3//10X," TOTAL GAMMA = ",F7.3)
      IF(IWALL.EQ.2) GO TO 80
      WRITE(LP,145) CIR
145   FORMAT(1H1///,7X,"STRAIGHT BOTTOM WALL STREAMLINE"/
     +13X,"LIFTING CYLINDER "//10X," CIRCULATION = ",F7.4)
      Y1= -SEMIH
      IWALL=2
      GO TO 70
80    CIRTOT=GAMMAT-GAMMA
      WRITE(LP,155) CIRTOT
155   FORMAT(///10X," CIRCULATION ESTIMATE = ",F7.4)
      END FILE LP
      STOP
      END
```

APPENDIX F.4 - SAMPLE OUTPUT OF SWFLOW

```
              STRAIGHT TOP WALL STREAMLINE
                   LIFTING CYLINDER

                 CIRCULATION =  0.5000

     X              Y         CP              PHI
  -30.750         6.5000    0.0016          0.0000
  -26.000         6.5000    0.0020          0.0000
  -20.250         6.5000    0.0026          0.0000
  -15.250         6.5000    0.0025          0.0000
  -11.250         6.5000   -0.0004          0.0000
   -8.250         6.5000   -0.0103          0.0000
   -6.250         6.5000   -0.0276          0.0000
   -4.750         6.5000   -0.0513          0.0000
   -3.250         6.5000   -0.0864          0.0000
   -1.750         6.5000   -0.1257          0.0000
   -0.250         6.5000   -0.1483          0.0000
    1.250         6.5000   -0.1364          0.0000
    2.750         6.5000   -0.0998          0.0000
    4.750         6.5000   -0.0513          0.0000
    6.750         6.5000   -0.0220          0.0000
    8.750         6.5000   -0.0077          0.0000
   11.750         6.5000    0.0003          0.0000
   15.750         6.5000    0.0026          0.0000
   20.750         6.5000    0.0026          0.0000

 CIRCULATION INTEGRAL

    PART 1 =   0.000
    PART 2 =  24.668

  TOTAL GAMMA =  24.668
```

```
           STRAIGHT BOTTOM WALL STREAMLINE
                  LIFTING CYLINDER

               CIRCULATION =  0.5000


    X            Y          CP          PHI
 -30.750      -6.5000     0.0058      0.0000
 -26.000      -6.5000     0.0078      0.0000
 -20.250      -6.5000     0.0117      0.0000
 -15.250      -6.5000     0.0175      0.0000
 -11.250      -6.5000     0.0239      0.0000
  -8.250      -6.5000     0.0271      0.0000
  -6.250      -6.5000     0.0234      0.0000
  -4.750      -6.5000     0.0131      0.0000
  -3.250      -6.5000    -0.0063      0.0000
  -1.750      -6.5000    -0.0309      0.0000
  -0.250      -6.5000    -0.0459      0.0000
   1.250      -6.5000    -0.0379      0.0000
   2.750      -6.5000    -0.0144      0.0000
   4.750      -6.5000     0.0131      0.0000
   6.750      -6.5000     0.0251      0.0000
   8.750      -6.5000     0.0270      0.0000
  11.750      -6.5000     0.0231      0.0000
  15.750      -6.5000     0.0168      0.0000
  20.750      -6.5000     0.0113      0.0000


CIRCULATION INTEGRAL

   PART 1 =    0.000
   PART 2 =   24.248

 TOTAL GAMMA =  24.248


 CIRCULATION ESTIMATE =  0.4196
```

APPENDIX G

Description of Subroutine WALCAL Computations

1. Introduction

The purpose of this program is to determine velocity distributions and associated streamline shapes in the farfield of thin airfoils. Linear potential flow is assumed with compressibility taken into account by use of the Prandtl-Glauert factor. Calculations are based on a simplified singularity distribution of the model. Its solid blockage is simulated by a Rankine Oval of same length and maximum thickness as the airfoil and is represented by a source and a sink of equal strength. The model lift is represented by a single vortex, the strength of which is given by the estimated section lift coefficient. The wake blockage is accounted for by a source. Its strength is derived from the estimated section drag coefficient.

The program is used to predict the wall deflections and corresponding pressure distributions (upper and lower wall) of the 0.3-m adaptive wall test section. The computed deflections are added to the aerodynamically straight contours to provide jack movements relative to flat walls.

2. Symbols

(x,y)	reference system with x-axis parallel to freestream velocity U_∞. Its origin is located at the airfoil center
(x',y')	'tunnel' based reference system x'-axis coincides with tunnel center line. The origin is located at the test section inlet
$x'_{1,\ldots,21}$	position of jacks along the top and bottom flexible walls of the 0.3-m adaptive wall test section [inch]
U_∞	freestream velocity
M_∞	freestream Mach number
$\beta = (1 - M_\infty^2)^{1/2}$	compressibility factor
ϕ	disturbance potential (normalized with U_∞)
ψ	disturbance stream function (normalized with U_∞)
Φ	$= x + \phi$, flow potential
Ψ	$= y + \psi$, stream function
u	$= \frac{\partial \phi}{\partial x}$ disturbance velocities
v	$= \frac{\partial \phi}{\partial y}$
c	airfoil chord length [inch]

t	airfoil maximum thickness [inch]
Q'	source strength (normalized with $2\pi U_\infty$)
Γ'	vortex strength (normalized with $2\pi U_\infty$)
C_L	section lift coefficient
C_D	section drag coefficient
Re_c	Reynolds number based on airfoil chord
O	Landau Symbol

Subscripts:

B	blockage-induced
L	lift-induced
W	wake-induced

Superscripts:

$\hat{}$	quantities in incompressible flow (related to compressible flow by Prandtl-Glauert transformation)

3. Analysis

3.1 Calculation of disturbance velocities

The disturbance velocities

$$\begin{aligned} u &= u_B + u_L + u_W \\ v &= v_B + v_L + v_W \end{aligned} \qquad (1)$$

in the farfield of the airfoil can be calculated individually using only global information about the model geometry and its aerodynamic behavior, |1|.

3.1.1 Blockage

The solid blockage is caused by the streamline displacement due to the airfoil cross-section area distribution. The main parameters therefore are the airfoil chord length and its maximum thickness, which are readily available for each airfoil.

These two geometrical quantities uniquely define a Rankine oval whose flow field can easily be generated by a source and a sink of equal strength Q' located at ($\mp$a, o) within the body.

In order to determine Q' and 'a' from the given values c and t, it is convenient to first reduce the compressible potential equation to the Laplace equation by the usual transformation, |2|

$$\begin{aligned} x &= \hat{x} \\ y &= \frac{1}{\beta}\hat{y} \\ \phi(x,y) &= \frac{1}{\beta}\hat{\phi}(\hat{x},\hat{y}) \end{aligned} \tag{2}$$

The resulting incompressible flow

$$\hat{\phi}_{\hat{x}\hat{x}} + \hat{\phi}_{\hat{y}\hat{y}} = 0 \tag{3}$$

is then related to the compressible flow field by the relations

$$\begin{aligned} \phi_x(x,y) &= \frac{1}{\beta}\hat{\phi}_{\hat{x}}(\hat{x},\hat{y}) \\ \phi_y(x,y) &= \hat{\phi}_{\hat{y}}(\hat{x},\hat{y}) \end{aligned} \tag{4}$$

The geometry of the body, especially its thickness, does not change under the transformation.

The velocity potential for the incompressible flow past the Rankine Oval is given by

$$\hat{\Phi}(\hat{x},\hat{y}) = \hat{x} + Q'\{\ln[(\hat{x}+a)^2 + \hat{y}^2]^{1/2} - \ln[(\hat{x}-a)^2 + \hat{y}^2]^{1/2}\} \tag{5}$$

Q' and a have now to be adjusted to match the prescribed geometry (length and thickness) of the oval:

In order to generate stagnation points at $(\hat{x}_s, \hat{y}_s) = (\pm\frac{c}{2},o)$, Q' and 'a' have to fulfill the equation

$$\begin{aligned} \hat{U}(\hat{x}_s,\hat{y}_s) &= \frac{\partial\hat{\Phi}}{\partial\hat{x}}(\hat{x}_s,\hat{y}_s) = 1 + \hat{u}(\hat{x}_s,\hat{y}_s) \\ &= 1 + Q'\left\{\frac{\hat{x}_s + a}{(\hat{x}_s + a)^2 + \hat{y}_s^2} - \frac{\hat{x}_s - a}{(\hat{x}_s - a)^2 + \hat{y}_s^2}\right\} \\ &\stackrel{!}{=} 0 \end{aligned} \tag{6}$$

and therefore

$$Q' = \frac{\hat{x}_s^2 - a^2}{2a} \tag{7}$$

A second relation can be derived from the prescribed maximum thickness t at

$\hat{x}_t = 0$. Application of Gauss' law immediately yields the equation

$$2\int_0^{\hat{y}_t} [1 + \hat{u}\ (o,\hat{y})]\ d\hat{y} = 2\pi Q' \tag{8}$$

with $\hat{y}_t = \frac{t}{2}$.

Using (6) we get

$$\hat{u}\ (o,\hat{y}) = Q' \frac{2a}{a^2 + \hat{y}^2} \tag{9}$$

and then

$$\begin{aligned}\int_0^{\hat{y}_t} [1 + \hat{u}\ (o,\hat{y})]\ d\hat{y} &= \hat{y}_t + 2aQ' \frac{1}{a} \text{ arc tan } \frac{\hat{y}}{a} \Big|_0^{\hat{y}_t} \\ &= \hat{y}_t + 2Q' \text{ arc tan } \frac{\hat{y}t}{a} \\ &\overset{!}{=} \pi Q'\end{aligned} \tag{10}$$

Assuming thin airfoils

$$\frac{t}{c} = \frac{\hat{y}_t}{\hat{x}_s} \approx \frac{\hat{y}_t}{a} << 1 \tag{11}$$

we can use the first order approximation

$$\text{arc tan } x = x + O\ (x^3\ ) \qquad , \ |x| < 1 \tag{12}$$

which transforms (10) into

$$\hat{y}_t + Q'\ (2\ \frac{\hat{y}_t}{a} - \pi) = 0 \tag{13}$$

Combining (7) and (13) finally yields the cubic equation

$$\begin{aligned} a^3 + pa + q &= 0 \\ p &= -\hat{x}_s^2 \\ q &= \frac{2\hat{x}_s^2\hat{y}_t}{\pi}\end{aligned} \tag{14}$$

Since

$$D = (\frac{p}{3})^3 + (\frac{q}{2})^2 = \hat{x}_s^4\ [(\frac{\hat{y}_t}{\pi})^2 - \frac{\hat{x}_s^2}{27}] < 0$$

according to (11), (14) has three real solutions, |3|

$$a_1 = r \cos\left(\frac{\delta}{3}\right)$$

$$a_2 = r \cos\left(\delta + \frac{2\pi}{3}\right) \qquad (15)$$

$$a_3 = r \cos\left(\delta + \frac{4\pi}{3}\right)$$

with
$$r = \frac{2\hat{x}_s}{\sqrt{3}} \qquad (16)$$

$$\delta = \text{arc cos}\left(- \frac{\hat{y}_t \sqrt{27}}{\pi \hat{x}_s}\right)$$

However, application of assumption (11), which may be specified as

$$\frac{t}{c} < 0.2 \qquad (17)$$

(airfoil thickness less than 20%), reveals

$$\frac{a_2}{\hat{x}_s} < 0 \ , \quad \frac{a_3}{\hat{x}_s} > 1 \qquad (18)$$

It is obvious that these two solutions do not generate the desired flow field and can therefore be disregarded. Inserting the 'correct' solution a = a, in (7) and using (5) and (4) we can calculate the blockage induced disturbance velocities everywhere in the flow field:

$$u_B(x,y) = \frac{1}{\beta}\hat{u}_B(\hat{x},\hat{y}) = \frac{Q'}{\beta}\left\{\frac{x + a}{(x + a)^2 + \beta^2 y^2} - \frac{x - a}{(x - a)^2 + \beta^2 y^2}\right\}$$

$$v_B(x,y) = \hat{v}_B(\hat{x},\hat{y}) = \beta\, Q' y \left\{\frac{1}{(x + a)^2 + \beta^2 y^2} - \frac{1}{(x - a)^2 + \beta^2 y^2}\right\} \qquad (19)$$

3.1.2 Lift

The airfoil lift is represented by a single vortex located at the 1/4 chord point. Its strength Γ' is related to the section lift coefficient by the Kutta-Youkowsky-Theorem, |2|

$$\Gamma' = \frac{cC_L}{4\pi} \qquad (20)$$

Application of the transformation (2) shows that the corresponding incompressible flow has circulation

$$\hat{\Gamma}' = \frac{1}{2\pi}\int\{\hat{\phi}_{\hat{x}}\, d\hat{x} + \hat{\phi}_{\hat{y}}\, d\hat{y}\} = \frac{1}{2\pi}\int\{\beta\phi_x\, dx + \phi_y \beta\, dy\} = \beta\Gamma' \qquad (21)$$

Differentiation of the incompressible vortex-potential

$$\hat{\phi}_L\ (\hat{x},\hat{y}) = -\ \hat{\Gamma}'\ \text{arc tan}\ \frac{\hat{y}}{\hat{x} + \frac{c}{4}} \tag{22}$$

and transforming back in the compressible plane immediately yields the lift induced disturbance velocities throughout the flow field

$$\begin{aligned} u_L\ (x,y) &= \beta\Gamma'\ \frac{y}{(x + \frac{c}{4})^2 + \beta^2 y^2} \\ v_L\ (x,y) &= -\beta\Gamma'\ \frac{x + \frac{c}{4}}{(x + \frac{c}{4})^2 + \beta^2 y^2} \end{aligned} \tag{23}$$

3.1.3 Wake Blockage

The source representing the model wake is located at the trailing edge of the airfoil Its (normalized) stength Q'_W can be derived from the section drag coefficient as follows, |1|

$$Q'_W = \frac{cC_D}{4\pi} \tag{24}$$

The resulting disturbance velocities are then according to (19)

$$\begin{aligned} u_W\ (x,y) &= \frac{Q'_W}{\beta}\ \frac{x - \frac{c}{2}}{(x - \frac{c}{2})^2 + \beta^2 y^2} \\ v_W\ (x,y) &= \beta Q'_W\ \frac{y}{(x - \frac{c}{2})^2 + \beta^2 y^2} \end{aligned} \tag{25}$$

3.2 Calculation of streamline shapes

The equation for the streamlines in the related incompressible flow is given by

$$\hat{\Psi} = \hat{y} + \hat{\psi}_B + \hat{\psi}_L + \hat{\psi}_W = C = \text{const.} \tag{26}$$

with

$$\begin{aligned} \hat{\psi}_B\ (\hat{x},\hat{y}) &= Q'\ \{\text{arc tan}\ \frac{\hat{y}}{\hat{x} + a} - \text{arc tan}\ \frac{\hat{y}}{\hat{x} - a}\} \\ \hat{\psi}_L\ (\hat{x},\hat{y}) &= \hat{\Gamma}'\ \ell n\ [(\hat{x} + \frac{c}{4}{}^2 + \hat{y}^2]^{1/2} \qquad , \hat{\Gamma}' = \beta\Gamma' \\ \hat{\psi}_W\ (\hat{x},\hat{y}) &= Q'_W\ \text{arc tan}\ \frac{y}{\hat{x} - \frac{c}{2}} \end{aligned} \tag{27}$$

The value of the constant is determined by inserting a known point $(\hat{x}_o, \hat{y}_o)$ from the streamline to be calculated

$$C = \hat{\Psi}\ (\hat{x}_o, \hat{y}_o) \tag{28}$$

Rearranging (26) leads to the following simple iteration formula

$$\hat{y}_{(N+1)} = \hat{\Psi}\ (\hat{x}_o, \hat{y}_o) - (\hat{\psi}_B + \hat{\psi}_L + \hat{\psi}_W)\ (\hat{x}, \hat{y}_{(N)}), \quad N = 0, 1, \ldots \tag{29}$$

with

$$\hat{y}_{(o)}\ (\hat{x}_k) = \hat{Y}(\hat{x}_{k-1}) \quad , \quad k = 1, 2, \ldots \text{ (jack locations)} \tag{30}$$

Iteration stops if

$$|\hat{y}_{(N+1)} - \hat{y}_{(N)}| < 10^{-3} \tag{31}$$

The corresponding compressible streamline then passes through the point $x_o = \hat{x}_o$, $y_o = \frac{1}{\beta}\hat{y}_o$. Its distortion remains unchanged

$$y - y_o = \hat{y} - \hat{y}_o \tag{32}$$

The programming of the equations is relatively straightforward. Only expression (26) has been implemented in such a way as to separately calculate the wall deflections due to the different singularities. The reformulated equation reads like

$$\begin{aligned} \hat{y} - \hat{y}_o &= \hat{\psi}_{B,o} - \hat{\psi}_B + \hat{\psi}_{L,o} - \hat{\psi}_L + \hat{\psi}_{W,o} - \hat{\psi}_W \\ &= \hat{y}_B - y_o + \hat{y}_L - y_o + \hat{y}_W - y_o \end{aligned} \tag{33}$$

with $\hat{\psi}_{B,o} = \hat{\psi}_B\ (\hat{x}_o, \hat{y}_o)$ and so forth.

Inserting the relations (27) for the stream functions leads to the iteration formulae:

solid blockage

$$\begin{aligned} \hat{y}_{B,(N+1)} - \hat{Y}_o = Q' \ \{&\text{arc tan}\ \frac{\hat{y}_o}{\hat{x}_o + a} - \text{arc tan}\ \frac{\hat{y}_o}{\hat{x}_o - a} \\ &+ \text{arc tan}\ \frac{\hat{y}_{(N)}}{\hat{x} - a} - \text{arc tan}\ \frac{\hat{y}_{(N)}}{\hat{x} + a}\} \\ = Q' \ \{&\text{arc tan}\ \frac{\hat{x}_o - a}{\hat{y}_o} - \text{arc tan}\ \frac{\hat{x}_o + a}{\hat{y}_o} \\ &+ \text{arc tan}\ \frac{\hat{x} + a}{\hat{y}_{(N)}} - \text{arc tan}\ \frac{\hat{x} - a}{\hat{y}_{(N)}}\} \end{aligned} \tag{34}$$

The use of the inverse arguments in the arc tan functions is preferable for the numerical computations since it avoids the otherwise occurring discontinuities when $\hat{x} \pm a$ changes sign.

lift effect

$$\hat{y}_{L,\,(N+1)} - \hat{y}_0 = \frac{1}{2}\beta\Gamma' \{\ell n\,[(\hat{x}_0 + \frac{c}{4})^2 + \hat{y}_0^2] - \ell n\,[(\hat{x} + \frac{c}{4})^2 + \hat{y}_{(N)}^2]\} \qquad (35)$$

wake blockage

$$\begin{aligned} \hat{y}_{w,(N+1)} - \hat{y}_0 &= Q_w' \quad \{\text{arc tan}\ \frac{\hat{y}_0}{\hat{x}_0 - \frac{c}{2}} - \text{arc tan}\ \frac{\hat{y}_{(N)}}{\hat{x} - \frac{c}{2}}\} \\ &= Q_w' \quad \{\text{arc tan}\ \frac{\hat{x} - \frac{c}{2}}{\hat{y}_{(N)}} - \text{arc tan}\ \frac{\hat{x}_0 - \frac{c}{2}}{\hat{y}_0}\} \end{aligned} \qquad (36)$$

The perturbation velocities along the walls, required by the wall adaptation procedure |4|, can be determined from (1), (19), (23), and (25) as follows

$$u_{wall} = [(1 + u)^2 + v^2]^{1/2} - 1 \qquad (37)$$

4. References

1. Mokry, M.; Chan, Y. Y.; and Jones, D. J.: Two-Dimensional Wind Tunnel Wall Interference, AGARDograph No. 281, 1983.

2. Ward, G. N.: Linearized Theory of Steady High-Speed Flow. Cambridge, At the University Press, 1955.

3. Bronshtein, I. N.; and Semendyayew, K. A.: Handbook of Mathematics, Van Nostrand.

NASA

Report Documentation Page

1. Report No.
NASA CR-181694

2. Government Accession No.

3. Recipient's Catalog No.

4. Title and Subtitle
Wall Adjustment Strategy Software for Use with the NASA Langley 0.3-Meter Transonic Cryogenic Tunnel Adaptive Wall Test Section

5. Report Date
November 1988

6. Performing Organization Code

7. Author(s)
Stephen W. D. Wolf

8. Performing Organization Report No.

10. Work Unit No.
505-61-01-02

9. Performing Organization Name and Address
Vigyan Research Associates, Inc.
Hampton, VA 23666

11. Contract or Grant No.
NAS1-17919, Task 1

13. Type of Report and Period Covered
Contractor Report

12. Sponsoring Agency Name and Address
National Aeronautics and Space Administration
Langley Research Center
Hampton, VA 23665-5225

14. Sponsoring Agency Code

15. Supplementary Notes
Langley Technical Monitor: Edward J. Ray

16. Abstract
This Wall Adjustment Strategy (WAS) software provides successful on-line control of the 2-D flexible walled test section of the Langley 0.3-m Transonic Cryogenic Tunnel. This software package allows the level of operator intervention to be regulated as necessary for research and production type 2-D testing using an Adaptive Wall Test Section (AWTS). The software is designed to accept modification for future requirements, such as 3-D testing, with a minimum of complexity. The WAS software described here is a unique attempt to provide a *user friendly* package which could be used to control any flexible walled AWTS. Control system constraints influence the details of data transfer, not the data type. Consequently, this entire software package could be used in different control systems, if suitable interface software is available.

A complete overview of the software highlights the data flow paths, the modular architecture of the software and the various operating and analysis modes available. A detailed description of the software modules includes listings of the code. We provide a user's manual to explain task generation, operating environment, user options and what to expect at execution. A typical output listing is shown as part of one of the software tests described in the text. Necessary utility software for data file management is only briefly described, since this software is system oriented.

The WAS software forms the major component of any AWTS control system. We intend this report to help those who need to know the details of controlling an AWTS with flexible walls and those involved with WAS software maintenance.

17. Key Words (Suggested by Author(s))
Adaptive wall wind tunnels
Wall adjustment strategy
Flexible walled test sections

18. Distribution Statement
Unclassified - Unlimited
Subject Category 02

19. Security Classif. (of this report)
Unclassified

20. Security Classif. (of this page)
Unclassified

21. No. of pages
195

22. Price
A09