NASA Contractor Report 165687

NASA-CR-165687
19810012529

AIRCL: A PROGRAMMED SYSTEM FOR GENERATING NC TAPES FOR AIRPLANE MODELS - USER'S MANUAL

Nuri Akgerman and C. F. Billhardt

BATTELLE, COLUMBUS LABORATORIES
Columbus, Ohio 43201

Contract NAS1-15090
March 1981

NASA
National Aeronautics and
Space Administration

Langley Research Center
Hampton, Virginia 23665

AIRCL

A PROGRAMMED SYSTEM FOR GENERATING NC TAPES FOR AIRPLANE MODELS

User's Manual

By

Nuri Akgerman and C. F. Billhardt

BATTELLE
Columbus Laboratories

INTRODUCTION

AIRCL is a computer program written in FORTRAN which calculates the coordinates needed to machine models of airplanes on numerically controlled (NC) machine tools. These models are typically advanced designs made for wind-tunnel testing. AIRCL runs on a Control Data Corp (CDC) 6500 or Cyber 73 computer using the NOS or NOS/BE operating system. The input to AIRCL is data representing coordinates on the fuselage bulkheads and wing chords. From this data and certain other optional parameters, AIRCL will determine the intersection between the wing and fuselage and the cutter locations needed to machine the fuselage and/or wing. The output of AIRCL is a file formatted to imitate the CL output of CDC's APT 3 compiler. This permits the output to be post-processed and then run on a specific machine tool. No special features such as linear or circular interpolation are assumed for the machine tool controller. The only requirement is that it must have mirror image (axis inversion) on the Y axis. Since airplanes are symmetrical about the X axis, AIRCL always produces the CL file for the right-hand side. Mirror image is then used during machining to produce the left side. Figure 1 is a photograph of a test model NC machined using tapes generated by AIRCL.

FIGURE 1. MODEL MACHINED USING TAPES GENERATED BY AIRCL

Program Structure

AIRCL runs as a batch program with data input from a file copied from cards or some other source. AIRCL uses CDC's Segmentation Loader to reduce the memory required for execution. AIRCL requires approximately 123,000(8) [42,500(10)] words of memory to run. It also requires two files for storing intermediate results. These are opened if and when needed, and closed when no longer required. The general structure of AIRCL is shown in Figure 2. For clarity, this figure does not include system utilities and library functions which are provided by the compiler or other CDC system programs.

The following summarizes the function of each program element shown in Figure 2. A more complete description of the operation of each element is given in the text.

I. Root

- A. WNGFSLG - Root program. Initializes data; calls level 2 elements; provides I/O buffers. This does no data processing of its own.
- B. INTPLT - Calls interpolation routines specified or allowed by amount of data.
- C. WRITWA - Writes to and reads from a scratch file used for data expansion in FCLPTS and WCLPTS.
- D. OPENWA - Opens or closes the scratch file used for data expansion.
- E. WAERR - Used if an error occurs in WRITWA.

II. Level 2 - MAJOR SUBROUTINES

- A. READFW - Reads namelist parameters, and wing and fuselage data.
- B. WINGCL - Finds cutter offset locations for wing surface.
- C. FSLGCL - Finds cutter offset locations for fuselage surface.
- D. INTRSC - Finds intersection of wing and fuselage CL data.
- E. WCLPTS - Formulates wing CL data, expanding data by interpolation as needed. Checks and compensates for the intersection of each wing pass with the W/F intersection curve.

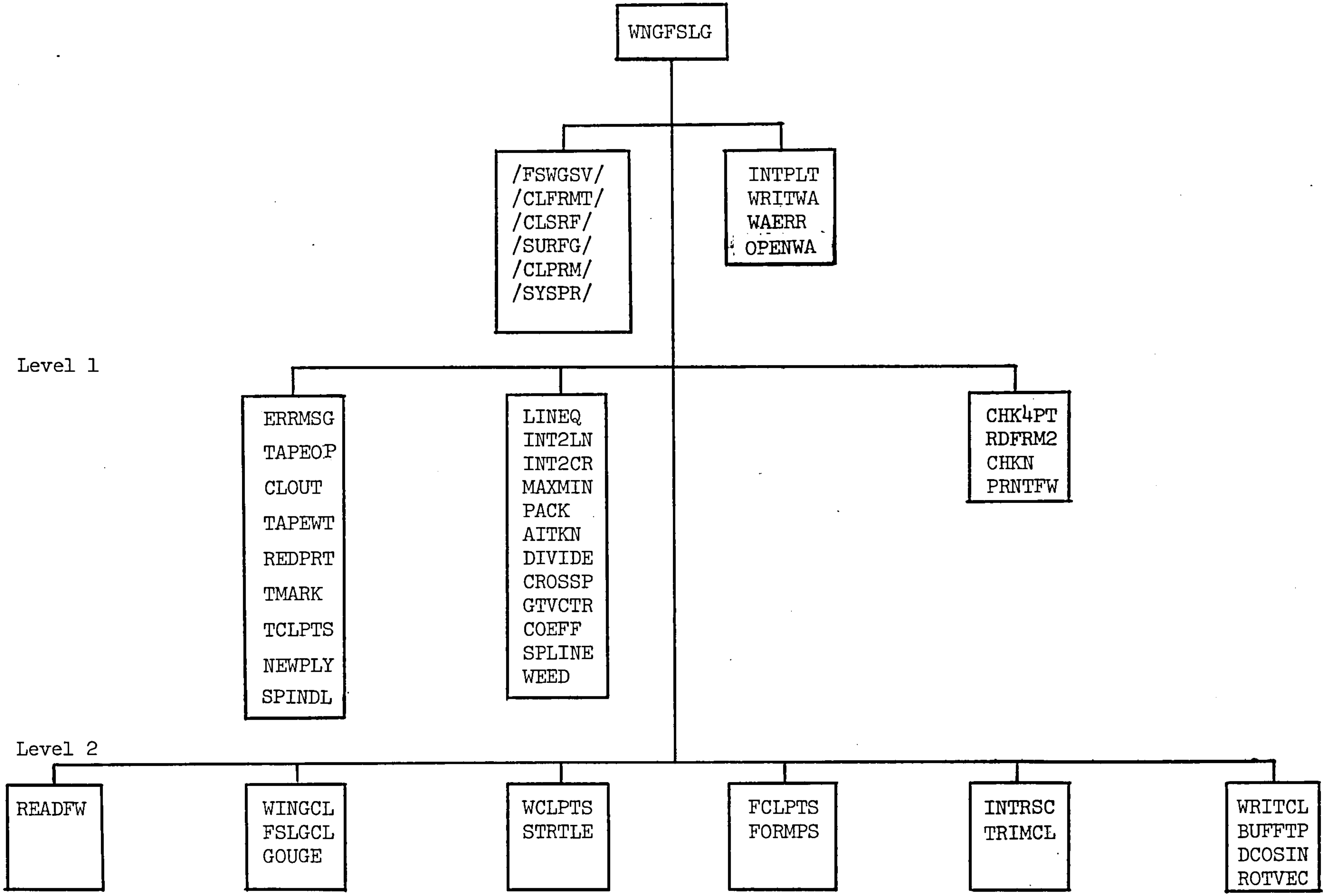

FIGURE 2. PROGRAM STRUCTURE OF AIRCL

F. FCLPTS - Generates cutter paths for fuselage.

G. WRITCL - Writes the output file.

H. TRIMCL - Provides a trim cut around the X-Y planview of the model.

I. STRTLE - Generates multiple passes at the wing leading edge to eliminate heavy starting cuts.

J. GOUGE - Checks to see if cutter will gouge at each cutter offset location.

K. FORMPS - Forms fuselage pass.

L. BUFFTP - Formatting routine for CLTAPE.

M. DCOSIN - Calculates the direction cosine matrix.

N. ROTVEC - Rotates a 3-D matrix in 3-D space.

III. Level 1 - WORKING ROUTINES

A. LINEQ - Determines the coefficients of a line setment.

B. INT2LN - Finds the intersection of two line segments.

C. MAXMIN - Finds the maximum and minimum of a linear array.

D. AITKN - Does polynomial interpolation of a curve.

E. DIVIDE - Divides a plane curve into equal size segments using a linear interpolation.

F. GTVCTR - Determines the coefficients of a vector.

G. CROSSP - Determines the cross product of two vectors.

H. INT2CR - Determines the intersection of two curves.

I. SPLINE - Does spline interpolation of a plane curve.

J. COEFF - Determines the spline fit coefficient for a set of points.

K. TCLPTS - Creates a cutter offset for a trim cut; limits trim cut data to +Y values.

L. NEWPLY - Creates cutter offset data to a plane polygon.

M. ERRMSG - Generates error messages during writing of CL file data.

N. TAPEOP - Writes end-of-file on CLTAPE.

O. CLOUT - Outputs CL data to CL file.

P. TAPEWT - Forms CLTAPE records and outputs them via BUFFTP.

Q. PACK/UNPACK - Formats arrays read from or written to a direct access file.

R. REDPRT - Outputs error message from BUFFTP.

S. CHKN - Checks number of points to be read against array limits.

T. PRNTFW - Prints wing and fuselage input data on line printer for user verification.

U. RDFRM2 - Reads data when format 2 is specified.

V. WEED - Deletes CL points that are within tolerance.

W. TMARK - Puts APT code on CLTAPE.

X. CHK4PT - Checks to see if a bulkhead is a point.

Y. SPINDL - Puts APT code on CLTAPE.

Table I lists, in alphabetical order, each program element, its size, the elements it calls, and the elements which call it.

Description of Operation

READFW is the first subroutine called when AIRCL is started. This reads data and makes certain validity checks. Data come from two sources. The first is from cards (INPUT) and is accessed by Namelist PRMTRS. Namelist variables include declarations as to which part surface is to be cut, the type and amount of interpolation to be performed, the location of the set point, etc. All of these variables have predefined default values. The variables in the namelist are described in detail elsewhere.

The second source of input is from a file with the local file name (lfn) TAPE4. TAPE4 contains the actual data describing the wing and fuselage to be machined. There are seven different card-image formats in which data may be coded on TAPE4. These formats are described in detail elsewhere. The first elements read for any of the seven formats is how many points per section and how many sections follow for both the wing and fuselage. CHKN is used to test these values against the array sizes specified for input data. If the number of points or sections exceeds the number allowed, an error is generated.

It is preferable to have sufficient data such that the fuselage extends past the intersection with the wing and the wing extends into the fuselage. This is not a fixed requirement however. If the wing or fuselage is defined short of the W/F intersection curve, the appropriate data will be extended. This is done using a linear extension of the coefficients at the points or station closest to the intersection curve. These coefficients will be based on the type of interpolation specified by the user for the direction in question.

After reading the data, if a bottom surface is to be cut, the Z axis data are inverted. The Y data values are then set to their absolute values to ensure the data lie above the X axis; that is, the data must represent the right-hand side of the model. The Y data values for bulkheads also are tested to ensure that they do not curve back toward the Y axis, thereby creating an undercut situation. CHK4PT is used to determine if the first or last fuselage bulkhead is a point. If it is, this condition would cause problems later in the program, so a point bulkhead is replaced by a circular bulkhead with a very small, but finite radius. When machined, this circle is small enough so that it will still appear to be a point. Finally, PRNTFW is called if the proper print option has been set. PRNTFW lists the input data and the slope at each point for the user's verification.

FSLGCL is called next. This generates the cutter offsets for the fuselage. To do this, DIVIDE and INTPLT are used to interpolate each bulkhead into as many pieces as was specified in the input Namelist. While doing this, the perimeter of each bulkhead is found. The maximum perimeter is used to determine the number of machining passes which will eventually be required to yield a surface with an inter-pass scallop height less than or equal to that specified in the Namelist. The effect of residual scallop height to the spacing between machining passes is illustrated in Figure 3. Table II shows the number of passes required per linear inch of surface for various cutter sizes to produce different scallop heights. It is interesting to observe that when going from 0.050 inch scallop height to 0.001 inch, a 50:1 decrease, the number of passes required increases by only about 6.5:1. If the number of passes required to produce the surfaces desired is greater than 1600, an error message is generated and the program halts.

FSLGCL next determines the required position of the cutter in order to machine the interpolated surface. This is done by finding two vectors tangent to the surface at each point. The vector normal to the surface at the point is the cross product of the two tangent vectors. The center of the cutter lies on the normal vector a distance of one cutter radius away from the true surface point. The coordinates used for the tangent vectors are the points which lie on either side of the point in question. This is shown in Figure 4. If the point in question is a terminal point of either a section or a pass, the point itself is substituted for the non-existent adjacent point. The coordinate values after interpolation and after cutter offset are printed if the appropriate print variable was set in the Namelist.

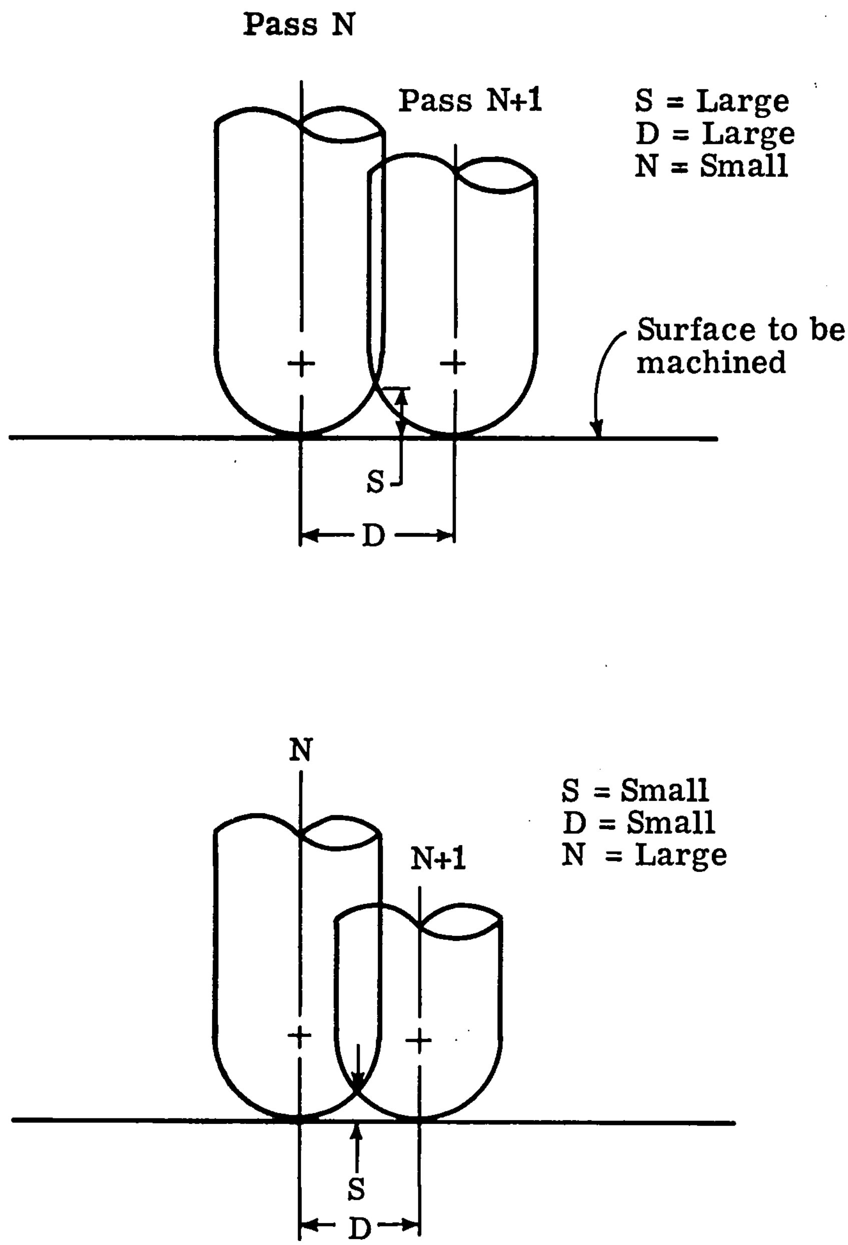

FIGURE 3. EFFECT OF PASS SPACING (D) ON RESIDUAL SCALLOP HEIGHT (S)

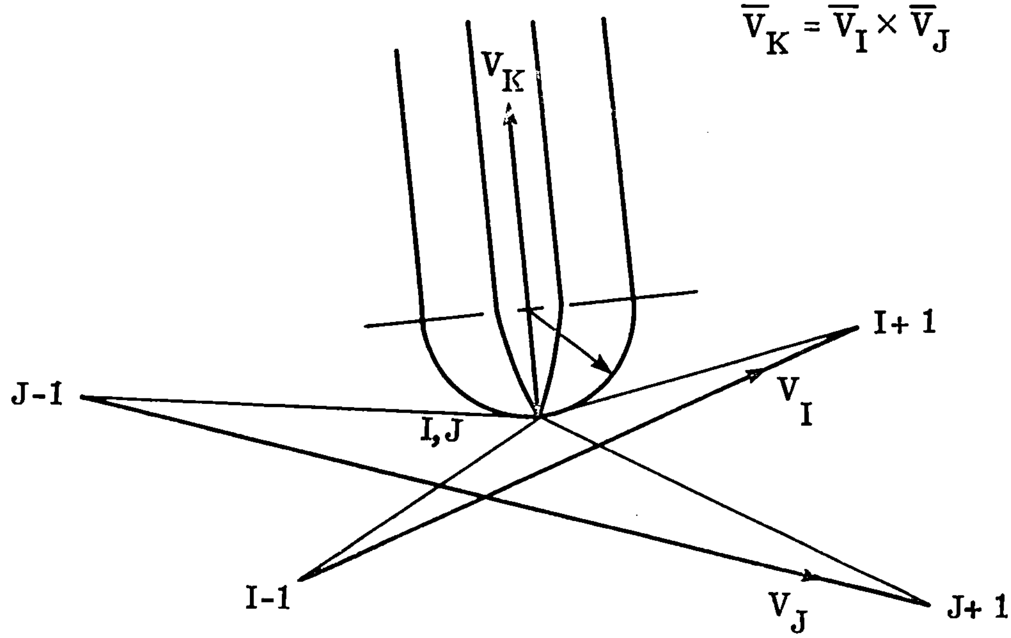

FIGURE 4. FINDING CUTTER OFFSET BY VECTOR CROSS PRODUCT

WINGCL is called next. This does the same thing for the wing that FSLGCL did for the fuselage. When it starts, however, WINGCL tests the data to ensure that the first point on each wing station has the smallest X value of any point on the station. If the first point does not have the smallest X coordinate, points are eliminated until the point with the least X value is found. This is shown in Figure 5. On the upper surface, the minimum X coordinate is at point K. Thus, points 1 through K-1 would be eliminated from the data for the upper wing. It should be noted that although these points are eliminated from the upper wing surface, they are not added to the lower wing surface. The test to insure that the data for each wing section increase monotonically in the x-direction is only conducted until any point passes the test; from then on the test is bypassed for that section.

WINGCL uses GOUGE to determine if the surface is concave or convex at each cutter offset location. If the surface is concave, the cutter will gouge the surface if it is offset along the normal by only the cutter radius. To prevent gouging, the cutter offset is made larger than the cutter radius specified. This condition is shown in Figure 6. With the cutter offset by more than the cutter radius, the surface will be cut tangent to the lines which intersect at the surface point. Uncut material will remain at the point, but the surface will not be undercut. Once WINGCL and FSLGCL are completed, there is no further need for the actual surface data, except for the last fuselage pass which is used for trim calculations. Data for the last fuselage pass and the wing leading and trailing cutter offsets are saved in Common/FSWGSV/. Common/SURFG/ which contained the original surface data is then used for a variety of storage purposes in subsequent sections of the program.

With the cutter offset location determined for the full fuselage and wing surfaces, the next step is to find the intersection of the wing and fuselage. This is done by subroutine INTRSC. As illustrated in Figure 7, a trial bulkhead, (X_I), which intersects the wing pass at Y = 0.0 is found by interpolation. The X value at X_I is compared to the X value of the wing pass and trial bulkhead intersection as seen in the Y-Z bulkhead. If the two X intersection values are equal, within a defined tolerance, the intersection point found lies on the true wing/fuselage intersection. If the two X intersections values are not equal, a new bulkhead, (X'_I), is generated at the X coordinate of the W/F intersection found in the Y-Z projection. This process continues until the X values converge to the true intersection point, or the limit on the number of trials is reached. The limit on the number of

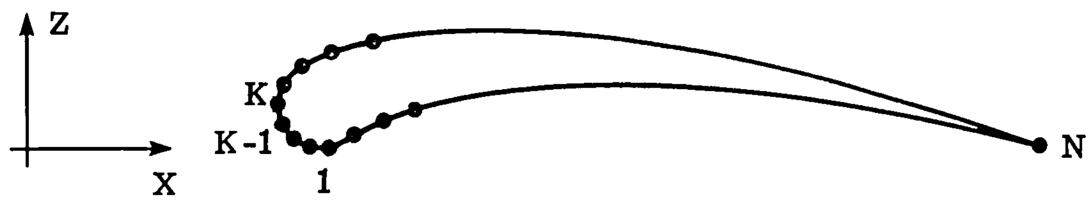

FIGURE 5. MAKING WING COORDINATES MONOTONIC IN X

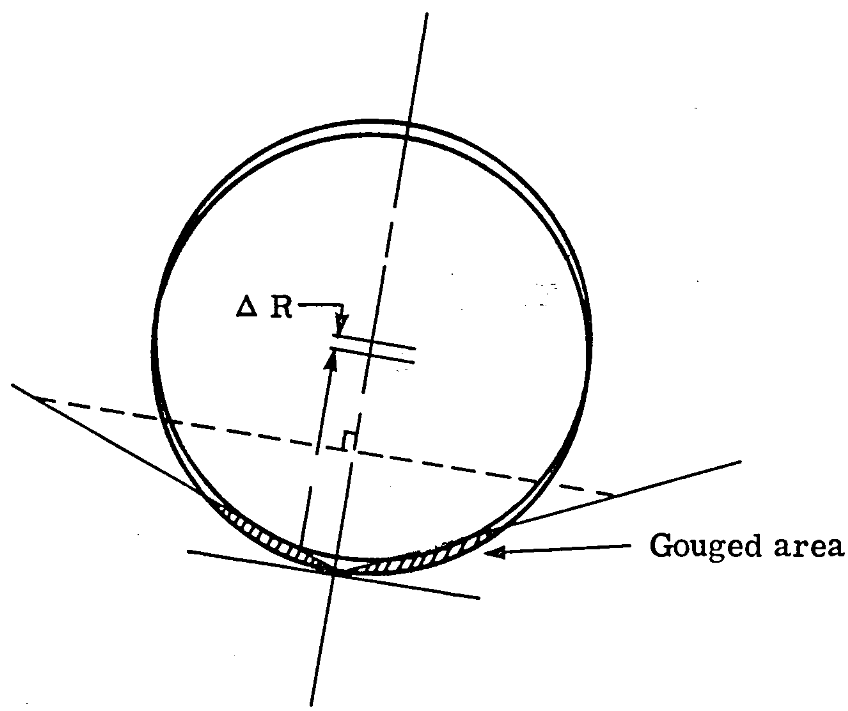

FIGURE 6. ELIMINATION OF SURFACE GOUGE

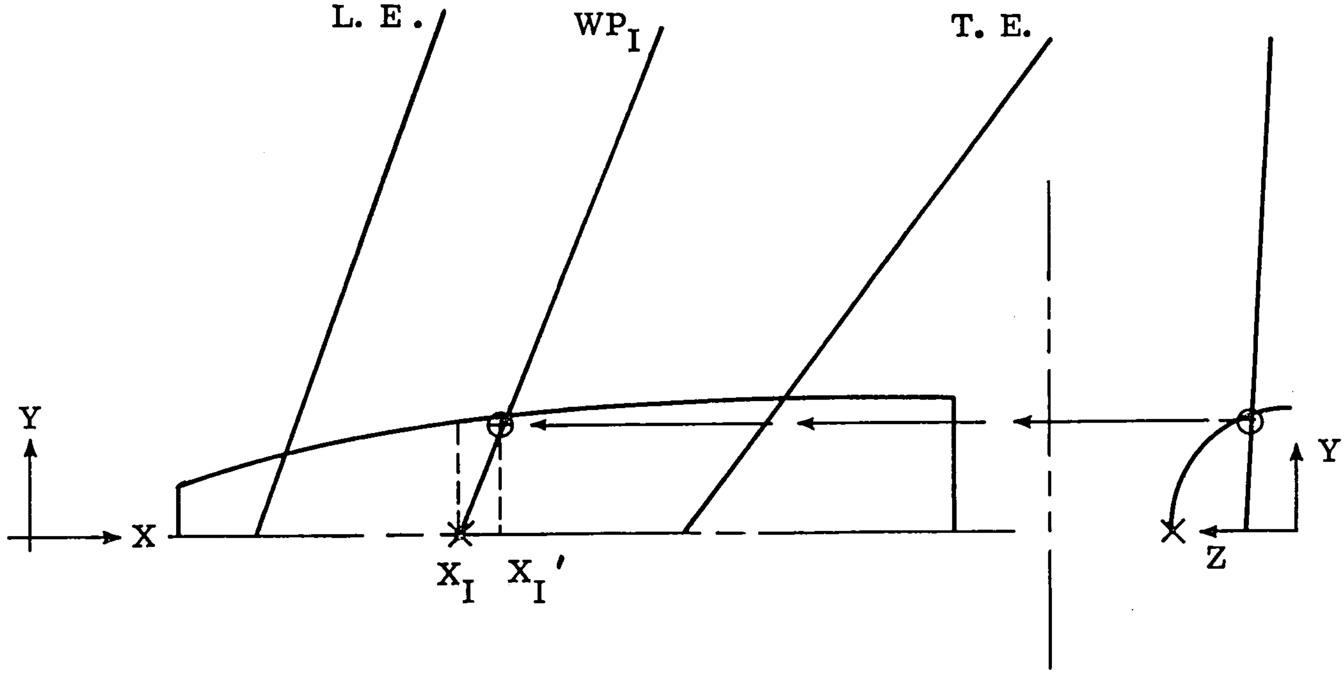

FIGURE 7. FINDING THE WING-FUSELAGE INTERSECTION

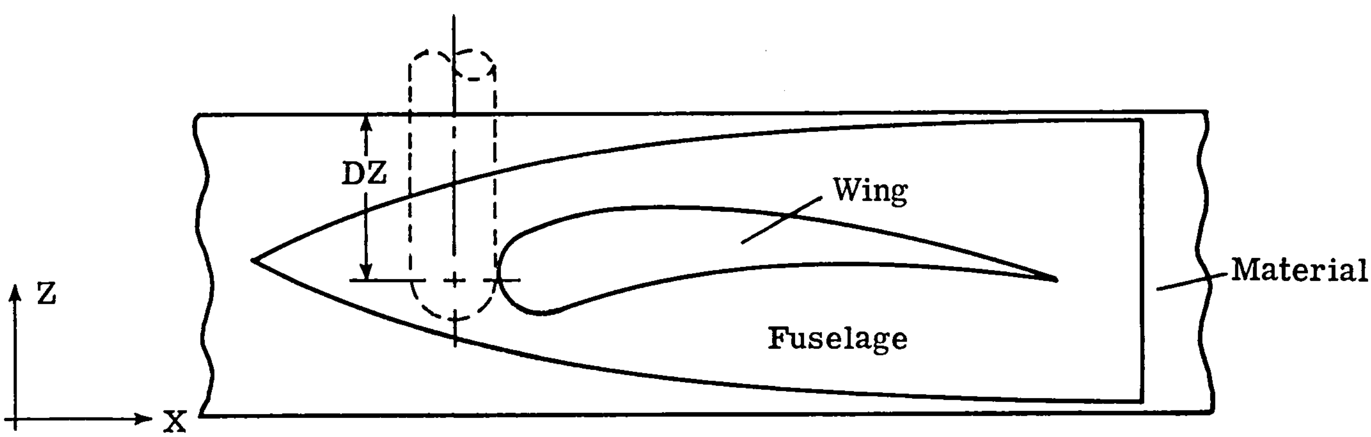

FIGURE 8. NEED FOR MULTIPLE PASSES AT THE LEADING EDGE

trials is provided to prevent being caught in an infinite loop should the intersection algorithm fail for some reason.

Once the W/F curve of intersection is found, FCLPTS and WCLPTS are called to generate the actual cutter locations necessary to machine the fuselage and wing, respectively. FCLPTS uses the intersection curve generated by INTRSC to bound the CL data for the entire fuselage which were generated by FSLGCL. That is, FCLPTS sorts out from the points which define the entire fuselage, those points which lie outside (above) the W/F intersection. In addition, FCLPTS expands the data by interpolation to provide both additional fuselage passes (more points on each bulkhead) and additional points along each pass (more bulkheads).

In operation, FCLPTS starts by expanding the number of points on each bulkhead. This is done to generate the number of passes required to meet the residual scallop height specified by the user. If the number of passes required is greater than 150, a direct access, word addressable scratch file is established. The expanded bulkheads are generated by using DIVIDE and INTPLT. The bulkhead points are generated and written to the file, one bulkhead at a time. When the data are read back in, it is done streamline by streamline. That is, when processing streamline N, the Nth point on each bulkhead is read from the file.

When the data for the desired streamline are prepared, either from in-core arrays or from the scratch file, the streamline is then expanded, if required. This expansion would be based on the specification given by the user as NCLF in the Namelist. Working forward from the nose, the leading intersection of the streamline and W/F curve is found. Likewise, working backward from the tail, the trailing intersection of the streamline and W/F curve is found. Arrays are then formulated which contain the streamline up to the leading intersection, the W/F intersection up to the trailing intersection, and then the remainder of the streamline. Special conditions are tested for and handled, such as when the fuselage is completely within the wing, or when the Z coordinate on the streamline at the intersection with the W/F curve is below the Z coordinate of the W/F curve itself. The resulting data are stored on a file for subsequent formatting to file CLTAPE. The order of writing to this file is reversed after each streamline is processed to provide the back and forth motion actually used in cutting. The last step in FCLPTS is to make one pass from nose to tail along the W/F intersection curve itself. This eliminates the steps on the curve which may be generated by the first part of FCLPTS.

WCLPTS is similar to FCLPTS in that WCLPTS limits the wing data to be cut to those points on the wing which lie outside of the fuselage, as bounded by the W/F intersection curve. WCLPTS also expands the data by interpolation both along all chords, and along the "stringers" connecting the same point on each curve. A maximum of 1900 wing passes may be generated. Using the wing data generated by WINGCL, after expansion as required, the intersection of each wing pass with the W/F intersection curve is determined. The intersection point itself and all points outboard of the intersection curve are written to a temporary file. This file is subsequently processed to generate the actual CL format output. The direction in which points are written to this file is reversed at the end of each pass to provide the back and forth motion desired when actually machining.

The first pass generated by WCLPTS has a special feature. Because the first pass on the wing is at the leading edge, and because the leading edge probably lies about at the center of the fuselage, in the X-Z plane, multiple passes are generated on the first pass. As shown in Figure 8, the depth of cut required at the leading edge can be considerable. Further, since this occurs on the very first pass, this depth of cut is required across the full width of the cutter. This could lead to excess tool loading, and as a consequence, tool damage or breakage. To eliminate this problem, the depth of cut at the leading edge is divided into a series of passes, each one below the previous pass. The distance between adjacent passes is specified by the user using variable ZSTEP in the Namelist input. After the leading edge pass is cut to its final depth, subsequent wing passes are cut to the full depth required for each. However, for these subsequent passes, the distance between passes is expected to be considerably less than the cutter size, so the tool loading is considerably less than it would be at the leading edge without multiple passes.

Following WCLPTS and FCLPTS, TRIMCL is used as specified by the user, to provide a trim cut around the X-Y plan view of the model. This trim cut starts at the leading point of the fuselage, or the center of the wing, if the wing lies completely ahead of the fuselage. It then proceeds along the fuselage until the intersection of the fuselage and leading edge is found, then out along the leading edge to the tip. The tool is retracted to the clearance plane and moved to the outboard tip of the trailing edge. This prevents cutting of the wing tip as a plane parallel to the X-Z plane which the tip most likely is not. The tool is then moved back towards the center line along the trailing edge, finding the W/F intersection as required.

TRIMCL is the last program element of AIRCL which is involved with analyzing the geometry of the model and formulating cutter positions. The actual last element of AIRCL is WRITCL. This takes the data written to temporary file NCDATA by FCLPTS, WCLPTS, and TRIMCL, and writes the actual output file. This output file is formatted to emulate the CL file format which would be produced by APT Section 2. Thus, CL data produced by either APT or AIRCL can be post-processed in the same way. The post-processor program produces the NC tape which is then run on a specific NC machine tool.

Data Input

There are two types of data input to AIRCL. The first type tells AIRCL what to do and how to get started. These data are read from card input when AIRCL is executed, and are input using CDC Fortran "Namelist" practices. The second type of data is the data which actually describes the fuselage and wing to be cut. These data are expected to be on a file with the local file name of TAPE4. The following sections describe these data types in detail.

I. Namelist Data

These data are input on cards. A '$' is punched in column 2 of the card, followed by the name of the Namelist. In the case of AIRCL, the name of the Namelist is PRMTRS. A space is skipped, and then the name of each variable which is to be set is given, followed by an '=' and the value to be given the variable. If a variable is not listed, its default value is used. As many cards as are necessary may be used to input the variables. After the final variable, another '$' is punched to indicate the end of data input.

The following variables may be input to AIRCL via Namelist PRMTRS.

A. INTYP - The type of interpolation to be used at various parts of the model. INTYP is a 10-element array, of which the first five elements are used as follows:

(1) INTYP(1): On the wing along the chords (X-Z plane).
(2) INTYP(2): On the wing from centerline to tip (X-Y plane).
(3) INTYP(3): The fuselage bulkheads (Y-Z plane).
(4) INTYP(4): The fuselage streamlines (X-Z plane).
(5) INTYP(5): The fuselage-wing intersection curve.

The actual values given to INTYP have the following meaning: If INTYP(I) =

0: Spline interpolation, if four or more points. If less than four points, linear interpolation is substituted.

1: Linear interpolation

2...N: Polynomial interpolation with order equal to value specified.

The following default values are used:

(1) INTYP(1) = 0 (wing along chords, spline)

(2) INTYP(2) = 0 (wing, center to tip, spline)

(3) INTYP(3) = 0 (fuselage along bulkheads, spline)

(4) INTYP(4) = 0 (fuselage streamlines, spline)

(5) INTYP(5) = 1 (W/F intersection curve, linear)

For INTYP(1) only

-99 will cause the program to skip the initial interpolation of input points in WINGCL, and do spline interpolation for increasing the number of CL paths to satisfy the scallop height criteria.

-1 skips initial interpolation of input points for the wing, and then uses linear interpolation to satisfy the scallop height criteria.

-2...-N skips initial interpolation of input points for the wing, and then uses Nth order interpolation to satisfy the scallop height criteria.

Use of a negative value for INTYP(1) for formats 5, 6, and 7

Because for input formats 5, 6, and 7 the wing can have different numbers of points for the various sections, the program calculates an equal number of points (NPSW) for each section in the initial interpolation routine. Since the initial interpolation is skipped if INTYP(1) is negative, the program will interpolate the input sections so that they each have the same number of points as the input section with the largest number of points, and at the same percent chord values as that input section.

B. NPSW - The number of interpolated points to be generated at each wing section. (i.e. the number of wing stringers). Maximum value = 150; default value = 100.

C. NPSF - The number of interpolated points to be generated at each fuselage section. (i.e. the number of fuselage stringers). Maximum value = 50; default value = 25.

D. SETPT - The starting setpoint of the tool, relative to the origin of the data. The setpoint is measured to the end of the cutter, not the center of the ball. The SETPT is a three-element array so the setpoint can be specified relative to each axis. Default value = 0.0, 0.0, 0.0.

E. FDRATE - The feed rate to be used for machining the model. Default value = 10 ipm.

F. CUTRAD - The radius of the cutter to be used to machine the model. Default value = 0.50 inch (1.0 inch diameter).

G. RAPID - The rapid traverse speed, ipm. Default value = 60 ipm.

H. WING - A logical variable indicating whether or not machining passes are to be generated along the wing. Default value = T (true).

I. FUSLG - A logical variable indicating whether or not machining passes are to be generated along the fuselage. Default value = T (true).

J. TRIM - A logical variable indicating whether a machining pass is to be made to trim the model. This would profile the model in the X-Y plane. Default value = F (false).

K. UPRSRF - A logical variable indicating that upper surface data are to be read and processed. Default value = F (false).

L. LWRSRF - A logical variable indicating that lower surface data are to be read and processed. Default value = T (true).

NOTE: UPSRF and LWRSRF are mutually exclusive. They cannot both be simultaneously true or false.
If specified to be the same, AIRCL will assume the UPRSRF = T, and LWRSRF = F.

M. IPRNT - This is a 10-element array used to control the printing of intermediate results. These values are primarily intended to be used as a debugging aid, and would not be used in normal operation of AIRCL. The particular variable used indicates the portion of the program for which intermediate results are to be printed. The variable assigned to each portion of AIRCL is as follows:

IPRNT(1)	Not used
" (2)	READFW
" (3)	WINGCL
" (4)	FSLGCL
" (5)	INTRSC
" (6)	FCLPTS
" (7)	WCLPTS
" (8)	WRITCL
" (9)	TRIMCL
" (10)	Not used.

The following values are used for each element of IPRNT:

IPRNT(2): If ≠ 0, prints Namelist value and input data*

*The Namelist is always printed out.

IPRNT(3) through IPRNT(9): If = 0, no printed output

If = 1, prints name of subroutine when entered

If > 1, prints various intermediate results applicable to the subroutine.

Default value for all elements = 0.

Even if none of the IPRNT variables are set by the Namelist, the following messages will always be printed:

- XXXX FUSELAGE PASSES WILL BE GENERATED IN FCLPTS. This message is printed if the number of fuselage passes required based on scallop height is greater than the value for NPSF. The larger of these two is the one which is used.
- YYYY WING PASSES WILL BE GENERATED IN WCLPTS. This message is printed if the number of wing passes required based on scallop height is greater than the value for NPSW. The larger of these two is the one which is used.
- COMPLETED CLTAPE FOR PARTNO ----- (uses part name entered with coordinate data)

A TOTAL OF ZZZ RECORDS WERE WRITTEN. (To CLTAPE).

N. SCALE - This allows the data to be scaled up or down as desired. If SCALE > 1.0, the data are enlarged to cut a model larger than specified by the data. Default value = 1.0.

O. ZSTEP - The size of the step to make in the Z-axis when making the first pass at the leading edge of the wing. The total depth required from the set point to finish depth will be divided by the value for the step size to determine the total number of cuts to be generated to reach final depth. Default value = 0.200 in.

P. SCLPHT - The height of the scallop which is to be left between adjacent machining passes. The smaller this value is, the closer the surface is to a true, smooth surface, but the more passes will be generated. Default value = 0.010 in.

Q. NCLW - The number of points generated by interpolation on the wing, from centerline to tip. The default is the number of wing sections in the input data.

R. NCLF - The number of points generated by interpolation on each fuselage streamline. The default value is the number of fuselage sections in the input data.

S. CLRPLN - coordinate of clearance plane. If not declared, SETPT(3) will be used.

T. XYANGL, YZANGL, ZXANGL - Rotation angles on respective coordinate planes, to be entered in degrees. Default values are zero.

U. TOLRNC - Tolerance for all wing and fuselage passes. If specified, points within tolerance are deleted. The result is a shorter NC tape. Default value is 0.0

A typical Namelist might appear as follows:

b$PRMTRS UPRSRF=F, LWRSRF=T, SETPT = 0.0, 0.0, 2.0, NPSF = 25, NPSW = 50, TRIM=T, IPRNT(2)=9 $.

This would process the data for the lower surface, generate 25 fuselage streamline passes, 50 wing stringer passes, and a trim cut. The set point would be 0., 0., 2, and the Namelist and input data would be printed.

Following the Namelist data on file INPUT (cards) is a NC machine designation data card. This would have two items on it.

A. Postprocessor name (i.e. BENDIX, SUNTRN, etc.) (Format A6, CC 1-6) Default=SUNTRN

B. Machine number (F10.4, CC 11-20) Default = 2.0.

Following the postprocessor card, wing stringer interpolation specifications are expected. If missing, stringer wise interpolation is performed according to NCLW and INTYP(2). Interpolation between wing sections can be specified to be spline or linear. The number of points to be calculated for an interval as well as interval bounds are variable. The first interval is assumed to start at the first wing section. Each interpolation data card contains three parameters.

1) Cross section at which the interpolation ends.
2) Interpolation type, "L" for linear, "S" for spline.
3) Number of points to be calculated.

The format of the card is (3X,I2,4X,A1,2X,I3). Spline interpolation requires at least four sections; otherwise, it will revert to linear interpolation. As an example, let's assume that we have a wing composed of 20 sections, and that between sections 7 and 14 we need spline interpolation; whereas, linear interpolation is sufficient for the rest. The interpolation data cards might read as follows:

```
 7  L  30
14  S  50
20  L  25
```

That is between sections 1 and 7 we would like to have 30 points calculated with linear interpolation. Between sections 7 and 14 we would like to have 50 points calculated with spline interpolation. And between sections 14 and 20, we would like to have another 25 points calculated via linear interpolation.

Please note that in order for the wing interpolation data cards be read in properly, the postprocessor card must appear after the namelist PRMTRS.

An example of a complete INPUT DECK is shown in figure 9.

```
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45
 $PRMTRS LWRSRF=F, IPRNT(2)=2, SCLPHT=.005,
 FDRATE=30., CUTRAD=.75, TRIM=F$
BENDIX    3.0
```

FIGURE 9. EXAMPLE NAMELIST AND MACHINE SPECIFICATION DATA

The Namelist parameters shown would develop a tape for the following conditions.

- Upper surface of model
- List input data
- Develop as many passes as necessary so residual scallop height is less than or equal to 0.005 inches
- Set the feedrate to 30 ipm.
- Use a ball end cutter with a 0.75 inch radius

The model will be cut on a tool with a #3 Bendix controller.

II. Section Data

Data for the actual model to be machined are expected to be on a local file named TAPE4. This data would be in card image format as follows.

A. Header Card

(1) IFRMT (CC 1-5, format I5) - The format to be used for reading the subsequent data cards. Valid range 1-7.

(2) PARTID (CC 11-80, format 9A8) - The part name, title, or other alphanumeric descriptive information.

B. Number of Data Points (4I5)

(1) NFPT (CC 1-5)- The number of points to be read for each fuselage section (bulkhead). Maximum allowable = 50.*

(2) NFSC (CC 6-10) - The number of fuselage sections for which data are to be read. Maximum allowable = 30.

(3) NWPT (CC 11-15) - The number of points to be read for each wing section (chord). Maximum allowable = 150.

(4) NWSC (CC 16-20) - The number of wing sections (chords) for which data are to be read. Maximum allowable = 30.

NFPT and NWPT should be left blank for formats 5, 6, and 7.

C. Data Points Data for the wing and fuselage may be read according to seven different formats. The format to be used is specified by IFRMT in the header card. The seven formats are as follows:

(1) IFRMT = 1

(a) For each fuselage bulkhead

1. X position of the fuselage bulkhead (F10.6)
2. Y, Z coordinates of each point on the bulkhead referred to by the previous card.

   If upper surface: format = 2F10.6, 20X

   If lower surface: format = 20X, 2F10.6

(b) For each wing chord

1. Y position of the chord (F10.6)
2. X, Z coordinates of each point on the chord referred to by the previous card.

   If upper surface: format = 2F10.6, 20X

   If lower surface: format = 20X, 2F10.6

(2) IFRMT = 2

(a) The X coordinates for all fuselage bulkheads (format 8F10.5)

*For Format 4, this is the number of points to be computed for the quarter-ellipse.

1. If upper surface

   a. The Y coordinates on the first upper bulkhead
      Format = 8F10.5

   b. The Z coordinates on the first upper bulkhead
      Format = 8F10.5

   c. Continue for the rest of the bulkheads.

   d. Then, skip the corresponding data for the Y
      and Z coordinates of the lower bulkheads.
      Format = 8F10.5

2. If lower surface

   a. Skip the Y and Z data for I points per section
      on J number of fuselage sections. Format = 8F10.5

   b. Y coordinates on the first lower bulkhead
      Format = 8F10.5

   c. Z coordinates on the first lower bulkhead
      Format = 8F10.5

   d. Continue for the rest of the bulkheads.

(b) The Y coordinate for all wing chords (Format = 8F10.5)

1. If upper surface

   a. The X coordinates on the first upper chord
      Format = 8F10.5

   b. The Z coordinates on the first upper chord
      Format = 8F10.5

   c. Continue for the rest of the chords

   d. Skip corresponding X, Z data for I points per
      chord on J lower chords. Format = 8F10.5

2. If lower surface

   a. Skip X, Z data for I points per chord on J upper
      chords. Format = 8F10.5

   b. The X coordinates on the first lower chord
      Format = 8F10.5

   c. The Z coordinates on the first lower chord.
      Format = 8F10.5

   d. Continue for the rest of the chords.

(3) IFRMT = 3

(a) X coordinate of each fuselage bulkhead for all
    bulkheads (format 8F10.5)

1. Y,Z coordinate pairs for each point on each bulkhead.

   2 pair per card

   a. If upper surface; format = 2F10.5, 20X, 2F10.5, 20X

   b. If lower surface, format = 20X, 2F10.5, 20X, 2F10.5

(b) Y coordinate of each wing chord for all chords (format 8F10.5)

1. X,Z coordinate pairs for each point on each chord. Two pair per card.

   a. If upper surface, format = 2F10.5, 20X, 2F10.5, 20X

   b. If lower surface, format = 20X, 2F10.5, 20X, 2F10.5

(4) IFRMT = 4 - This applies to the fuselage only. This format allows fuselage bulkheads to be entered as parametric ellipses with the actual points on each bulkhead generated internally. The format for the wing is the same for IFRMT = 3 or 4. When IFRMT = 4, the fuselage is read (in format 4F10.6) as:

(a) The X position of each fuselage bulkhead.

(b) The Y-axis width of the fuselage bulkhead.

(c) The Z-axis height of the fuselage bulkhead.

(d) The offset of the ellipse along the Z axis from the X-Y axis.

Note: 1. If Y = Z, the bulkhead is a circle.

2. If Y = Z = 0, the bulkhead is a point.

(5) IFRMT = 5 - This format is for a single surface only.

(a) For each bulkhead

1. The X position of a bulkhead, and the number of points on this bulkhead (maximum of 50 points per bulkhead) (F10.4,I5)

2. The Y coordinates of the points on this bulkhead (7F10.4)

3. The Z coordinates of the points on this bulkhead (7F10.4)

(b) After all bulkheads are defined, wing data are entered as follows for each wing station:

1. The Y position of a wing station, the number of points on the wing at this station (maximum = 150), and the X-axis reference dimension. The reference dimension is added to all subsequent X-values. This permits the wing station data to be defined along the X-axis in a relative manner, with the leading edge typically being at X = 0.0 (F10.4, I5, F10.4).

2. The X coordinates of the points on this wing station, absolute or relative to the reference dimension (7F10.4).

3. The Z coordinate of the points on this wing station (7F10.4).

(6) IFRMT = 6

(a) For each bulkhead

1. The X position of the bulkhead and the number of points (upper or lower surface) on the bulkhead (maximum = 50) (F10.4, I5)

2. The Y coordinates of the points on the bulkhead (7F10.4)

3. The Z coordinates of the points on the upper portion of the bulkhead (7F10.4).

4. The Z coordinates of the points on the lower portion of the bulkhead (7F10.4).

Note: The number of Z coordinates must be the same on both the upper and lower portion of a given fuselage bulkhead.

(b) After all fuselage points are defined, the wing data are entered as follows for each wing section:

1. The Y position of a wing station, the number of points on the station (maximum = 150), and the X-axis reference dimension. (F10.4, I5, F10.4).

2. The X coordinates of the points on the wing station, absolute or relative to the reference value (7F10.4).

3. The Z coordinates of the upper wing surface (7F10.4).

4. The Z coordinates of the lower wing surface (7F10.4).

Note: The number of Z coordinates must be the same on both the upper and lower surface of a given wing station.

(7) IFRMT = 7.

(a) For each bulkhead

1. The X-position of the bulkhead, and the number of points (upper or lower surface) on the bulkhead. (F10.4, I5).

2. The Y coordinates of points on the upper portion of the bulkhead (7F10.4).
3. The Z coordinates of points on the upper portion of the bulkhead. (7F10.4).
4. The Y coordinates of points on the lower portion of the bulkhead. (7F10.4).
5. The Z coordinates of points on the lower portion of the bulkhead (7F10.4)

   Note: The number of Y and Z coordinates must be the same on both the upper and lower surface at a given bulkhead.

(b) After all the fuselage points are defined, the wing data are entered as follows for each wing station:

1. The Y position of a wing station, the number of points on the station (maximum = 150), and the X-axis reference dimension. (F10.4, I5, F10.4).
2. The X coordinates (absolute or relative) of the points on the upper wing surface. (7F10.4).
3. The Z coordinates of the points on the upper wing surface. (7F10.4).
4. The X coordinates (absolute or relative) of the points on the lower wing surface. (7F10.4).
5. The Z coordinates of the points on the lower wing surface. (7F10.4).

   Note: The number of X and Z coordinates must be the same on both the upper and lower surface at a given wing station.

The following conventions apply to all input formats:

Data must be ascending order. For the fuselage, this means the X coordinate of the bulkheads must be in left to right order. At each bulkhead, the data are to be in ascending order by Y. That is, a bulkhead is described from top (or bottom) to the midpoint. For wings, the Y coordinates run from the center to the tip, in ascending order. Wing stations are then defined from least X to greatest X.

It is generally assumed that the nose of the plane is towards the left side and the tail towards the right. This is not a requirement, however, and the nose can be either on the right or left side.

Data are always assumed to lie in the first or second quadrant. That is, the data are symmetrical about the X axis, and all Y values are positive.

Fuselage sections are defined in the Y-Z plane only, with each section at a unique X coordinate. Wing sections are defined in the X-Z plane, with each station at a unique Y coordinate.

Figure 10 shows the generalized geometry of a fuselage. There are NFPT points on each bulkhead, and NFSC bulkheads. Within AIRCL, fuselage geometry is referenced by two-dimensional arrays, i.e., FSLG(I,J). I refers to a particular point on some fuselage bulkhead. The range of I would be from 1 to NFPT. J refers to a specific bulkhead and would run from 1 to NFSC. Figure 11 illustrates the geometry for a wing. There are NWPT points on each wing chord and NWSC chords. Wings are also described by two-dimensional arrays, i.e. WING (I,J). I refers to the points on a wing chord, and J refers to the chords themselves.

A composite view of a wing and fuselage is shown in the X-Y plane, in Figure 12. Several points mentioned above should be reiterated:

- The model is symmetrical about the X-axis, and all elements are described by positive Y values. If there are points with negative Y values, the absolute value of the Y component will replace the original value.
- Z coordinates are entered with the sign appropriate for their position relative to the origin of the data. When a bottom surface is to be machined, all Z coordinates are inverted automatically.
- The X and Z components of the origin of the data may lie anywhere along these axes, respectively.
- The nose of the model does not have to lie to the left, relative to the wing. However, all references to nose and tail, or leading and trailing edges assume the nose to be on the left.
- The leading edge of the wing must lie to the left.

Figure 12 also illustrates an important restriction in AIRCL which must be adhered to. That is, the first bulkhead must lie completely ahead of or completely behind the leading edge. That is, the leading edge cannot

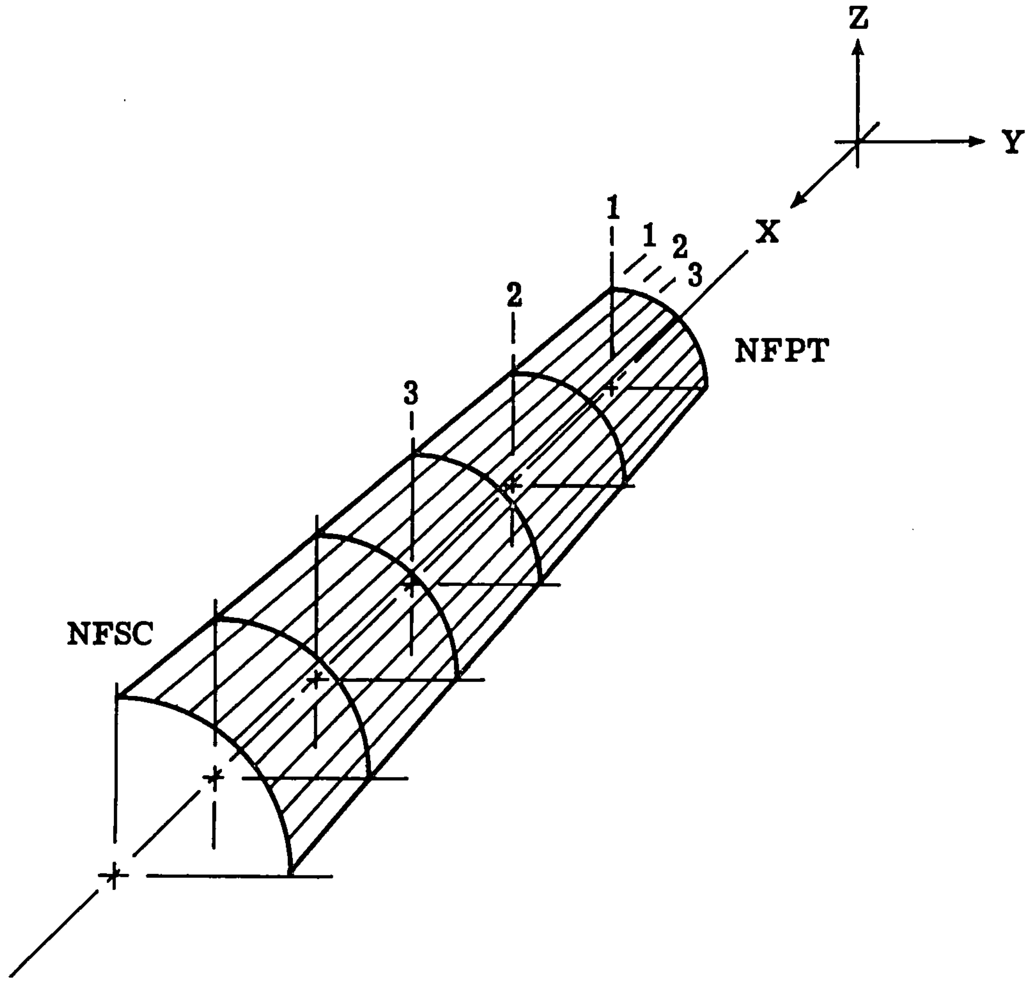

FIGURE 10. FUSELAGE NOMENCLATURE

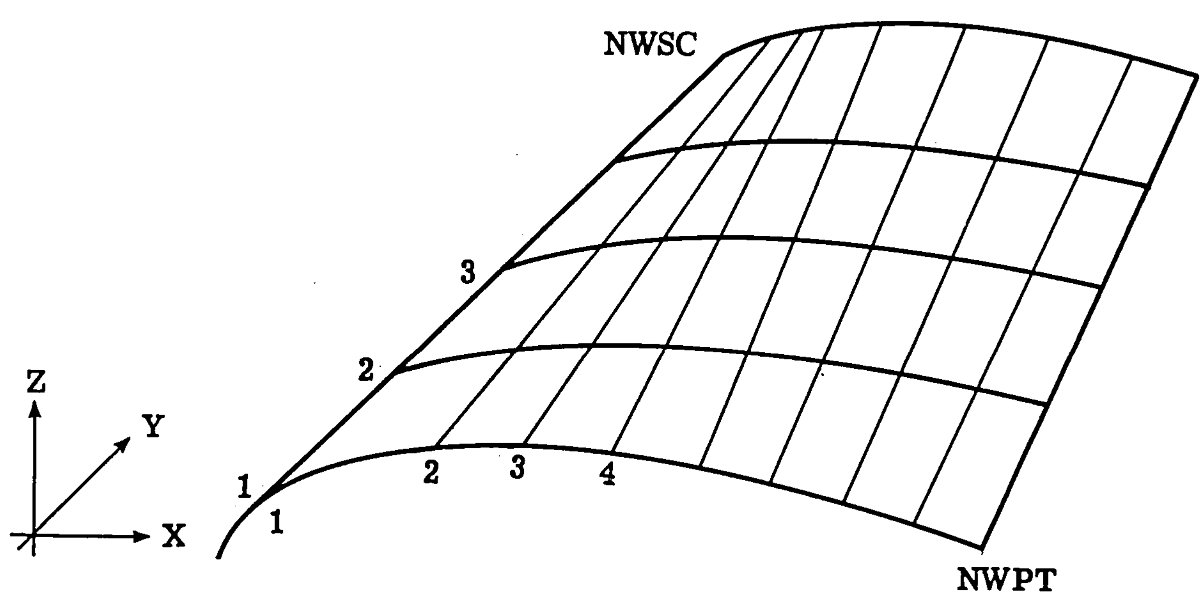

FIGURE 11. WING NOMENCLATURE

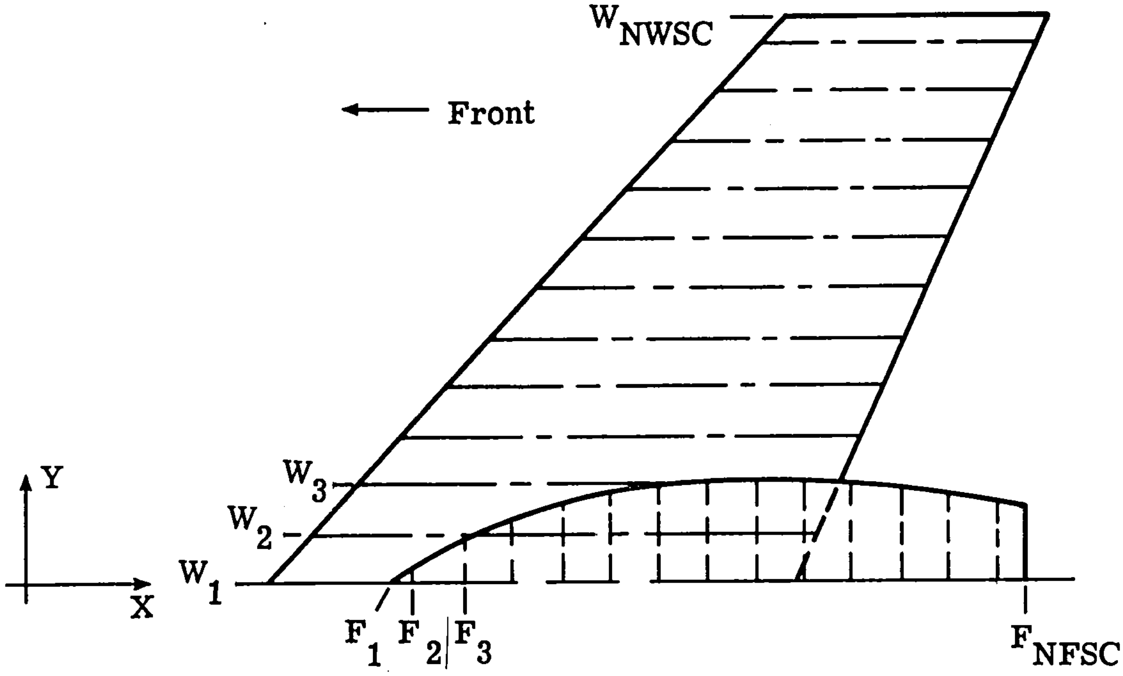

FIGURE 12. PLAN VIEW OF WING/FUSELAGE GEOMETRY

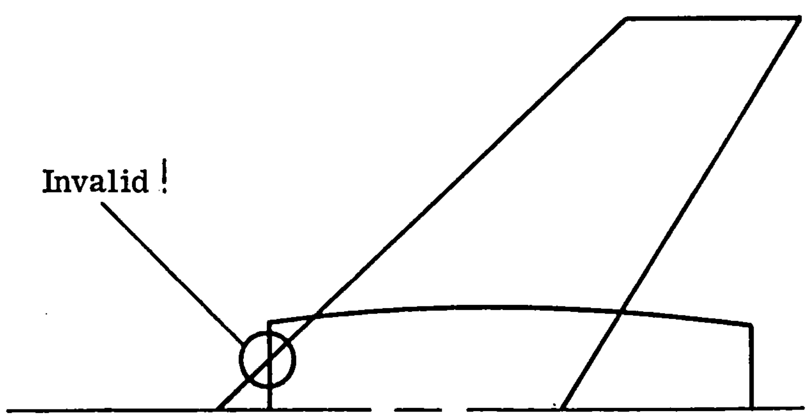

FIGURE 13. GEOMETRY RESTRICTIONS

intersect the first bulkhead. Likewise, the trailing edge must lie completely ahead of or completely behind the last bulkhead. An invalid arrangement is shown in Figure 13 where the leading edge intersects the first bulkhead.

Using AIRCL

The following description of how to use AIRCL is divided into two parts. The first part is for the NC part programmer who wants to use AIRCL to generate a tape for machining a model. The second part is for a systems programmer who wants to revise or add new code to AIRCL.

Executing AIRCL

For the NC part programmer, it is assumed an absolute load module of AIRCL exists with the name AIRABS. The procedure he then would use is as follows (Figure 14):

```
  Job cards, account number, etc.
  T50, CM150000 should be adequate
GET(AIRABS) -- Attach absolute file
GET(TAPE4=AIRDAT)               Attach data from disk or cards;
[COPYBR(INPUT,TAPE4)]
[REWIND,TAPE4.      ]           use one or the other.
AIRABS.  Execute AIRCL with input from TAPE4; output
         will go to CLTAPE and printer
RETURN, AIRABS.  Release AIRCL
REWIND,CLTAPE
LABEL(APTABS,NT,D=1600,VSN=ND1051,FI=APTABSFL,PO=R,FA=H)
REWIND,APTABS.
COPYBF,APTABS,APT.
RETURN,APTABS.
REWIND,APT.
     Use 5 lines above if loading APT from magnetic tape; otherwise
GET,APT.
RFL,100000.  Re-set memory required.
APT.  Execute APT, input from CLTAPE, output to printer & TAPE66.
RETURN, APT.  Release APT file.
REWIND,TAPE66.
```

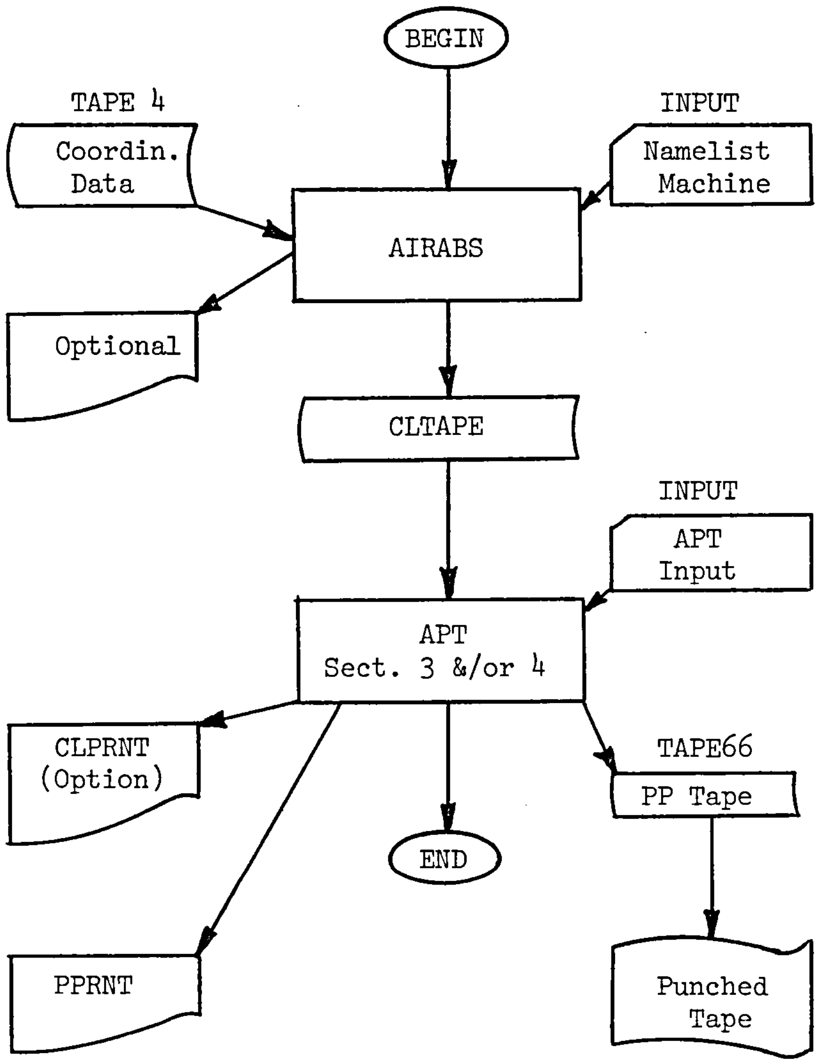

FIGURE 14. AIRCL AND APT EXECUTION PROCESS

```
  PTP(TAPE66).  Dump APT output to punched tape.
EXIT.
7/8/9 (EOR)
     Coordinate input to AIRCL, if from cards
7/8/9 (EOR)
     $PRMTRS.......(namelist input to AIRCL)
     Post processor name and number
7/8/9 (EOR)
PARTNO.....
MACHIN/.....
CLPRNT (if desired)      input to APT
PTONLY/2
FINI
6/7/8/9 (EOF)
```

```
PARTNØ          THIS IS A DUMMY TITLE
MACHIN/CINACI,203,LINEAR
CLPRNT        Optional
PTØNLY/2
FINI
```

FIGURE 15. EXAMPLE INPUT TO APT SECTIONS 3 AND 4

The input required by APT is shown in Figure 15. This takes CLTAPE, the output of AIRABS, and processes it through section 3 and/or 4 of APT. If CLPRNT is requested, APT Section 3 is used to list CLTAPE. The MACHIN and PARTNO statements shown in Figure 15 are required by APT even though this information is already on the CLTAPE. No part name, number, etc. has to follow the PARTNO statement: The machine make and number on the MACHIN statement likewise do not have to be the same as were included in the input to AIRABS. These statements must be included, however, to present the proper syntax to APT.

Updating AIRCL

When creating or modifying the AIRCL program itself, four files will be used. These are:

1. AIRSRC - The source file of AIRCL in update format; i.e. COMDECK's are declared at the beginning and then CALL'ed where needed.
2. AIRNPL - The program library created by using CDC's UPDATE.
3. AIRRLB - The relocatable binary file of compiled AIRCL segments.
4. AIRABS - The load image, absolute file of AIRCL; this is what is executed by the NC programmer to generate CLTAPE. CLTAPE is the input to APT.

To create AIRNPL, AIRRLB, and AIRABS for the first time, starting from AIRSRC, the following procedures are used:

```
  Job cards, account number, etc.
  T50, CM150000
GET(AIRSRC)
UPDATE(F,N,I=AIRSRC)  Make full update and create new program
  library, NEWPL, and COMPILE file.
REPLACE(NEWPL=AIRNPL)  Save new program library
REWIND,COMPILE.
RETURN, NEWPL,AIRSC.  Release files
FTN(I=COMPILE,B=AIRRLB,R=3)  Compile Fortran modules
REPLACE(AIRRLB).  Save relocatable binary.
MAP,PART.
SEGLOAD(I=INPUT,B=AIRABS)  Use Segmentation Loader with
  loader directives from cards (input).  See Table III.
  Create AIRABS as absolute load image file (output)
LOAD(AIRRLB)
NOGO.  Do not execute AIRCL
REPLACE(AIRABS).  Save absolute load file.
EXIT.
7/8/9 (EOR)
Loader directives (See Table III)
6/7/8/9 (EOF)
```

Modifying the source and creating a new absolute load file is done as follows. This is illustrated by Figure 16.

Job cards, account number, etc.

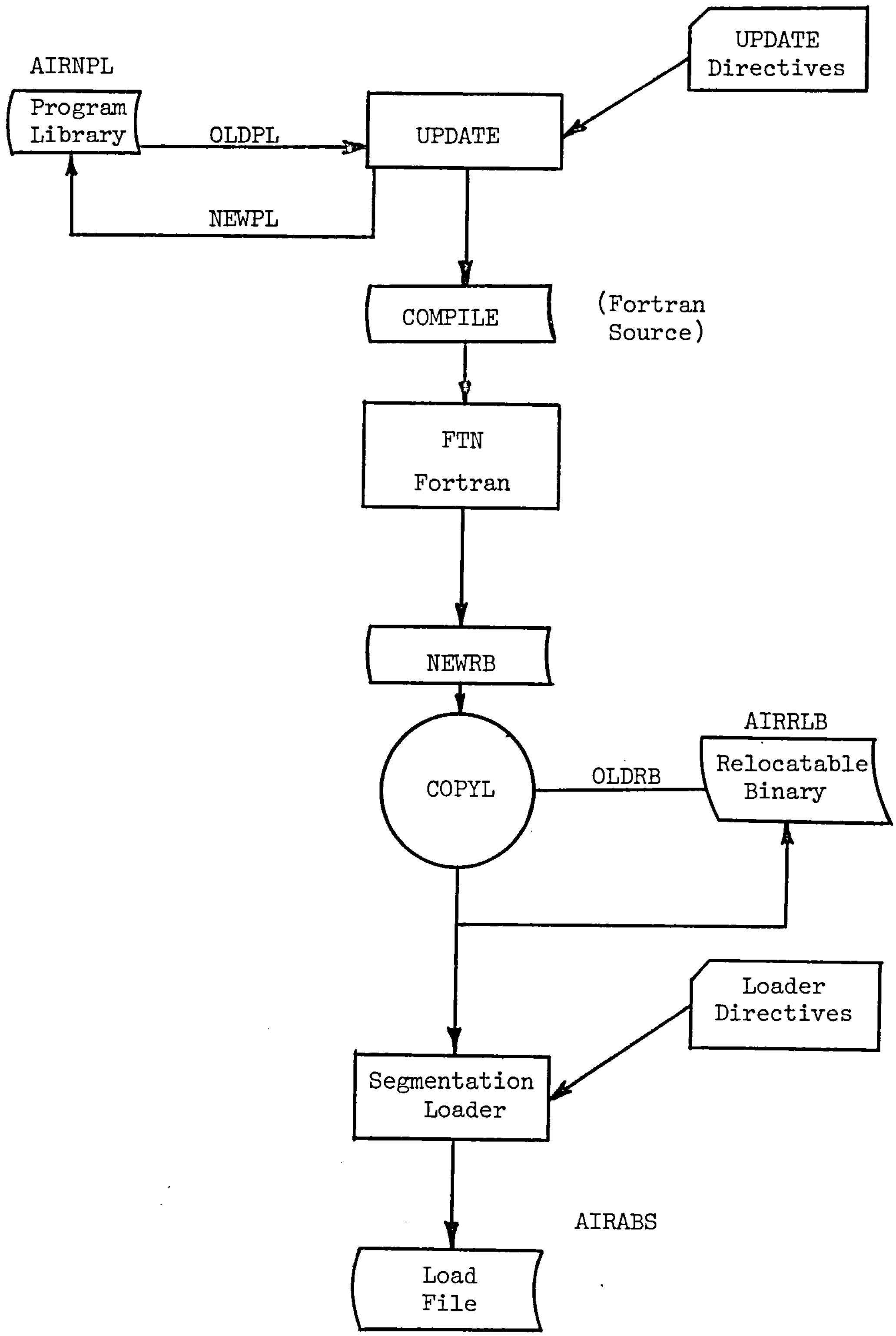

FIGURE 16. AIRCL UPDATE PROCESS

```
GET(OLDPL=AIRNPL)
UPDATE(L=A12,N)  Update directives will be from INPUT (cards).
REPLACE(NEWPL=AIRNPL)  Save revised program library.
RETURN(NEWPL)  Release file
REWIND(COMPILE)  This is Fortran source of only those modules
   which were updated.
FTN(I=COMPILE,B=NEWRB,R=3).  Compile
REWIND, NEWRB.
GET(OLDRB=AIRRLB)  Attach old relocatable binary file
COPYL(OLDRB, NEWRB, AIRRLB,,A)
  Generate new relocatable binary file by replacing only those
  modules which were UPDATE'd and re-compiled
REPLACE(AIRRLB)
RETURN(COMPILE,NEWRB)
MAP,PART.
SEGLOAD(I=INPUT,B=AIRABS)
LOAD(AIRRLB)
REPLACE(AIRABS)
NOGO.
7/8/9 (EOR)
  Update directives
7/8/9 (EOR)
  Segmentation directives (See Table III.)
6/7/8/9 (EOF).
```

TABLE 1. AIRCL PROGRAM ELEMENTS

(Alphabetical Order)

Element Name	Size (Octal)	Calls	Called by
AITKN	213		INTPLT, INTRSC
BUFFTP	1027	REDPRT	TAPEOP
CHK4PT	132		READFW
CHKN	111		READFW
COEFF	23		SPLINE
CLOUT	301	ERRMSG, TAPEWT	WRITCL
CROSSP	71		WINGCL, FSLGCL
DCOSIN			WRITCL
DIVIDE	147		WINGCL, FSLGCL WCLPTS, FCLPTS
ERRMSG	23		CLOUT, WRITCL
FCLPTS	1304	INTPLT, INT2CR INT2LN, LINEQ, DIVIDE, MAXMIN WEED	WNGFSLG
FORMPS			FCLPTS
FSLGCL	311	GTVCTR, CROSSP, DIVIDE, INTPLT	WNGFSLG
GOUGE			WINGCL
GTVCTR	56		WINGCL, FSLGCL
INT2CR	564	INTPLT, INT2LN, LINEQ	TRIMCL, FCLPTS, WCLPTS
INT2LN	41		INT2CR, FCLPTS, INTRSC
INTPLT	141	AITKN, SPLINE	INTRSC, FSLGCL WINGCL, FCLPTS, WCLPTS
INTRSC	702	INTPLT, INT2LN, LINEQ	WNGFSLG
LINEQ	63		INTRSC, INT2CR, FCLPTS,
MAXMIN	25		TRIMCL, STRTLE, FCLPTS, WCLPTS
NEWPLY	132		TCLPTS
OPENWA/CLOSWA	73		FCLPTS, WCLPTS
PACK/UNPACK			FCLPTS, WCLPTS
PRNTFW	134		READFW
RDFRM2	154		READFW
READFW	1077	CHK4PT, CHKN, PRNTFW, RDFRM2	

TABLE I. AIRCL PROGRAM ELEMENTS (CONTINUED)

Element Name	Size (Octal)	Calls	Called by
REDPRT	15		BUFFTP
ROTVEC			CLOUT, WRITCL
SPINDL		TAPEWT, ERRMSG	TMARK, CLOUT, WRITCL
SPLINE	307	COEFF	
STRTLE	717	MAXMIN, WEED	WCLPTS
TAPEOP	25		WRITCL
TAPEWT	135		CLOUT, TAPEWT
TCLPTS	561	NEWPLY	TRIMCL, WINGCL
TMARK			CLOUT
TRINCL	564	INT2CR, TCLPTS MAXMIN	WNGFSLG
WAERR	43		OPENWA
TRIMCL	564	INT2CR, TCLPTS MAXMIN, WEED	WNGFSLG
WCLPTS	332	STRTLE, DIVIDE, WEED, INTPLT	WNGFSLG
WINGCL	452	INTPLT, DIVIDE, CROSSP, GTVCTR, TCLPTS, GOUGE	WNGFSLG
WNG WNGFSLG	1530	READFW, FSLGCL, WINGCL, INTRSC, FCLPTS, WCLPTS, TRIMCL, WRITCL	
WRITCL	305	CLOUT, ERRMSG, TAPEOP, TAPEWT	WNGFSLG
WRITWA/READWA	122		FCLPTS, WCLPTS
WEED			FCLPTS, STRTLE WCLPTS, TRIMCL

TABLE II. PASSES PER INCH REQUIRED TO PRODUCE SURFACE WITH SCALLOP HEIGHTS AS SPECIFIED

Scallop Height (in.)	Passes Per Inch for Given Cutter Diameter D = 0.250	D = 0.500	D = 0.750	D = 1.000
0.050	5.00	3.33	2.67	2.29
0.040	5.46	3.68	2.97	2.55
0.030	6.16	4.21	3.40	2.93
0.020	7.37	5.10	4.14	3.57
0.015	8.42	5.86	4.76	4.11
0.010	10.21	7.14	5.81	5.03
0.005	14.27	10.05	8.19	7.09
0.002	22.45	15.84	12.93	11.19
0.001	31.69	22.38	18.27	15.82

TABLE III. SEGMENTATION LOADER DIRECTIVES

```
* SEGLOAD DIRECTIVES FOR WNGFSLG              -   JULY 1978
* ROOT SEGMENT
       TREE    WNGFSLG
WNGFSLG INCLUDE W.SQ,INTPLT,WRITWA,WAERR,OPENWA
       GLOBAL  SYSPR,CLPRM,SURFG,CLSRF,Q8.IO.,FCL.C.,GET.RT,CON.RM
       GLOBAL  AOB.RM,IO.BUF.,CLFRMT,FSWGSV
* FIRST LEVEL STORING WORKING ROUTINES
       LEVEL
        TREE    LINEQ
LINEQ   INCLUDE INT2LN,MAXMIN,SPLINE,AITKN,DIVIDE,CROSSP,GTVCTR,INT2CR
LINEQ   INCLUDE COEFF,PACK,WEED
        TREE    CHK4PT
CHK4PT  INCLUDE RDFRM2,INPC=,GOTOER=,NAMOUT=,NAMIN=,INCOM=,KRAKER=
CHK4PT  INCLUDE FLTIN=,CHKN,PRNTFW
        TREE    ERRMSG
ERRMSG  INCLUDE TAPEOP,CLOUT,TAPEWT,REDPRT,REWIND=,ENCODE=,INPB=,TMARK
ERRMSG  INCLUDE NEWPLY,TCLPTS,SPINDL
* SECOND LEVEL FOR MAJOR SUBROUTINE TREES
       LEVEL
        TREE    READFW
        TREE    WINGCL
WINGCL  INCLUDE FSLGCL,GOUGE
        TREE    INTRSC
INTRSC  INCLUDE TRIMCL
        TREE    FCLPTS
FCLPTS  INCLUDE FORMPS
        TREE    WRITCL
WRITCL  INCLUDE BUFFTP,DCOSIN,ROTVEC
        TREE    WCLPTS
WCLPTS  INCLUDE STRTLE
       END
```

1. Report No.	2. Government Accession No.	3. Recipient's Catalog No.
NASA CR-165687		

4. Title and Subtitle	5. Report Date
AIRCL, A Programmed System for Generating NC Tapes for Airplane Models, User's Manual	March 1981
	6. Performing Organization Code
7. Author(s)	8. Performing Organization Report No.
N. Akgerman and C. F. Billhardt	
	10. Work Unit No.
9. Performing Organization Name and Address BATTELLE, Columbus Laboratories 505 King Avenue Columbus, OH 43201	11. Contract or Grant No. NAS1-15090
	13. Type of Report and Period Covered Contractor Report
12. Sponsoring Agency Name and Address National Aeronautics & Space Administration Washington, DC 20546	14. Sponsoring Agency Code

15. Supplementary Notes

Langley Technical Monitor: D. G. Bowen

Final Report

16. Abstract

A computer program has been written which calculates the cutter location file needed to machine models of airplane wings or wing-fuselage combinations on numerically controlled machine tools. Input to the program is a data file consisting of coordinates on the fuselage and wing. From this data file, the program calculates tool offsets, determines the intersection between wing and fuselage tool paths, and generates additional information needed to machine the fuselage and/or wing. Output from the program can be post-processed for use on a variety of milling machines. This report contains information on program structure and methodology, and the user's manual for implementation of the program.

The computer program is available from COSMIC, 112 Barrow Hall, University of Georgia, Athens, GA, 30602. The program identification number is LAR-12494.

17. Key Words (Suggested by Author(s))	18. Distribution Statement
Wind Tunnel Model Construction Numerical Control Machining	Unclassified - Unlimited Subject Category - 37, 05

19. Security Classif. (of this report)	20. Security Classif. (of this page)	21. No. of Pages	22. Price*
Unclassified	Unclassified	38	A03

*For sale by the National Technical Information Service, Springfield, Virginia 22161