

AN ABSTRACT OF THE THESIS OF

Michael G. Babb for the degree of Master of Science

in Forest Products presented on May 2, 1985

Title: Simulation Analysis of Green Veneer Recovery

as Influenced by Different Clipping Strategies

Abstract approved:

Redacted for Privacy

Dr. James W. Funck

Several studies have identified the green veneer clipper as the largest contributor to volume loss during veneer production. A recent project reported that merchantable veneer volumes could be increased anywhere from 1.5 to four percent. No conclusions were made in regard to the mix of veneer achieved with the additional recovery. Through computer simulation techniques, this study focuses on management selected clip control adjustments in an attempt to identify specific changes in veneer mix as well as the impact on overall recovery.

Clipped veneer from fifteen Douglas-fir (*Pseudotsuga*

menziesii) number 2 sawlog blocks were digitized in the laboratory. A computer program was written to reconnect the clipped pieces thereby reconstructing the peeled ribbon of veneer. The logic of a clipper scanner was incorporated in another program to enable a computer simulated clip of the reconstructed veneer. Specific conclusions based on changes in clip control adjustments and subsequent changes in veneer mix and/or economic recovery become possible.

By varying a clip control input during each run, significant changes in the complexion of the veneer mix as well as wood yield were noted. For example, a 1.5 inch reduction in margin setting produced a 3.3 percent increase in overall merchantable veneer recovery. With each change in clip control a corresponding increase in volume and/or economic recovery was possible. The introduction of a double fishtail option to clip control did not prove economically feasible.

^c
Copyright by Michael G. Babb

May 2, 1985

All Rights Reserved

Simulation Analysis of Green Veneer Recovery
as Influenced by Different Clipping Strategies

by

Michael G. Babb

A THESIS

submitted to

Oregon State University

in partial fulfillment of
the requirements for the
degree of

Master of Science

Completed May 2, 1985

Commencement June 1985

APPROVED:

Redacted for Privacy

Professor of Forest Products in charge of major

Redacted for Privacy

Head of Department of Forest Products

Redacted for Privacy

Dean of Graduate School

Date thesis presented: May 2, 1985

ACKNOWLEDGEMENTS

Without the guidance of my major professor, Dr. James Funck, this "no problem" thesis would have died years ago. Thanks Jim for your instruction, patience, and sense of humor.

Thanks to the Plywood Research Foundation, the cooperating mill, the Department of Forest Products, and the Milne Computer Center.

I would also like to thank Mr. Dave Jordan for his diligence and commitment to the tedious job of digitizing the veneer.

DEDICATION

This thesis is dedicated to my family: Deanna, Rachel, and Rebecca. We got the job done. Now let's get back to the business of being a family once more.

TABLE OF CONTENTS

INTRODUCTION	1
OBJECTIVE	9
GREEN-END REVIEW	10
A CLOSER LOOK AT CLIPPER SETTINGS	19
PROJECT OVERVIEW	27
PROCEDURE	29
Digitizing the Veneer	29
Digitizer Computer Programming	33
Reconstructing the Ribbon of Veneer	35
Matching Points	35
Input Data File	37
Considerations	39
Programming	44
Simulating the Green-End Clipper/Scanner	50
Understanding the Model	50
Reclip Programming	55
RESULTS AND DISCUSSION	64
CONCLUSIONS	71
RECOMMENDATIONS	74
Veneer Recovery	74
Computer Simulation	75
IMPLICATIONS FOR FUTURE STUDY	77

GLOSSARY	78
----------	----

LITERATURE CITED	79
------------------	----

APPENDICES

A Digitizing Computer Program	83
B Reconstruction Computer Program	92
C Plotting Computer Program	132
D Reclip Computer Program	136

LIST OF FIGURES

<u>Figure</u>	<u>Page</u>
1. Stumpage costs increasing faster than veneer revenues	2
2. Potential yearly increase in veneer revenue	3
3. Components of Douglas-fir peeler blocks determined by Woodfin	5
4. Block volume components found by Sheffield	5
5. Typical veneer production process	11
6. Major inputs required for a clip decision	16
7. Simplified through-beam, optical scanner logic	18
8. Possible fishtail measures	20
9. Margin adjustment to clip control	20
10. Clip with panel switch on	23
11. Clip with random mode selected	23
12. Example of an accurate clip resulting in wood loss	25
13. Example of an inaccurately placed clip producing wood loss	25
14. Project overview	28
15. Laboratory configuration of equipment	30
16. Digitizer platen	31
17. Graphical presentation of reconstruction programming logic	36
18. Filmed veneer on the digitizer platen	38

19. Computer generated plot of the film	38
20. Skewed clips required adjustments to make them parallel	41
21. Match points must be corrected	41
22. Four general veneer types	46
23. Reclip programming simulates the optical scanner functions	52
24. Clip logic control in the computer simula- tion	54
25. "Look ahead" clip logic with replacement	61

LIST OF TABLES

<u>TABLE</u>	<u>PAGE</u>
1. Control run scanner settings	65
2. Summary of veneer mix	66
3. Summary of percent distribution in veneer mix	68
4. Percent change in mix against control	69

SIMULATION ANALYSIS OF GREEN VENEER RECOVERY AS INFLUENCED BY DIFFERENT CLIPPING STRATEGIES

INTRODUCTION

The economic reality of today's wood products market has forced many plywood mills to shut down, while others continue to hang on, working less hours with fewer people. Forecasters present a double-edged sword for the future. Demand for construction materials is expected to strengthen [1,2,3]. However, stumpage prices are expected to increase at a faster rate than veneer revenues [4,5] (Figure 1). In addition, a timber shortfall is predicted by the year 2000 [4,5,6]. Raw material costs still represent the largest portion of production costs [7]. This mixed bag of news presents the mill manager with the paradox of needing to get more for less. Recovery must be increased and costs must be reduced by utilizing more efficient methods of wood conversion. Generally, recovery means maximizing wood material and/or profit [8,9]. Often the two go hand in hand. For example, a one percent increase in volume recovery in a mill producing seventy million square feet, 3/8 inch basis, and selling at \$30/thousand, would increase annual revenue by \$70,000 (Figure 2).

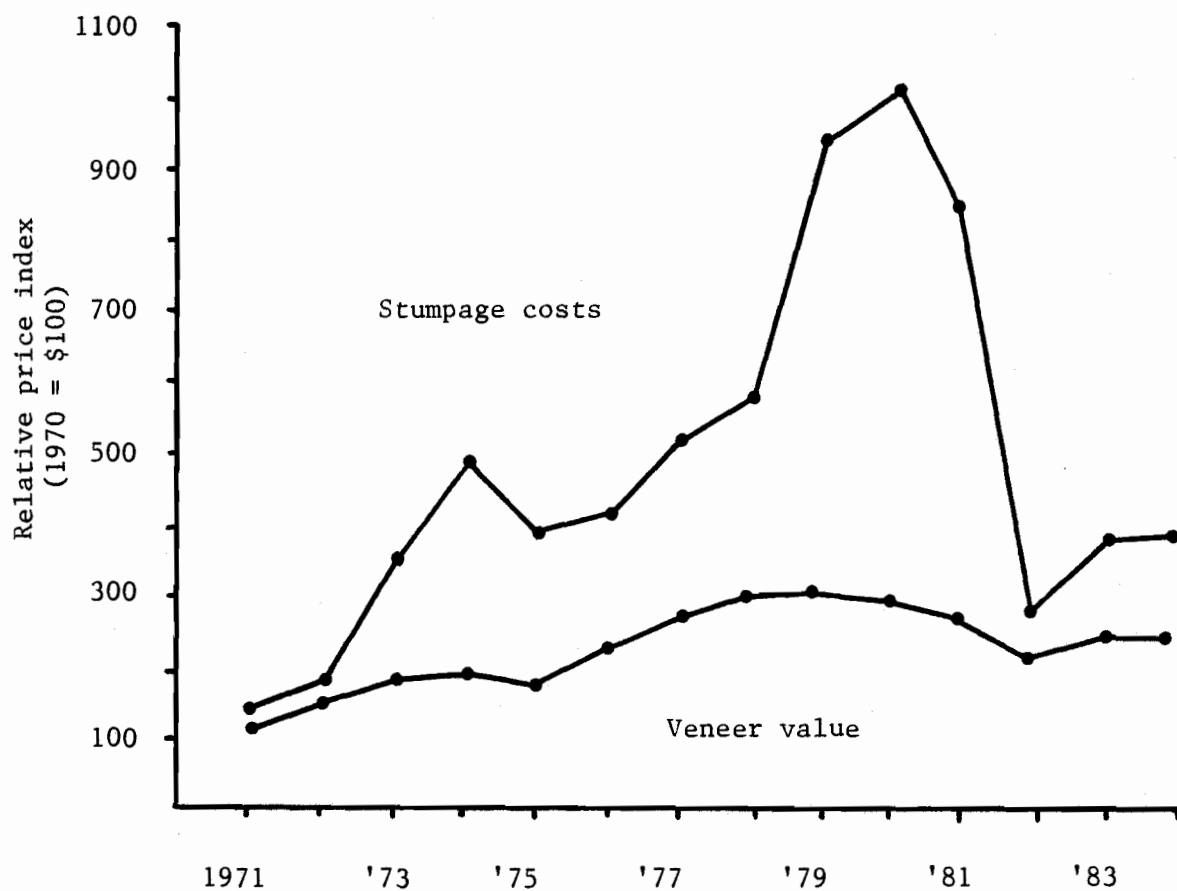


Figure 1. Stumpage costs increasing faster than veneer revenues

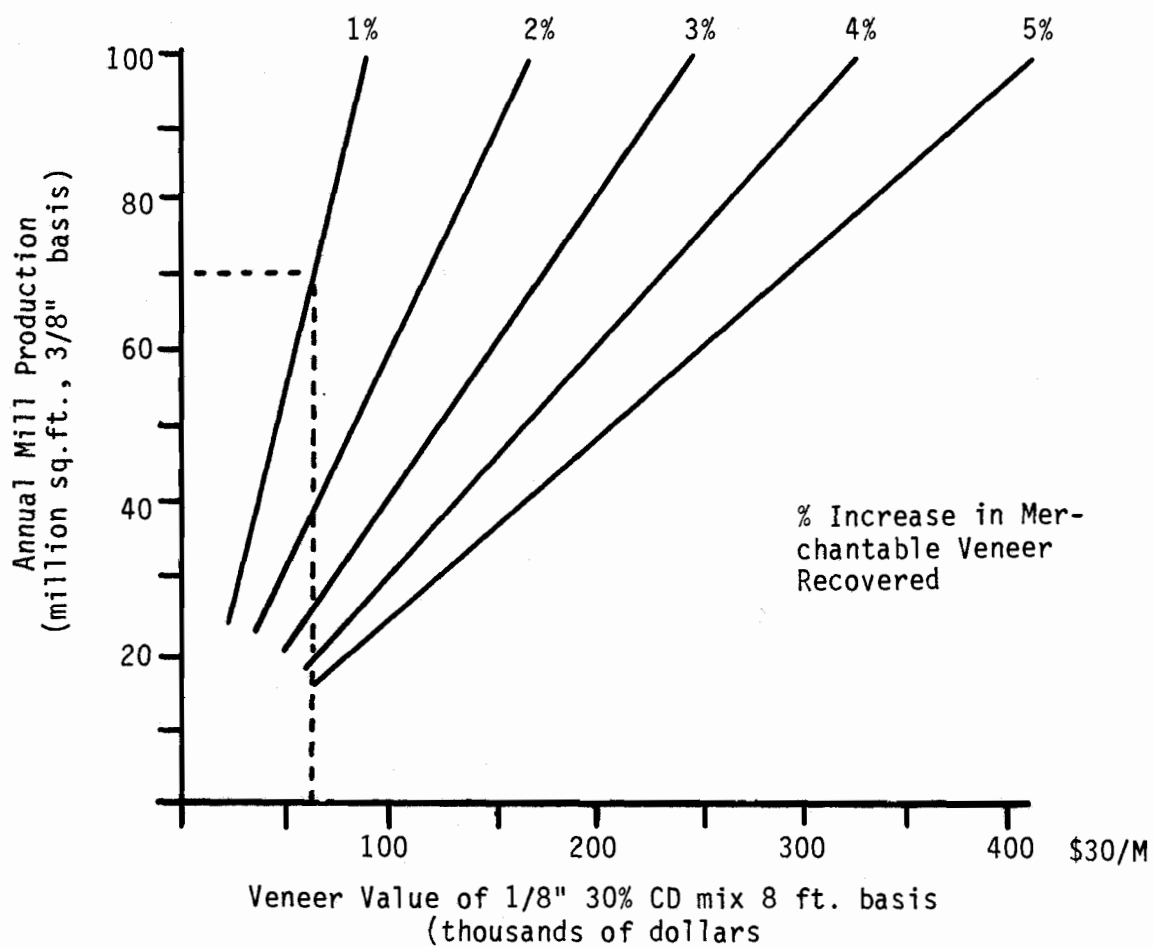


Figure 2. Potential yearly increase in veneer revenue.

Plywood manufacturing is inherently wasteful. In fact, only about half of the block volume is converted into the final product [10,11]. Several studies have identified the green veneer clipping center as the largest contributor to block volume loss [12,13]. Estimates range from a low of 7 to a high of 21 percent (Figures 3 & 4). These studies go on to identify the sources of loss and conclude with specific recommendations for improvement. In many cases, mill managers can implement immediate corrective action with a resulting increase in veneer yield. Research methodology can also serve as a model for quality control checks, leading to a uniform product with reliable process control. Opportunities for recovery improvement fall into four major categories:

1. operator skill, knowledge, and experience
2. clipper control design and operation
3. mechanical functioning of the clipper/scanner
4. management clipping strategy.

Managers and researchers have joined together in an effort to identify and overcome the obstacles inherent in each clip component. Routine maintenance schedules provide immediate yield improvement by maintaining machine design capabilities. Sophisticated scanning devices interfaced with computers enable precise clip control. Introduction of the rotary clipper has overcome many of the problems experienced

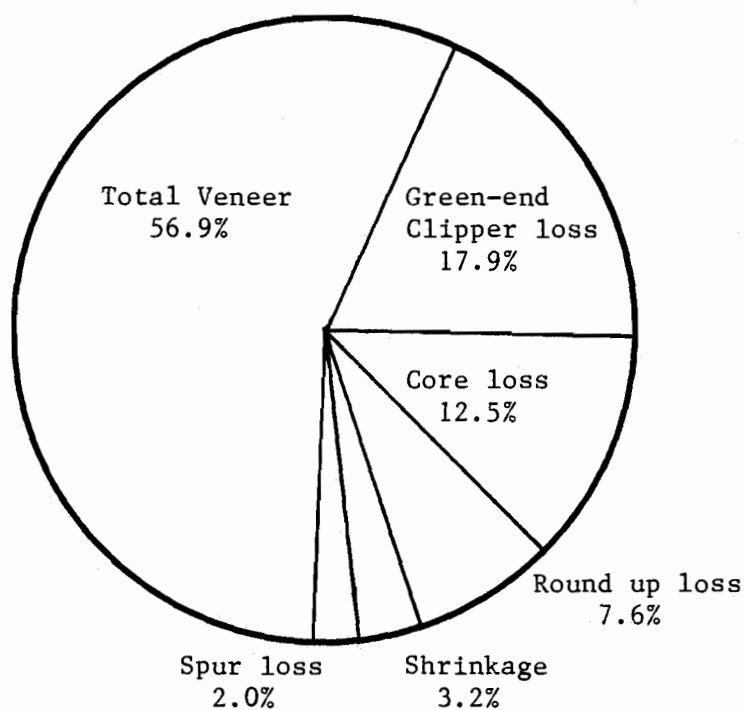


Figure 3. Components of Douglas-fir peeler blocks determined by Woodfin

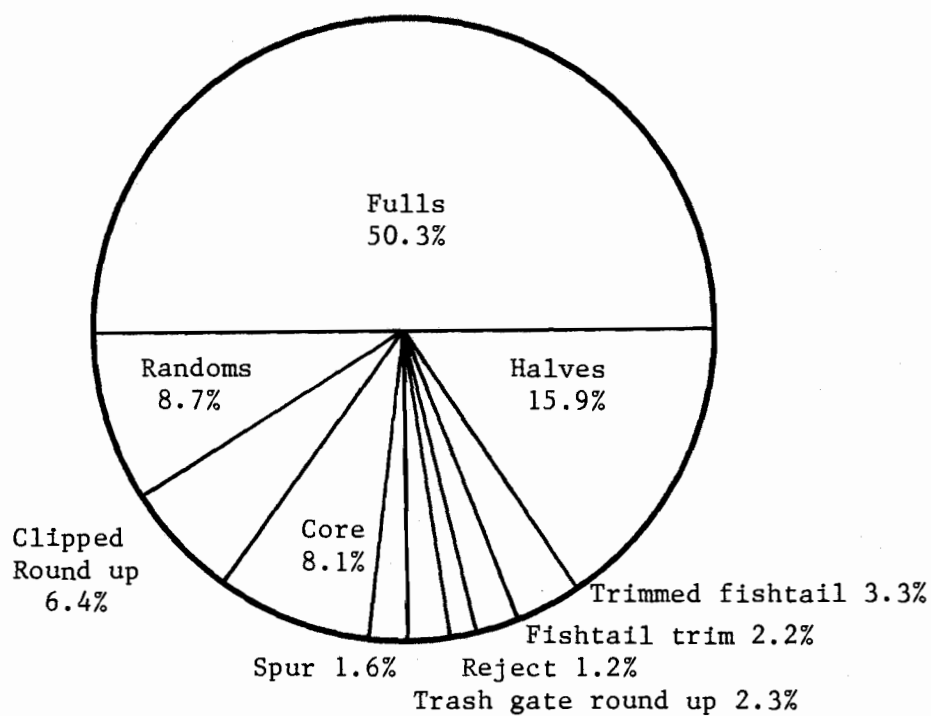


Figure 4. Block volume components found by Sheffield

with the guillotine clipper [13,14]. New delivery systems maintain a constant stream of raw materials resulting in faster throughput. Finally, management has realized the role clip strategy plays in ultimate recovery as influenced by margin setting, flaw limits, minimum strip, etc..

It is hard to justify major capital outlays for state of the art equipment today, when tomorrow may never come. Consequently, managers must look to existing equipment and processes in the search for higher recovery. Therein lies the justification for this paper. Mills can increase recovery through a better understanding of clipping strategies and subsequent control adjustments.

To understand the effect a change in clip component has on overall veneer yield, a green veneer recovery analysis must be performed. A measure of the merchantable veneer peeled is compared to incoming net block volume. This veneer recovery factor(VRF), estimates yield. The VRF is only an approximate measure of recovery. Loss volumes are calculated by subtracting the sum of all losses (core, shrinkage, spur, trashgate, etc.) from the total block volume rather than by direct measurement. In addition, error is inherent in the VRF equation by virtue of the use of mixed measuring units.

A review of the literature reveals several studies involving the application of computers for simulation and veneer recovery analysis. Numerous programs have been written to analyze the effect precision X-Y lathe chargers have on recovery [15,16]. Since the role defects play on the occurrence of clips is not a component of these models, they have little impact on the actual clipping sequence.

Tobin and Bethel conducted an innovative study utilizing filmed veneer from four peeler blocks [17]. The filmed images were digitized in order to convert the pictorial record into numerical data for computer application and manipulation. Individual pieces were reconnected by a computer program to simulate the continuous ribbon of veneer. Additional programming incorporated various clip strategies to effect a "reclip" of the ribbon. The objective of the study was to maximize grade rather than merchantable veneer yield. Any attempt to extrapolate their research findings to actual mill situations is impossible for several reasons.

First, the study predated the introduction of scanner controlled clippers. Consequently, the impact of critical management inputs could not be measured. The simulation model assumed the veneer to be subdivided into one inch cross grain widths, implying defects could only be clipped out to a minimum one inch interval.

Another shortcoming was the fact that after the block was rounded up, the ribbon was clipped to four by eight foot sheets without regard to quality characteristics. Also, round up was classified as trash material. With today's smaller diameter blocks, round up is an important source of additional veneer yield. Finally, the aim of their study was to clip for grade; few mills have this capability. Perhaps the greatest value of Tobin and Bethel's work was to demonstrate the importance of the digitizing procedure as well as the power of computer simulation models for veneer recovery studies.

Sheffield also utilized a digitizer for the acquisition of data from which a detailed study in the types and volumes of green-end veneer recovery losses were reported [13]. An 8MM movie camera recorded all wood material passing downstream from the green-end clipper. Actual clips were analyzed and categorized for clip accuracy and recoverability. Areas were computed for each classification of veneer. Potentially recoverable veneer was reported to be 1.1 to 2.6 percent of the total block volume. This study made no attempt to reconstruct and reclip the ribbon of veneer. Sheffield's research pointed to the potential for increases in veneer yield as well as reinforcing the value of the digitizer computer interface in veneer recovery studies.

OBJECTIVE

The primary purpose of this study is to determine the effect changes in mill management scanner inputs have on veneer recovery. User inputs include:

1. margin limits
2. minimum strip
3. clipping for panel versus randoms
4. double fishtail option.

The double fishtail option is not currently available to most managers. However, this research will determine the economic feasibility as judged by a potential increase in recovery if the option was at hand. Also, this project will establish a database containing a complete record of the veneer ribbon resulting from the peel of several blocks. This information can be used as a basis for clipping veneer for grade recovery.

GREEN-END REVIEW

Sheffield [13] found that his study mills could indeed increase wood yield. He judged the occurrence of clips as either accurate or inaccurate. That is, clips were considered accurate if they fell in a correct response to scanner settings. Conversely, inaccurate clips meant that scanner-directed clips were not made at the ordered location. In both cases, good wood was removed from merchantable pieces of veneer, resulting in lower veneer recovery. A detailed explanation of this seemingly contradictory finding will follow later in this report. Three to five percent additional merchantable veneer could have been recovered. One must look at the in-process manufacture of veneer to understand how accurate and/or inaccurate clips occur (Figure 5).

The typical veneer mill or plywood green-end is divided into eight processes or procedures. First, logs are bucked into peelable lengths. Depending on the mill's capability and order file, peeler blocks will measure from seven to ten feet in length. Next, the bark is removed. Then the blocks are positioned in the lathe. Positioning aids, like the X-Y charger, are becoming increasingly popular. Otherwise, the block is usually geometrically centered. Once the position is set, the

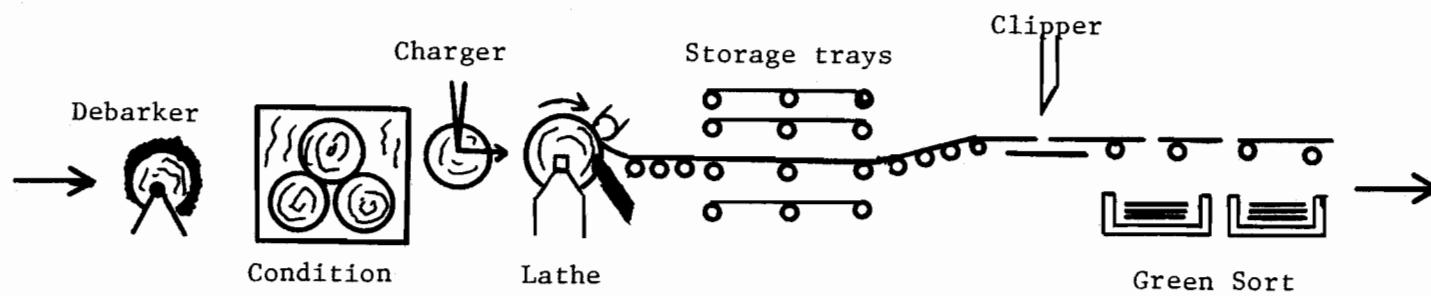


Figure 5. Typical veneer production process

lathe carriage moves forward as the spindles rotate the block. As a full-length knife slices through the wood, the turning block becomes a ribbon of veneer. Following the peel, the ribbon is conveyed downstream where it is clipped into selected widths based on the appearance of specific open flaws and/or maximum widths. Almost all mills clip with the aid of an optical scanner. The clipped pieces are then sorted for size and moisture content. Wood can be lost at each stage of this process. This study will attempt to reduce losses due to below-grade veneer and round-up at the green-end clipper by focusing on selected clip control inputs.

The primary purpose of the veneer clipper is to clip the widest useable lengths of wood. As a consequence, this ultimately leads to the removal of specific open flaws. This strategy usually results in maximizing the number of full-sized sheets. However, economic considerations may sacrifice yield for higher value. For example, would it be best to clip a random and lose a fishtail, or take some random wood to allow the recovery of both piece types? An economic decision is made when yield recovery is the target. This comes in the form of a panel versus random switch which is set on the clip control. To get a clear picture of how management inputs impact on the clip, a cursory description of the

scanner controlled clipping machine center is presented. Equipment in place at the study mill will be described. Several other configurations are available, but all perform under similar rules of logic.

Debarker: Ring

Charger: PMI Superior Precision X-Y

Lathe: Premier

Conveyor: Direct coupled

Scanner: Morvue

Line clock: Morvue

Blade position indicator: Morvue

Clipper: Elliot Bay Semi-Auto (guillotine)

Before a clip is ordered, notwithstanding operator override, the clipper must have specific information concerning the piece of veneer. First, the defect size and location must be known. Also, the speed that the veneer moves under the clipper must be tracked. With regard to the clipper, knife position must be ascertained. The final and often most important clip variables are management selected limits and options. These include flaw limits, margin and minimum strip settings, panel versus random, and others. All the aforementioned components contribute to clip accuracy. An examination into the collection and action based on this information reveals a complex interaction among separate components, that together, cause a clip to be ordered.

Veneer speed is typically measured with a timing wheel or line clock. This is a direct contact, free spinning tracking device. It is located about 30 inches ahead of the clipper. Various factors, such as wheel wear, loss of wood contact, and physical placement of the apparatus can introduce error. Consequently, this measure is only a good approximation and some clipping accuracy may be sacrificed.

The position of the clipper blade must be known at all times. The blade must be ready to fall upon command. Sometimes, a clip is requested before the blade has completed its full stroke cycle, that is, prior to returning to the top-most position. The ability of the system to react to this "quick clip" order is very important to veneer recovery [14]. This capability may limit the minimum strip selection. Air quality and pressure, valve condition, belt wear, and hose line configuration and condition all contribute to clipper response time.

With the introduction of optical scanners, significant improvements in veneer clipping have followed. Standard sized sheet accuracy is maintained. Defect removal is accomplished with more precision. Much more veneer can be processed in a given time frame. Increased revenue is guaranteed through a reduction in

waste. Optical, through-beam, defect detectors are the most commonly used. However, research with x-ray, ultrasonic, microwave, and neutron devices may give rise to a new generation of defect detection systems [18]. Regardless of the method employed, a defect detection system must possess the following attributes:

1. able to sense, classify, measure, and determine relative defect position
2. consistency and reliability
3. real-time basis
4. durability to withstand mill conditions
5. easy maintenance
6. cost effectiveness.

The typical through-beam, optical scanner relies on the reception of light-dark messages to locate open voids. No sound or intact veneer flaws can be detected. An overhead light source, such as an ordinary incandescent or florescent bulb, shines down on the moving veneer. A row of sensors beneath the veneer detects light that may pass through the voids. A simplified illustration of this arrangement is depicted in Figures 6 and 7. Optical data is gathered left and right hand simultaneously, allowing differentiation of left and right fishtails. Detector heads are placed at half-inch intervals and read at a rate which produces about a one-tenth inch cross-grain resolution at a 200 fpm line

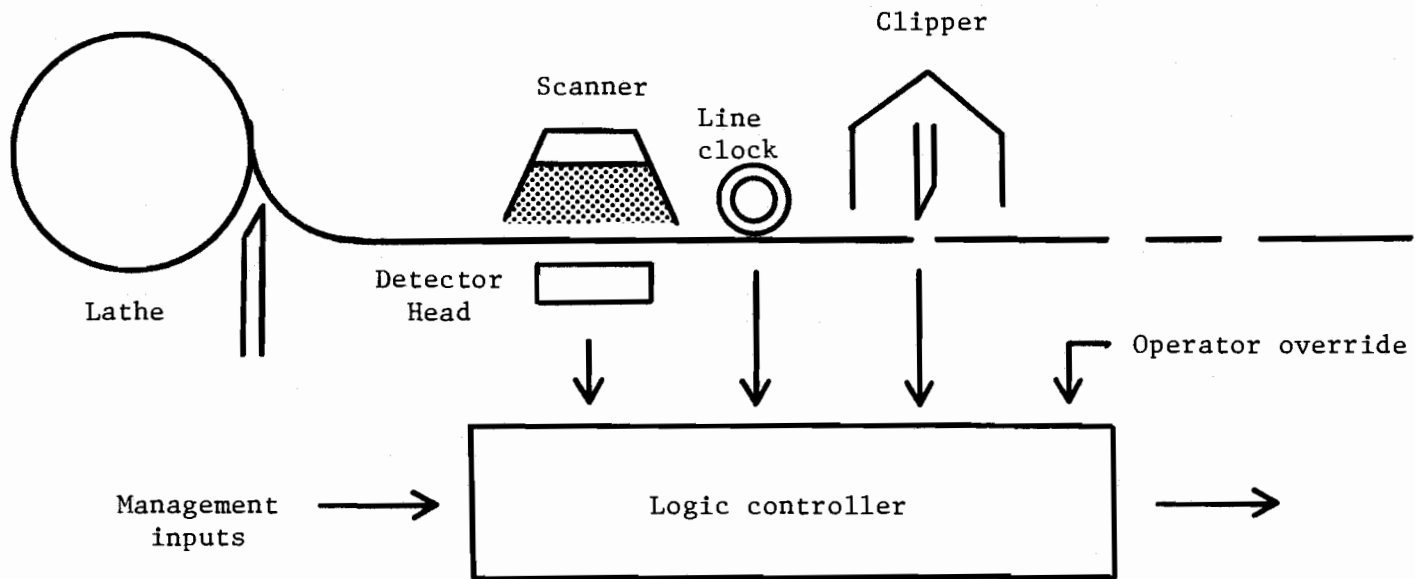


Figure 6. Major inputs required for a clip decision

speed.

The various clip logic inputs: scanner data, knife position, and line speed, are fed to the logic controller. This unit maybe a minicomputer or a hard-wired, read-only memory device. This information, along with management selected clip options, determine the location and occurence of clips.

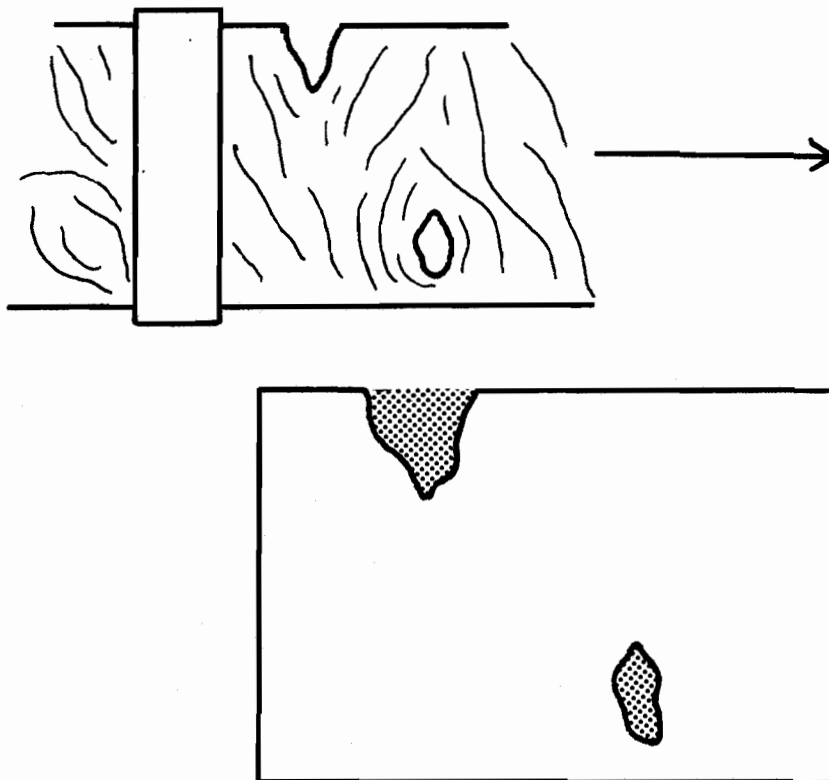
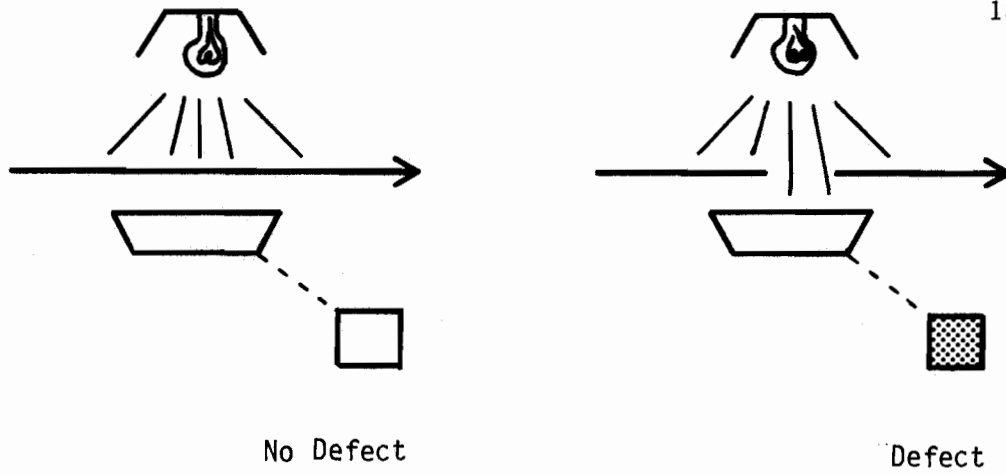


Figure 7. Simplified through-beam, optical scanner logic

A CLOSER LOOK AT CLIPPER SETTINGS

Depending on the make and model, several management selected options are available that allow the user to tailor clip decisions to specific needs. These include veneer size, flaw limits, and clip control adjustments.

Veneer size refers to both with and across grain measures. For the remainder of the paper, with grain and across grain are synonymous with veneer length and width respectively. Standard sheet sizes as well as with grain fishtail limits are set. Fishtail widths are dependent on the minimum strip setting at the lower end and the half sheet width setting in the upper dimension (Figure 8). Flaw limits encompass cracks, edge flaws, and open defects. Open voids must exceed both the with and cross grain tolerances before a clip is ordered. Margin, flaw centering, and minimum strip fall in the clip control adjustment category. The margin setting determines the clip pull back, or the closeness of a clip to the defect or edge of a piece (Figure 9). As the name implies, flaw centering forces the leading and trailing clips to fall an equal distance from a defect. Often, this setting is used to compensate for line clock wheel wear. Minimum strip sets the lower limit for

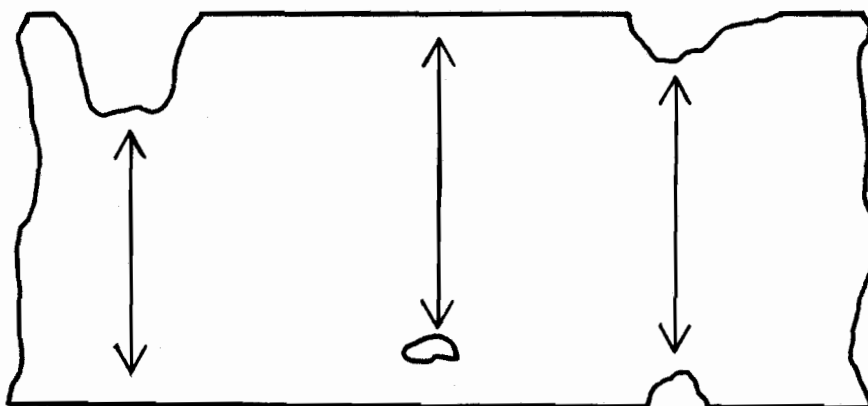
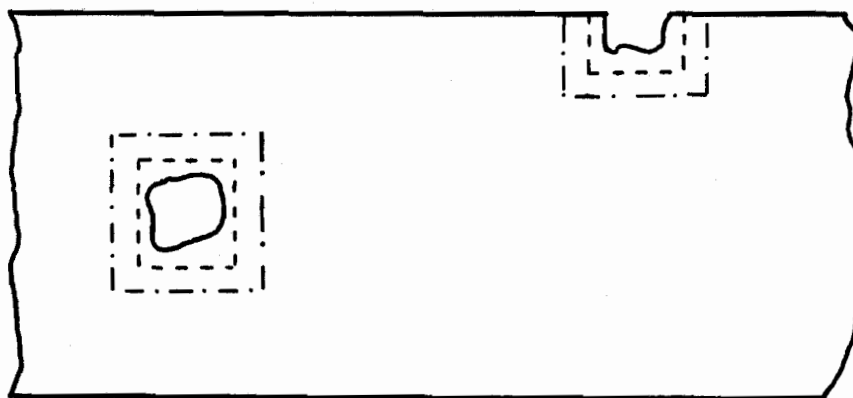


Figure 8. Possible fishtail measures



.--- Commonly set margin
---- Necessary margin limit

Figure 9. Margin adjustment to clip control

width of a piece. Any piece less than this setting is not considered economically feasible for further conversion.

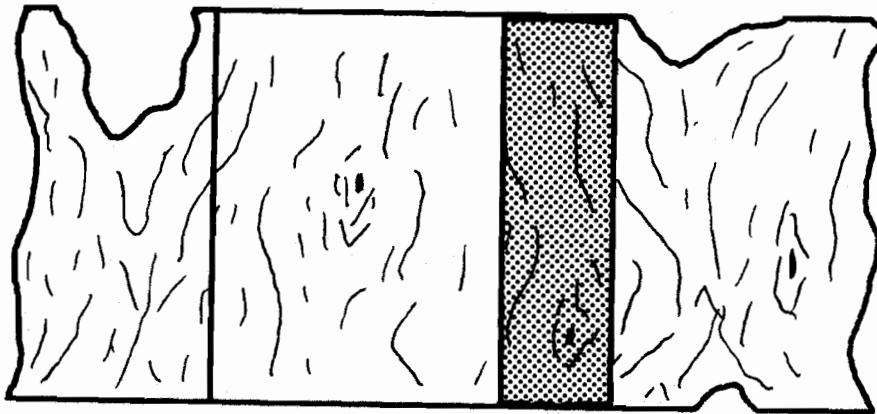
The margin adjustment was originally designed as a method to overcome the inability of early clippers to place clips where the scanner ordered. Now, margin input is often used as an indirect compensation for wane expected around a defect or on the edges of round up. The amount of wane actually present is a function of veneer thickness and location within the ribbon. Because of this, no explicit adjustment for veneer thickness is necessary, just a change in the margin setting. As a consequence of the peeler block roundup, a larger amount of wane is present at the beginning of the peel. Once a cylinder is formed, lesser amounts of wane can be expected. Margin control adjusts to this fact by allowing for an alternate selection, which reduces the limit. Following the appearance of a specific number of full sheets, usually two or three, the alternate selection takes priority. A similar option is also available for minimum strip width.

Even though a specific input is required for the minimum strip setting, in an actual clipping sequence, the amount of good wood must meet this input plus the margin setting. For example, if the minimum strip is set at five inches and the margin is 1.5 inches, then

the total amount of good wood needed for a random strip must be eight inches. An additional setting, panel versus random, determines whether maximum wood yield or the recovery of all possible half sheets will take precedence.

The panel/random option lets the mill manager select a strategy for maximum wood yield (random), or maximum economic recovery (panel). This clip option is activated when a full width measure is greater than the half sheet but less than the half sheet plus minimum strip setting. In the panel mode, a half is clipped and the full width extra wood is lost (Figure 10). Two equal randoms are cut if the random option is selected (Figure 11).

Returning to the seemingly contradictory terms of accurate and inaccurate clips, one can now see the potential for losses within each category. Good wood is lost because of unnecessarily large margin and/or minimum strip settings. Conversely, wood is lost as a consequence of inaccurately placed clips caused by mechanical or electrical malfunctions. What follows is a hypothetical clip sequence presented to illustrate the accurate and inaccurate clip classifications. Management has decided on the following clip settings:



Attached to
fishtail

Figure 10. Clip with panel switch on

Random

Random

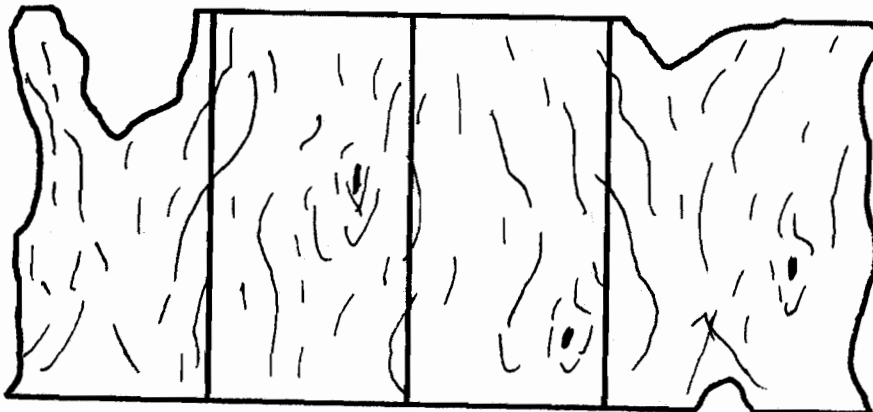


Figure 11. Clip with random mode selected

full sheet ----- 53 (inches)

half sheet ----- 26.6

min. strip ----- 7

margin ----- 1.5

fishtail length -- 54.

Clips accurately placed are a consequence of these instructions. A frequent occurrence that results in lost wood even though clips were made correctly, is depicted in Figure 12. The void extends too far into the wood, therefore, the fishtail length cannot be met; (remember, before a random is clipped, minimum strip plus margin must be found). In this example, seven inches of good wood is lost even though two halves are also recovered. Another case for review is presented in Figure 13. Here, good wood is lost by virtue of an inaccurately placed clip. The downstream clip should have been placed 1.5 inches from the void. The resulting random strip is reduced in size and four inches of good wood is lost when the fishtail is trimmed. Taking into account a spur knife setting of 101 inches, a total of 73.66 square inches is lost from merchantable veneer recovery. A corresponding decrease in revenue can be expected.

With the information presented in the general discussion on the green-end and the clip process, one can see the potential for increasing wood yield. This

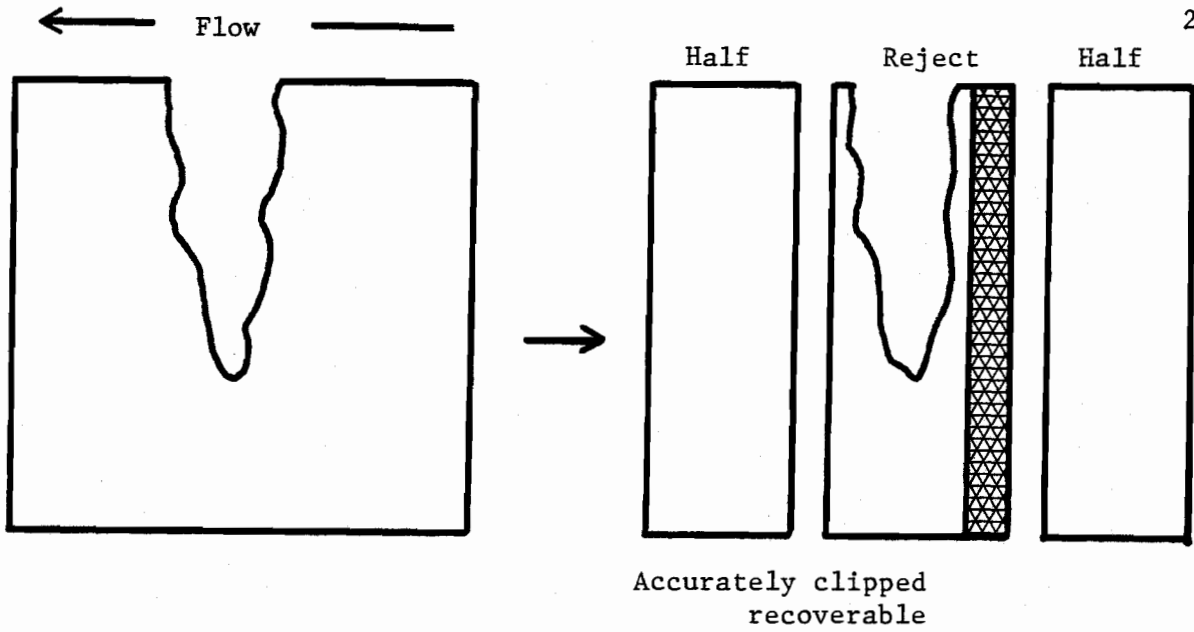


Figure 12. Example of an accurate clip resulting in wood loss

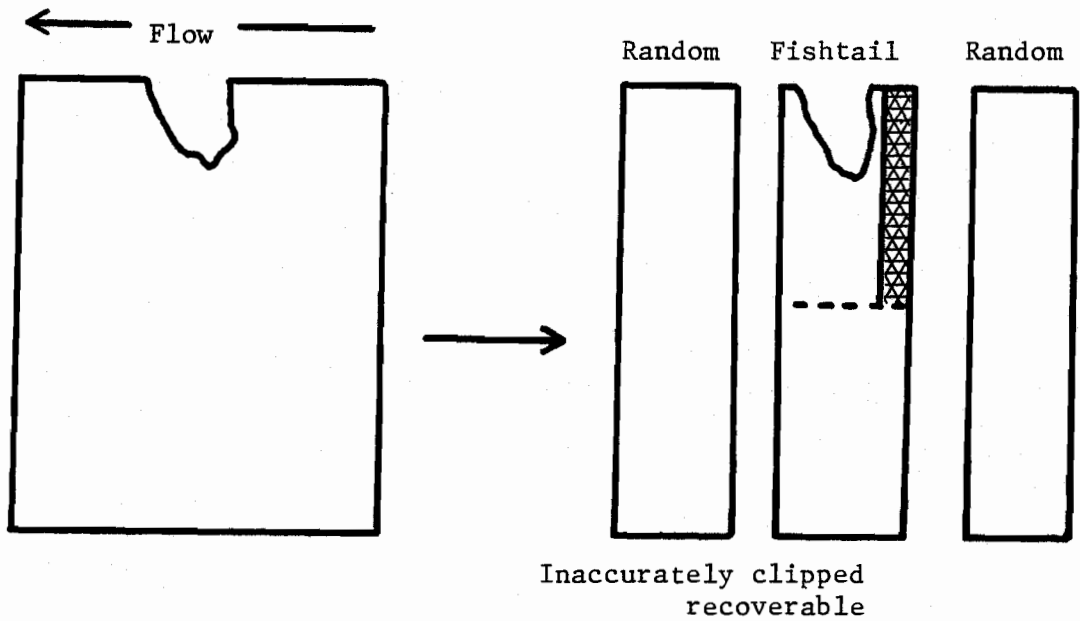


Figure 13. Example of an inaccurately placed clip producing wood loss

brings us to the main emphasis of this study: what is the impact on recovery with changes in management clip instructions? Can waste be reduced? Will a computer model provide sufficient parameters and flexibility to incorporate the dynamic aspects of wood as well as specific mill management requirements in an attempt to simulate the green veneer clipper?

PROJECT OVERVIEW

A simplified version of the mill study is presented in Figure 14. Specific mill data was gathered by Sheffield in a previous study [13]. Therefore, only a cursory description of this phase of the project is presented. See the cited material for a complete narrative of the on-site preparations and procedures. Similarly, the laboratory digitizing sequence with regard to projector alignment, skew correction, and scalar factor determination is identical to the cited study.

An 8mm movie camera synchronized with high speed flash units filmed all veneer immediately after passing under the green-end clipper. The filmed images were then projected onto a digitizer platen to enable transformation to digital format. With the data as input, computer programs were written to reconstruct the ribbon of veneer and then "reclip" the strip according to various clip hierarchies. A comparison of actual mill clip to the simulated clip was analyzed for differences in yield. Recovery figures are also compared between simulated clip runs with varying clip logic inputs.

Filming done by Sheffield and Funck

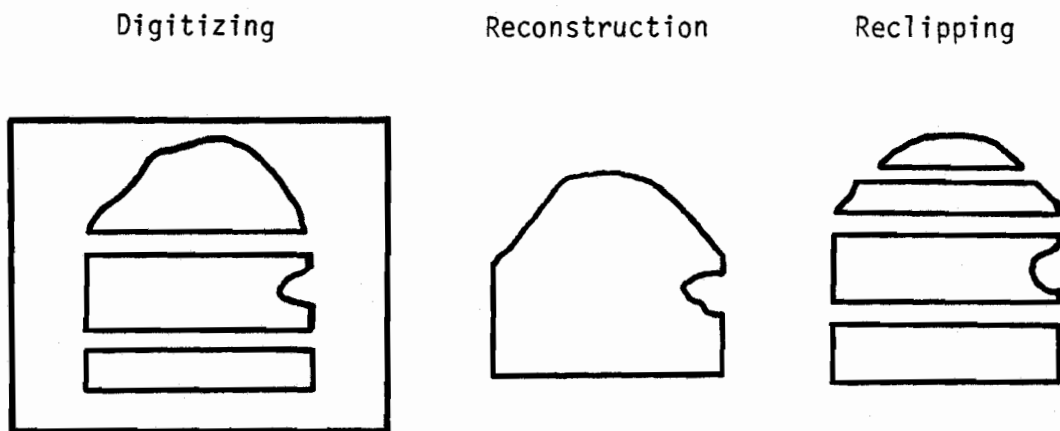
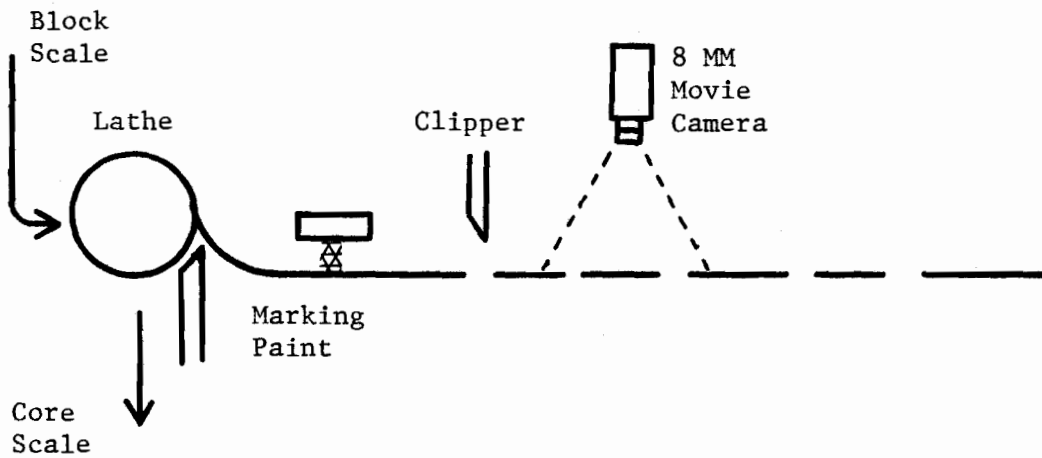


Figure 14. Project overview

PROCEDURES

Digitizing the Veneer

A GTCO DEMI-PAD 5 digitizer was used in this study. As previously mentioned, the digitizer converts analog data (the filmed veneer), to digital data (X-Y coordinate pairs) (Figure 15). The digitizer platen can be thought of as a Cartesian coordinate system, with all points falling in the upper right hand quadrant (Figure 16). A bank of specific coordinates is created by tracing an image with the cursor. These data points faithfully represent all aspects of the image. In this manner, veneer shapes and their defects, as well as wane, were described. Rectangular pieces, such as fulls and randoms, are referenced by their corner points. Discrete-read digitizing was performed in these instances. Because of their irregular shape, round up was continuously digitized. This is a physical trace of the entire perimeter of the piece. Cracks were described by four points. Open voids exhibiting a wane border were also continuously digitized so that actual shapes could be preserved. Voids with no wane were represented as having wane, but the wane coincided with the actual edge of the defect. In other words, the defect edge and the wane



Figure 15. Laboratory configuration of equipment

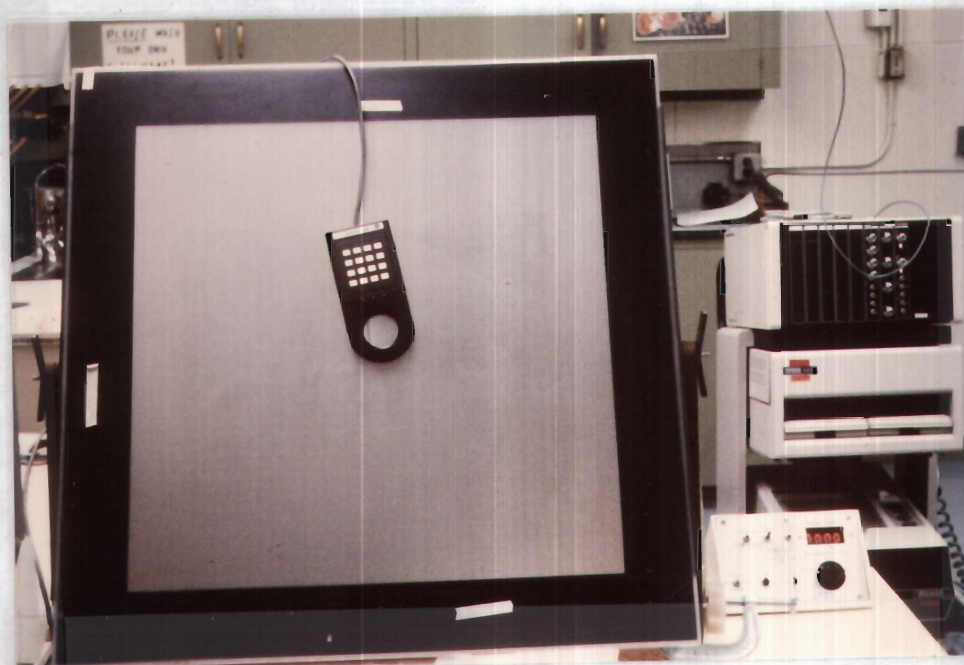


Figure 16. Digitizer platen

edge were the same points. The reasoning for this will be made clear in a later section of this paper. Veneer shapes were handled in a similar fashion with respect to wane.

Even though the location of other types of veneer flaws like thin wood, sound knots, pitch streaks, white spec, etc., are not tracked during standard scanning, they were located during the digitizing process. With all defect information at hand, standard clipped pieces could be assigned a grade, or the entire clip sequence could be guided by grade recovery criteria. The current effort is limited to the capacity of the optical scanners. Future study into the economics of grade clipping may provide the incentive for developing other means of veneer scanning equipment for commercial application. At any rate, the data base and general simulation logic for this type of study is available with the current research.

Other pertinent information was also gathered during digitizing. Defects were coded by type. The terminal points of contact or match points between adjacent pieces were also identified. These are key points used in the reconstruction program. A true representation of each piece was attempted. However, sometimes one piece of veneer would hide another beneath it; especially during round up. In these cases, a reasonable guess as

to the hidden shape was produced on tracing paper. The traced outline was then digitized, with defect patterns exhibited by surrounding pieces reproduced. Some joining of pieces was also necessary. For example, when jam ups occurred at the clipper, often the pieces were clipped, then reclipped, the belt was reversed, and then the pieces were reclipped again. No logical pattern for reconstruction was possible. The mass of clipped veneer was sorted out on tracing paper, with each piece rejoined during the trace. Then the tracing was digitized in the normal fashion. Breaks in the veneer not causing complete separation were coded as cracks.

Digitizer Computer Programming

The digitizer was interfaced with a Digital Equipment Corporation MINC microcomputer. A short program written in FORTRAN-IV controlled the digitizer responses. This was an interactive program which guided the user through the digitizing process by a series of video prompts. Questions, such as: "Is this a new piece?", or, "Does this piece have any defects?", required a specific action by the user. Depending on the response, control is transferred to the appropriate program address. The program also decodes the signals received from the digitizer and formats this information. The information is easily retrieved and manipulated by

other programs that follow.

Another aspect of the programming involves the correcting of digitized points so each piece's base or clip line is parallel to the platen's. This skew correction factor is computed for each piece and is applied to all points within the piece. When the piece happens to be the last piece in a ribbon section, the base and trailing clip lines do not correspond, in all other instances, these lines are the same. This fact makes the rejoining of the pieces less cumbersome. Initial data storage was on floppy disc.

Reconstructing the Ribbon of Veneer

Matching points

Reconstruction means joining the individual digitized pieces of veneer back together. When this is performed on a peeler block basis, then the entire peel can be seen prior to clipping. Basically, the program identifies corresponding match points coded during digitizing, then adjusts the coordinates of one piece of veneer to the next piece, and so on until all pieces are aligned. Figure 17 shows two pieces of veneer. Coordinates (X_1, Y_1) and (X_2, Y_2) represent matching points. If all other points on the two pieces; defects, wane, etc., are referenced to these match points, one may change the location of the match points, then relocate all other points by the same X and Y distances moved by the match points without changing their relative position within the piece. If point pairs (X_2, Y_2) are equated to point pairs (X_1, Y_1) , then one can say match point 2 has been moved by -5 in the X direction and +4 in the Y. If all other points on sheet II are moved by the same X and Y distances, then the pieces become "joined" together. Once joined, the coordinates of piece II are reassigned values in ascending order, starting with the match points. Successive pieces are reconnected in the same manner. With each natural break

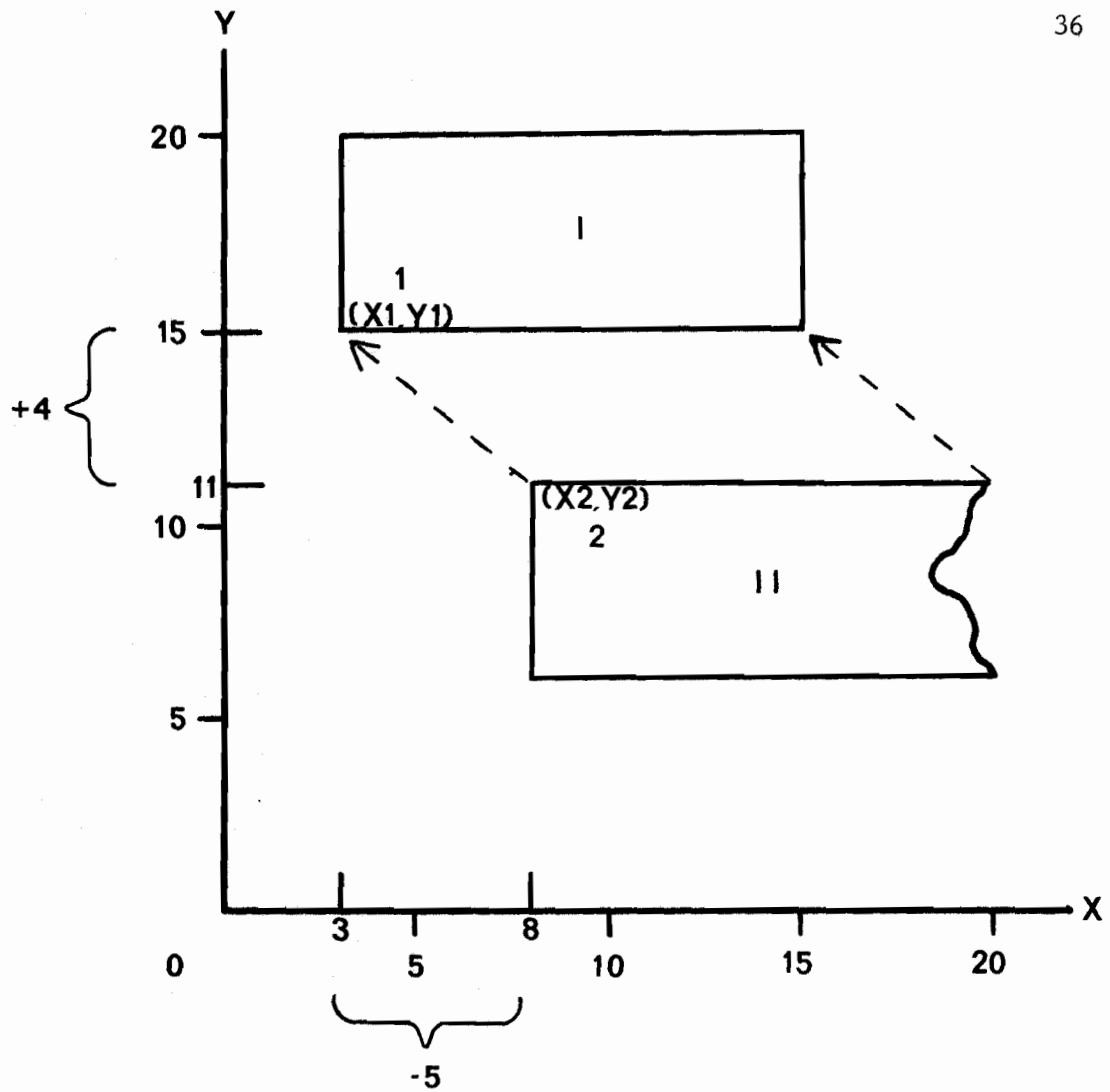


Figure 17. Graphical presentation of reconstruction programming logic

in the ribbon, the numbering sequence begins again at (1,1). An example of the digitized veneer followed by a computer generated plot for the same ribbon section appears in Figures 18 and 19.

Input data files

During digitizing, each piece of veneer is treated separately, with no regard to adjacent pieces. Similar point sets are generated for each piece. A general starting point is required, as well as a specific order for each digitizing sequence. That is, first digitize the veneer perimeter, then the wane (if present). Next digitize the defects. All this data is sequentially stored in a very large file. Groups of data for each clipped piece are segregated from another by a unique code. Defects are separated by -44, pieces by -99, etc.. In this way, control is passed to specific programming sections depending on the delimiter encountered. Defects are coded by type:

- 1 = sound knot
- 2 = open void
- 3 = crack
- 4 = pitch streak
- 5 = white spec.

When, in the defect manipulation program section, a

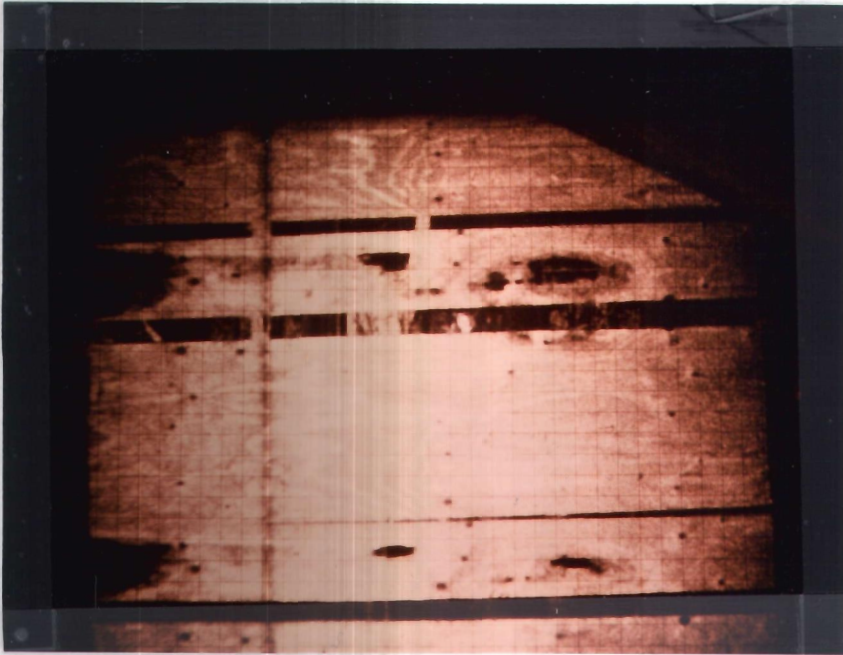


Figure 18. Filmed veneer on the digitizer platen

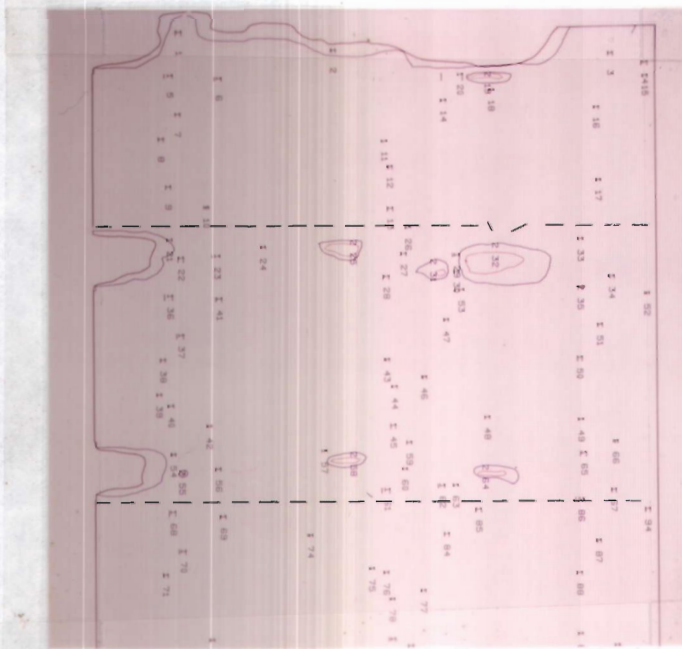


Figure 19. Computer generated plot of the film

certain defect code is read, control is transferred accordingly.

Manipulation of these very large data files was not possible on the MINC computer due to memory limitations. To accomplish this task, the Oregon State University host computer, a mainframe CDC CYBER 170/720, was used.

Considerations

Transfer of digitized data to the CYBER was accomplished with the aid of DATALINK. This is a software package which runs under different computer hardware and/or software configurations for the purpose of file transfer.

The programming logic as layed out appears very straightforward. However, nuances inherent in the digitizing process and equipment proved to be formidable obstacles.

Error introduced by the human element required large amounts of corrective programming. It was impossible to precisely represent the filmed veneer in digital form. In all cases, the distances between adjacent matching points were different. Similarly, the cross grain measure was always unequal. If one assumed the initial piece in the ribbon to be accurately digitized with regard to match points, all successive pieces could be adjusted accordingly. Straight, smooth, continuous

edges on the ribbon could be guaranteed. Unfortunately, closure at the end of the ribbon would vary by several inches. In addition to human error, optical distortion in the movie camera, and the fact that some pieces were actually clipped skewed, made the correction and adjustment portion of the program even more complicated. Pieces had to be forced into shapes approximating rectangles. Similar measures had to be applied to defects as well. Of course, if the piece was not a rectangle to begin with, the programming should not make it one.

These problems were tackled in two stages. First, if two clips existed, the lines were forced parallel to one another. The edges were then adjusted in or out to reflect a rectangular shape. Key information regarding these adjustments was stored and applied to defects when necessary (Figures 20 & 21).

Another problem arising from sources already noted was manifested by a series of short length pieces. When scaled up to full mill size, pieces obviously clipped as full-length measured less than the 101 inches required. The reclip programming logic would interpret these pieces as fishtails and clips would fall accordingly. It is possible that some pieces were actually less than full length, however, they would probably be pulled during the green sort as full length. The assumption was made

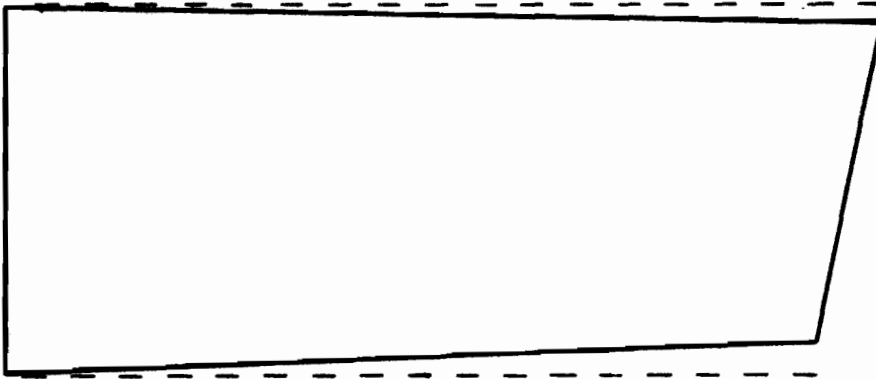


Figure 20. Skewed clips required adjustments to make them parallel

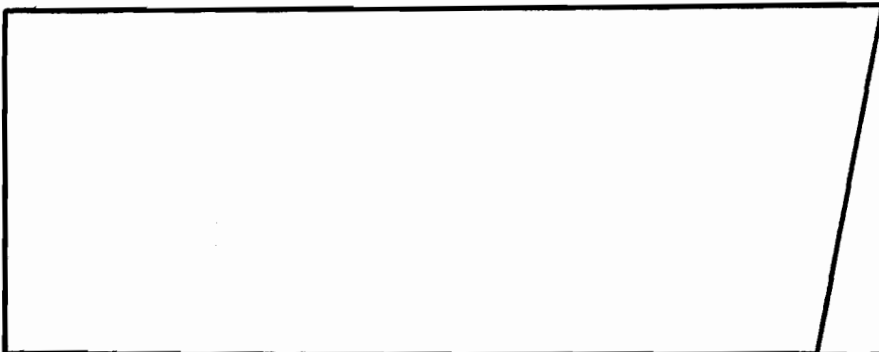


Figure 21. Match points must be corrected

that if a piece was clipped to a standard-sized dimension, then it must have met the across grain criteria when scanned in the mill. Coordinate adjustments were made to reflect this assumption.

So far, this discussion has focused primarily on standard-sized piece adjustments. What about round up where good edges are seldom found? To affect the adjustments and corrections previously described, the clipped veneer had to be somewhat rectangular in appearance. Remembering that if one assumes the initial piece in each ribbon section to be accurately digitized, then trailing pieces could be forced into the correct position by matching already adjusted points. In other words, apply within sheet adjustments independent of other pieces, then equate adjacent match points. The initial clip is critical for the positioning of all trailing, matched pieces. One may assume that an original clip occurs only in an attempt to recover a merchantable piece of veneer. Therefore, the length of the piece must be at least the fishtail setting. Unfortunately, round up is oftentimes clipped to convenient sized lengths to facilitate chip recovery. Some of these problems are overcome by the tracing procedure described in the digitizing section. Even so, the with grain measure may still be too short. To overcome this problem, specific tolerances were applied. For example,

if the width grain measure at a clip was at least 96 inches, then, at this point in the piece, the distance was assumed to be full length. A correction factor was computed and applied to the X values contained in the matching point pairs. This adjustment was halved, with a plus value applied to the right hand side and a corresponding minus value to the left side.

A plot of the reconstructed ribbon was generated on the Zeta Drum plotter located at the O.S.U. Computer Center. This proved a valuable tool for assessing digitizing and the application of reconstruction adjustments. The plot program is written in FORTRAN V and requires several CYBER system subroutine calls. With each break in the ribbon, numbering returns to (1,1). To prevent one ribbon section from being superimposed over another on the plot, a constant value was applied to each coordinate point following the initial ribbon section. Ribbon pieces then appear to be an equal distance apart.

To review, first width grain clips are tested for length and adjusted if necessary. Next, parallel clips are insured. Finally, right and left edges are made perpendicular to the clip plane. Any adjustment to shape is proportionately applied to defects.

Programming

The main purpose of the reconstruction program is to prepare the digitized data for use by the "reclip" program. Since the clip logic will simulate the actual scanner controlled clipping sequence, the laboratory data must be reasonably close to scanner generated information. Wood "passing" through the clipping program must begin at the start of the peel, with all associated wood characteristics presented in a continuous, ascending flow. Like the scanner, the clip program logic must be able to recognize left and right fishtails as well as relative location of defects within the ribbon. In addition to performing these tasks, the reconstruction program creates files that enable the production of the Zeta plot. After comparing the plot to the filmed veneer, any differences noted on the plot are located in the digitized data file and edited to conform to the film. If necessary, pieces are redigitized.

To simulate the scanner's ability to distinguish right and left hand sides of a piece, all data files are separated into left and right sides. The defects and shapes are sorted on the basis of minimum and maximum Y values. Points falling to the right and left of the plane connecting these Y's are contained in the right and left side files respectively. In the case of

cracks, the upper edge of the defect goes in the right and the lower to the left side file. After the left/right sort is completed, a rough outline of the piece shape is determined. Four general veneer shapes must be considered (Figure 22). It is impossible to precisely record each veneer shape in the digitizing process. Start and stop points never coincide. Often, redundant points are recorded. When pieces have match points, coordinates extending beyond the clip line are incorrect and must be removed from the data files. Also, if the digitizer is allowed to drift, or remains in one place for a moment, then extra points are returned and written to the data file. Again, these coordinates must be edited out.

If a piece is described by just four point pairs, say a half sheet, then the left and right side shape file contain only two coordinates. The match points describe the shape (Shape III). They are also the minimum and maximum Y of the piece. Only the X values are different. Similarly, when the first clip occurs on an intact ribbon section, only one clip (two match points) are exhibited by the piece (Shape II). The Y values at the clip line are equal, but the X's are not. Conversely, the last piece clipped on a ribbon section has but one clip and it is located on the leading edge rather

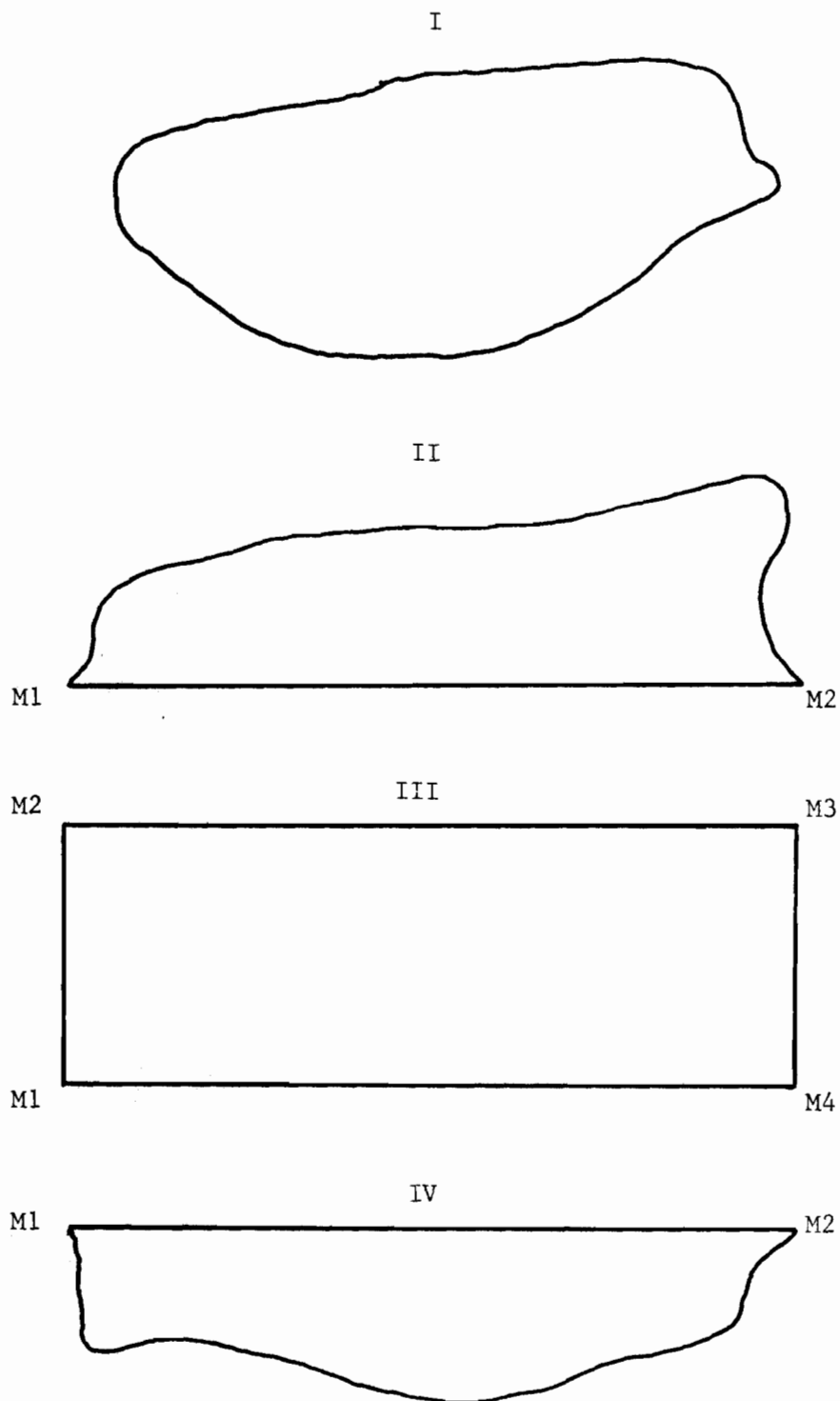


Figure 22. Four general veneer shapes

than the trailing edge of the piece (Shape IV). Shape I has no clips, thus no match points. Regardless of the shape, all pieces are sorted into left and right sides. Following the sort, all Y values are renumbered; reversing the minimum and maximum Y's. Remember, the digitizer sets the origin at the lower, left corner of the platen. The pieces fall within the upper right quadrant. Visualize the shapes as moving up the platen as digitizing continues. Now if one watches the veneer as it passes under the clipper, the first points seen would be the leading edge or the maximum Y of the piece. To correctly portray the action of the veneer moving over the scanner heads, the digitized Y's, representing cross grain measures, must be reordered. The X points do not change, except where skew correction adjustments are necessary. Defects are just sorted to their respective sides. Proportional adjustments are made if similar corrections were required on the shape.

Up to this point, the program is working with values expressed in terms of the digitizer. A standard size, full sheet measures only about 16 by 8 inches. Before the clip program can work with the data, the coordinates must be scaled up to full mill size. Full length pieces are tested for full sheet tolerances and adjusted if required. Those obviously clipped as full length but for reasons previously outlined fall below

this measure, are "stretched" to the required distance. During the sort, maximum and minimum X coordinates are tracked for each piece. If the difference between these points is greater than or equal to 96 inches, then the piece is expanded in the width grain dimension to meet full length standards. This adjustment is crucial for the initial piece of each ribbon section, since all successive pieces are joined to the first.

As the pieces are scaled up, the points are moved farther apart. New coordinates must be computed to fill the gaps. The new coordinates are added at .02 inch increments to insure a "scanning" accuracy at least as good as the optical scanner at a line speed of 200 fpm.

Information is stored on file in a form similar to the digitized data, except the reconstructed data is segregated by ribbon sections rather than by individual clipped pieces. If one were to watch the peel at the lathe, imagine the data file as one intact section of wood. The largest segment of data will correspond to the peel after rounding up of the block is accomplished.

Twenty-six separate data files are created by the reconstruction program. Seven are "scratch" files that disappear after program execution. The CYBER system sort requires the creation of these temporary files. Four defect files, left and right shape and their cor-

responding wane, are necessary for the plotting program. Once the plot is generated, these files are purged. Additional files store sorted defect and shape coordinates in a form necessary for grading purposes. An information file is also created that holds pertinent data used as the basis for adjustments and corrections. The remaining six files serve as input to the clip program. Four files contain shape and wane coordinates for both right and left sides. The principal defect parameters, such as minimum and maximum X and Y as well as data specific to defect type, are stored in the final two files.

Simulating the Green-end Clipper/Scanner

Understanding the computer model

An attempt was made to simulate the reading process performed by an optical scanner as a ribbon of veneer passes by. As specific criteria are met, clips are made. Since this is a simulation, exact functions cannot be reproduced. However, a reasonable facsimile can be produced and specific judgements and conclusions can be made on the basis of the simulation.

The clip logic begins by considering the type of edge exhibited by the piece of veneer. In keeping with the actual scanner logic, a piece must have at least one good edge to be a potential fishtail, unless double fishtails are allowed. The program begins by reading shape and wane data until this condition is met. Next, the width grain length is computed and tested against the preselected fishtail minimum length. If two good edges are found, the program looks for standard length measure. Even with two good edges, the piece may fall short of the required distance. In this case the piece is considered a left fishtail for purposes of accountability. Once a piece has been identified as potentially merchantable, the defect summary file is read to select the initial open flaw falling on, or downstream from the current "scan" position. The scan line to the computer

is just the current Y value of a specific point pair read from the shape file. Locations of defects are determined by their minimum Y value in relation to the current scan. This logic simulates the across grain read of the optical scanner as the veneer moves by (Figure 23).

Once a defect is found to fall within the purview of the scan, specific parameters of the flaw are temporarily stored in memory. If a part of the defect falls on the current scan, then immediate action is required to determine whether the flaw should be removed. As the scan continues, the cross grain distance is continuously computed and tested against the maximum piece length allowed. The scan halts when this measure is met, when a defect is identified as clippable, or when the potential piece type changes.

When the scan is stopped, a clip may not necessarily be ordered. Perhaps the clippable defect is located close to an edge. If the distance from the defect to the other edge is at least fishtail length the scan will continue. If the piece type changes but the minimum across grain length has not been met, then a clip ordered would generate an unmerchantable piece of veneer (Figure 24). Specific economic information must be considered before a clip decision is made. Unless this

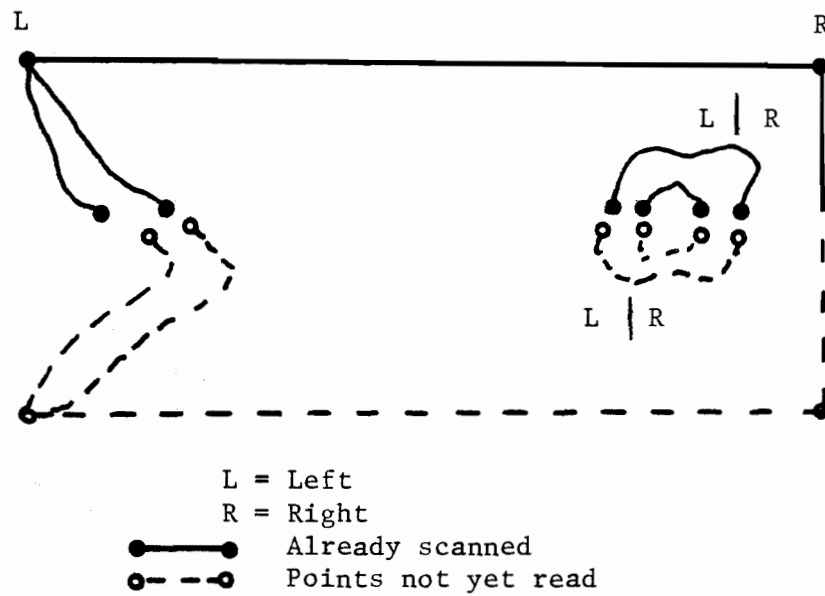
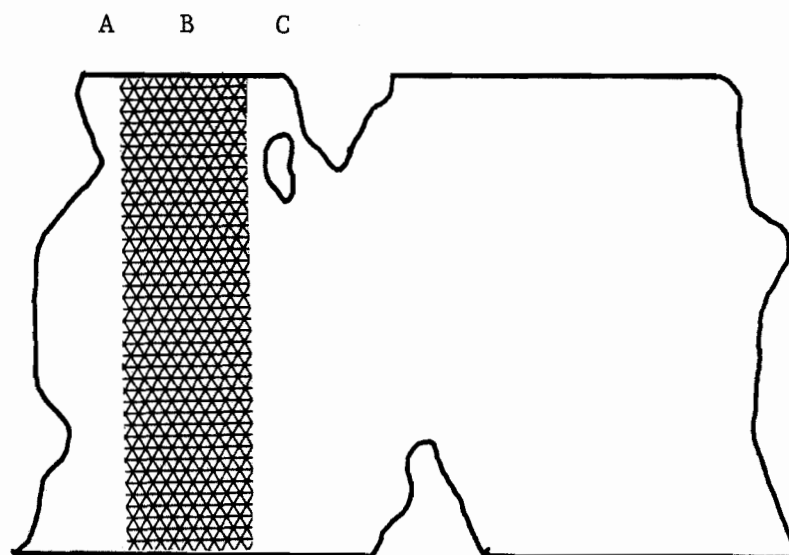


Figure 23. Reclip programming simulates the optical scanner function

function is cancelled by other management inputs, the piece with the highest value should be recovered. For example, let's say we have a random strip measuring eight inches in length. Suddenly a large defect is encountered on one end of the ribbon. Should a clip be made at this point? The answer depends on whether fishtail requirements are met. If a fishtail is possible, will it remain a fishtail long enough to meet minimum length? If not, could some of the random wood be left attached to the fishtail to enable recovery of both types?

To select the appropriate action raised by these questions, economic criteria must be included as input to the clip decision. Also, an ability to "look ahead" on the ribbon must be possible before the choices presented can be resolved. Since the optical scanner is located upstream from the clipper, a preview of veneer is available as clipper input logic. About thirty inches of veneer can be seen ahead of the clipper. The clip model emulates this function by virtue of computer memory. However, much more wood material can be seen before a clip is ordered. Theoretically, the entire ribbon could be stored in memory and a perfect clip sequence could be made. In practice only a distance equivalent to two full sized sheets plus minimum strip need be considered. All possible clip combinations can



- A. Trash
- B. Full width
- C. Left fishtail

Figure 24. Clip logic control in the computer simulation

be made, resulting in a clip pattern producing the highest value.

Since this is a computer simulation, actual conditions present in the mill which have a bearing on the clipping operation are not considered in the model. No belt slippage, line clock wear, or jam ups occur. The computer never loses track of defect position; consequently, no flaw centering need be applied. All of these facts must be kept in mind when comparisons are made between the simulated clip and the actual mill clip.

A running tally of piece types and sizes is maintained. This data can be presented in any number of ways to express value, wood yield, within block clip comparisons with each run, and between block summaries.

Reclip programming

The term "reclip" refers to the computer simulated clip run rather than the actual mill clip. As mentioned earlier, the program begins by considering the characteristics of the leading edge of the peel. Unless the double fishtail option is selected, at least one edge must be good. Good means that the piece exhibits a straight side, and the side is perpendicular to a potential clip line. Usually, this situation is one in which no wane is present on the edge. The ability of the program to determine the straightness of an edge re-

quires successive tests in the movement of the X coordinate with each read of the shape data. If the current X does not differ from the last X read by more than 0.10 inch, then the side is considered to be straight. A similar window of tolerance is established to test the presence of wane. Piece shape and wane files are both read at the current Y scan. If the difference in X values at the scan is less than 0.10 inch, then the piece is considered to be free of wane. Flags are set indicating the status of the edge for both parameters. In this way, the shape pattern can be tracked for all edges of the veneer ribbon. Wane flags are stored in memory and are not used for the determination of clips. Later, the simulated clips are checked to see if they fell on any wane portion of a merchantable piece. This procedure provides a type of sensitivity check on a particular margin setting selected by the user. By varying this setting through several runs, an optimum tolerance can be determined for a specific peeler block. If the double fishtail option is selected, length measurements commence immediately, otherwise, at least one good edge must first be found. The length or width grain measure is tested against the desired distance selected by the user. When the minimum length required for a merchantable piece is met, a potential

recovery flag is set for the particular veneer type identified.

The margin setting moves the clip by a predetermined distance to account for the presence of wane around an open flaw. For the purpose of this program, it is also used as a measure for testing the actual length of wood from a good edge to an open flaw if this condition allows a fishtail measure. In the first case, the clip is adjusted in the Y direction. For the fishtail measure, the margin is applied in the X plane. Two settings are available for margin adjustments. The smaller, alternate adjustment is applied after two full sheets have been clipped. Usually by the time this condition has been satisfied, the scan has reached the continuous ribbin section of the peel.

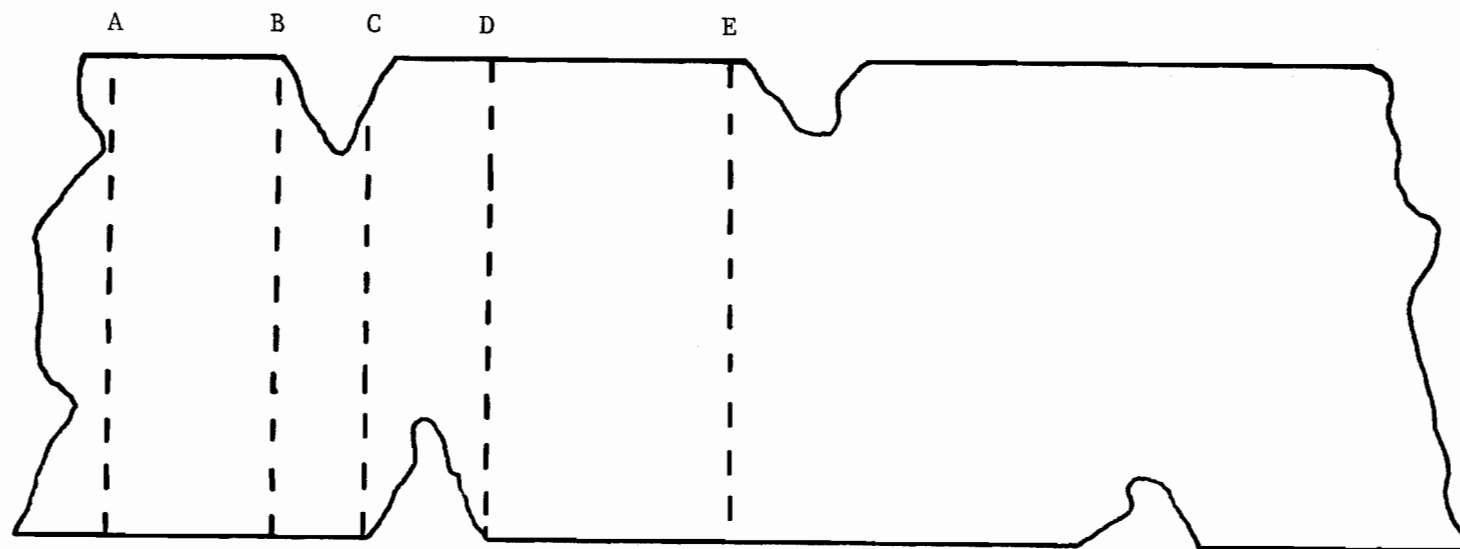
Once a potential merchantable piece has been identified, the scan or Y value at this point is stored in memory. As long as the wood continues to meet the length measure, the piece flag remains unchanged. The width or across grain measure is continuously monitored by subtracting the initial scan from the current sweep. When the maximum scan length is achieved (2 fulls + margin) the scan halts and a clip decision is made. A full or fishtail may be produced depending on the type flag currently set. When the beginning or end of a de-

fect is encountered a clip determination may also be necessary. If the defect falls short of the tolerances, then no change in piece type occurs and the scan continues. However, depending on the location of the new defect, another piece flag may be set. If the current flag is set to a left fishtail and the new defect falls on the extreme right edge of the piece, no new piece flag may be required. No change in flag is necessary unless the length requirements are not met. Now if the new defect is situated such that the current piece is no longer available, then the piece flag changes and a clip must be considered.

When trash is identified as the initial wood type, then a recoverable piece is encountered, a clip is ordered. Similarly, when the piece flag changes from a merchantable piece to trash, a clip is also ordered. In these two cases, no consideration is given to the length measure. A clip is always ordered when moving from trash to good or vice versa. Later, the scan distance is tested for merchantable length. If trash is not a component of the clip condition, the choice of piece type to recover comes down to which one is more desirable in terms of revenue. If nothing changes during the scan, the clip decision is just a matter of clipping to the maximum allowable piece length.

Let's say that a full width piece flag has been set. With the next scan, a fishtail is found. This is a frequent occurrence in round up. Memory has stored the initial piece type as well as the beginning and end, (start and stop Y). The fishtail flag is set and the scan continues until some other condition is presented that would stop the scan. In this way, a "look ahead" or preview of coming wood function is activated. Several clip options are possible in the above scenario. If both piece types meet their respective length requirements, no further decisions are needed. A clip is ordered to separate the pieces. Conversely, if neither piece meets the measure, then the random length may be joined to the fishtail if this results in meeting the fishtail setting. An economically based decision is made to determine the recoverability of both types when the fishtail is less than the minimum fishtail setting. When the random length is in excess of minimum requirements, the extra wood may be enough to make the fishtail recoverable if it were attached to the fishtail. If the value of recovering both types is less than the value of the clipped random and the loss of the fishtail, then this clip pattern is preferable. These circumstances are found many times throughout the round up portion of the peel. Of course, the situation could be reversed, with the random following the fishtail.

The look-ahead clip logic continues throughout the peel. An initial piece flag is set, the look ahead flag is set with a change in piece type, a clip decision is made, then the current piece flag becomes the look ahead flag and the new piece becomes the new current piece flag (Figure 25). This exchange procedure is not necessary if the piece type remains the same until maximum piece length is found. For example, once the scan reaches the continuous ribbon section of the peel, fewer clippable defects are encountered so successive full sheets are generally clipped. With the selection of the double fishtail option, a departure is made from the standard clipper/scanner logic. The good edge requirement is no longer necessary. The scan logic remains the same with this exception. However, a new situation must be considered in the look ahead scheme. Under standard clip logic, if two fishtail types are adjacent, the clip is automatic. But when the double option is selected, a more comprehensive effort is made to recover all possible merchantable wood. The programming logic remains basically the same. Without the double fishtail, the lowest value wood recovered is the left or right fishtail. With the double option allowed, additional trim requirements needed to convert the piece make this the worst case. In other words, a hierarchy of preferred piece



A INL = 4
PFLAG = 1 Clip Trash

B INL = 1
PFLAG = 2 Wait

C LH = 1
INL = 2
PFLAG = 3 Wait

D LH = 2
INL = 3
PFLAG = 1

Clip Left Fishtail
Clip Random

E LH = 3
INL = 1
PFLAG = 2

Clip Right Fishtail

1 = Full Width
2 = Left Fishtail
3 = Right Fishtail
4 = Trash

Figure 25. "Look ahead" clip logic with replacement

types is established. Trash falls into the lowest category.

Like the mill clipper, the simulation program accepts numerous user input parameters. These inputs are treated as variables within the program. Their value during a clip run depends on initial assignment statements prior to execution of the program. This data is actually read from another input file. In this way, several simulated clip runs can be produced successively with a new read of the parameter input file. This file also contains the value of veneer types.

Clip data is updated for each natural break in the peel. All tracking information is zeroed. Piece flags are reinitialized. The program logic returns to initial piece shape considerations. A clip sequence sort is performed following each peel. Those pieces flagged and clipped as merchantable pieces are checked for length requirements. Remember, when trash is encountered, a clip is ordered without regard to the preceeding piece length. If the clipped piece does not meet minimum length requirements, a reassignment to trash is made. Several trash assignments may follow in succession. The program merges these into one clip.

The raw data clip file consists of four columns. The first column holds the piece flag. Columns two and three reflect the Y coordinates for the leading and

trailing clips. The intact ribbon section separated by natural breaks is found in column four. From this simple arrangement, any number of possible output formats can be produced. Total count and volume can be output for each block. Comparisons between successive runs on the same block or between blocks can be generated.

In addition to clip information, a cumulative tally of the number of times a clip falls on the waney portion of the veneer can be output. Also, if any short blocks are peeled, a message to that effect is also available. A short block is defined as one that exhibits two straight edges, but fails to meet the full length sheet requirement.

RESULTS AND DISCUSSION

Increases in the value of the veneer mix and/or improvements in total wood recovery were found in each simulated clip run. As expected, volume recovery increased anywhere from 0.5 to 3.3 percent. Maximization of the higher value sheets was also accomplished. Even though an increase in volume recovery was noted with the double fishtail option, an economic analysis will probably show this strategy falling short of break even.

It is important to keep in mind that each run is compared to the base or control clip when judging improvements in veneer value or wood yield. Control run parameters, except veneer values, are identical to those used in the study mill. Table I lists the scanner settings for the control run. With each successive run, only one clip control parameter was changed. Note that the panel option was turned off for the control run. One can expect a higher count of half sheets had this mode been selected. A summary of all runs with piece count and volume recovery for each component of the veneer mix is presented in Table II. The percent distribution of veneer type is found in Table III. Table IV reflects the percent change in veneer mix against the

 Table 1. Control run scanner settings (inches)

Sheet Size	With grain	Across grain
Full	100.0	54.0
Half	100.0	26.6
Minimum Strip	100.0	6.0
Fishtail	51.0	6.0

Flaw Limits

Open Flaws	2.0	0.6
Edge Flaws	2.0	3.0
Cracks	36.0	2.0

Clip Control

Margin	
Normal	2.0
Alternate	0.6
Panel	No
Double Fishtail	No

Veneer Values (\$/M sq.ft., 1/10" basis)

Fulls	25.50
Halves	24.25
Randoms	14.75
Fishtails	12.00

Table 2. Summary of veneer mix (count/sq. ft./1/10" basis)

Scanner Setting	Fulls	Halves	Randoms	Left/Right Fishtails	Double Fishtail	Total
Control	198 7425.0	14 258.608	140 1647.398	130 532.215	---	--- 9863.221
53" Full	203 7484.0	10 184.721	140 1662.509	131 536.102	---	--- 9867.332
5" Random	198 7425.0	15 277.080	144 1647.696	143 561.507	---	--- 9911.283
1" Margin	198 7425.0	16 295.552	158 1840.625	135 521.204	---	--- 10082.381
0.5" Margin	199 7462.5	15 277.080	170 1928.239	138 516.157	---	--- 10183.976
Panel	198 7425.0	25 461.800	118 1432.187	130 532.215	---	--- 9851.202
Double Fishtail	198 7425.0	14 258.608	138 1646.874	111 470.965	55 202.375	--- 10003.822

base run for each adjustment to clip control.

Reducing the full sheet size (across grain distance) one inch to 53 inches did not produce a significant increase in volume recovery. However the value of the clip is greater as evidenced by the addition of five more full sheets. A large spread in the price of fulls and halves would reflect a substantial increase in economic value. The extra inch of full length good wood was redistributed to produce the highest value piece. Fishtail recovery remained unchanged.

When the minimum strip is reduced to five inches, a 0.5 percent increase in volume recovery occurred. This figure is lower than expected. With a base run margin setting of two inches, at least nine inches (two times margin plus minimum strip) must be found before a random is clipped. Since the strip measure is reduced, an extra inch of full length wood is available. In this study the additional wood produced a seven percent increase in halves, while random strip recovery remained unchanged. A 5.5 percent increase was found in fishtails. Apparently the full length wood remaining after clipping a half was not enough to meet minimum strip requirements. Consequently the excess wood was left attached to an adjacent fishtail.

As expected, increased wood yield followed with reductions in margin setting. Again, an economic emphasis

Table 3. Summary of percent distribution in veneer mix

Scanner Setting	Fulls	Halves	Randoms	Left/Right Fishtails	Double Fishtails	Total
Control	75.28	2.62	16.70	5.40	---	---
53" Full	75.85	1.87	16.85	5.43	---	0.042
5" Random	74.91	2.80	16.62	5.67	---	0.487
1" Margin	73.64	2.93	18.26	5.17	---	2.222
0.5" Margin	73.28	2.72	18.93	5.07	---	3.252
Panel	75.37	4.69	14.54	5.40	---	-0.122
Double Fishtail	74.22	2.59	16.46	4.71	2.02	1.426

Table 4. Percent change in veneer mix against base run

Scanner Setting	Fulls	Halves	Randoms	Fishtails
53" Full	0.007	-28.57	0.09	0.73
5" Random	0.00	7.14	0.00	5.50
1" Margin	0.00	14.28	11.17	11.73
0.5" Margin	0.005	7.14	17.05	-0.02
Panel	0.00	78.57	-13.06	0.00
Double Fishtail	0.00	0.00	-0.00	26.52

is placed on the recovery of highest value pieces.

When the panel option is selected, a 78% increase in half sheet recovery occurred. Randoms were reduced by thirteen percent. The primary objective of the scanner is to maximize the recovery of halves without regard to volume recovery loss. This strategy produces a higher value veneer mix but lower wood yield.

A 26.5 percent increase in fishtails was found when the double fishtail option was allowed. However, total recovery only increased by 1.4 percent. These results suggest that standard right and left fishtail widths were reduced with the additional wood going into double fishtails.

CONCLUSIONS

Conclusions are first reported with regard to veneer recovery implications. A general discussion of the computer simulation follows.

Wood material currently being lost at the clipper can be recovered. The improvement in wood yield follows a pattern of maximizing the higher value full length pieces. No additional processing is necessary as is required for fishtail trim. Double fishtails would require even more handling since each piece would be individually placed in the trim saw for conversion to a usable piece.

Minimum strip and margin settings are more critical in the round up portion of the peel. By reducing these limits, more round up material is recovered. Once into the continuous section of the ribbon, the frequency of bad edges and clippable defects declines. These facts are well known as evidenced by the availability of the alternate clip control selection once a specific number of fulls has been clipped. The 5.5 percent increase in fishtail recovery is a direct result of reducing the minimum strip and margin limits. The minimum strip setting also applies to minimum cross grain fishtail width. It is difficult to ascertain which setting

is more sensitive to the recovery of a specific piece type. Additional runs allowing the change of both clip parameters is necessary.

As the margin limit is reduced, clips will fall within the actual waney portion of the round up. The overwhelming volume of fishtails and randoms are also clipped from round up. Since wane is normally found on the edge of pieces, the amount of thin wood remaining will be reduced as further conversion takes place. Fishtails will be used as core stock. Randoms may be trimmed to facilitate edge glueing and stringing. These pieces are almost always used as inner plys in the finished plywood panel. In the event some wane is still present on the piece, the amount must be within allowable specification tolerances. For example, areas of wane up to 1.5 inches across the grain by eight inches along the grain are permitted in C grade faces, backs, and inner plys. Further study is needed to determine the actual amount of wane left on merchantable pieces as the margin adjustment is reduced.

When the clip is controlled in part by the panel switch more half sheets are produced. Of course, randoms and overall veneer recovery may decrease. Mill managers must look to their order files as well as the spread in prices for veneer types when deciding on the panel or random selection.

The increased recovery of 1.4 percent and 26.5 percent in fishtail yield occurring when the double fishtail option is selected, probably falls short of the economic break even point to warrant introduction. Once chip prices and additional labor needed for the two trims on the double fishtail are considered, the benefits of additional recovery may be offset by lost revenue and increased conversion costs.

Computer simulation techniques provide an excellent opportunity for analyzing the implications of management selected scanner inputs on veneer recovery. Wood material is not sacrificed as would be the case in actual trial and error mill clips. However, the acquisition of the data bank is a tedious and time consuming procedure. Because of the dynamic nature of wood, programming was extremely lengthy and intricate. Consequently, program execution requires a lot of time and money. But over the long run, the ability to play "what if" games can result in an optimum clip strategy that guarantees maximum volume and/or economic recovery.

RECOMMENDATIONS

Veneer Recovery

The following recommendations must be weighed by each mill manager against the condition and capabilities of their green veneer production facility. Furthermore, the assumption is made that the clipper can respond consistently to scanner directives.

In the general case, clip control adjustments must be reduced by as much as possible. If a mill can consistently clip 53 inch fulls and still meet finished panel dimensions, an increase in the full component of the veneer mix will be realized. Similarly, the minimum strip setting should be reduced to a point where it is no longer economically feasible to handle the smaller pieces. Managers must decide how much thin wood will be tolerated and reduce margin limits appropriately. If the clipper scanner is operating correctly, the incidence of clips falling on the actual flaw is very small. The panel mode should not be selected as the preferred strategy in clip control just because more halves are produced. This mode depends on veneer requirements needed to fill existing order files and opportunities to sell excess inventories of specific piece types. The double

fishtail option does not appear cost effective in the purview of this study involving the small diameter class log.

It is imperative that mill managers understand the interaction of clip control adjustments on overall veneer recovery. No single clip strategy can be applied across the board to all mills. An optimum clip hierarchy depends on a specific log mix, veneer market value, equipment, and operator knowledge found at each mill.

Computer Simulation

This project has been agonizingly long for several reasons. First, the Tymstudy projector used in the digitizing procedure proved woefully inadequate. At least nine months were lost due to down time. Many hours of re-digitizing were necessary to correct for the inability of the projector to hold a single frame long enough to record all aspects of a piece. Often the shape was completed, then the frame advanced leaving the defects in an entirely new location on the platen. Another method for projecting the veneer image onto the platen, or a different make of projector must be secured. On a related subject, a finer grained color film would greatly speed up the identification and discrimination of defects during digitizing.

Run times were totally dependent on other time-sharing commitments operating simultaneously on the host computer. Often a run was initiated early in the morning and not completed for six to eight hours. Under optimum conditions the average run required approximately ten minutes. Reconstruction execution often ran over several days. Efficient use of research materials and time could be enhanced if research activities were conducted on a computer reserved for this purpose.

For the sake of conserving programming requirements as well as digitizing time, some assumptions may be appropriate with regard to selected veneer pieces. In those cases where the shape of a piece in no way controls a clip decision, an exact duplication of the veneer may not be necessary. Small pieces and irregular shapes found in round up could be excluded.

IMPLICATIONS FOR FUTURE STUDY

Further research is needed for a complete understanding of veneer recovery implications when several clip control variables are changed simultaneously. Sensitivity measures for each input may be forthcoming. More peels from other diameter classes need to be investigated. Perhaps a change in margin setting is more critical in larger blocks than in smaller diameter class blocks.

This study concentrated on simulating the functions of the optical through beam scanner. However, during digitizing all parameters of the peel were recorded. With information on pitch streaks, white spec, and sound knots, a clip simulation could be guided by grade criteria. When scanners are able to detect these "intact" flaws other areas, such as automatic sorts and veneer upgrading with automated controls may become feasible.

A logical extension of the current reconstruction effort would be to rewind the peel electronically to original block form. Assuming scanners are able to locate interior flaws, blocks could be positioned in the lathe to enhance the recovery of a specific veneer mix at the clipper.

GLOSSARY

Accurately clipped recoverable - veneer lost as a consequence to a properly placed clip.

Digitize - conversion of analog information to digital form (filmed veneer to X and Y coordinates).

Full ribbon width - minimum acceptable with-grain dimension.

Full sheet - determines the across-grain dimension of the large sheet.

Fishtail width - amount of good wood that must appear at one edge or the other of the ribbon.

Half sheet - determines the across-grain dimension of the small sheet when the available good wood is less than a full. It also sets maximum fishtail size.

Margin - sets clip pull back allowing the mill to adjust the closeness of a clip to a defect or edge.

Minimum strip - sets the smallest across-grain measure allowed for a random strip or fishtail.

Round up - veneer produced prior to the formation of a cylindrical peeler block. Usually that part of the peel preceeding the recovery of two full sheets.

Wane - thin wood surrounding defects or on the edge of round up veneer.

LITERATURE CITED

1. Kokus, J. and Marcin, T.C. Forecasting Longterm Housing Demand in an Environment of Uncertainty. Paper presented at the Southern Regional Science Association Annual Meeting. Rosslyn, Virginia (April 1974).
2. Marcin, T.C. 1977. Outlook for Housing by Type of Unit and Region: 1978 - 2020. USDA Forest Service Research Paper FPL 304. Forest Products Lab., Madison, Wisconsin. 11 pp.
3. Ruderman, F.K. 1981. Production, Prices, Employment and Trade in Northwest Forest Industries. Second Quarter 1981. USDA Forest Service. Pacific Northwest Forest and Range Experiment Station, Portland, Oregon. 61 pp.
4. Phelps, R.B. 1977. The Demand and Price Situation for Forest Products 1976 - 77. USDA Forest Service Miscellaneous Publication No. 1357 37 pp.
5. Adams, D.M. 1977. Effects of National Forest Timber Harvest on Softwood Stumpage, Lumber and Plywood Markets: An Economic Analysis. Research Bulletin 15. OSU Forest Products Lab., Corvallis, Oregon. 50 pp.
6. Bueter, J.H.; Johnson, N.K. and Scheurman, H.L. 1976. Timber for Oregon's Tomorrow: An Analysis of Reasonably Possible Occurrences. Research Bulletin 19. OSU Forest Products Lab., Corvallis, Oregon. 111 pp.
7. Baldwin, R.F. 1981. Plywood Manufacturing Practices, 2nd ed. Miller Freeman Publications, San Francisco, California. 326 pp.

8. Clapp, V.W. 1982. Lumber Recovery: How Does Your Mill's Performance Rate? Forest Industries 109(3):26-27.
9. Bruce, David. 1970. Predicting Product Recovery from Logs and Trees. USDA Forest Service Research Paper PNW-107. Pacific Northwest Forest and Range Experiment Station, Portland, Oregon. 15 pp.
10. Lane, P.H.; Woodfin, R.O.; Henley, J.W. and Plank, M.E. 1973. Veneer Recovery from Old- Growth Douglas-fir. USDA Forest Service Research Paper PNW-162. Pacific Northwest Forest and Range Experiment Station, Portland, Oregon. 44 pp.
11. Fahey, T.D. 1974. Veneer Recovery from Second Growth Douglas-fir. USDA Forest Service PNW-173. Pacific Northwest Forest and Range Experiment Station, Portland, Oregon. 22 pp.
12. Woodfin, R.O. 1973. Wood Losses in Plywood Production -- Four Species. Forest Products Journal 23(9):98-106.
13. Sheffield, Thomas H. 1983. Green-End Veneer Recovery and Losses: A Detailed Study in Types and Volumes. A Thesis submitted to Oregon State University in partial fulfillment of the requirements for the degree of Master of Science. Corvallis, Oregon. 216 pp.
14. Maxey, C. 1977. Veneer Clipper Knife Speed. Oregon State University Forest Research Lab. Project F-993 Review. OSU, Corvallis, Oregon. 8 pp.
15. Anonymous. 1974. Computer Program Reveals Potential Veneer Losses. Crows 52(9):16-18
16. Foschi, R.O. 1976. Log Centering Errors and Veneer Yield. Forest Industries Journal 26(2):52-56.

17. Tobin, L.R. and J.S. Bethel. 1969. Veneer Recovery Prediction Analysis Through Computer Simulation. Wood and Fiber 1(2):98-106.
18. Szymani, Ryszard and Kent A. McDonald. Date. Defect Detection in Lumber: State of the Art. Forest Products Journal 31(11):34-44
19. McMillin, Charles W. 1982. Application of Image Analysis to Wood Science. Wood Science 14(3):97-105.

APPENDICES

APPENDIX A

Digitizing Computer Program

```
*****
* THIS PROGRAM IS WRITTEN TO FACILITATE THE CONVERSION OF FILMED *
* VENEER IMAGES TO DIGITAL DATA BY USING A GTCO DEMI-PAD 5      *
* DIGITIZER INTERFACED WITH A DEC MINC MICROCOMPUTER            *
* PROGRAMMER: MIKE BABB                                          *
*****
```

LIST OF VARIABLES:

```
AX,AY,BX,BY,CX,CY,DX,DY = SKEW CORRECTED COORDINATES
BLX,BLY = LOWER LEFT CORNER MATCH POINTS
BN = BLOCK NUMBER
BRX,BRY = LOWER RIGHT CORNER MATCH POINTS
DC = DIAMETER CLASS
ID = DEFECT CODE
M,N = DIGITIZER CONTROL VARIABLES
MBL = MATCH POINT CODE BOTTOM LEFT CORNER
MBR = MATCH POINT CODE BOTTOM RIGHT CORNER
MTL = MATCH POINT CODE TOP LEFT CORNER
MTR = MATCH POINT CODE TOP RIGHT CORNER
NCHAR,IRCNT,BUF,TBUF = SYSTEM ENCODE/DECODE STRING
MANIPULATION OF DATA
P = PIECE NUMBER
R1-R9 = PROMPT RESPONSE VARIABLES
TLX,TLY = UPPER LEFT CORNER MATCH POINTS
TRX,TRY = UPPER RIGHT CORNER MATCH POINTS
CIN,COPYB = SYSTEM DATA MANAGEMENT SUBROUTINES
  BYTE BUF(20),TBUF(11)
  INTEGER DC,BN,P,MBL,MBR,MTL,MTR,M,ID,R1,R2,R3,R4,R5,R6,R7,R8,R9
  REAL BLX,BLY,BRX,BRY,RC,F1,F2,AX,AY,BX,BY,TLX,TLY,TRX,TRY,CX,CY,DX
  *,DY
  DATA NCHAR,IRCNT,P,MBL,MBR,MTL,MTR,M,R1,R2,R3,R4,R5,R6,R7,R8,R9/17
  *0/
```

C
C

```
3 WRITE(7,5)
5 FORMAT(' THIS PROGRAM IS DESIGNED TO DIGITIZE THE',/, ' FILMED IMAG
*ES OF VENEER SHEETS USING A',/, ' GTCO DEMI-PAD 5 DIGITIZER AND DEC
*',/, ' MINC MICROCOMPUTER.',/,/, ' THE DEFECT CODES ARE:',/,5X,' 1
* SOUND KNOTS',/,5X,' 2 OPEN FLAWS',/,5X,' 3 SPLITS',/,5X,' 4
* PITCH STREAKS',/,5X,' 5 WHITE SPECK',/,/,/, ' SHOULD PROBLEMS OC
*CUR AT ANY TIME, PLEASE',/, ' PRESS THE "8" BUTTON ON THE CURSOR.',
*/, ' THIS WILL ALLOW YOU TO RE-ENTER THE PROGRAM',/, ' AT VARIOUS PL
*ACES.',/,/,/, ' MATCH POINT CODES (1-6)',/,/,4X,'CX,CY',6X,'DX,DY'
*,/,4X,'TLX,TLY',4X,'TRX,TRY',/,5X,' -----',/,5X,'!2
*6 3!',/,5X,'!',14X,'!',/,5X,'!',14X,'!',/,5X,'!1 5 4
*!',/,5X,' -----',/,4X,'BLX,BLY BRX,BRY',/,4X,'AX,AY
* BX,BY',/,/,/, ' RECTANGULAR DEFECT DIGITIZING',/,/,12X,'X2,Y2',
*,/9X,'-----',/,9X,'! !',/,9X,'-----',/,6X,'X1,Y1',/,/,/,/,/)
```

C
C
C

GET OUTPUT DATA FILE NAME

```
WRITE(7,10)
10 FORMAT(' THE DATA FILE NAMES SHOULD FOLLOW THIS',/, ' FORMAT--BLK1A
```



```

*.DAT,BLK1B.DAT,BLK2A.DAT',/, ' BLK3A.DAT, ETC. IN THIS EXAMPLE, TH
*e',/, ' DIGITIZING SESSION WAS INTERRUPTED WHILE',/, ' DIGITIZING BL
*OCK NUMBER 1, HENCE THE A AND B.',/, ' BLOCKS 1 THROUGH 30 ARE DIAM
*ETER CLASS 1',/, ' AND BLOCKS 31 THROUGH 60 ARE DIAMETER',/, ' CLASS
* 2.',/,/,/,/,/)
13 WRITE(7,15)
15 FORMAT(' NAME OF DATA FILE: ', $)
   READ(5,20)N,BUF
20 FORMAT(Q,20A1)
C
C OPEN FILE FOR OUTPUT
C
   CALL ASSIGN(10,BUF,N,'NEW','CC',2) !OPEN FILE
C
C ENTER FILE IDENTIFIERS
C
23 WRITE(7,25)
25 FORMAT(' PLEASE ENTER THE DIAMETER CLASS FROM THE',/,5X,'TERMINAL
*KEYBOARD.',/,/,/,/)
   READ(5,30)DC
30 FORMAT(I1)
   WRITE(7,40)
40 FORMAT(' USING AN 01-30 FORMAT, PLEASE ENTER THE',/,5X,'BLOCK NUMB
*ER USING THE TERMINAL KEYBOARD.',/,/,/,/)
   READ(5,45)BN
45 FORMAT(I2)
   WRITE(7,50)
50 FORMAT(' USING A 001-999 FORMAT, PLEASE ENTER THE',/, ' PIECE NUMBE
*R FROM THE TERMINAL KEYBOARD.',/,/,/,/)
   READ(5,55)P
55 FORMAT(I3)
58 WRITE(7,60)DC,BN,P
60 FORMAT(' DIAMETER=',I1,' BLOCK=',I2,' PIECE=',I3,/,/,/,/)
   WRITE(7,63)
63 FORMAT(' ARE THESE ENTRIES CORRECT? 0=YES, 1=NO.',/, ' FROM NOW ON,
* PLEASE USE THE CURSOR BUTTONS',/, ' TO ANSWER ALL QUESTIONS.',/,/,
*/)
   CALL DATIN1(BUF,NCHAR,IRCNT,TBUF,R1,X,Y)
   IF(R1.EQ.0)GO TO 70
   IF(R1.EQ.1)GO TO 23
   IF(R1.EQ.8)GO TO 500
   WRITE(7,65)
65 FORMAT(' ILLEGAL QUESTION RESPONSE. PLEASE TRY AGAIN.',/,/,/,/)
   GO TO 58
70 WRITE(10,75)DC,BN,P
75 FORMAT(3I4)
   C
   C READ BOTTOM LEFT AND RIGHT POINTS OF THE SHEET TO ESTABLISH THE
   C SKEW CORRECTION FACTORS.
   C
78 WRITE(7,80)

```

```
80 FORMAT(' SKEW CORRECTION ROUTINE.  READ BOTTOM LEFT',/, ' CORNER.
*WHILE DOING SO, PLEASE PRESS THE',/, ' APPROPRIATE MATCH CODE ON
*THE CURSOR.',/, ' PRESS THE "0" BUTTON IF THERE IS NO',/,
*' MATCH POINT.',/,/,/)
CALL DATIN1(BUF,NCHAR,IRCNT,TBUF,MBL,BLX,BLY)
IF(MBL.GE.0.AND.MBL.LE.6)GO TO 83
IF(MBL.EQ.8)GO TO 500
WRITE(7,65)
GO TO 78
83 WRITE(7,85)
85 FORMAT(' WHILE PRESSING APPROPRIATE MATCH CODE',/, ' READ BOTTOM
*RIGHT CORNER.',/,/,/)
CALL DATIN1(BUF,NCHAR,IRCNT,TBUF,MBR,BRX,BRY)
IF(MBR.GE.0.AND.MBR.LE.6)GO TO 87
IF(MBR.EQ.8)GO TO 500
WRITE(7,65)
GO TO 83
```

C

C CALCULATION OF SKEW CORRECTION FACTORS

C

```
87 RC=ATAN((BRY-BLY)/(BRX-BLX))
F1=cos(RC)      !SKEW FACTOR
F2=sin(RC)      !SKEW FACTOR
```

C

C MATCHING POINT ROUTINE

C

```
88 WRITE(7,90)
90 FORMAT(' ARE THERE MATCH POINTS? 0=YES, 1=NO',/,/,/)
CALL DATIN1(BUF,NCHAR,IRCNT,TBUF,R3,X,Y)
IF(R3.EQ.0)GO TO 95
IF(R3.EQ.1)GO TO 133
IF(R3.EQ.8)GO TO 500
WRITE(7,65)
GO TO 88
95 WRITE(7,100)
100 FORMAT(' ARE THERE TRAILING EDGE MATCH POINTS? 0=YES,1=NO',/,/,/)
CALL DATIN1(BUF,NCHAR,IRCNT,TBUF,R4,X,Y)
IF(R4.EQ.0)GO TO 105
IF(R4.EQ.1)GO TO 112
IF(R4.EQ.8)GO TO 500
WRITE(7,65)
GO TO 95
105 AX=BLX*F1+BLY*F2      !NOW SKEW CORRECTED
AY=BLY*F1-BLX*F2      !NOW SKEW CORRECTED
BX=BRX*F1+BRY*F2      !NOW SKEW CORRECTED
BY=BRY*F1-BRX*F2      !NOW SKEW CORRECTED
108 WRITE(10,110)MBL,AX,AY,MBR,BX,BY
110 FORMAT(I3,2F7.3,/,I3,2F7.3)
112 WRITE(7,115)
115 FORMAT(' ARE THERE LEADING EDGE MATCH POINTS? 0=YES, 1=NO',/,/,/)
CALL DATIN1(BUF,NCHAR,IRCNT,TBUF,R5,X,Y)
```

```

        IF(R5.EQ.0)GO TO 120
        IF(R5.EQ.1)GO TO 133
        IF(R5.EQ.8)GO TO 500
        WRITE(7,65)
        GO TO 112
120 WRITE(7,125)
125 FORMAT(' WHILE PRESSING THE APPROPRIATE MATCH CODE',/, ' KEY,
      *READ TOP LEFT MATCH POINT.',/,/,/)
      CALL DATIN1(BUF,NCHAR,IRCNT,TBUF,MTL,TLX,TLY)
      IF(MTL.GE.1.AND.MTL.LE.6)GO TO 128
      IF(MTL.EQ.8)GO TO 500
      WRITE(7,65)
      GO TO 120
128 WRITE(7,130)
130 FORMAT(' WHILE PRESSING THE APPROPRIATE MATCH CODE',/, ' KEY,
      * READ THE TOP RIGHT MATCH POINT.',/,/,/)
      CALL DATIN1(BUF,NCHAR,IRCNT,TBUF,MTR,TRX,TRY)
      IF(MTR.GE.1.AND.MTR.LE.6)GO TO 132
      IF(MTR.EQ.8) GO TO 500
      WRITE(7,65)
      GO TO 128
132 CX=TLX*F1+TLY*F2           !NOW SKEW CORRECTED
      CY=TLY*F1-TLX*F2           !NOW SKEW CORRECTED
      DX=TRX*F1+TRY*F2           !NOW SKEW CORRECTED
      DY=TRY*F1-TRX*F2           !NOW SKEW CORRECTED
      WRITE(10,110)MTL,CX,CY,MTR,DX,DY
133 WRITE(10,135)
135 FORMAT(' -22 0 0')

C
C SHAPE DIGITIZING ROUTINE
C
138 WRITE(7,140)
140 FORMAT(' DO THE MATCH POINTS DESCRIBE THE SHAPE?',/, ' 0=YES,
      * 1=NO',/,/,/)
      CALL DATIN1(BUF,NCHAR,IRCNT,TBUF,R6,X,Y)
      IF(R6.EQ.0)GO TO 153
      IF(R6.EQ.1)GO TO 145
      IF(R6.EQ.8)GO TO 500
      WRITE(7,65)
      GO TO 138
145 WRITE(7,150)
150 FORMAT(' WHILE PRESSING ANY "NUMBER" CURSOR BUTTON',/, ' OTHER THAN
      * "8" OR "9", BEGIN TRACING THE',/, ' SHAPE. STRAIGHT LINES CAN BE
      *DIGITIZED',/, ' BY SINGLE POINT READS AT EACH END. SWITCH',/, ' MOD
      *ES BY PRESSING THE "F" BUTTON. PRESS THE',/, ' "9" KEY WHEN FINISH
      *ED.',/,/,/)
      CALL DATIN2(BUF,NCHAR,IRCNT,TBUF,F1,F2,M)
      IF(M.EQ.8)GO TO 500
153 WRITE(10,155)
155 FORMAT(' -77 0')

```

C

C SHAPE DIGITIZING ROUTINE

C

```
158 WRITE(7,160)
160 FORMAT(' IS A SHAPE MARGIN PRESENT? 0=YES, 1=NO',/,/,/)
    CALL DATIN1(BUF,NCHAR,IRCNT,TBUF,R7,X,Y)
    IF(R7.EQ.0)GO TO 235
    IF(R7.EQ.1)GO TO 238
    IF(R7.EQ.8)GO TO 500
    WRITE(7,65)
    GO TO 158
235 CALL DATIN2(BUF,NCHAR,IRCNT,TBUF,F1,F2,M)
    IF(M.EQ.8)GO TO 500
238 WRITE(10,239)
239 FORMAT(' -55 0')
```

C

C DEFECT DIGITIZING ROUTINE

C

```
240 WRITE(7,245)
245 FORMAT(' DOES THIS PIECE HAVE ANY DEFECTS? 0=YES, 1=NO',/,/,/)
    CALL DATIN1(BUF,NCHAR,IRCNT,TBUF,R3,X,Y)
    IF(R3.EQ.0)GO TO 250
    IF(R3.EQ.1)GO TO 248
    IF(R3.EQ.8)GO TO 500
    WRITE(7,65)
    GO TO 240
248 WRITE(10,290)
    GO TO 320
250 WRITE(7,255)
255 FORMAT(' USING THE CURSOR, PLEASE ENTER THE DEFECT CODE.',/,/,/)
    CALL DATIN1(BUF,NCHAR,IRCNT,TBUF,ID,X,Y)
    IF(ID.EQ.8)GO TO 500
257 WRITE(7,260)
260 FORMAT(' CAN THE DEFECT BE DESCRIBED BY A RECTANGLE?',/,
    *' 0=YES=DISCRETE, 1=NO=CONTINUOUS',/,/,/)
    CALL DATIN1(BUF,NCHAR,IRCNT,TBUF,R4,X,Y)
    IF(R4.EQ.8)GO TO 500
    WRITE(7,265)ID,R4
265 FORMAT(' ID=',I2,' R4=',I2,/,/,/)
270 WRITE(7,63)
    CALL DATIN1(BUF,NCHAR,IRCNT,TBUF,R5,X,Y)
    IF(R5.EQ.0)GO TO 275
    IF(R5.EQ.1)GO TO 250
    IF(R5.EQ.8)GO TO 500
    WRITE(7,65)
    GO TO 270
275 WRITE(10,280)ID,R4
280 FORMAT(2I2)
    WRITE(7,285)
285 FORMAT(' IF DISCRETE DIGITIZING, PLEASE READ JUST',/, ' THE LOWER L
    *EFT AND THEN UPPER RIGHT CORNERS.',/, ' IF CONTINUOUS, USE THE "F"
    *BUTTON TO SWITCH',/, ' MODES. USE ANY "NUMBER" KEY OTHER THAN',/, '
```

```
* "8" OR "9".  PRESS THE "9" KEY WHEN FINISHED.',/,/,/)
CALL DATIN2(BUF,NCHAR,IRCNT,TBUF,F1,F2,M)
IF(M.EQ.8)GO TO 500
WRITE(10,290)
290 FORMAT(' -88 0')
293 WRITE(7,295)
295 FORMAT(' DOES THE DEFECT HAVE A MARGIN? 0=YES, 1=NO',/,/,/)
CALL DATIN1(BUF,NCHAR,IRCNT,TBUF,R6,X,Y)
IF(R6.EQ.0)GO TO 300
IF(R6.EQ.1)GO TO 305
IF(R6.EQ.8)GO TO 500
WRITE(7,65)
GO TO 293
300 CALL DATIN2(BUF,NCHAR,IRCNT,TBUF,F1,F2,M)
IF(M.EQ.8)GO TO 500
305 WRITE(10,310)
310 FORMAT(' -44 0')
313 WRITE(7,315)
315 FORMAT(' ARE THERE MORE DEFECTS? 0=YES, 1=NO',/,/,/)
CALL DATIN1(BUF,NCHAR,IRCNT,TBUF,R7,X,Y)
IF(R7.EQ.0)GO TO 250
IF(R7.EQ.1)GO TO 320
IF(R7.EQ.8)GO TO 500
WRITE(7,65)
GO TO 313
320 WRITE(10,325)
325 FORMAT(' -99 0')
328 WRITE(7,330)
330 FORMAT(' DO YOU WANT TO TAKE A BREAK? 0=YES, 1=NO',/,/,/)
CALL DATIN1(BUF,NCHAR,IRCNT,TBUF,R8,X,Y)
IF(R8.EQ.1)GO TO 350
IF(R8.EQ.0)GO TO 333
IF(R8.EQ.8)GO TO 500
WRITE(7,65)
GO TO 328
333 WRITE(7,335)
335 FORMAT(' ARE YOU SURE YOU WANT TO STOP? 0=YES, 1=NO',/,/,/)
CALL DATIN1(BUF,NCHAR,IRCNT,TBUF,R9,X,Y)
IF(R9.EQ.8)GO TO 500
IF(R9.NE.0)GO TO 328
CLOSE(UNIT=10)          !CLOSE FILE
WRITE(7,340)
340 FORMAT(' THE FILE HAS BEEN CLOSED AND THE SESSION',/, ' ENDED.
*GOOD-BYE!',/,/,/)
CALL EXIT              !EXIT PROGRAM
350 WRITE(7,355)
355 FORMAT(' IS THIS A NEW BLOCK? 0=YES, 1=NO',/,/,/)
CALL DATIN1(BUF,NCHAR,IRCNT,TBUF,R1,X,Y)
IF(R1.EQ.1)GO TO 360
IF(R1.EQ.0)GO TO 370
IF(R1.EQ.8)GO TO 500
```

```

        WRITE(7,65)
        GO TO 350
360 P=P+1
363 WRITE(7,365)
365 FORMAT(' IS THERE ENOUGH DISK STORAGE LEFT FOR',/, ' ANOTHER PIECE?
      * 0=YES, 1=NO',/,/,/)
      CALL DATIN1(BUF,NCHAR,IRCNT,TBUF,R2,X,Y)
      IF(R2.EQ.0)GO TO 58
      IF(R2.EQ.1)GO TO 370
      IF(R2.EQ.8)GO TO 500
      WRITE(7,65)
      GO TO 363
370 CLOSE(UNIT=10)                !CLOSE FILE
      WRITE(7,375)
375 FORMAT(' THE FILE HAS NOW BEEN CLOSED.',/, ' A NEW FILE WILL NOW BE
      *OPENED.',/,/,/)
      GO TO 13
500 WRITE(10,510)
510 FORMAT(' -13 0')
      WRITE(7,515)
515 FORMAT(5X,' SPECIAL HELP KEY',/,5X,' ----- ----',/,/,/, ' PLE
      *ASE CHOOSE FROM THE FOLLOWING LIST THE',/, ' APPROPRIATE CURSOR BUT
      *TON TO PRESS TO',/, ' RETURN TO THE PROGRAM.',/,/,
      *' 0= START OF PROGRAM',/,
      *' 1= SKEW CORRECTION ROUTINE',/,
      *' 2= MATCH POINTS?',/,
      *' 3= DIGITIZE SHAPE',/,
      *' 4= DIGITIZE SHAPE MARGIN',/,
      *' 5= DEFECTS PRESENT?',/,
      *' 6= DEFECT MARGIN PRESENT?',/,
      *' 7= MORE DEFECTS PRESENT?',/,
      *' 8= TAKE A BREAK AND CLOSE FILE?',/,
      *' 9= NOT ENOUGH DISK STORAGE?',/,/,/)
      CALL DATIN1(BUF,NCHAR,IRCNT,TBUF,M,X,Y)
      IF(M.EQ.0)GO TO 3
      IF(M.EQ.1)GO TO 78
      IF(M.EQ.2)GO TO 88
      IF(M.EQ.3)GO TO 138
      IF(M.EQ.4)GO TO 158
      IF(M.EQ.5)GO TO 240
      IF(M.EQ.6)GO TO 293
      IF(M.EQ.7)GO TO 313
      IF(M.EQ.8)GO TO 328
      IF(M.EQ.9)GO TO 363
      END

```

C
C

```

*****
* SUBROUTINE DATAIN1 ACCEPTS AND DECODES                                *
* DIGITIZER INPUT IN DISCRETE MODE.                                     *
* SUBROUTINE DATIN1(BUF,NCHAR,IRCNT,TBUF,M,X,Y)                        *
*****

```

```

      CALL CIN(BUF,NCHAR,IRCNT)      !INITIATE DATA ACQUISITION
10  IF(NCHAR.GT.0)GO TO 20           !ANY DIGITIZER DATA?
      GO TO 10
20  N=NCHAR      !SAVE STRING SIZE COUNTER
      NCHAR=0    !CLEAR FLAG
      CALL COPYB(N-1,BUF,TBUF)      !COPY BUFFER
      DECODE(11,30,TBUF)M,X,Y      !DECODE THE STRING
30  FORMAT(I1,2F5.3)
      RETURN
      END

```

C
C

```

*****
* SUBROUTINE DATAIN2 ACCEPTS AND DECODES INPUT *
* FROM THE DIGITIZING CURSOR WHEN CONTINUOUS *
* MODE IS NECESSARY. *
* SUBROUTINE DATIN2(BUF,NCHAR,IRCNT,TBUF,F1,F2,M) *
*****
      WRITE(7,5)
      5  FORMAT(' PLEASE BEGIN DIGITIZING.',/,/,/,/)
      CALL CIN(BUF,NCHAR,IRCNT)      !DATA ACQUISITION
10  IF(NCHAR.GT.0)GO TO 20           !ANY DATA?
      GO TO 10
20  N=NCHAR      !SAVE COUNTER
      NCHAR=0    !CLEAR FLAG
      CALL COPYB(N-1,BUF,TBUF)      !COPY BUFFER
      DECODE(11,30,TBUF)M,X,Y      !DECODE STRING
30  FORMAT(I1,2F5.3)
      IF(M.EQ.8.OR.M.EQ.9)GO TO 50
      CX=X*F1+Y*F2    !SKEW CORRECTED
      CY=Y*F1-X*F2    !SKEW CORRECTED
      WRITE(10,40)CX,CY      !WRITE TO DISK
40  FORMAT(2F7.3)
      GO TO 10
50  RETURN
      END

```

APPENDIX B

Reconstruction Computer Program


```
*****
* THIS FORTRAN 77 PROGRAM IS DESIGNED TO RECONSTRUCT *
* INDIVIDUAL PIECES OF VENEER INTO THE ORIGINAL      *
* RIBBON USING DIGITIZED DATA                      *
* PROGRAMMER: MIKE BABB                             *
*****
```

LIST OF VAVIABLES:

```
BDIS = UNCORRECTED TRAILING CLIP X DISTANCE
BN = BLOCK NUMBER
DC = DIAMETER CLASS
DKNT = OPEN DEFECT COUNTER
DMPTS = DEFECT WANE POINTS COUNTER
DPTS = DEFECT POINT COUNTER
DUMB(4,2) = DUMMY SKEW CORRECTION ARRAY
FULLDIF = DIFFERENCE BETWEEN EXPECTED FULL SHEET WIDTH AND
          DIGITIZED PIECE WIDTH
F1,F3 = SKEW CORRECTION VARIABLES
ID = DEFECT CODE
L(2750,2),R(2750,2) = LEFT/RIGHT COORDINATES ARRAY
LPTS = LEFT SIDE POINT COUNTER
LXCHNG,RXCHNG,LBY,RBY,LFACT,RFACT,LSKEW,RSKEW = SHEET
          ADJUSTMEMNT FACTORS
M,MX,MY = MATCH POINT VARIABLES
M1PTS = MATCH POINTS INITIAL PIECE
M2PTS = MATCH POINTS ALL OTHER PIECES
MAT1(4,3) = INITIAL PIECE MATCH POINT ARRAY
MAT2(4,3) = MATCH POINT ARRAY FOR ALL OTHER PIECES
MGPTS = WANE POINT COUNTER
MKNT = TOTAL DEFECT COUNTER
P = PIECE NUMBER
PADJ = PIECE ADJUSTMENT FLAG
P1 = PIECE COUNT TEST VARIABLE
RPTS = RIGHT SIDE POINT COUNTER
SCALAR = FULL MILL SCALE FACTOR
SFLAG = DEFECT LOCATION FLAG
SKEW(4,2) = SKEW FACTORS CORRECTION ARRAY
SPTS = TOTAL POINT COUNTER
SUM(2) = MINIMUM AND MAXIMUM Y POINTER ARRAY
SX,SY = X AND Y COORDINATES
T(2750,2) = TEMPORARY COORDINATE ARRAY
TDIS = UNCORRECTED LEADING EDGE X DISTANCE
TEMP(4,3) = TEMPORARY MATCH POINT ARRAY
UN = FILE NUMBER VARIABLE
XC = X ADJUST VARIABLE
XDIS,DIS = X DISTANCES BETWEEN MATCH POINTS
XMIN,XMAX = MINIMUM AND MAXIMUM DEFECT X COORDINATES
XN,MINY,XX,MAXY,MINX,MAXX = CRACK PARAMETERS
YDIS = Y DISTANCE BETWEEN MATCH POINTS
YSTRT = MINIMUM Y VALUE ON A PIECE
PROGRAM RECON(INPUT,OUTPUT)
INTEGER DC,BN,P,M1PTS,SPTS,LPTS,RPTS,MGPTS,DPTS,ID,
* DKNT,DMPTS,MKNT,M2PTS,SUM(2),UN,P1,PADJ,SFLAG
```

```

      REAL M,MX,MY,SX,SY,F1,MAT1(4,3),MAT2(4,3),TEMP(4,3),
      * L(2750,2),R(2750,2),S(2750,2),YSTRT,XC,F3,YDIS,DUMB(4,2),
      * SCALAR,XMIN,XMAX,DIS,YHLD,LSKEW,RSKEW,SKEW(4,2),
      * LXCHNG,RXCHNG,LBY,RBY,LFACT,RFACT,FULLDIS,XDIS,
      * T(2750,2),TDIS,BDIS,FULLDIF,XN,MINY,XX,MAXY,MINX,MAXX
      COMMON T
      DATA DKNT,MKNT,P1/0,0,0/
      DATA ((MAT1(I,J),J=1,3),I=1,4)/12*0.0/
      DATA ((MAT2(I,J),J=1,3),I=1,4)/12*0.0/
      DATA SCALAR/6.129/
      DATA ((TEMP(I,J),J=1,3),I=1,4)/12*0.0/
      OPEN(1,FILE='BLKINP')
      OPEN(2,FILE='LSHAPT')
      OPEN(3,FILE='RSHAPT')
      OPEN(4,FILE='LSMART')
      OPEN(5,FILE='RSMART')
      OPEN(6,FILE='LDFECT')
      OPEN(7,FILE='RDFECT')
      OPEN(8,FILE='LDMART')
      OPEN(9,FILE='RDMART')
      OPEN(10,FILE='LSPEKT')
      OPEN(11,FILE='RSPEKT')
      OPEN(12,FILE='LCRAKT')
      OPEN(13,FILE='RCRAKT')
      OPEN(14,FILE='GDSUMT')
      OPEN(15,FILE='SDSUMT')
      OPEN(16,FILE='LDCLPT')
      OPEN(17,FILE='RDCLPT')
      OPEN(18,FILE='LMCLPT')
      OPEN(19,FILE='RMCLPT')
      OPEN(20,FILE='SSORT')
      OPEN(21,FILE='LDCLIP')
      OPEN(22,FILE='RDCLIP')
      OPEN(23,FILE='LMCLIP')
      OPEN(24,FILE='RMCLIP')
      OPEN(25,FILE='ERRORT')
      OPEN(26,FILE='SSUM')
      OPEN(27,FILE='GSUM')
      FULLDIS=101.0/SCALAR
5 DO 6 I6=1,4
      DO 6 I7=1,2
          SKEW(I6,I7)=0.0
6 CONTINUE
10 READ(1,*)DC,BN,P
      IF(DC.EQ.-999)THEN
C
C  END OF BLOCK DELIMITER.  WRITE "-999 0 0" ON EACH FILE.
C
      DO 20 UN=2,9
          WRITE(UN,15)
15      FORMAT(' -999 0 0')
```

```

20  CONTINUE
    WRITE(10,28)
    WRITE(11,28)
    WRITE(12,28)
    WRITE(13,28)
    WRITE(14,*)' -999 0 0 0 0 0 0 0 0 0'
    WRITE(15,*)' -999 0 0 0 0 0 0 0 0 0'
    DO 22 UN=16,19
        WRITE(UN,21)
21  FORMAT(' -999 0 0 0')
22  CONTINUE
    GOTO 10
    ELSEIF(DC.EQ.-111)THEN
C
C  END OF FILE DELIMITER. WRITE "-111 0 0" ON EACH FILE.
C
    DO 25 UN=2,9
        WRITE(UN,23)
23  FORMAT(' -111 0 0')
25  CONTINUE
    WRITE(10,29)
    WRITE(11,29)
    WRITE(12,29)
    WRITE(13,29)
    WRITE(14,*)' -111 0 0 0 0 0 0 0 0 0'
    WRITE(15,*)' -111 0 0 0 0 0 0 0 0 0'
    DO 27 UN=16,19
        WRITE(UN,26)
26  FORMAT(' -111 0 0 0')
27  CONTINUE
    STOP
ENDIF
28 FORMAT(' -999 0')
29 FORMAT(' -111 0')
DIS=0.0
WRITE(25,*)'PIECE # = ',P
FULLDIF=0.0
M1PTS=0
F1=0.0
F3=0.0
PADJ=0
30 READ(1,*)M,MX,MY
    IF(M.NE.-22.)THEN
        M1PTS=M1PTS+1
        MAT1(M1PTS,1)=M
        MAT1(M1PTS,2)=MX+5.0
        MAT1(M1PTS,3)=MY
        GOTO 30
    ENDIF
    IF(M1PTS.EQ.0)GOTO 45
    IF(MAT1(1,1).NE.1.0)THEN

```

```

        DO 40 I2=1,M1PTS
          MAT1(I2,2)=MAT1(I2,2)+10.0
40    CONTINUE
      ENDIF
45    SPTS=0
50    READ(1,*)SX,SY
      IF(SX.NE.-77.)THEN
        SPTS=SPTS+1
        IF(SPTS.GT.2750)THEN
          WRITE(25,60)P,BN,DC,SX,SY
          STOP
        ENDIF
        S(SPTS,1)=SX+5.0
        S(SPTS,2)=SY
        GOTO 50
      ENDIF
C
C  SHAPE POINT COUNT EXCEEDS ARRAY DIMENSION.  WRITE THIS
C  INFORMATION ON ERROR FILE 25.  THEN STOP THE PROGRAM.
C
60    FORMAT(' THE SHAPE POINT COUNT EXCEEDS THE ARRAY DIMENSION
      * OF 2750 ON',/, ' PIECE ',I4, ' BLOCK ',I3, ' DIAMETER ',I3,/,
      * ' SX= ',F8.3, ' SY= ',F8.3)
      IF(M1PTS.EQ.0)GOTO 76
      IF(MAT1(1,1).NE.1.0)THEN
        DO 75 I5=1,SPTS
          S(I5,1)=S(I5,1)+10.0
75    CONTINUE
      ENDIF
76    IF(SPTS.EQ.4)THEN
      LPTS=2
      RPTS=2
      L(1,1)=S(2,1)
      L(1,2)=S(2,2)
      L(2,1)=S(1,1)
      L(2,2)=S(1,2)
      R(1,1)=S(3,1)
      R(1,2)=S(3,2)
      R(2,1)=S(4,1)
      R(2,2)=S(4,2)
    ELSE
      CALL MINMAX(S,SPTS,SUM)
      CALL SORTS(S,L,R,SUM,SPTS,LPTS,RPTS)
    ENDIF
    IF(M1PTS.EQ.2)THEN
      CALL SORT1(L,R,MAT1,LPTS,RPTS)
      XDIS=R(RPTS,1)-L(LPTS,1)
      FULLDIF=FULLDIS-XDIS
      IF(FULLDIF.NE.0.0)THEN
        IF(FULLDIF.LE.(4.0/SCALAR))THEN
          PADJ=1

```

```

        CALL STRETCH(FULLDIF,L,LPTS,R,RPTS)
      ENDIF
    ENDIF
  ENDIF
  CALL REVERSE(L,R,LPTS,RPTS)
  F1=L(LPTS,1)
  F3=R(RPTS,1)
  YSTRT=L(LPTS,2)
  WRITE(25,*) ' XDIS= ',XDIS,' FULLDIF= ',FULLDIF,' PADJ= ',
    *PADJ
  WRITE(25,*) ' F1= ',F1,' F3= ',F3
C
C  FILL IN WHERE DIGITIZED POINTS ARE TOO FAR APART.
C
  WRITE(25,*) ' SHAPE POINTS'
  WRITE(25,80)LPTS,RPTS
  CALL PTSFILL(L,LPTS)
  CALL PTSFILL(R,RPTS)
  WRITE(25,*) ' SHAPE POINTS PTSFILL'
  WRITE(25,80)LPTS,RPTS
80  FORMAT(' LPTS= ',I5,2X,' RPTS= ',I5)
  P1=P1+1
  WRITE(2,491)DC,BN,P1
  WRITE(3,491)DC,BN,P1
C
C  WRITE THE FIRST PIECE SHAPE POINTS TO THE LEFT AND RIGHT SHAPE
C  FILES.  POINTS ARE CORRECTED UP TO FULL MILL SCALE UNITS.
C
  DO 85 I1=1,LPTS
    L(I1,1)=L(I1,1)*SCALAR
    L(I1,2)=L(I1,2)*SCALAR
    WRITE(2,490)L(I1,1),L(I1,2)
85  CONTINUE
  DO 90 I2=1,RPTS
    R(I2,1)=R(I2,1)*SCALAR
    R(I2,2)=R(I2,2)*SCALAR
    WRITE(3,490)R(I2,1),R(I2,2)
90  CONTINUE
  MGPTS=0
95  READ(1,*)SX,SY
  IF(SX.NE.-55.)THEN
    MGPTS=MGPTS+1
    IF(MGPTS.GT.2750)THEN
      WRITE(25,105)P,BN,DC,SX,SY
      STOP
    ENDIF
    S(MGPTS,1)=SX+5.0
    S(MGPTS,2)=SY
    GOTO 95
  ENDIF
C
C  SHAPE MARGIN POINT COUNT EXCEEDS ARRAY DIMENSION.  WRITE THIS

```

```

C  INFORMATION ON ERROR FILE 25.  THEN STOP THE PROGRAM.
C
105 FORMAT(' THE SHAPE MARGIN POINT COUNT EXCEEDS THE ARRAY
      * DIMENSION OF 2750 ON',/, ' PIECE ',I4,' BLOCK ',I3,
      * ' DIAMETER ',I3,/, 'SX= ',F8.3,' SY= ',F8.3)
C
C  IF MGPTS EQUALS ZERO, THEN THERE IS NO SHAPE MARGIN.
C
      IF(MGPTS.EQ.0)THEN
        WRITE(4,491)DC,BN,P1
        WRITE(5,491)DC,BN,P1
        DO 111 I1=1,LPTS
          WRITE(4,490)L(I1,1),L(I1,2)
111      CONTINUE
        DO 112 I2=1,RPTS
          WRITE(5,490)R(I2,1),R(I2,2)
112      CONTINUE
        GOTO 135
      ENDIF
      IF(M1PTS.EQ.0)GOTO 120
      IF(MAT1(1,1).NE.1.0)THEN
        DO 115 I6=1,MGPTS
          S(I6,1)=S(I6,1)+10.0
115      CONTINUE
        ENDIF
120  IF(MGPTS.EQ.4)THEN
        LPTS=2
        RPTS=2
        L(1,1)=S(2,1)
        L(1,2)=S(2,2)
        L(2,1)=S(1,1)
        L(2,2)=S(1,2)
        R(1,1)=S(3,1)
        R(1,2)=S(3,2)
        R(2,1)=S(4,1)
        R(2,2)=S(4,2)
      ELSE
        CALL MINMAX(S,MGPTS,SUM)
        CALL SORTS(S,L,R,SUM,MGPTS,LPTS,RPTS)
      ENDIF
      IF(M1PTS.EQ.2)THEN
        CALL SORT1(L,R,MAT1,LPTS,RPTS)
        IF(PADJ.NE.0)THEN
          CALL STRETCH(FULLDIF,L,LPTS,R,RPTS)
        ENDIF
      ENDIF
      CALL REVERSE(L,R,LPTS,RPTS)
      WRITE(25,*)' MARGIN POINTS'
      WRITE(25,80)LPTS,RPTS
      CALL PTSFILL(L,LPTS)
      CALL PTSFILL(R,RPTS)

```

```

        WRITE(25,*)' MARGIN POINTS PTSFILL'
        WRITE(25,80)LPTS,RPTS
        WRITE(4,491)DC,BN,P1
        WRITE(5,491)DC,BN,P1
        DO 125 I1=1,LPTS
            L(I1,1)=L(I1,1)*SCALAR
            L(I1,2)=L(I1,2)*SCALAR
            WRITE(4,490)L(I1,1),L(I1,2)
125    CONTINUE
        DO 130 I2=1,RPTS
            R(I2,1)=R(I2,1)*SCALAR
            R(I2,2)=R(I2,2)*SCALAR
            WRITE(5,490)R(I2,1),R(I2,2)
130    CONTINUE
135    DPTS=0
136    READ(1,*)ID,SY
C
C    IF ID EQUALS -88 ON THE FIRST READ, THERE ARE NO DEFECTS ON
C    THIS PIECE.
C
        IF(ID.EQ.-88.)GOTO 136
        IF(ID.EQ.-44.)GOTO 136
        IF(ID.EQ.-99.)GOTO 271
140    READ(1,*)SX,SY
        IF(SX.NE.-88.)THEN
            DPTS=DPTS+1
            IF(DPTS.GT.2750)THEN
                WRITE(25,150)P,BN,DC,SX,SY,ID
                STOP
            ENDIF
            S(DPTS,1)=SX+5.0
            S(DPTS,2)=SY
            GOTO 140
        ENDIF
        MKNT=MKNT+1
C
C    DEFECT POINT COUNT EXCEEDS THE ARRAY DIMENSION.  WRITE THIS
C    INFORMATION ON ERROR FILE 25.  THEN STOP THE PROGRAM
C
150    FORMAT(' THE DEFECT POINT COUNT EXCEEDS THE ARRAY DIMENSION
        *OF 500 ON',/, ' PIECE ',I4,' BLOCK ',I3,' DIAMETER ',I3,/,
        * ' SX= ',F8.3,' SY= ',F8.3,' ID= ',F8.3)
        IF(M1PTS.EQ.0)GOTO 161
        IF(MAT1(1,1).NE.1.)THEN
            DO 160 I8=1,DPTS
                S(I8,1)=S(I8,1)+10.0
160    CONTINUE
        ENDIF
161    IF(DPTS.EQ.2)THEN
C
C    TWO POINT DIGITIZING.

```

C

```

      LPTS=2
      RPTS=2
      L(1,1)=S(1,1)
      L(1,2)=S(2,2)
      L(2,1)=S(1,1)
      L(2,2)=S(1,2)
      R(1,1)=S(2,1)
      R(1,2)=S(2,2)
      R(2,1)=S(2,1)
      R(2,2)=S(1,2)
      IF(PADJ.NE.0)THEN
        CALL STRETCH(FULLDIF,L,LPTS,R,RPTS)
      ENDIF
      L(1,2)=ABS(L(1,2)-22.0)
      L(2,2)=ABS(L(2,2)-22.0)
      R(1,2)=ABS(R(1,2)-22.0)
      R(2,2)=ABS(R(2,2)-22.0)
      IF(ID.EQ.2)THEN
        DKNT=DKNT+1
        WRITE(6,492)ID,MKNT
        WRITE(7,492)ID,MKNT
        WRITE(26,493)P1,L(1,1)*SCALAR,L(1,2)*SCALAR,L(2,1)*SCALAR,
*      L(2,2)*SCALAR,L(1,1)*SCALAR,R(2,1)*SCALAR,ID,MKNT
      ELSE
        WRITE(8,492)ID,MKNT
        WRITE(9,492)ID,MKNT
        WRITE(27,493)P1,L(1,1)*SCALAR,L(1,2)*SCALAR,L(2,1)*SCALAR,
*      L(2,2)*SCALAR,L(1,1)*SCALAR,R(2,1)*SCALAR,ID,MKNT
      ENDIF
      ELSEIF(ID.EQ.3)THEN
        LPTS=2
        RPTS=2
        L(1,1)=S(1,1)
        L(1,2)=S(1,2)
        L(2,1)=S(2,1)
        L(2,2)=S(2,2)
        R(1,1)=S(3,1)
        R(1,2)=S(3,2)
        R(2,1)=S(4,1)
        R(2,2)=S(4,2)
        IF(PADJ.NE.0)THEN
          IF((L(1,1)-F1).LT.(F3-L(1,1)))THEN
            DO 170 I7=1,2
              L(I7,1)=L(I7,1)-FULLDIF*.5
              R(I7,1)=R(I7,1)-FULLDIF*.5
170          CONTINUE
          ELSE
            DO 175 I5=1,2
              L(I5,1)=L(I5,1)+FULLDIF*.5
              R(I5,1)=R(I5,1)+FULLDIF*.5

```



```

175      CONTINUE
      ENDIF
    ENDIF
    CALL REVERSE(L,R,LPTS,RPTS)
    IF(L(1,1).LT.L(2,1))THEN
      MINX=L(1,1)
      MAXX=L(2,1)
    ELSE
      MINX=L(2,1)
      MAXX=L(1,1)
    ENDIF
    IF(L(1,2).GT.L(2,2))THEN
      XN=L(1,1)
      MINY=L(1,2)
    ELSE
      XN=L(2,1)
      MINY=L(2,2)
    ENDIF
    IF(R(1,2).GT.R(2,2))THEN
      XX=R(1,1)
      MAXY=R(1,2)
    ELSE
      XX=R(2,1)
      MAXY=R(2,2)
    ENDIF
    DKNT=DKNT+1
    WRITE(26,493)P1,XN*SCALAR,MINY*SCALAR,XX*SCALAR,
*   MAXY*SCALAR,MINX*SCALAR,MAXX*SCALAR,ID,MKNT
    WRITE(6,492)ID,MKNT
    WRITE(7,492)ID,MKNT
    WRITE(27,493)P1,XN*SCALAR,MINY*SCALAR,XX*SCALAR,
*   MAXY*SCALAR,MINX*SCALAR,MAXX*SCALAR,ID,MKNT
    WRITE(8,492)ID,MKNT
    WRITE(9,492)ID,MKNT
    WRITE(12,492)ID,MKNT
    WRITE(13,492)ID,MKNT
  ELSE
    CALL MINMAX(S,DPTS,SUM)
    CALL SORTS(S,L,R,SUM,DPTS,LPTS,RPTS)
    IF(PADJ.NE.0)THEN
      CALL STRETCH(FULLDIF,L,LPTS,R,RPTS)
    ENDIF
    CALL REVERSE(L,R,LPTS,RPTS)
    CALL DMINMAX(L,R,LPTS,RPTS,XMIN,XMAX)
    IF(ID.EQ.2)THEN
      DKNT=DKNT+1
      WRITE(26,493)P1,L(1,1)*SCALAR,L(1,2)*SCALAR,
*   L(LPTS,1)*SCALAR,L(LPTS,2)*SCALAR,XMIN*SCALAR,
*   XMAX*SCALAR,ID,MKNT
      WRITE(6,492)ID,MKNT
      WRITE(7,492)ID,MKNT
    
```

```
      ELSE
        WRITE(27,493)P1,L(1,1)*SCALAR,L(1,2)*SCALAR,
*      L(LPTS,1)*SCALAR,L(LPTS,2)*SCALAR,XMIN*SCALAR,
*      XMAX*SCALAR,ID,MKNT
        WRITE(8,492)ID,MKNT
        WRITE(9,492)ID,MKNT
      ENDIF
    ENDIF
    CALL PTSFILL(L,LPTS)
    CALL PTSFILL(R,RPTS)
    DO 180 I1=1,LPTS
      L(I1,1)=L(I1,1)*SCALAR
      L(I1,2)=L(I1,2)*SCALAR
      IF(ID.NE.2)THEN
        WRITE(8,490)L(I1,1),L(I1,2)
        WRITE(23,494)L(I1,1),L(I1,2),P1,MKNT
        IF(ID.EQ.3)THEN
          WRITE(6,490)L(I1,1),L(I1,2)
          WRITE(12,490)L(I1,1),L(I1,2)
          WRITE(21,494)L(I1,1),L(I1,2),P1,MKNT
        ELSEIF(ID.EQ.5)THEN
          WRITE(10,490)L(I1,1),L(I1,2)
        ENDIF
      ELSE
        WRITE(6,490)L(I1,1),L(I1,2)
        WRITE(21,494)L(I1,1),L(I1,2),P1,MKNT
      ENDIF
180  CONTINUE
    DO 190 I2=1,RPTS
      R(I2,1)=R(I2,1)*SCALAR
      R(I2,2)=R(I2,2)*SCALAR
      IF(ID.NE.2)THEN
        WRITE(9,490)R(I2,1),R(I2,2)
        WRITE(24,494)R(I2,1),R(I2,2),P1,MKNT
        IF(ID.EQ.3)THEN
          WRITE(7,490)R(I2,1),R(I2,2)
          WRITE(13,490)R(I2,1),R(I2,2)
          WRITE(22,494)R(I2,1),R(I2,2),P1,MKNT
        ELSEIF(ID.EQ.5)THEN
          WRITE(11,490)R(I2,1),R(I2,2)
        ENDIF
      ELSE
        WRITE(7,490)R(I2,1),R(I2,2)
        WRITE(22,494)R(I2,1),R(I2,2),P1,MKNT
      ENDIF
190  CONTINUE
    IF(ID.EQ.2)THEN
      WRITE(6,200)
      WRITE(7,200)
    ELSE
      WRITE(8,200)
```

```

        WRITE(9,200)
    ENDIF
    IF(ID.EQ.3)THEN
        WRITE(6,200)
        WRITE(7,200)
        WRITE(12,200)
        WRITE(13,200)
    ELSEIF(ID.EQ.5)THEN
        WRITE(10,200)
        WRITE(11,200)
    ENDIF
    IF(ID.EQ.2)THEN
        WRITE(8,492)ID,MKNT
        WRITE(9,492)ID,MKNT
        DMPTS=0
        READ(1,*)SX,SY
        IF(SX.EQ.-44.)THEN
            WRITE(27,493)P1,L(1,1),L(1,2),L(LPTS,1),L(LPTS,2),
*           L(1,1),R(RPTS,1),ID,MKNT
            DO 195 I1=1,LPTS
                WRITE(8,490)L(I1,1),L(I1,2)
                WRITE(23,494)L(I1,1),L(I1,2),P1,MKNT
195          CONTINUE
            DO 199 I2=1,RPTS
                WRITE(9,490)R(I2,1),R(I2,2)
                WRITE(24,494)R(I2,1),R(I2,2),P1,MKNT
199          CONTINUE
200          FORMAT(' -88 0')
        ELSE
            DMPTS=DMPTS+1
            S(DMPTS,1)=SX+5.0
            S(DMPTS,2)=SY
210          READ(1,*)SX,SY
            IF(SX.EQ.-44.)GOTO 225
            DMPTS=DMPTS+1
            IF(DMPTS.GT.2750)THEN
                WRITE(25,221)P,BN,DC,SX,SY,ID
                STOP
            ENDIF
            S(DMPTS,1)=SX+5.0
            S(DMPTS,2)=SY
            GOTO 210
        C
        C DEFECT MARGIN POINT COUNT EXCEEDS THE ARRAY DIMENSION.
        C WRITE THIS INFORMATION ON ERROR FILE 25. THEN STOP THE
        C PROGRAM.
        C
221      FORMAT(' THE DEFECT MARGIN POINT COUNT EXCEEDS THE
*          ARRAY DIMENSION OF 2750 ON',/, 'PIECE ',I4, 'BLOCK ',I3,
*          'DIAMETER ',I3,/,
*          'SX= ',F8.3, 'SY= ',F8.3, 'ID= ',F3.0)
225      IF(M1PTS.EQ.0)GOTO 235

```

```

                IF(MAT1(1,1).NE.1.)THEN
                    DO 230 I1=1,DMPTS
                        S(I1,1)=S(I1,1)+10.0
230                CONTINUE
                ENDIF
235            IF(DMPTS.EQ.2)THEN
                LPTS=2
                RPTS=2
                L(1,1)=S(1,1)
                L(1,2)=S(2,2)
                L(2,1)=S(1,1)
                L(2,2)=S(1,2)
                R(1,1)=S(2,1)
                R(1,2)=S(2,2)
                R(2,1)=S(2,1)
                R(2,2)=S(1,2)
                IF(PADJ.NE.0)THEN
                    CALL STRETCH(FULLDIF,L,LPTS,R,RPTS)
                ENDIF
                L(1,2)=ABS(L(1,2)-22.0)
                L(2,2)=ABS(L(2,2)-22.0)
                R(1,2)=ABS(R(1,2)-22.0)
                R(2,2)=ABS(R(2,2)-22.0)
                WRITE(27,493)P1,L(1,1)*SCALAR,L(1,2)*SCALAR,L(2,1)
*                *SCALAR,L(2,2)*SCALAR,L(1,1)*SCALAR,R(2,1)*SCALAR,
*                ID,MKNT
            ELSE
                CALL MINMAX(S,DMPTS,SUM)
                CALL SORTS(S,L,R,SUM,DMPTS,LPTS,RPTS)
                IF(PADJ.NE.0)THEN
                    CALL STRETCH(FULLDIF,L,LPTS,R,RPTS)
                ENDIF
                CALL REVERSE(L,R,LPTS,RPTS)
                CALL DMINMAX(L,R,LPTS,RPTS,XMIN,XMAX)
                WRITE(27,493)P1,L(1,1)*SCALAR,L(1,2)*SCALAR,
*                L(LPTS,1)*SCALAR,L(LPTS,2)*SCALAR,XMIN*SCALAR,
*                XMAX*SCALAR,ID,MKNT
            ENDIF
            CALL PTSFILL(L,LPTS)
            CALL PTSFILL(R,RPTS)
            DO 250 I1=1,LPTS
                L(I1,1)=L(I1,1)*SCALAR
                L(I1,2)=L(I1,2)*SCALAR
                WRITE(8,490)L(I1,1),L(I1,2)
                WRITE(23,494)L(I1,1),L(I1,2),P1,MKNT
250            CONTINUE
            DO 260 I2=1,RPTS
                R(I2,1)=R(I2,1)*SCALAR
                R(I2,2)=R(I2,2)*SCALAR
                WRITE(9,490)R(I2,1),R(I2,2)
                WRITE(24,494)R(I2,1),R(I2,2),P1,MKNT
260            CONTINUE

```

```
ENDIF
WRITE(8,200)
WRITE(9,200)
ENDIF
GOTO 135
270 FORMAT(' -99 0 0 0')
271 IF(M1PTS.NE.0)GOTO 291
275 FORMAT(' -99 0')
IF(DKNT.NE.0)THEN
  CLOSE(26)
  OPEN(26,FILE='SSUM')
  REWIND 26
  CALL SM5SORT(0)
  CALL SM5FAST
  CALL SM5FROM('SSUM')
  CALL SM5TO('SSORT')
  CALL SM5KEY(13,7,'ASCII6','A')
  CALL SM5END
  CLOSE(20)
  OPEN(20,FILE='SSORT')
  REWIND 20
  DO 276 I1=1,DKNT
    READ(20,493)I4,D2,D3,D4,D5,D6,D7,I8,I9
    WRITE(15,493)I4,D2,D3,D4,D5,D6,D7,I8,I9
276  CONTINUE
  WRITE(15,277)
277  FORMAT(' -99 0 0 0 0 0 0 0 0')
  CLOSE(26)
  OPEN(26,FILE='SSUM')
  REWIND 26
  CLOSE(20)
  OPEN(20,FILE='SSORT')
  REWIND 20
  CLOSE(21)
  OPEN(21,FILE='LDCLIP')
  REWIND 21
  CALL SM5SORT(0)
  CALL SM5FAST
  CALL SM5FROM('LDCLIP')
  CALL SM5TO('SSORT')
  CALL SM5KEY(10,16,'ASCII6','A')
  CALL SM5KEY(1,7,'ASCII6','A')
  CALL SM5END
  CLOSE(20)
  OPEN(20,FILE='SSORT')
  REWIND 20
  CLOSE(21)
  OPEN(21,FILE='LDCLIP')
  REWIND 21
278  READ(20,494,END=279)D1,D2,I1,I2
  WRITE(16,494)D1,D2,I1,I2
```

```
      GOTO 278
279  WRITE(16,270)
      CLOSE(22)
      OPEN(22,FILE='RDCLIP')
      REWIND 22
      CLOSE(20)
      OPEN(20,FILE='SSORT')
      REWIND 20
      CALL SM5SORT(0)
      CALL SM5FAST
      CALL SM5FROM('RDCLIP')
      CALL SM5TO('SSORT')
      CALL SM5KEY(10,16,'ASCII6','A')
      CALL SM5KEY(1,7,'ASCII6','A')
      CALL SM5END
      CLOSE (20)
      OPEN(20,FILE='SSORT')
      REWIND 20
      CLOSE(22)
      OPEN(22,FILE='RDCLIP')
      REWIND 22
280  READ(20,494,END=281)D1,D2,I1,I2
      WRITE(17,494)D1,D2,I1,I2
      GOTO 280
281  WRITE(17,270)
      ELSE
        WRITE(6,200)
        WRITE(7,200)
        WRITE(16,270)
        WRITE(17,270)
        WRITE(15,277)
      ENDIF
      DKNT=0
      CLOSE(20)
      OPEN(20,FILE='SSORT')
      REWIND 20
      IF(MKNT.NE.0)THEN
        CLOSE(27)
        OPEN(27,FILE='GSUM')
        REWIND 27
        CALL SM5SORT(0)
        CALL SM5FAST
        CALL SM5FROM('GSUM')
        CALL SM5TO('SSORT')
        CALL SM5KEY(13,7,'ASCII6','A')
        CALL SM5END
        CLOSE(20)
        OPEN(20,FILE='SSORT')
        REWIND 20
        DO 283 I3=1,MKNT
          READ(20,493)I1,D2,D3,D4,D5,D6,D7,I8,I9
```

```
      WRITE(14,493)I1,D2,D3,D4,D5,D6,D7,I8,I9
283  CONTINUE
      WRITE(14,277)
      CLOSE(27)
      OPEN(27,FILE='GSUM')
      REWIND 27
      CLOSE(20)
      OPEN(20,FILE='SSORT')
      REWIND 20
      CLOSE(23)
      OPEN(23,FILE='LMCLIP')
      REWIND 23
      CALL SM5SORT(0)
      CALL SM5FAST
      CALL SM5FROM('LMCLIP')
      CALL SM5TO('SSORT')
      CALL SM5KEY(10,16,'ASCII6','A')
      CALL SM5KEY(1,7,'ASCII6','A')
      CALL SM5END
      CLOSE(20)
      OPEN(20,FILE='SSORT')
      REWIND 20
      CLOSE(23)
      OPEN(23,FILE='LMCLIP')
      REWIND 23
284  READ(20,494,END=285)D1,D2,I1,I2
      WRITE(18,494)D1,D2,I1,I2
      GOTO 284
285  WRITE(18,270)
      CLOSE(24)
      OPEN(24,FILE='RMCLIP')
      REWIND 24
      CLOSE(20)
      OPEN(20,FILE='SSORT')
      REWIND 20
      CALL SM5SORT(0)
      CALL SM5FAST
      CALL SM5FROM('RMCLIP')
      CALL SM5TO('SSORT')
      CALL SM5KEY(10,16,'ASCII6','A')
      CALL SM5KEY(1,7,'ASCII6','A')
      CALL SM5END
      CLOSE(20)
      OPEN(20,FILE='SSORT')
      REWIND 20
      CLOSE(24)
      OPEN(24,FILE='RMCLIP')
      REWIND 24
286  READ(20,494,END=287)D1,D2,I1,I2
      WRITE(19,494)D1,D2,I1,I2
      GOTO 286
```

```
287  WRITE(19,270)
      MKNT=0
      CLOSE(20)
      OPEN(20,FILE='SSORT')
      REWIND 20
      ELSE
        WRITE(18,270)
        WRITE(19,270)
        WRITE(14,277)
      ENDIF
      DO 290 UN=2,13
        WRITE(UN,275)
290  CONTINUE
      GOTO 5

C
C  NOW STARTING ON ALL PIECES FOLLOWING THE FIRST PIECE OF
C  THE RIBBON.
C
291  DIS=0.0
      PADJ=0
      TDIS=0.0
      BDIS=0.0
      LFACT=0.0
      RFACT=0.0
      FULLDIF=0.0
      LSKEW=0.0
      RSKEW=0.0
      LXCHNG=0.0
      RXCHNG=0.0
      XC=0.0
      XDIS=0.0
      READ(1,*)DC,BN,P
      WRITE(25,*)'PIECE # = ',P
      M2PTS=0
292  READ(1,*)M,MX,MY
      IF(M.NE.-22.)THEN
        M2PTS=M2PTS+1
        TEMP(M2PTS,1)=M
        TEMP(M2PTS,2)=MX+5.0
        TEMP(M2PTS,3)=MY
        GO TO 292
      ENDIF
      IF(M2PTS.EQ.2)THEN
        MAT2(1,1)=TEMP(1,1)
        MAT2(1,2)=TEMP(1,2)
        MAT2(1,3)=TEMP(1,3)
        MAT2(2,1)=TEMP(2,1)
        MAT2(2,2)=TEMP(2,2)
        MAT2(2,3)=TEMP(2,3)
      ELSE
        MAT2(1,1)=TEMP(1,1)
        MAT2(1,2)=TEMP(1,2)
```



```
      MAT2(1,3)=TEMP(1,3)
      MAT2(2,1)=TEMP(3,1)
      MAT2(2,2)=TEMP(3,2)
      MAT2(2,3)=TEMP(3,3)
      MAT2(3,1)=TEMP(4,1)
      MAT2(3,2)=TEMP(4,2)
      MAT2(3,3)=TEMP(4,3)
      MAT2(4,1)=TEMP(2,1)
      MAT2(4,2)=TEMP(2,2)
      MAT2(4,3)=TEMP(2,3)
ENDIF
SPTS=0
DO 295 I5=1,4
  DO 295 I6=1,2
    SKEW(I5,I6)=0.0
    DUMB(I5,I6)=0.0
295 CONTINUE
300 READ(1,*)SX,SY
  IF(SX.NE.-77.)THEN
    SPTS=SPTS+1
    IF(SPTS.GT.2750)THEN
      WRITE(25,60)P,BN,DC,SX,SY
      STOP
    ENDIF
    S(SPTS,1)=SX+5.0
    S(SPTS,2)=SY
    GOTO 300
  ENDIF
  IF(SPTS.LE.4)THEN
    IF(SPTS.EQ.4)THEN
      L(1,1)=S(2,1)
      L(1,2)=S(2,2)
      L(2,1)=S(1,1)
      L(2,2)=S(1,2)
      R(1,1)=S(3,1)
      R(1,2)=S(3,2)
      R(2,1)=S(4,1)
      R(2,2)=S(4,2)
    ELSE
      L(1,1)=MAT2(2,2)
      L(1,2)=MAT2(2,3)
      L(2,1)=MAT2(1,2)
      L(2,2)=MAT2(1,3)
      R(1,1)=MAT2(3,2)
      R(1,2)=MAT2(3,3)
      R(2,1)=MAT2(4,2)
      R(2,2)=MAT2(4,3)
    ENDIF
    SKEW(3,1)=L(1,1)
    SKEW(3,2)=L(1,2)
    SKEW(1,1)=L(2,1)
```

```

      SKEW(1,2)=L(2,2)
      SKEW(4,1)=R(1,1)
      SKEW(4,2)=R(1,2)
      SKEW(2,1)=R(2,1)
      SKEW(2,2)=R(2,2)
      LPTS=2
      RPTS=2
      GOTO 320
    ELSE
      CALL MINMAX(S,SPTS,SUM)
      CALL SORTS(S,L,R,SUM,SPTS,LPTS,RPTS)
    ENDIF
    IF(M2PTS.EQ.2)THEN
      CALL SORT2(L,R,MAT2,LPTS,RPTS)
    ELSE
      SKEW(1,1)=MAT2(1,2)
      SKEW(1,2)=MAT2(1,3)
      SKEW(2,1)=MAT2(4,2)
      SKEW(2,2)=MAT2(4,3)
      SKEW(3,1)=MAT2(2,2)
      SKEW(3,2)=MAT2(2,3)
      SKEW(4,1)=MAT2(3,2)
      SKEW(4,2)=MAT2(3,3)
      CALL SORT3(L,R,MAT2,LPTS,RPTS,SKEW)
    ENDIF
320 LBY=L(LPTS,2)
    RBY=R(RPTS,2)
    IF(M2PTS.EQ.4)THEN
      CALL PCSKEW(L,R,LPTS,RPTS,LFACT,RFACT,LSKEW,
*      RSKEW,LXCHNG,RXCHNG,DIS,SKEW)
      TDIS=R(1,1)-L(1,1)
      BDIS=R(RPTS,1)-L(LPTS,1)
      XDIS=MAX(TDIS,BDIS)
      FULLDIF=FULLDIS-XDIS
      IF(FULLDIF.NE.0.0)THEN
        IF(FULLDIF.LE.(4.0/SCALAR))THEN
          PADJ=1
          CALL STRETCH(FULLDIF,L,LPTS,R,RPTS)
        ENDIF
      ENDIF
      IF((MAT2(1,1)+MAT2(2,1)).EQ.3.)THEN
        XC=F1-L(1,1)
      ELSEIF((MAT2(3,1)+MAT2(4,1)).EQ.7.)THEN
        XC=F3-R(1,1)
      ELSEIF(MAT2(1,1).EQ.1.)THEN
        XC=F1-L(1,1)
      ELSE
        XC=F3-R(1,1)
      ENDIF
    ELSE
      XDIS=R(1,1)-L(1,1)

```

```

        FULLDIF=FULLDIS-XDIS
        IF(FULLDIF.NE.0.0)THEN
            IF(FULLDIF.LE.(4.0/SCALAR))THEN
                PADJ=1
                CALL STRETCH(FULLDIF,L,LPTS,R,RPTS)
            ENDIF
        ENDIF
        IF(MAT2(1,1).EQ.2.)THEN
            XC=F1-L(1,1)
        ELSE
            XC=F3-R(1,1)
        ENDIF
    ENDIF
    YDIS=L(1,2)
    WRITE(25,*)' XDIS= ',XDIS,' TDIS= ',TDIS,' BDIS= ',BDIS
    WRITE(25,*)' FULLDIF= ',FULLDIF,' XC= ',XC,' PADJ= ',PADJ
    CALL CONNECT(L,R,LPTS,RPTS,YDIS,YSTRT,XC)
    IF(M2PTS.EQ.4)THEN
        YHLD=L(LPTS,2)
        F1=L(LPTS,1)
        F3=R(RPTS,1)
    ENDIF
    WRITE(25,*)' F1= ',F1,' F3= ',F3
    WRITE(25,*)' SHAPE POINTS:'
    WRITE(25,80)LPTS,RPTS
    CALL PTSFILL(L,LPTS)
    CALL PTSFILL(R,RPTS)
    WRITE(25,*)' SHAPE POINTS PTSFILL:'
    WRITE(25,80)LPTS,RPTS
    DO 330 I1=1,LPTS
        L(I1,1)=L(I1,1)*SCALAR
        L(I1,2)=L(I1,2)*SCALAR
        WRITE(2,490)L(I1,1),L(I1,2)
330 CONTINUE
    DO 340 I2=1,RPTS
        R(I2,1)=R(I2,1)*SCALAR
        R(I2,2)=R(I2,2)*SCALAR
        WRITE(3,490)R(I2,1),R(I2,2)
340 CONTINUE
    MGPTS=0
345 READ(1,*)SX,SY
    IF(SX.NE.-55.)THEN
        MGPTS=MGPTS+1
        IF(MGPTS.GT.2750)THEN
            WRITE(25,105)P,BN,DC,SX,SY
            STOP
        ENDIF
        S(MGPTS,1)=SX+5.0
        S(MGPTS,2)=SY
        GOTO 345
    ENDIF

```

```

C
C IF MGPTS EQUALS ZERO, THEN THERE IS NO SHAPE MARGIN.
C
      IF(MGPTS.EQ.0)THEN
        DO 348 I1=1,LPTS
          WRITE(4,490)L(I1,1),L(I1,2)
348    CONTINUE
        DO 349 I2=1,RPTS
          WRITE(5,490)R(I2,1),R(I2,2)
349    CONTINUE
        GOTO 355
      ENDIF
      IF(MGPTS.EQ.4)THEN
        LPTS=2
        RPTS=2
        L(1,1)=S(2,1)
        L(1,2)=S(2,2)
        L(2,1)=S(1,1)
        L(2,2)=S(1,2)
        R(1,1)=S(3,1)
        R(1,2)=S(3,2)
        R(2,1)=S(4,1)
        R(2,2)=S(4,2)
      ELSE
        CALL MINMAX(S,MGPTS,SUM)
        CALL SORTS(S,L,R,SUM,MGPTS,LPTS,RPTS)
      ENDIF
      IF(M2PTS.EQ.4)THEN
        CALL SORT3(L,R,MAT2,LPTS,RPTS,DUMB)
        CALL PCSKEW(L,R,LPTS,RPTS,LFACT,RFACT,LSKEW,RSKEW,
*      LXCHNG,RXCHNG,DIS,SKEW)
      ELSE
        CALL SORT2(L,R,MAT2,LPTS,RPTS)
      ENDIF
      IF(PADJ.NE.0)THEN
        CALL STRETCH(FULLDIF,L,LPTS,R,RPTS)
      ENDIF
      CALL CONNECT(L,R,LPTS,RPTS,YDIS,YSTRT,XC)
      WRITE(25,*)' MARGIN POINTS: '
      WRITE(25,80)LPTS,RPTS
      CALL PTSFILL(L,LPTS)
      CALL PTSFILL(R,RPTS)
      WRITE(25,*)' MARGIN POINTS PTSFILL: '
      WRITE(25,80)LPTS,RPTS
      DO 353 I1=1,LPTS
        L(I1,1)=L(I1,1)*SCALAR
        L(I1,2)=L(I1,2)*SCALAR
        WRITE(4,490)L(I1,1),L(I1,2)
353    CONTINUE
      DO 354 I2=1,RPTS
        R(I2,1)=R(I2,1)*SCALAR

```

```

        R(I2,2)=R(I2,2)*SCALAR
        WRITE(5,490)R(I2,1),R(I2,2)
354 CONTINUE
355 DPTS=0
        SFLAG=0
360 READ(1,*)ID,SY
C
C IF ID EQUALS -88 THE FIRST TIME, THEN THERE ARE NO DEFECTS.
C
        IF(ID.EQ.-88.)GOTO 360
        IF(ID.EQ.-44.)GOTO 360
        IF(ID.EQ.-99.)GO TO 485
365 READ(1,*)SX,SY
        IF(SX.NE.-88.)THEN
            DPTS=DPTS+1
            IF(DPTS.GT.2750)THEN
                WRITE(25,150)P,BN,DC,SX,SY,ID
                STOP
            ENDIF
            S(DPTS,1)=SX+5.0
            S(DPTS,2)=SY
            GOTO 365
        ENDIF
        MKNT=MKNT+1
        IF(DPTS.EQ.2)THEN
C
C TWO POINT DIGITIZING.
C
            LPTS=2
            RPTS=2
            L(1,1)=S(1,1)
            L(1,2)=S(2,2)
            L(2,1)=S(1,1)
            L(2,2)=S(1,2)
            R(1,1)=S(2,1)
            R(1,2)=S(2,2)
            R(2,1)=S(2,1)
            R(2,2)=S(1,2)
            IF(M2PTS.EQ.4)THEN
                IF((L(1,1)-F1).LT.(F3-L(1,1)))THEN
                    SFLAG=1
                ENDIF
                CALL DEFSKEW(L,R,LPTS,RPTS,LFACT,RFACT,LSKEW,RSKEW,
*           LXCHNG,RXCHNG,DIS,LBY,RBY,SFLAG)
            ENDIF
            IF(PADJ.NE.0)THEN
                CALL STRETCH(FULLDIF,L,LPTS,R,RPTS)
            ENDIF
            CALL CONNECT(L,R,LPTS,RPTS,YDIS,YSTRT,XC)
            IF(ID.EQ.2)THEN
                DKNT=DKNT+1

```

```

        WRITE(6,492)ID,MKNT
        WRITE(7,492)ID,MKNT
        WRITE(26,493)P1,L(1,1)*SCALAR,L(1,2)*SCALAR,L(2,1)*
*       SCALAR,L(2,2)*SCALAR,L(1,1)*SCALAR,R(2,1)*SCALAR,ID,
*       MKNT
        ELSE
        WRITE(8,492)ID,MKNT
        WRITE(9,492)ID,MKNT
        WRITE(27,493)P1,L(1,1)*SCALAR,L(1,2)*SCALAR,L(2,1)*
*       SCALAR L(2,2)*SCALAR,L(1,1)*SCALAR,R(2,1)*SCALAR,ID,
*       MKNT
        ENDIF
    ELSEIF(ID.EQ.3)THEN
        LPTS=2
        RPTS=2
        L(1,1)=S(1,1)
        L(1,2)=S(1,2)
        L(2,1)=S(2,1)
        L(2,2)=S(2,2)
        R(1,1)=S(3,1)
        R(1,2)=S(3,2)
        R(2,1)=S(4,1)
        R(2,2)=S(4,2)
        IF(M2PTS.EQ.4)THEN
            IF((L(1,1)-F1).LT.(F3-L(1,1)))THEN
                SFLAG=1
            ENDIF
            CALL DEFSKEW(L,R,LPTS,RPTS,LFACT,RFACT,LSKEW,RSKEW,
*           LXCHNG,RXCHNG,DIS,LBY,RBY,SFLAG)
        ENDIF
        IF(PADJ.NE.0)THEN
            IF(SFLAG.EQ.0)THEN
                DO 380 I8=1,2
                    L(I8,1)=L(I8,1)+FULLDIF*.5
                    R(I8,1)=R(I8,1)+FULLDIF*.5
380          CONTINUE
                ELSE
                    DO 381 I9=1,2
                        L(I9,1)=L(I9,1)-FULLDIF*.5
                        R(I9,1)=R(I9,1)-FULLDIF*.5
381          CONTINUE
                ENDIF
            ENDIF
            CALL CONNECT(L,R,LPTS,RPTS,YDIS,YSTRT,XC)
            IF(L(1,1).LT.L(2,1))THEN
                MINX=L(1,1)
                MAXX=L(2,1)
            ELSE
                MINX=L(2,1)
                MAXX=L(1,1)
            ENDIF
            IF(L(1,2).GT.L(2,2))THEN
                XN=L(1,1)

```

```

        MINY=L(1,2)
    ELSE
        XN=L(2,1)
        MINY=L(2,2)
    ENDIF
    IF(R(1,2).GT.R(2,2))THEN
        XX=R(1,1)
        MAXY=R(1,2)
    ELSE
        XX=R(2,1)
        MAXY=R(2,2)
    ENDIF
    DKNT=DKNT+1
    WRITE(26,493)P1,XN*SCALAR,MINY*SCALAR,XX*SCALAR,
*   MAXY*SCALAR,MINX*SCALAR,MAXX*SCALAR,ID,MKNT
    WRITE(6,492)ID,MKNT
    WRITE(7,492)ID,MKNT
    WRITE(27,493)P1,XN*SCALAR,MINY*SCALAR,XX*SCALAR,
*   MAXY*SCALAR,MINX*SCALAR,MINY*SCALAR,ID,MKNT
    WRITE(8,492)ID,MKNT
    WRITE(9,492)ID,MKNT
    WRITE(12,492)ID,MKNT
    WRITE(13,492)ID,MKNT
ELSE
    CALL MINMAX(S,DPTS,SUM)
    CALL SORTS(S,L,R,SUM,DPTS,LPTS,RPTS)
    IF(M2PTS.EQ.4)THEN
        IF((L(1,1)-F1).LT.(F3-L(1,1)))THEN
            SFLAG=1
        ENDIF
        CALL DEFSKEW(L,R,LPTS,RPTS,LFACT,RFACT,LSKEW,RSKEW,
*   LXCHNG,RXCHNG,DIS,LBY,RBY,SFLAG)
    ENDIF
    IF(PADJ.NE.0)THEN
        CALL STRETCH(FULLDIF,L,LPTS,R,RPTS)
    ENDIF
    CALL CONNECT(L,R,LPTS,RPTS,YDIS,YSTRT,XC)
    CALL DMINMAX(L,R,LPTS,RPTS,XMIN,XMAX)
    IF(ID.EQ.2)THEN
        DKNT=DKNT+1
        WRITE(26,493)P1,L(1,1)*SCALAR,L(1,2)*SCALAR,
*   L(LPTS,1)*SCALAR,L(LPTS,2)*SCALAR,XMIN*SCALAR,
*   XMAX*SCALAR,ID,MKNT
        WRITE(6,492)ID,MKNT
        WRITE(7,492)ID,MKNT
    ELSE
        WRITE(27,493)P1,L(1,1)*SCALAR,L(1,2)*SCALAR,
*   L(LPTS,1)*SCALAR,L(LPTS,2)*SCALAR,XMIN*SCALAR,
*   XMAX*SCALAR,ID,MKNT
        WRITE(8,492)ID,MKNT
        WRITE(9,492)ID,MKNT
    
```

```
      ENDIF
    ENDIF
    CALL PTSFILL(L,LPTS)
    CALL PTSFILL(R,RPTS)
    DO 385 I1=1,LPTS
      L(I1,1)=L(I1,1)*SCALAR
      L(I1,2)=L(I1,2)*SCALAR
      IF (ID.NE.2) THEN
        WRITE(8,490)L(I1,1),L(I1,2)
        WRITE(23,494)L(I1,1),L(I1,2),P1,MKNT
        IF (ID.EQ.3) THEN
          WRITE(6,490)L(I1,1),L(I1,2)
          WRITE(12,490)L(I1,1),L(I1,2)
          WRITE(21,494)L(I1,1),L(I1,2),P1,MKNT
        ELSEIF (ID.EQ.5) THEN
          WRITE(10,490)L(I1,1),L(I1,2)
        ENDIF
      ELSE
        WRITE(6,490)L(I1,1),L(I1,2)
        WRITE(21,494)L(I1,1),L(I1,2),P1,MKNT
      ENDIF
385  CONTINUE
    DO 390 I2=1,RPTS
      R(I2,1)=R(I2,1)*SCALAR
      R(I2,2)=R(I2,2)*SCALAR
      IF (ID.NE.2) THEN
        WRITE(9,490)R(I2,1),R(I2,2)
        WRITE(24,494)R(I2,1),R(I2,2),P1,MKNT
        IF (ID.EQ.3) THEN
          WRITE(7,490)R(I2,1),R(I2,2)
          WRITE(13,490)R(I2,1),R(I2,2)
          WRITE(22,494)R(I2,1),R(I2,2),P1,MKNT
        ELSEIF (ID.EQ.5) THEN
          WRITE(11,490)R(I2,1),R(I2,2)
        ENDIF
      ELSE
        WRITE(7,490)R(I2,1),R(I2,2)
        WRITE(22,494)R(I2,1),R(I2,2),P1,MKNT
      ENDIF
390  CONTINUE
    IF (ID.EQ.2) THEN
      WRITE(6,200)
      WRITE(7,200)
    ELSE
      WRITE(8,200)
      WRITE(9,200)
    ENDIF
    IF (ID.EQ.3) THEN
      WRITE(6,200)
      WRITE(7,200)
      WRITE(12,200)
```



```

        WRITE(13,200)
    ELSEIF(ID.EQ.5)THEN
        WRITE(10,200)
        WRITE(11,200)
    ENDIF
    IF(ID.EQ.2)THEN
        WRITE(8,492)ID,MKNT
        WRITE(9,492)ID,MKNT
        DMPTS=0
        READ(1,*)SX,SY
        IF(SX.EQ.-44.)THEN
            WRITE(27,493)P1,L(1,1),L(1,2),L(LPTS,1),L(LPTS,2),
*           L(1,1),R(RPTS,1),ID,MKNT
            DO 400 I1=1,LPTS
                WRITE(8,490)L(I1,1),L(I1,2)
                WRITE(23,494)L(I1,1),L(I1,2),P1,MKNT
400          CONTINUE
            DO 410 I2=1,RPTS
                WRITE(9,490)R(I2,1),R(I2,2)
                WRITE(24,494)R(I2,1),R(I2,2),P1,MKNT
410          CONTINUE
        ELSE
            DMPTS=DMPTS+1
            S(DMPTS,1)=SX+5.0
            S(DMPTS,2)=SY
420          READ(1,*)SX,SY
            IF(SX.EQ.-44.)GO TO 425
            DMPTS=DMPTS+1
            IF(DMPTS.GT.2750)THEN
                WRITE(25,150)P,BN,DC,SX,SY,ID
                STOP
            ENDIF
            S(DMPTS,1)=SX+5.0
            S(DMPTS,2)=SY
            GOTO 420
425          IF(DMPTS.EQ.2)THEN
                LPTS=2
                RPTS=2
                L(1,1)=S(1,1)
                L(1,2)=S(2,2)
                L(2,1)=S(1,1)
                L(2,2)=S(1,2)
                R(1,1)=S(2,1)
                R(1,2)=S(2,2)
                R(2,1)=S(2,1)
                R(2,2)=S(1,2)
                IF(M2PTS.EQ.4)THEN
                    CALL DEFSKEW(L,R,LPTS,RPTS,LFACT,RFACT,LSKEW,
*                   RSKEW,LXCHNG,RXCHNG,DIS,LBY,RBY,SFLAG)
                ENDIF
                IF(PADJ.NE.0)THEN

```

```

        CALL STRETCH(FULLDIF,L,LPTS,R,RPTS)
    ENDIF
    CALL CONNECT(L,R,LPTS,RPTS,YDIS,YSTRT,XC)
    WRITE(27,493)P1,L(1,1)*SCALAR,L(1,2)*SCALAR,L(2,1),
*   SCALAR,L(2,2)*SCALAR,L(1,1)*SCALAR,R(2,1)*SCALAR,ID,
*   MKNT
    ELSE
        CALL MINMAX(S,DMPTS,SUM)
        CALL SORTS(S,L,R,SUM,DMPTS,LPTS,RPTS)
        IF(M2PTS.EQ.4)THEN
            CALL DEFSKEW(L,R,LPTS,RPTS,LFACT,RFACT,LSKEW,RSKEW,
*   LXCHNG,RXCHNG,DIS,LBY,RBY,SFLAG)
        ENDIF
        IF(PADJ.NE.0)THEN
            CALL STRETCH(FULLDIF,L,LPTS,R,RPTS)
        ENDIF
        CALL CONNECT(L,R,LPTS,RPTS,YDIS,YSTRT,XC)
        CALL DMINMAX(L,R,LPTS,RPTS,XMIN,XMAX)
        WRITE(27,493)P1,L(1,1)*SCALAR,L(1,2)*SCALAR,
*   L(LPTS,1)*SCALAR,L(LPTS,2)*SCALAR,XMIN*SCALAR,
*   XMAX*SCALAR,ID,MKNT
    ENDIF
    CALL PTSFILL(L,LPTS)
    CALL PTSFILL(R,RPTS)
    DO 430 I1=1,LPTS
        L(I1,1)=L(I1,1)*SCALAR
        L(I1,2)=L(I1,2)*SCALAR
        WRITE(8,490)L(I1,1),L(I1,2)
        WRITE(23,494)L(I1,1),L(I1,2),P1,MKNT
430    CONTINUE
    DO 440 I2=1,RPTS
        R(I2,1)=R(I2,1)*SCALAR
        R(I2,2)=R(I2,2)*SCALAR
        WRITE(9,490)R(I2,1),R(I2,2)
        WRITE(24,494)R(I2,1),R(I2,2),P1,MKNT
440    CONTINUE
    ENDIF
    WRITE(8,200)
    WRITE(9,200)
    ENDIF
    GO TO 355
485 IF(M2PTS.EQ.2)THEN
    M1PTS=0
    GOTO 271
    ELSE
    YSTRT=YHLD
    GOTO 291
    ENDIF
490 FORMAT(F8.3,2X,F8.3)
491 FORMAT(I2,2X,I2,2X,I3)
492 FORMAT(I2,2X,I4)
493 FORMAT(I3,1X,F8.3,2X,F8.3,2X,F8.3,2X,F8.3,2X,F8.3,2X,

```

```

      *F8.3,2X,I2,1X,I4)
494  FORMAT(F8.3,4X,F8.3,4X,I3,2X,I4)
      END

```

```

*****
*  THIS SUBROUTINE SORTS CONTINUOSLY DIGITIZED DATA  *
*  INTO LEFT AND RIGHT SIDES.                        *
*****

```

```

      SUBROUTINE SORTS(S,L,R,SUM,SPTS,LPTS,RPTS)
      INTEGER SUM(2),SPTS,LPTS,RPTS
      REAL L(2750,2),R(2750,2),S(2750,2)
      LPTS=0
      RPTS=0
      IF(SUM(1).GT.SUM(2))THEN
        DO 10 I1=SUM(2),1,-1
          LPTS=LPTS+1
          L(LPTS,1)=S(I1,1)
          L(LPTS,2)=S(I1,2)
10      CONTINUE
        DO 20 I2=SPTS,SUM(1),-1
          LPTS=LPTS+1
          L(LPTS,1)=S(I2,1)
          L(LPTS,2)=S(I2,2)
20      CONTINUE
        ELSE
          DO 30 I3=SUM(2),SUM(1),-1
            LPTS=LPTS+1
            L(LPTS,1)=S(I3,1)
            L(LPTS,2)=S(I3,2)
30      CONTINUE
        ENDIF
        IF(SUM(1).GT.SUM(2))THEN
          DO 40 I4=SUM(2),SUM(1)
            RPTS=RPTS+1
            R(RPTS,1)=S(I4,1)
            R(RPTS,2)=S(I4,2)
40      CONTINUE
          ELSE
            DO 50 I5=SUM(2),SPTS
              RPTS=RPTS+1
              R(RPTS,1)=S(I5,1)
              R(RPTS,2)=S(I5,2)
50      CONTINUE
            DO 60 I6=1,SUM(1)
              RPTS=RPTS+1
              R(RPTS,1)=S(I6,1)
              R(RPTS,2)=S(I6,2)
60      CONTINUE
          ENDIF
        RETURN
      END

```

```

*****
* THIS SUBROUTINE SORTS SHAPE POINTS WHERE THE PIECE *
* HAS ONLY TWO LEADING MATCH POINTS. *
*****
      SUBROUTINE SORT1(L,R,MAT,LPTS,RPTS)
      INTEGER LPTS,RPTS,PTS
      REAL MAT(4,3),L(2750,2),R(2750,2),T(2750,2)
      COMMON T
      PTS=0
      DO 20 I2=1,LPTS
        IF(L(I2,2).GT.MAT(1,3))THEN
          PTS=PTS+1
          IF(PTS.GT.2750)THEN
            WRITE(25,*)'ARRAY OVERFLOW IN SORT1 LEFT SIDE.'
            STOP
          ENDIF
          T(PTS,1)=L(I2,1)
          T(PTS,2)=L(I2,2)
        ELSE
          GOTO 30
        ENDIF
      20 CONTINUE
      30 IF(L(I2,1).LE.(MAT(1,2)-.10))THEN
        PTS=PTS+1
        T(PTS,1)=L(I2,1)
        T(PTS,2)=MAT(1,3)
      ENDIF
      PTS=PTS+1
      T(PTS,1)=MAT(1,2)
      T(PTS,2)=MAT(1,3)
      DO 35 I5=1,PTS
        L(I5,1)=T(I5,1)
        L(I5,2)=T(I5,2)
      35 CONTINUE
      LPTS=PTS
      PTS=0
      DO 40 I4=1,RPTS
        IF(R(I4,2).GT.MAT(2,3))THEN
          PTS=PTS+1
          IF(PTS.GT.2750)THEN
            WRITE(25,*)'ARRAY OVERFLOW IN SORT1 RIGHT SIDE.'
            STOP
          ENDIF
          T(PTS,1)=R(I4,1)
          T(PTS,2)=R(I4,2)
        ELSE
          GOTO 50
        ENDIF
      40 CONTINUE
      50 IF(R(I4,1).GE.(MAT(2,2)+.10))THEN
        PTS=PTS+1
        T(PTS,1)=R(I4,1)

```

```

      T(PTS,2)=MAT(2,3)
    ENDIF
    PTS=PTS+1
    T(PTS,1)=MAT(2,2)
    T(PTS,2)=MAT(2,3)
    DO 53 I3=1,PTS
      R(I3,1)=T(I3,1)
      R(I3,2)=T(I3,2)
53  CONTINUE
    RPTS=PTS
    RETURN
  END
*****
*  THIS SUBROUTINE SORTS SHAPE POINTS WHERE THE PIECE  *
*  HAS ONLY TWO TRAILING EDGE MATCH POINTS.           *
*****
  SUBROUTINE SORT2(L,R,MAT,LPTS,RPTS)
    INTEGER PTR,PTS,LPTS,RPTS
    REAL MAT(4,3),L(2750,2),R(2750,2),T(2750,2)
    COMMON T
    PTS=0
    PTR=0
    PTS=PTS+1
    T(PTS,1)=MAT(1,2)
    T(PTS,2)=MAT(1,3)
10  PTR=PTR+1
    IF(L(PTR,2).GE.MAT(1,3))THEN
      IF(L(PTR,1).LE.(MAT(1,2)-.10))THEN
        PTS=PTS+1
        T(PTS,1)=L(PTR,1)
        T(PTS,2)=MAT(1,3)
      ENDIF
      GOTO 10
    ENDIF
    DO 20 I2=PTR,LPTS
      PTS=PTS+1
      IF(PTS.GT.2750)THEN
        WRITE(25,*)'ARRAY OVERFLOW IN SORT2 LEFT SIDE.'
        STOP
      ENDIF
      T(PTS,1)=L(I2,1)
      T(PTS,2)=L(I2,2)
20  CONTINUE
    DO 25 I5=1,PTS
      L(I5,1)=T(I5,1)
      L(I5,2)=T(I5,2)
25  CONTINUE
    LPTS=PTS
    PTS=0
    PTR=0
    PTS=PTS+1

```

```

      T(PTS,1)=MAT(2,2)
      T(PTS,2)=MAT(2,3)
30  PTR=PTR+1
      IF(R(PTR,2).GE.MAT(2,3))THEN
        IF(R(PTR,1).GE.(MAT(2,2)+.10))THEN
          PTS=PTS+1
          T(PTS,1)=R(PTR,1)
          T(PTS,2)=MAT(2,3)
        ENDIF
        GOTO 30
      ENDIF
      DO 40 I4=PTR,RPTS
        PTS=PTS+1
        IF(PTS.GT.2750)THEN
          WRITE(25,*)'ARRAY OVERFLOW IN SORT2 RIGHT SIDE.'
          STOP
        ENDIF
        T(PTS,1)=R(I4,1)
        T(PTS,2)=R(I4,2)
40  CONTINUE
      DO 43 I3=1,PTS
        R(I3,1)=T(I3,1)
        R(I3,2)=T(I3,2)
43  CONTINUE
      RPTS=PTS
      RETURN
      END

```

```

*****
*  THIS SUBROUTINE SORTS SHAPE POINTS WHERE THE PIECE  *
*  HAS FOUR MATCHING POINTS.                          *
*****

```

```

      SUBROUTINE SORT3(L,R,MAT,LPTS,RPTS,SKEW)
      INTEGER PTR,PTS,LPTS,RPTS
      REAL MAT(4,3),L(2750,2),R(2750,2),SKEW(4,2),
      *T(2750,2)
      COMMON T
      PTS=0
      PTR=0
      PTS=PTS+1
      T(PTS,1)=MAT(2,2)
      T(PTS,2)=MAT(2,3)
10  PTR=PTR+1
      IF(L(PTR,2).GE.MAT(2,3))THEN
        IF(L(PTR,1).LE.(MAT(2,2)-.10))THEN
          PTS=PTS+1
          T(PTS,1)=L(PTR,1)
          T(PTS,2)=MAT(2,3)
          IF(L(PTR,1).LT.SKEW(3,1))THEN
            SKEW(3,1)=L(PTR,1)
          ENDIF
        ENDIF
      ENDIF

```

```

      GOTO 10
    ELSEIF(L(PTR,2).GE.MAT(1,3))THEN
      PTS=PTS+1
      T(PTS,1)=L(PTR,1)
      T(PTS,2)=L(PTR,2)
      IF(L(PTR,1).LT.SKEW(1,1))THEN
        SKEW(1,1)=L(PTR,1)
      ENDIF
    ELSEIF(L(PTR,1).LE.(MAT(1,2)-.10))THEN
      PTS=PTS+1
      T(PTS,1)=L(PTR,1)
      T(PTS,2)=MAT(1,3)
      IF(L(PTR,1).LT.SKEW(1,1))THEN
        SKEW(1,1)=L(PTR,1)
      ENDIF
      GOTO 30
    ELSE
      GOTO 30
    ENDIF
20 PTR=PTR+1
    IF(L(PTR,2).GE.MAT(1,3))THEN
      PTS=PTS+1
      IF(PTS.GT.2750)THEN
        WRITE(25,*)'ARRAY OVERFLOW IN SORT3 LEFT SIDE.'
        STOP
      ENDIF
      T(PTS,1)=L(PTR,1)
      T(PTS,2)=L(PTR,2)
      IF(L(PTR,1).LT.SKEW(1,1))THEN
        SKEW(1,1)=L(PTR,1)
      ENDIF
      GOTO 20
    ELSEIF(L(PTR,1).LE.(MAT(1,2)-.10))THEN
      PTS=PTS+1
      T(PTS,1)=L(PTR,1)
      T(PTS,2)=MAT(1,3)
      IF(L(PTR,1).LT.SKEW(1,1))THEN
        SKEW(1,1)=L(PTR,1)
      ENDIF
    ENDIF
30 PTS=PTS+1
    T(PTS,1)=MAT(1,2)
    T(PTS,2)=MAT(1,3)
    DO 35 I5=1,PTS
      L(I5,1)=T(I5,1)
      L(I5,2)=T(I5,2)
35 CONTINUE
    LPTS=PTS
    PTS=0
    PTR=0
    PTS=PTS+1

```

```
T(PTS,1)=MAT(3,2)
T(PTS,2)=MAT(3,3)
40 PTR=PTR+1
IF(R(PTR,2).GE.MAT(3,3))THEN
  IF(R(PTR,1).GE.(MAT(3,2)+.10))THEN
    PTS=PTS+1
    T(PTS,1)=R(PTR,1)
    T(PTS,2)=MAT(3,3)
    IF(R(PTR,1).GT.SKEW(4,1))THEN
      SKEW(4,1)=R(PTR,1)
    ENDIF
  ENDIF
  GOTO 40
ELSEIF(R(PTR,2).GE.MAT(4,3))THEN
  PTS=PTS+1
  IF(PTS.GT.2750)THEN
    WRITE(25,*)'ARRAY OVERFLOW IN SORT3 RIGHT SIDE.'
    STOP
  ENDIF
  T(PTS,1)=R(PTR,1)
  T(PTS,2)=R(PTR,2)
  IF(R(PTR,1).GT.SKEW(2,1))THEN
    SKEW(2,1)=R(PTR,1)
  ENDIF
  ELSEIF(R(PTR,1).GE.(MAT(4,2)+.10))THEN
    PTS=PTS+1
    T(PTS,1)=R(PTR,1)
    T(PTS,2)=MAT(4,3)
    IF(R(PTR,1).GT.SKEW(2,1))THEN
      SKEW(2,1)=R(PTR,1)
    ENDIF
    GOTO 60
  ELSE
    GOTO 60
  ENDIF
50 PTR=PTR+1
IF(R(PTR,2).GE.MAT(4,3))THEN
  PTS=PTS+1
  IF(PTS.GT.2750)THEN
    WRITE(25,*)'ARRAY OVERFLOW IN SORT3 RIGHT SIDE.'
    STOP
  ENDIF
  T(PTS,1)=R(PTR,1)
  T(PTS,2)=R(PTR,2)
  IF(R(PTR,1).GT.SKEW(2,1))THEN
    SKEW(2,1)=R(PTR,1)
  ENDIF
  GOTO 50
ELSEIF(R(PTR,1).GE.(MAT(4,2)+.10))THEN
  PTS=PTS+1
  T(PTS,1)=R(PTR,1)
```



```

      T(PTS,2)=MAT(4,3)
      IF(R(PTR,1).GT.SKEW(2,1))THEN
        SKEW(2,1)=R(PTR,1)
      ENDIF
    ENDIF
  60 PTS=PTS+1
    T(PTS,1)=MAT(4,2)
    T(PTS,2)=MAT(4,3)
    DO 63 I3=1,PTS
      R(I3,1)=T(I3,1)
      R(I3,2)=T(I3,2)
  63 CONTINUE
    RPTS=PTS
    WRITE(25,*)' SORT3----SKEW'
    WRITE(25,65)SKEW(1,1),SKEW(1,2),SKEW(2,1),SKEW(2,2)
    WRITE(25,70)SKEW(3,1),SKEW(3,2),SKEW(4,1),SKEW(4,2)
  65 FORMAT(' (1,1)= ',F8.3,2X,'(1,2)= ',F8.3,2X,'(2,1)= ',
    *F8.3,2X,'(2,2)= ',F8.3)
  70 FORMAT(' (3,1)= ',F8.3,2X,'(3,2)= ',F8.3,2X,'(4,1)= ',
    *F8.3,2X,'(4,2)= ',F8.3)
    RETURN
  END

```

```

*****
* THIS SUBROUTINE RENUMBERS THE INITIAL PIECE SO THAT *
* THE Y VALUES ARE IN ASCENDING ORDER BEGINNING *
* AT THE TRAILING EDGE. *
*****

```

```

      SUBROUTINE REVERSE(L,R,LPTS,RPTS)
      INTEGER LPTS,RPTS
      REAL L(2750,2),R(2750,2)
      DO 10 I1=1,LPTS
        L(I1,2)=ABS(L(I1,2)-22.0)
  10 CONTINUE
      DO 20 I2=1,RPTS
        R(I2,2)=ABS(R(I2,2)-22.0)
  20 CONTINUE
      RETURN
  END

```

```

*****
* THIS SUBROUTINE FILLS GAPS BETWEEN DIGITIZED POINTS *
* AT 0.01 UNIT INCREMENTS *
*****

```

```

      SUBROUTINE PTSFILL(TS,TP)
      INTEGER TP,PTS,PTR,K
      REAL TS(2750,2),T(2750,2),XDIS,YDIS,M
      COMMON T
      PTR=0
      PTS=1
  10 PTR=PTR+1
      K=1
      IF(PTS.GT.2750)GOTO 70

```

```
IF(PTR.EQ.TP)GO TO 80
T(PTS,1)=TS(PTR,1)
T(PTS,2)=TS(PTR,2)
PTS=PTS+1
XDIS=TS(PTR+1,1)-TS(PTR,1)
YDIS=TS(PTR+1,2)-TS(PTR,2)
20 IF(YDIS.EQ.0.)GO TO 10
IF((ABS(XDIS).LE.0.01).AND.(ABS(YDIS).GT.0.01))THEN
30 IF(YDIS.LT.0.0)THEN
    T(PTS,1)=TS(PTR,1)
    T(PTS,2)=TS(PTR,2)-K*0.01
ELSE
    T(PTS,1)=TS(PTR,1)
    T(PTS,2)=TS(PTR,2)+K*0.01
ENDIF
IF((K*0.01).LT.(ABS(YDIS)-0.01))THEN
    PTS=PTS+1
    IF(PTS.GT.2750)GOTO 70
    K=K+1
    GOTO 30
ELSE
    PTS=PTS+1
    GOTO 10
ENDIF
ELSEIF((ABS(YDIS).LE.0.01).AND.(ABS(XDIS).GT.0.01))THEN
40 IF(XDIS.LT.0.0)THEN
    T(PTS,1)=TS(PTR,1)-K*0.01
    T(PTS,2)=TS(PTR,2)
ELSE
    T(PTS,1)=TS(PTR,1)+K*0.01
    T(PTS,2)=TS(PTR,2)
ENDIF
IF((K*0.01).LT.(ABS(XDIS)-0.01))THEN
    K=K+1
    PTS=PTS+1
    IF(PTS.GT.2750)GOTO 70
    GOTO 40
ELSE
    PTS=PTS+1
    GOTO 10
ENDIF
ELSEIF((ABS(YDIS).GT.0.01).AND.(ABS(XDIS).GT.0.01))THEN
    M=ABS(YDIS/XDIS)
    IF((XDIS.LT.0.0).AND.(YDIS.LT.0.0))THEN
        T(PTS,1)=TS(PTR,1)-ABS(0.01/M)
        T(PTS,2)=TS(PTR,2)-0.01
        IF(T(PTS,1).GT.(TS(PTR+1,1)+0.01))GO TO 50
        IF(T(PTS,2).GT.(TS(PTR+1,2)+0.01))GO TO 50
        GOTO 60
    ELSEIF((XDIS.LT.0.0).AND.(YDIS.GT.0.0))THEN
        T(PTS,1)=TS(PTR,1)-ABS(0.01/M)
```

```

      T(PTS,2)=TS(PTR,2)+0.01
      IF(T(PTS,1).GT.(TS(PTR+1,1)+0.01))GO TO 50
      IF(T(PTS,2).LT.(TS(PTR+1,2)-0.01))GO TO 50
      GOTO 60
    ELSEIF((XDIS.GT.0.0).AND.(YDIS.LT.0.0))THEN
      T(PTS,1)=TS(PTR,1)+ABS(0.01/M)
      T(PTS,2)=TS(PTR,2)-0.01
      IF(T(PTS,1).LT.(TS(PTR+1,1)-0.01))GO TO 50
      IF(T(PTS,2).GT.(TS(PTR+1,2)+0.01))GO TO 50
      GOTO 60
    ELSEIF((XDIS.GT.0.0).AND.(YDIS.GT.0.0))THEN
      T(PTS,1)=TS(PTR,1)+ABS(0.01/M)
      T(PTS,2)=TS(PTR,2)+0.01
      IF(T(PTS,1).LT.(TS(PTR+1,1)-0.01))GO TO 50
      IF(T(PTS,2).LT.(TS(PTR+1,2)-0.01))GO TO 50
      GOTO 60
    ENDIF
50  XDIS=TS(PTR+1,1)-T(PTS,1)
    YDIS=TS(PTR+1,2)-T(PTS,2)
    TS(PTR,1)=T(PTS,1)
    TS(PTR,2)=T(PTS,2)
    PTS=PTS+1
    IF(PTS.GT.2750)GOTO 70
    GOTO 20
60  PTS=PTS+1
    ENDIF
    GOTO 10
70  WRITE(25,71)PTS,TP
71  FORMAT('ARRAY OVERFLOW IN PTSFILL. PTS= ',I5,' ',
    *'SIDE POINTS= ',I5)
    STOP
80  T(PTS,1)=TS(PTR,1)
    T(PTS,2)=TS(PTR,2)
    DO 90 I9=1,PTS
      TS(I9,1)=T(I9,1)
      TS(I9,2)=T(I9,2)
90  CONTINUE
    TP=PTS
    RETURN
    END

```

```

*****
*  THIS SUBROUTINE ADJUSTS TRAILING SHEET POINTS TO  *
*  TO FORM A CONTINUOUS RIBBON BEHIND THE INITIAL  *
*  PIECE                                           *
*****

```

```

      SUBROUTINE CONNECT(L,R,LPTS,RPTS,YDIS,YSTRT,XC)
      INTEGER LPTS,RPTS
      REAL YDIS,YSTRT,XC,D,L(2750,2),R(2750,2)
      D=0.0
      DO 10 I1=1,LPTS
        D=YDIS-L(I1,2)

```

```

      L(I1,2)=YSTRT+D
      L(I1,1)=L(I1,1)+XC
10  CONTINUE
      DO 20 I2=1,RPTS
          D=YDIS-R(I2,2)
          R(I2,2)=YSTRT+D
          R(I2,1)=R(I2,1)+XC
20  CONTINUE
      RETURN
      END

```

```

*****
*   THIS SUBROUTINE LOCATES THE MINIMUM AND MAXIMUM "Y"   *
*   VALUES OF SHAPE AND SHAPE MARGIN                       *
*****

```

```

      SUBROUTINE MINMAX(S,SPTS,SUM)
      INTEGER SPTS,SUM(2),LYMIN,LYMAX
      REAL YMIN,YMAX,S(2750,2)
      YMIN=S(1,2)
      YMAX=S(1,2)
      LYMIN=1
      LYMAX=1
      DO 10 I=2,SPTS
          IF(S(I,2).LT.YMIN)THEN
              LYMIN=I
              YMIN=S(I,2)
          ENDIF
          IF(S(I,2).GT.YMAX)THEN
              LYMAX=I
              YMAX=S(I,2)
          ENDIF
10  CONTINUE
      SUM(1)=LYMIN
      SUM(2)=LYMAX
      RETURN
      END

```

```

*****
*   THIS SUBROUTINE CORRECTS PIECES FOR SKEWED CLIPS.     *
*****

```

```

      SUBROUTINE PCSKEW(L,R,LPTS,RPTS,LFACT,RFACT,LSKEW,RSKEW,
      *LXCHNG,RXCHNG,DIS,SKEW)
      INTEGER LPTS,RPTS
      REAL LFACT,RFACT,LSKEW,RSKEW,LXCHNG,L(2750,2),R(2750,2),
      *RXCHNG,XADJ,DIS,LSY,RSY,YDIS,RSIZE,
      *LSIZE,SFACT,SKEW(4,2)

```

```

C
C   FORCE CLIP LINES TO PARALLEL Y VALUES.
C

```

```

      SFACT=(L(1,2)+R(1,2))* .5
      LSIZE=L(1,2)-L(LPTS,2)
      RSIZE=R(1,2)-R(RPTS,2)
      YDIS=(LSIZE+RSIZE)* .5

```

```

    LFACT=YDIS/LSIZE
    RFACT=YDIS/Rsize
    LSY=L(1,2)*LFACT
    RSY=R(1,2)*RFACT
    LSKEW=SFACT-LSY
    RSKEW=SFACT-RSY
    IF(LFACT.NE.0.0)THEN
      DO 10 I1=1,LPTS
        L(I1,2)=L(I1,2)*LFACT+LSKEW
10    CONTINUE
      ENDIF
    IF(RFACT.NE.0.0)THEN
      DO 20 I2=1,RPTS
        R(I2,2)=R(I2,2)*RFACT+RSKEW
20    CONTINUE
      ENDIF
C
C  FORCE SHAPE TO A RECTANGLE.
C
    DIS=L(1,2)-L(LPTS,2)
    LXCHNG=SKEW(3,1)-SKEW(1,1)
    RXCHNG=SKEW(4,1)-SKEW(2,1)
    WRITE(25,*)' LXCHNG= ',LXCHNG,' RXCHNG= ',RXCHNG,' DIS= ',
    *DIS
    IF(LXCHNG.NE.0.0)THEN
      IF(ABS(LXCHNG).LE..5)THEN
        DO 30 I3=1,LPTS
          XADJ=LXCHNG+(-LXCHNG*((L(I3,2)-L(LPTS,2))/DIS))
          L(I3,1)=L(I3,1)+XADJ
30    CONTINUE
        ENDIF
      ENDIF
    IF(RXCHNG.NE.0.0)THEN
      IF(ABS(RXCHNG).LE..5)THEN
        DO 40 I4=1,RPTS
          XADJ=RXCHNG+(-RXCHNG*((R(I4,2)-R(RPTS,2))/DIS))
          R(I4,1)=R(I4,1)+XADJ
40    CONTINUE
        ENDIF
      ENDIF
    RETURN
  END
*****
*  THIS SUBPROGRAM LOCATES THE MINIMUM AND MAXIMUM X      *
*  OF DEFECTS AND THEIR SURROUNDING WANE                  *
*****
  SUBROUTINE DMINMAX(L,R,LPTS,RPTS,XMIN,XMAX)
    INTEGER LPTS,RPTS
    REAL XMIN,XMAX,L(2750,2),R(2750,2)
    XMIN=L(1,1)
    XMAX=R(1,1)

```

```

      DO 10 I1=2,LPTS
        IF(L(I1,1).LT.XMIN)THEN
          XMIN=L(I1,1)
        ENDIF
      10 CONTINUE
      DO 20 I2=2,RPTS
        IF(R(I2,1).GT.XMAX)THEN
          XMAX=R(I2,1)
        ENDIF
      20 CONTINUE
      RETURN
      END
*****
* THIS SUBROUTINE ADJUSTS DEFECT POINTS FOR ANY TYPE *
* OF CORRECTION REQUIRED OF SHAPE BY PCSKEW *
*****
      SUBROUTINE DEFSKEW(L,R,LPTS,RPTS,LFACT,RFACT,LSKEW,RSKEW,
      *LXCHNG,RXCHNG,DIS,LBY,RBY,SFLAG)
      INTEGER LPTS,RPTS,SFLAG
      REAL LFACT,RFACT,LSKEW,RSKEW,LXCHNG,L(2750,2),R(2750,2),
      *RXCHNG,DIS,XADJ,LBY,RBY
C
C ADJUST POINTS TO MOVEMENT OF CLIP LINES TO PARALLEL Y.
C
      IF(SFLAG.EQ.0)THEN
        IF(RFACT.NE.0.0)THEN
          DO 10 I1=1,LPTS
            L(I1,2)=L(I1,2)*RFACT+RSKEW
          10 CONTINUE
          DO 20 I2=1,RPTS
            R(I2,2)=R(I2,2)*RFACT+RSKEW
          20 CONTINUE
        ENDIF
        ELSEIF(LFACT.NE.0.0)THEN
          DO 30 I3=1,LPTS
            L(I3,2)=L(I3,2)*LFACT+LSKEW
          30 CONTINUE
          DO 40 I4=1,RPTS
            R(I4,2)=R(I4,2)*LFACT+LSKEW
          40 CONTINUE
        ENDIF
C
C ADJUST POINTS TO MOVEMENT OF SHAPE TO RECTANGLE.
C
      IF(SFLAG.EQ.0)THEN
        IF(RXCHNG.NE.0.0)THEN
          IF(ABS(RXCHNG).LE..5)THEN
            DO 50 I5=1,LPTS
              XADJ=RXCHNG+(-RXCHNG*((L(I5,2)-LBY)/DIS))
              L(I5,1)=L(I5,1)+XADJ
            50 CONTINUE

```

```

        DO 60 I6=1,RPTS
            XADJ=RXCHNG+(-RXCHNG*((R(I6,2)-RBY)/DIS))
            R(I6,1)=R(I6,1)+XADJ
60      CONTINUE
        ENDIF
    ENDIF
ELSEIF(LXCHNG.NE.0.0)THEN
    IF(ABS(LXCHNG).LE..5)THEN
        DO 70 I7=1,LPTS
            XADJ=LXCHNG+(-LXCHNG*((L(I7,2)-LBY)/DIS))
            L(I7,1)=L(I7,1)+XADJ
70      CONTINUE
        DO 80 I8=1,RPTS
            XADJ=LXCHNG+(-LXCHNG*((R(I8,2)-RBY)/DIS))
            R(I8,1)=R(I8,1)+XADJ
80      CONTINUE
        ENDIF
    ENDIF
RETURN
END

```

```

*****
*  THIS SUBROUTINE STRETCHES THE WITH GRAIN DISTANCE      *
*  FULL SHEET LIMIT IF NECESSARY                          *
*****

```

```

SUBROUTINE STRETCH(FULLDIF,L,LPTS,R,RPTS)
INTEGER LPTS,RPTS
REAL FULLDIF,L(2750,2),R(2750,2)

```

C
C

```

        DO 10 I1=1,LPTS
            L(I1,1)=L(I1,1)-FULLDIF*.5
10      CONTINUE
        DO 20 I2=1,RPTS
            R(I2,1)=R(I2,1)+FULLDIF*.5
20      CONTINUE
RETURN
END

```

APPENDIX C

Plotting Computer Program


```
*****
* THIS PROGRAM UTILIZES RECONSTRUCTED DATA FILES TO *
* GENERATE A PLOT OF THE ENTIRE BLOCK PEEL OF VENEER *
* PROGRAMMER: JIM FUNCK *
*****
```

LIST OF VARIABLES:

GETPARM = CYBER SYSTEM SUBROUTINE

ID,DID = DEFECT CODE

IPEN = PLOT PEN CONTROL

KYWRD,KYVAL = CYBER CONTROL LANGUAGE PARAMETERS USED

FOR FILE ACCESS WHEN SUCCESSIVE RUNS ARE MADE.

LABX(2) = PLOT TITLE ARRAY

LASTY = MAXIMUM Y VALUE OF CURRENT PIECE

ND,I,J,JT = CONTROL TRANSFER FLAGS

PLOTTYPE,PLOTLUN,SIZE,SCALE,VECTORS,COLOR,SYMBOL,AXISL =

SYSTEM COMMANDS

XC = X CORRECT TO CENTER PIECE ON PLOT

X,Y,XP,YP = X AND Y COORDINATES

PROGRAM VENPLT(INPUT,OUTPUT,TAPE13=OUTPUT)

REAL LASTY

INTEGER PKNT

CHARACTER*10 KYWRD,KYVAL

DIMENSION LABX(2)

OPEN(1,FILE='LSHAPT')

OPEN(2,FILE='RSHAPT')

OPEN(3,FILE='LSMART')

OPEN(4,FILE='RSMART')

OPEN(5,FILE='LDMART')

OPEN(6,FILE='RDMART')

OPEN(7,FILE='LDFECT')

OPEN(8,FILE='RDFECT')

CALL GETPARM(KYWRD,KYVAL)

C

XC=0.0

PKNT=0

C

C

C

CALL PLOTTYPE(0)

CALL PLOTLUN(13)

CALL SIZE(1500.0,34.0)

CALL SCALE(0.10,0.10,1.0,0.75,0.0,-30.0)

CALL VECTORS

5 I=0

J=1

JT=1

CALL COLOR(J)

C

C

7 IF(I.EQ.2)GO TO 15

I=I+1

```
      READ(I,*)IDC,IBN,IP
      IF(IDC.EQ.-999)GO TO 100
      IPEN=0
      READ(I,*)X,Y
      IF(PKNT.EQ.0)THEN
        IF(I.EQ.2)PKNT=1
        GOTO 11
      ELSEIF(I.EQ.2)THEN
        GOTO 11
      ELSE
        XC=(LASTY+10.0)-Y
        GOTO 11
      ENDIF
10  READ(I,*)X,Y
11  IF(X.EQ.-99)GOTO 7
      XP=Y+XC
      LASTY=XP
      YP=X+20
      CALL PLOT(XP,YP,IPEN,0)
      IPEN=1
      GO TO 10
15  J=J+1
      CALL COLOR(J)
20  IF(I.EQ.4)GO TO 27
      I=I+1
      READ(I,*)IDC,IBN,IP
      IPEN=0
25  READ(I,*)X,Y
      XP=Y+XC
      YP=X+20
      IF(X.EQ.-99.)GO TO 20
      CALL PLOT(XP,YP,IPEN,0)
      IPEN=1
      GO TO 25
27  J=3
      CALL COLOR(J)
      I=6
30  IF(I.EQ.8)GO TO 43
      I=I+1
35  READ(I,*)ID,MKNT
      IF(ID.EQ.-99)GO TO 30
      IF(ID.EQ.3)THEN
37    READ(I,*)X,Y
        IF(X.EQ.-88.)GO TO 35
        GO TO 37
      ENDIF
      IPEN=0
40  READ(I,*)X,Y
      XP=Y+XC
      YP=X+20
      IF(X.EQ.-88.)GO TO 35
```

```
      IF(X.EQ.-99.)GO TO 30
      CALL PLOT(XP,YP,IPEN,0)
      IPEN=1
      GO TO 40
43  I=4
45  IF(I.EQ.6)GO TO 5
      I=I+1
50  IPEN=0
      READ(I,*)ID,DKNT
      IF(ID.EQ.-99)GO TO 45
      IF(DKNT.GT.999)THEN
          DKNT=(DKNT-1000)+1
      ENDIF
      IF(ID.EQ.1.OR.ID.EQ.4)J=4
      IF(ID.EQ.2)J=2
      IF(ID.EQ.3)J=3
      IF(ID.EQ.5)J=4
      ND=1
      IF(J.EQ.JT)GO TO 55
      CALL COLOR(J)
      JT=J
55  READ(I,*)X,Y
      XP=Y+XC
      YP=X+20
      IF(X.EQ.-88.)GO TO 50
      IF(X.EQ.-99.)GO TO 45
      CALL PLOT(XP,YP,IPEN,0)
      IF(ND.EQ.1.AND.I.EQ.6)THEN
          DID=ID
          YPI=YP-0.90
          CALL NUMBER(XP,YPI,0.0,0.1,-1,DID)
          XPI=XP+1.2
          CALL PLOT(XPI,YPI,0,0)
          CALL NUMBER(XPI,YPI,0.0,0.1,-3,DKNT)
          CALL PLOT(XP,YP,0,0)
          ND=0
      ENDIF
      IPEN=1
      GO TO 55
100 CALL AXISL(0.0,1400.0,0.0,0.0,0.0,0.0,100.0,0.0,9,0,1,0,
      *1.0,1.0,0.20,0)
      ENCODE(20,200,LABX)KYVAL
200 FORMAT('BLOCK ',A2,' DIAMETER 1')
      CALL SYMBOL(120.0,10.0,0.0,0.5,20,LABX)
      CALL PLOTEND
      STOP
      END
```

APPENDIX D

Reclip Computer Program

```
*****
* THIS PROGRAM SIMULATES THE OPTICAL SCANNER/CLIPPER *
* LOGIC IN READING RECONSTRUCTED DATA FILES AND THEN *
* PRODUCES A RECLIPPED RIBBON OF VENEER ACCORDING TO *
* MANAGEMENT SELECTED LIMITS AND OPTIONS. *
* PROGRAMMER: MIKE BABB *
*****
```

LIST OF VARIABLES:

```
BEGDEF/ENDEF = BEGINNING AND ENDING DEFECT Y VALUE
BEND = END OF BLOCK FLAG
BN = BLOCK NUMBER
CDE = CLIP ACTION FLAG
CKNT = CLIP COUNTER
CF = ACROSS GRAIN FLAW SETTING
CL = CRACK LENGTH SETTING
CLP(500,3) = CLIP ARRAY
CW = CRACK WIDTH SETTING
D,DPTS,DKNT = DEFECT COUNTERS
DC = DIAMETER CLASS
DCLIP(20,9) = CLIPPABLE DEFECT ARRAY
DEND = END OF DEFECT FILE FLAG
DHIT = COUNTER ON NUMBER OF TIMES CLIP FALLS ON DEFECT WANE
DNXT(10) = NEXT CLIPPABLE DEFECT ARRAY
EC = EDGE FLAW ACROSS GRAIN SETTING
EHIT = COUNTER ON NUMBER OF TIMES CLIPS FALLS ON PIECE WANE
EW = EDGE FLAW WITH GRAIN SETTING
FC = FULL SHEET ACROSS GRAIN SETTING
FT = FISHTAIL WITH GRAIN SETTING
FULL = FULL SHEET CLIP COUNTER
FULL2,HALF2,RAND2,LFISH2,RFISH2,DFISH2 = PIECE SQUARE FEET
FW = FULL RIBBON WITH GRAIN SETTING
HALF = HALF SHEET CLIP COUNTER
I = DEPARTING DEFECT FROM CLIP CONSIDERATION
INL = INITIAL POTENTIAL PIECE FLAG
INLDIS = INITIAL PIECE Y DISTANCE
ITER = ITERATION COUNT
JUMP = READ ACTION FLAG
LBND,RBND = LEFT AND RIGHT SIDE PIECE X COORDINATES
LDC(20,2),RDC(20,2) = LEFT/RIGHT CLIPPABLE DEFECT ARRAY
LFSHP,RTSHP,LFMAR,RTMAR = INPUT DATA FILES
LFT,RFT = CURRENT LEFT AND RIGHT SIDE PIECE X VALUES
LHDIS = LOOK AHEAD PIECE Y DISTANCE
LH(3) = LOOK AHEAD PIECE ARRAY
LHLBND,LHRBND = LOOK AHEAD LEFT AND RIGHT SIDE X VALUES
LMAR,RMAR = LEFT AND RIGHT SIDE EDGE CONDITION FLAGS
LMX,LMY,RMX,RMY = LEFT AND RIGHT SIDE WANE COORDINATES
LM(10,3),RM(10,3) = LEFT AND RIGHT SIDE WANE ARRAY
LX,LY,RX,RY = LEFT AND RIGHT SIDE SHAPE ARRAY
M = DUMMY MARGIN
MDC,MBN,MPCS = WANE FILE DIAMETER CLASS, BLOCK NUMBER, AND
PIECE
MG = MARGIN SETTING
MMR(30,3) = SUMMARY LEFT AND RIGHT SIDE WANE COORDINATES ARRAY
```

```

MROW = RECOV ARRAY POINTER
MS = MINIMUM STRIP ACROSS GRAIN SETTING
ND = NEXT DEFECT FLAG
PANEL = PANEL/RANDOM OPTION FLAG
PASS = INITIAL MARGIN ADJUSTMENT FLAG
PCLIP = POTENTIAL CLIP FLAG
PCS = PIECE NUMBER
PFLAG = CURRENT POTENTIAL PIECE FLAG
Q,R = ECONOMIC DECISION VARIABLES
RECOV(5,3) = POTENTIAL PIECE TRACKING ARRAY
RD1,RB2,RP3 = TEST VARIABLES FOR DIAMETER CLASS, BLOCK NUMBER,
AND PIECE
RD3 = ALTERNATE MARGIN CONTROL FLAG
RM1,RM2,RM3 = TEST VARIABLES FOR DIAMETER CLASS, BLOCK NUMBER,
AND PIECE
SC = HALF SHEET ACROSS GRAIN SETTING
SCAN = PREVIOUS Y SCANNER LINE
SEND = END OF RIBBON SECTION FLAG
SW = BLOCK SCORE WIDTH SETTING
SWEEP = CURRENT Y SCANNER LINE
TCLIP(20,9) = TEMPORARY CLIPPABLE DEFECT ARRAY
TOT = TOTAL SQUARE FEET
VEN(3,4) = VENEER VALUE ARRAY
WF = WITH GRAIN FLAW SETTING
PROGRAM VENCLIP(INPUT,OUTPUT)
REAL LX,LY,RX,RY,SWEEP,SCAN,LM(10,3),RM(10,3),LBND,LMX,
*TCLIP(20,9),DCLIP(20,9),LDC(20,2),RDC(20,2),DFISH2,LMY,
*DNXT(10),RECOV(5,3),INL,MMR(30,3),RAND2,HALF2,RBND,RMX,
*VEN(3,4),LH(3),CLP(500,4),FW,FT,FC,SC,SW,CL,CW,FULL2,M,
*CF,WF,EW,EC,MS,LHDIS,INLDIS,R,Q,LFT,RFT,PFLAG,MG,TOT,
*RMY,BEGDEF,ENDEF,LFISH2,RFISH2,LHLBND,LHRBND
INTEGER SEND,LMAR,RMAR,LMR,DKNT,I,RMR,D,PCLIP,ITER,STPCS,
*DPTS,FIN,LHEAD,DHIT,DEND,EHIT,HALF,CDE,JUMP,FULL,RB2,RP3,
*ND,MROW,SBLK,PANEL,CKNT,LFSHP,DC,MR,MDC,MBN,MPCS,BEND,PASS,
*RTSHP,LFMAR,RTMAR,BN,PCS,RD1,RDKNT,LFISH,RFISH,DFISH,RAND,
*FULLKNT,RM1,RM2,RM3,P
COMMON/SUM/EC,EW,CF,WF,SW,FW,ND,DNXT,SWEEP,PCS,MG,DEND
DATA LFISH,RFISH,DFISH,RAND,HALF,FULL,DEND,BEND/8*0/
DATA LFISH2,RFISH2,DFISH2,RAND2,FULL2,HALF2,TOT/7*0.0/
OPEN(1,FILE='LSHAP15')
OPEN(2,FILE='RSHAP15')
OPEN(3,FILE='LSMAR15')
OPEN(4,FILE='RSMAR15')
OPEN(6,FILE='SDSUM15')
OPEN(7,FILE='CLIPS15')
OPEN(8,FILE='CPDAT15')
OPEN(9,FILE='PARAMIP')
ITER=0
STPCS=0
DKNT=1
LFSHP=1
RTSHP=2
LFMAR=3

```

```
      RTMAR=4
C
C  READ IN CLIP COMMAND VARIABLES.
C
10  READ(9,*)FW,FT,FC,SC,MS,MG,CL,CW,WF,CF
      READ(9,*)EW,BS,SW,PANEL,MROW,VEN(1,1),VEN(1,2),
      *VEN(1,3),VEN(1,4),VEN(2,1)
      READ(9,*)VEN(2,2),VEN(2,3),VEN(2,4),VEN(3,1),VEN(3,2),
      *VEN(3,3),VEN(3,4)
      EC=BS+WF
      D=0
      ND=0
      SEND=0
      RDKNT=3
      LMAR=0
      FIN=0
      DPTS=0
      RMAR=0
      LMR=0
      RMR=0
      LH(1)=0.0
      LH(2)=0.0
      LH(3)=0.0
      LHEAD=0
      CKNT=0
      DHIT=0
      EHIT=0
      MR=0
      FULLKNT=0
      JUMP=0
      SBLK=0
      LBND=0.0
      RBND=300.0
      LFT=0.0
      RFT=0.0
      I=0
      P=0
      DO 20 I2=1,500
        DO 20 I3=1,4
          CLP(I2,I3)=0.0
20  CONTINUE
      DO 30 I3=1,5
        DO 30 I2=1,3
          RECOV(I3,I2)=0.0
30  CONTINUE
      DO 40 I4=1,20
        DO 40 I5=1,2
          LDC(I4,I5)=0.0
          RDC(I4,I5)=0.0
40  CONTINUE
      DO 50 I5=1,10
```

```

        DNXT(I5)=0.0
50 CONTINUE
        DO 60 I6=1,20
            DO 60 I7=1,7
                DCLIP(I6,I7)=0.0
                TCLIP(I6,I7)=0.0
60 CONTINUE
        IF(ITER.NE.0)GOTO 110
C
C READ SHAPE HEADER, TEST FOR END OF BLOCK/FILE.
C
        READ(LFSHP,*)DC,BN,PCS
        WRITE(7,*)'BLOCK NUMBER: ',BN
        WRITE(7,*)'FULL WIDTH= ',FC,' MIN STRIP= ',MS,
        *' MARGIN= ',MG
        IF(PANEL.EQ.0)THEN
            WRITE(7,*)'PANEL = NO'
        ELSE
            WRITE(7,*)'PANEL = YES'
        ENDIF
        IF(MROW.EQ.4)THEN
            WRITE(7,*)'DOUBLE FISHTAIL = NO'
        ELSE
            WRITE(7,*)'DOUBLE FISHTAIL = YES'
        ENDIF
        WRITE(7,*)
        WRITE(7,*)
        READ(RTSHP,*)RD1,RB2,RP3
        READ(LFMAR,*)MDC,MBN,MPCS
        READ(RTMAR,*)RM1,RM2,RM3
        IF(((DC+RD1+MDC+RM1)/4).NE.DC)THEN
            WRITE(8,*)'DIAMETER CLASSES DO NOT MATCH.'
            WRITE(8,105)DC,RD1,MDC,RM1
105  FORMAT(I3,2X,F3.0,2X,I3,2X,I3)
            STOP
        ENDIF
        IF(((BN+RB2+MBN+RM2)/4).NE.BN)THEN
            WRITE(8,*)'BLOCK NUMBERS DO NOT MATCH.'
            WRITE(8,105)BN,RB2,MBN,RM2
            STOP
        ENDIF
        IF(((PCS+RP3+MPCS+RM3)/4).NE.PCS)THEN
            WRITE(8,*)'PIECE NUMBERS DO NOT MATCH.'
            WRITE(8,105)PCS,RP3,MPCS,RM3
            STOP
        ENDIF
110 READ(LFMAR,*)LMX,LMY
        READ(RTMAR,*)RMX,RMY
        READ(LFSHP,*)LX,LY
        READ(RTSHP,*)RX,RY
        IF(LY.LT.RY)THEN

```



```
        SCAN=LY
      ELSE
        SCAN=RY
      ENDIF
      SWEEP=SCAN
      RECOV(4,1)=1.0
      RECOV(4,2)=SCAN
      INL=4.0
      PFLAG=4.0
      PASS= 1
C
C  UNLESS THEN RIBBON SECTION ENDS, EACH PASS THROUGH THE PROGRAM
C  RETURNS HERE, OTHERWISE, CONTROL TRANSFERS TO STATEMENT 10
C
270 CDE=0
      IF(PASS.EQ.0)THEN
        SCAN=SWEEP
        PFLAG=0.0
      ENDIF
      MR=0
      IF(RY.GT.LY)THEN
        SWEEP=RY
      ELSEIF(LY.GT.RY)THEN
        SWEEP=LY
      ELSE
        SWEEP=SWEEP+.10
      ENDIF
      EHIT=0
      LMR=0
      RMR=0
C
C  INCREMENT SWEEP.
C
      CALL READS(LX,LY,RX,RY,LFSHP,RTSHP,FIN,SWEEP)
      IF(FIN.NE.0)THEN
        CALL REND(FIN,DC,BN,PCS,BEND,SWEEP,MG)
        SEND=1
        PFLAG=INL
        GOTO 520
      ELSE
        CALL READS(LMX,LMY,RMX,RMY,LFMAR,RTMAR,FIN,SWEEP)
      ENDIF
      IF(FIN.NE.0)THEN
        CALL REND(FIN,DC,BN,PCS,BEND,SWEEP,MG)
        SEND=1
        PFLAG=INL
        GOTO 520
      ENDIF
C
C  SUBROUTINE "NSTOP" DETERMINES THE NEXT REQUIRED ACTION BASED ON
C  THE START OF A NEW DEFECT, THE END OF AN EXISTING DEFECT, OR NO
```

```

C  DEFECT.
C
C  JUMP: -1 = DROP A DEFECT
C          0 = NO DEFECTS
C          1 = ADD A DEFECT
C
C  SUBPROGRAM STDSUM READS THE DEFECT SUMMARY FILE AND IDENTIFIES
C  CLIPPABLE DEFECTS WITHIN THE CURRENT RIBBON SECTION.  DEFECTS
C  ARE CODED AND FLAGGED FOR APPROPRIATE ACTION.
C
      IF(DKNT.EQ.PCS)THEN
        IF(DEND.EQ.O)THEN
          CALL STDSUM(D,DKNT,LX,RX,CL,CW,TCLIP)
        ENDIF
      ENDIF
      IF(JUMP.EQ.O)THEN
        IF((DPTS+D).NE.O)THEN
          CALL NSTOP(TCLIP,D,DCLIP,DPTS,JUMP,I,ENDEF,BEGDEF,
*          ND,DNXT,MG)
        ENDIF
      ENDIF
      IF(JUMP.EQ.1)THEN
        IF(BEGDEF.LE.SWEEP)THEN
          CALL DADD(LDC,RDC,LX,RX,DPTS,DCLIP,TCLIP,D,JUMP)
        ENDIF
      ELSEIF(JUMP.EQ.-1)THEN
        IF(ENDEF.LE.SWEEP)THEN
          CALL DDROP(DCLIP,DPTS,I,LDC,RDC,JUMP)
        ENDIF
      ENDIF
      IF(ABS(LMX-LX).LE..10)THEN
        LMAR=0
      ELSE
        LMAR=1
      ENDIF
      IF(ABS(RMX-RX).LE..10)THEN
        RMAR=0
      ELSE
        RMAR=1
      ENDIF
C
C  SUBPROGRAM WTHGRN MEASURES THE WITH GRAIN DISTANCES
C  BETWEEN DEFECTS LYING ON THE SAME Y PLANE.
C  A POTENTIAL RECOVERY FLAG IS ASSIGNED.
C
      IF(DPTS.NE.O)THEN
        CALL WTHGRN(LMAR,RMAR,LDC,RDC,PFLAG,LFT,RFT,LX,
*        RX,FT,MROW,DPTS,LMX,RMX,LMR,RMR,SWEEP,MG,LM,RM)
        GOTO 450
      ENDIF
      IF(LMAR.EQ.O)THEN

```

```

      LFT=LX
    ELSE
      LFT=LX+MG
    ENDIF
    IF(RMAR.EQ.0)THEN
      RFT=RX
    ELSE
      RFT=RX-MG
    ENDIF

```

```

C
C PFLAG CODES REFLECT THE CURRENT PIECE TYPE.
C

```

```

C PFLAG:  1 = FULL WIDTH
C          2 = LEFT FISHTAIL
C          3 = RIGHT FISHTAIL
C          4 = TRASH
C          5 = DOUBLE FISHTAIL
C

```

```

      IF((RX-LX).LT.FT)THEN
        PFLAG=4.0
      ELSEIF((LMAR+RMAR).EQ.0)THEN
        IF((RX-LX).GE.FW)THEN
          IF(ABS(LX-LFT).LE..10)THEN
            IF(ABS(RX-RFT).LE..10)THEN

```

```

PFLAG=1.0

```

```

      ELSE

```

```

PFLAG=2.0

```

```

      ENDIF
      ELSEIF(ABS(RX-RFT).LE..10)THEN
        PFLAG=3.0
      ELSEIF(MROW.EQ.5)THEN
        PFLAG=5.0
      ELSE
        PFLAG=4.0
      ENDIF
      ELSEIF(ABS(LX-LFT).LE..10)THEN
        PFLAG=2.0
      ELSEIF(ABS(RX-RFT).LE..10)THEN
        PFLAG=3.0
      ELSEIF(MROW.EQ.5)THEN
        PFLAG=5.0
      ELSE
        PFLAG=4.0
      ENDIF
      ELSEIF((LMAR+RMAR).EQ.2)THEN
        IF(MROW.EQ.5)THEN
          IF(((RX-MG)-(LX+MG)).GE.FT)THEN
            PFLAG=5.0
          ELSE
            PFLAG=4.0
          ENDIF

```

```
      ELSE
        PFLAG=4.0
      ENDIF
    ELSEIF(LMAR.EQ.0)THEN
      IF(ABS(LX-LFT).LE..10)THEN
        IF(((RX-MG)-LX).GE.FT)THEN
          PFLAG=2.0
        ELSE
          PFLAG=4.0
        ENDIF
      ELSEIF(MROW.EQ.5)THEN
        IF(((RX-MG)-LX).GE.FT)THEN
          PFLAG=5.0
        ELSE
          PFLAG=4.0
        ENDIF
      ELSE
        PFLAG=4.0
      ENDIF
    ELSEIF(ABS(RX-RFT).LE..10)THEN
      IF((RX-(LX+MG)).GE.FT)THEN
        PFLAG=3.0
      ELSE
        PFLAG=4.0
      ENDIF
    ELSEIF(MROW.EQ.5)THEN
      IF((RX-(LX+MG)).GE.FT)THEN
        PFLAG=5.0
      ELSE
        PFLAG=4.0
      ENDIF
    ELSE
      PFLAG=4.0
    ENDIF
450 IF((LMAR+RMAR).EQ.0.AND.(RX-LX).LT.FW)STPCS=1
    IF(PASS.NE.0)THEN
      IF(SWEEP.GE.(SCAN+MG))THEN
        PASS=0
        RECOV(4,3)=SWEEP
      ELSE
        GOTO 270
      ENDIF
    ENDIF
    IF(PFLAG.EQ.0.0)THEN
      CALL PMDLOAD
      STOP
    ENDIF
    IF(RECOV(PFLAG,1).EQ.1.0)THEN
      RECOV(PFLAG,3)=SWEEP
    ELSE
      RECOV(PFLAG,1)=1.0
```

```

        RECOV(PFLAG,2)=SWEEP
        RECOV(PFLAG,3)=SWEEP
    ENDIF
    IF(PFLAG.NE.4.0)THEN
        IF((RMX-LMX).LT.FT)EHIT=1
        MR=MR+1
        MMR(MR,1)=PFLAG
        MMR(MR,2)=SWEEP
        MMR(MR,3)=EHIT
    ENDIF
    IF(MROW.EQ.5)THEN
        IF(LFT.GT.LBND)THEN
            LBND=LFT
        ENDIF
        IF(RFT.LT.RBND)THEN
            RBND=RFT
        ENDIF
    ENDIF
C
C THE "LHEAD" FLAG IS SET IN THE EVENT A CLIP HAS THE POTENTIAL
C TO CAUSE THE FOLLOWING PIECE TO BE LOST BECAUSE OF THE LACK
C OF REQUIRED CROSS-GRAIN DISTANCE.
C
520 IF(SEND.NE.0)THEN
    CDE=1
ELSE
    CDE=0
ENDIF
C
C DETERMINATION OF POTENTIAL CLIPS BASED ON CROSS-GRAIN DISTANCE.
C SET LOOK AHEAD FLAG IF APPROPRIATE. MAKE CLIP ADJUSTMENTS FOR WANE.
C
    IF(INL.NE.PFLAG)THEN
        IF(DPTS.EQ.0)THEN
            M=MG
        ELSE
            M=0.0
        ENDIF
        CALL MGNADJ(RECOV,INL,PFLAG,SWEEP,CDE,MG,M)
    ENDIF
ENDIF
525 IF(LHEAD.NE.0)GOTO 630
    IF((SEND+CDE).NE.0)GOTO 550
    IF((RECOV(INL,3)-RECOV(INL,2)).GE.(2*FC+MS))THEN
        CDE=2
        GOTO 550
    ENDIF
    IF(MROW.EQ.5)THEN
        IF((RBND-LBND).LT.FT)THEN
            CDE=1
            GOTO 550

```

```

        ENDIF
    ENDIF
    IF(PFLAG.EQ.INL)GOTO 270
    LHEAD=1
    LH(1)=INL
    LH(2)=RECOV(INL,2)
    LH(3)=RECOV(INL,3)
    RECOV(INL,1)=0.0
    RECOV(INL,2)=0.0
    RECOV(INL,3)=0.0
    INL=PFLAG
    IF(MROW.EQ.5)THEN
        LHLBND=LBND
        LHRBND=RBND
        LBND=LFT
        RBND=RFT
    ENDIF
    GOTO 270

C
C  ENTERING STANDARD CLIP ROUTINE.
C
550 IF(INL.EQ.1.0)THEN
C
C  CLIP FULL-WIDTH PIECE(S).
C
        IF(RDKNT.EQ.3)THEN
            IF(FULLKNT.GT.1)THEN
                READ(9,*)MG
                RDKNT=0
            ENDIF
        ENDIF
        CALL FLCLIP(CLP,CKNT,FC,SC,MS,PANEL,CDE,RECOV(1,2),
        * RECOV(1,3),FULLKNT,PCS,SEND,BEND)
        ELSE
C
C  CLIP FISHTAIL(S) OR TRASH.
C
        CALL FTCLIP(CLP,CKNT,SC,MS,CDE,INL,RECOV(INL,2),
        * RECOV(INL,3),PCS,SEND,BEND)
        ENDIF
        IF(CDE.EQ.0)THEN
            CALL PMDLOAD
            STOP
        ENDIF
        IF(SEND.NE.0)GOTO 870
        IF(CDE.EQ.1)THEN
            RECOV(INL,1)=0.0
            RECOV(INL,2)=0.0
            RECOV(INL,3)=0.0
            INL=PFLAG
            IF(MROW.EQ.5)THEN

```

```
        LBND=LFT
        RBND=RFT
    ENDIF
    ELSE
        RECOV(INL,2)=CLP(CKNT,3)
    ENDIF
    GOTO 270
C
C ENTERING CLIP ROUTINE WITH "LOOK AHEAD" FLAG SET.
C
630 IF((SEND+CDE).NE.0)GOTO 635
    IF((RECOV(INL,3)-RECOV(INL,2)).GE.(2*FC+MS))THEN
        CDE=1
        GOTO 635
    ENDIF
    IF(INL.NE.PFLAG)GOTO 635
    GOTO 270
635 LHDIS=LH(3)-LH(2)
    INLDIS=RECOV(INL,3)-RECOV(INL,2)
    IF(LH(1).EQ.1.0)GOTO 810
    IF(INL.EQ.1.0)GOTO 720
    IF(MROW.EQ.5)GOTO 670
C
C FISHTAIL FOLLOWED BY A FISHTAIL.
C
640 CDE=1
    CALL FTCLIP(CLP,CKNT,SC,MS,CDE,LH(1),LH(2),
    *LH(3),PCS,SEND,BEND)
    IF(MROW.EQ.5)THEN
        LHLBND=LBND
        LHRBND=RBND
        LBND=LFT
        RBND=RFT
    ENDIF
    GOTO 840
C
C TRY TO RECOVER BOTH FISHTAIL TYPES; POSSIBLE
C ONLY WHEN DOUBLE FISHTAIL OPTION IS SELECTED.
C
670 IF((RBND-LHLBND).LT.FT)GOTO 640
    IF((LHRBND-LBND).LT.FT)GOTO 640
    IF(LHDIS.GE.MS)THEN
        IF(INLDIS.GE.MS)THEN
            GOTO 640
        ELSEIF(INL.EQ.5.0)THEN
            IF((INLDIS+LHDIS).GE.(2*MS))THEN
                RECOV(5,2)=RECOV(5,3)-MS
                LH(3)=RECOV(5,2)
                GOTO 640
            ENDIF
        ENDIF
    ENDIF
```

```

ELSEIF(INLDIS.GE.MS)THEN
  IF(LH(1).EQ.5.0)THEN
    IF((INLDIS+LHDIS).GE.(2*MS))THEN
      LH(3)=LH(2)+MS
      RECOV(INL,2)=LH(3)
      GOTO 640
    ENDIF
  ENDIF
ENDIF
IF(LHLBND.GT.LBND)THEN
  LBND=LHLBND
ENDIF
IF(LHRBND.LT.RBND)THEN
  RBND=LHRBND
ENDIF
RECOV(5,1)=1.0
RECOV(5,2)=LH(2)
RECOV(5,3)=RECOV(INL,3)
LHEAD=0
IF(INL.NE.5.0)THEN
  RECOV(INL,1)=0.0
  RECOV(INL,2)=0.0
  RECOV(INL,3)=0.0
ENDIF
INL=5.0
IF(PFLAG.EQ.4.0)GOTO 525
GOTO 270
C
C FISHTAIL FOLLOWED BY A FULL.
C
720 IF(INLDIS.LT.MS)THEN
  RECOV(LH(1),1)=1.0
  RECOV(LH(1),2)=LH(2)
  RECOV(LH(1),3)=RECOV(1,3)
  INL=LH(1)
  RECOV(1,1)=0.0
  RECOV(1,2)=0.0
  RECOV(1,3)=0.0
  LHEAD=0
  GOTO 525
ENDIF
IF(LHDIS.GE.MS)GOTO 640
IF(LHDIS.EQ.0.0)GOTO 640
IF(INLDIS.LE.SC)THEN
  R=INLDIS
ELSE
  Q=INLDIS/SC
  R=INLDIS-INT(Q)*SC
ENDIF
C
C DETERMINE IF ECONOMICALLY FEASIBLE TO TAKE SOME OF FULL

```


C SECTION WOOD AND ADD TO THE FISHTAIL TO ALLOW BOTH PIECES
C TO BE RECOVERED.

C

```
      CALL POTCLIP(R,PCLIP,FW,FT,MS,VEN,LHDIS)
      IF(PCLIP.NE.0)THEN
        RECOV(1,2)=RECOV(1,2)+R
        LH(3)=RECOV(1,2)
      ENDIF
      GOTO 640
```

C

C FULL FOLLOWED BY A FISHTAIL.

C

```
810 IF(LHDIS.LT.MS)THEN
      RECOV(INL,2)=LH(2)
      LHEAD=0
      RECOV(1,1)=0.0
      RECOV(1,2)=0.0
      RECOV(1,3)=0.0
      GOTO 525
    ELSEIF(INLDIS.EQ.0.0)THEN
      GOTO 820
    ELSEIF(INLDIS.LT.MS)THEN
      IF(LHDIS.GT.SC)THEN
        Q=LHDIS/SC
        R=LHDIS-INT(Q)*SC
      ELSE
        R=LHDIS
      ENDIF
      CALL POTCLIP(R,PCLIP,FW,FT,MS,VEN,INLDIS)
      IF(PCLIP.NE.0)THEN
        RECOV(INL,2)=RECOV(INL,2)-R
        LH(3)=RECOV(INL,2)
      ENDIF
    ENDIF
820 CDE=1
      IF(RDKNT.EQ.3)THEN
        IF(FULLKNT.GT.1)THEN
          READ(9,*)MG
          RDKNT=0
        ENDIF
      ENDIF
      CALL FLCLIP(CLP,CKNT,FC,SC,MS,PANEL,CDE,LH(2),LH(3),
        *FULLKNT,PCS,SEND,BEND)
      IF(MROW.EQ.5)THEN
        LHLBND=LBND
        LHRBND=RBND
        LBND=LFT
        RBND=RFT
      ENDIF
840 IF(SEND.NE.0)GOTO 870
      IF(PFLAG.EQ.INL)THEN
```

```
      LHEAD=0
    ELSE
      LH(1)=INL
      LH(2)=RECOV(INL,2)
      LH(3)=RECOV(INL,3)
      RECOV(INL,1)=0.0
      RECOV(INL,2)=0.0
      RECOV(INL,3)=0.0
      INL=PFLAG
    ENDIF
    GOTO 270
C
C  WRITE CLIPS
C
870  P=P+1
      IF(CKNT.EQ.0)THEN
        CKNT=CKNT+1
        CLP(CKNT,1)=4.0
        CLP(CKNT,2)=RECOV(4,1)
        CLP(CKNT,3)=SWEEP
        CLP(CKNT,4)=P
      ENDIF
      CALL CLPRITE(CLP,CKNT,MS,LFISH,RFISH,DFISH,RAND,
        *HALF,FULL,LFISH2,RFISH2,DFISH2,RAND2)
      MR=0
      WRITE(7,*)'SHAPE MARGIN HITS = ',EHIT
      WRITE(7,*)
      IF(BEND.NE.0)GOTO 1000
      REWIND 9
      ITER=1
      SBLK=SBLK+STPCS
      GOTO 10
1000  SBLK=SBLK+STPCS
      SBLK=SBLK/4
      IF(SBLK.GT.1)THEN
        SBLK=1
      ELSE
        SBLK=0
      ENDIF
      WRITE(7,1005)
1005  FORMAT(50('*'))/)
      WRITE(7,1006)
1006  FORMAT(T19,'CLIP SUMMARY'/)
      WRITE(7,1007)
1007  FORMAT(50('*'))/)
      WRITE(7,*)'FULL SHEETS = ',FULL
      FULL2=FULL*100.0*FC/144.0
      WRITE(7,1008)FULL2
1008  FORMAT(T10,'SQUARE FEET = ',F7.3,/)
      WRITE(7,*)'HALF SHEETS = ',HALF
      HALF2=HALF*100.0*SC/144.0
```

```

WRITE(7,1008)HALF2
WRITE(7,*)'RANDOMS = ',RAND
WRITE(7,1008)RAND2
WRITE(7,*)'LEFT FISHTAILS = ',LFISH
WRITE(7,1008)LFISH2
WRITE(7,*)'RIGHT FISHTAILS = ',RFISH
WRITE(7,1008)RFISH2
IF(MROW.EQ.5)THEN
  WRITE(7,*)'DOUBLE FISHTAILS = ',DFISH
  WRITE(7,1008)DFISH2
ENDIF
TOT=FULL2+HALF2+LFISH2+RFISH2+RAND2+DFISH2
WRITE(7,1009)TOT
1009 FORMAT(' TOTAL SQUARE FEET = ',F7.3,/)
END
*****
* THIS SUBROUTINE DETERMINES THE MOST VALUABLE CLIP *
* SEQUENCE IF THE PIECE DOES NOT MEET WIDTH TOLERANCE *
*****
SUBROUTINE POTCLIP(R,PCLIP,FW,FT,MS,VEN,DIS)
REAL VEN(3,4),RVAL,FVAL,VNR,DIF,RVAL1,FVAL1,
*R,FW,FT,MS,DIS,RFVAL
INTEGER PCLIP
PCLIP=0.0
RVAL=0.0
FVAL=0.0
RFVAL=0.0
RVAL1=0.0
FVAL1=0.0
RVAL=VEN(1,3)*R*FW/144*.001
FVAL=VEN(1,4)*(R+DIS)*FT/144*.001
IF(RVAL.LT.FVAL)THEN
  VNR=FVAL
ELSE
  VNR=RVAL
ENDIF
DIF=R-MS+DIS
IF((R-DIF).GE.MS)THEN
  RVAL1=VEN(1,3)*(R-DIF)*FW/144*.001
  FVAL1=VEN(1,4)*MS*FT/144*.001
  RFVAL=RVAL1+FVAL1
  IF(RFVAL.GT.VNR)VNR=RFVAL
ENDIF
IF(VNR.EQ.RVAL)GOTO 10
PCLIP=1
IF(VNR.EQ.FVAL)THEN
  R=R+DIS
ELSE
  R=R-DIF
ENDIF
10 CONTINUE

```

```

      RETURN
      END
*****
* THIS SUBROUTINE DROPS DEFECT FROM CONSIDERATION AS *
* THEY DEPART FROM THE CLIP DETERMINATION *
*****
      SUBROUTINE DDROP(DCLIP,DPTS,I,LDC,RDC,JUMP)
      REAL DCLIP(20,9),LDC(20,2),RDC(20,2),TEMP(20,9),TLDC(20,2),
      *TRDC(20,2)
      INTEGER I,DPTS,JUMP,PTS
      PTS=0
      DO 20 I1=1,DPTS
        IF(I1.NE.I)THEN
          PTS=PTS+1
          TLDC(PTS,1)=LDC(I1,1)
          TLDC(PTS,2)=LDC(I1,2)
          TRDC(PTS,1)=RDC(I1,1)
          TRDC(PTS,2)=RDC(I1,2)
          DO 10 I2=1,9
            TEMP(PTS,I2)=DCLIP(I1,I2)
10      CONTINUE
        ENDIF
20    CONTINUE
      DO 30 I3=1,PTS
        LDC(I3,1)=TLDC(I3,1)
        LDC(I3,2)=TLDC(I3,2)
        RDC(I3,1)=TRDC(I3,1)
        RDC(I3,2)=TRDC(I3,2)
        DO 30 I4=1,9
          DCLIP(I3,I4)=TEMP(I3,I4)
30    CONTINUE
      DPTS=PTS
      JUMP=0
      I=0
      RETURN
      END
*****
* THIS SUBROUTINE LOCATES APPLICABLE OPEN FLAWS FOR *
* CLIP CONSIDERATION AND ADDS THEM TO THE DEFECT TRACK *
*****
      SUBROUTINE STDSUM(D,DKNT,LX,RX,CL,CW,TCLIP)
      COMMON/SUM/EC,EW,CF,WF,SW,FW,ND,DNXT,SWEEP,PCS,MG,DEND
      REAL XN,YN,XX,YX,MINX,MAXX,CD,WD,LX,RX,EC,EW,CL,CW,
      *TCLIP(20,9),DNXT(10),CF,WF,SW,FW,SWEEP,MG,ID,P,N
      INTEGER C,D,ND,PCS,DKNT,DEND
      C=0
      D=0
10  IF(ND.EQ.0)THEN
      READ(6,*)P,XN,YN,XX,YX,MINX,MAXX,ID,N
      IF(P.EQ.-99.0)THEN
        DKNT=DKNT+1

```

```
      GOTO 30
    ELSEIF(P.EQ.-999.0)THEN
      DEND=1
      GOTO 30
    ELSEIF(P.EQ.-111)THEN
      GOTO 30
    ENDIF
  ELSE
    N=DNXT(1)
    ID=DNXT(2)
    XN=DNXT(4)
    YN=DNXT(5)
    XX=DNXT(6)
    YX=DNXT(7)
    MINX=DNXT(8)
    MAXX=DNXT(9)
    P=DNXT(10)
  ENDIF
  ND=0
  IF((YX+MG).LT.SWEEP)GOTO 10
  CD=YX-YN
  WD=MAXX-MINX
  IF(ID.EQ.3.0)THEN
    IF(CD.GT.CW.AND.WD.GT.CL)THEN
      C=1
    ENDIF
  ELSEIF(ID.EQ.2.0)THEN
    IF((RX-MAXX).LT.(SW-FW).OR.(MINX-LX).LT.(SW-FW))THEN
      IF(CD.GT.EC.AND.WD.GT.EW)THEN
        C=1
      ENDIF
    ELSEIF(CD.GT.CF.AND.WD.GT.WF)THEN
      C=1
    ENDIF
  ELSEIF(ID.GT.5.0)THEN
    WRITE(8,*)'DEFECT CODE ERROR IN STDSUM.'
    WRITE(8,*)'ID= ',ID,' NO= ',N,' PCS= ',PCS
    STOP
  ENDIF
  IF(C.EQ.0)GOTO 10
  IF((YN-MG).GT.SWEEP)THEN
    ND=1
    DNXT(1)=N
    DNXT(2)=ID
    DNXT(3)=0.0
    DNXT(4)=XN
    DNXT(5)=YN
    DNXT(6)=XX
    DNXT(7)=YX
    DNXT(8)=MINX
    DNXT(9)=MAXX
```

```

        DNXT(10)=PCS
    ELSE
        D=D+1
        TCLIP(D,1)=N
        TCLIP(D,2)=ID
        TCLIP(D,3)=0.0
        TCLIP(D,4)=XN
        TCLIP(D,5)=YN-MG
        TCLIP(D,6)=XX
        TCLIP(D,7)=YX+MG
        TCLIP(D,8)=MINX-MG
        TCLIP(D,9)=MAXX+MG
        C=0
        GOTO 10
    ENDIF
30 CONTINUE
    RETURN
END
*****
* THIS SUBROUTINE ADDS APPROPRIATE DEFECTS TO THE      *
* DEFECT TRACK FOR CLIP CONSIDERATION.                *
*****
        SUBROUTINE DADD(LDC,RDC,LX,RX,DPTS,DCLIP,TCLIP,D,JUMP)
        REAL LDC(20,2),RDC(20,2),DCLIP(20,9),TLDC(20,2),TRDC(20,2),
        *TCLIP(20,9),LX,RX,TEMP(20,9)
        INTEGER PTS,PTR,DPTS,D,KNT,JUMP
        PTS=0
        KNT=1
        PTR=0
        IF(DPTS.EQ.0)THEN
            DO 10 I1=1,D
                LDC(I1,1)=TCLIP(I1,1)
                IF((TCLIP(I1,8)).LT.LX)THEN
                    LDC(I1,2)=LX
                ELSE
                    LDC(I1,2)=TCLIP(I1,8)
                ENDIF
                RDC(I1,1)=TCLIP(I1,1)
                IF((TCLIP(I1,9)).GT.RX)THEN
                    RDC(I1,2)=RX
                ELSE
                    RDC(I1,2)=TCLIP(I1,9)
                ENDIF
10 CONTINUE
                DO 20 I5=1,D
                    DO 20 I6=1,9
                        DCLIP(I5,I6)=TCLIP(I5,I6)
20 CONTINUE
                DPTS=D
                GOTO 110
            ENDIF

```

```
30 PTR=PTR+1
40 PTS=PTS+1
  IF((TCLIP(KNT,8)).LT.LDC(PTR,2))THEN
    TLDC(PTS,1)=TCLIP(KNT,1)
    TRDC(PTS,1)=TCLIP(KNT,1)
    IF((TCLIP(KNT,8)).LT.LX)THEN
      TLDC(PTS,2)=LX
    ELSE
      TLDC(PTS,2)=TCLIP(KNT,8)
    ENDIF
    IF(TCLIP(KNT,9).GT.RX)THEN
      TRDC(PTS,2)=RX
    ELSE
      TRDC(PTS,2)=TCLIP(KNT,9)
    ENDIF
    DO 50 I5=1,9
      TEMP(PTS,I5)=TCLIP(KNT,I5)
50  CONTINUE
    IF(KNT.EQ.D)THEN
      DO 70 I6=PTR,DPTS
        PTS=PTS+1
        TLDC(PTS,1)=LDC(I6,1)
        TLDC(PTS,2)=LDC(I6,2)
        TRDC(PTS,1)=RDC(I6,1)
        TRDC(PTS,2)=RDC(I6,2)
        DO 60 I7=1,9
          TEMP(PTS,I7)=DCLIP(I6,I7)
60  CONTINUE
70  CONTINUE
      ELSE
        KNT=KNT+1
        GOTO 40
      ENDIF
    ELSE
      TLDC(PTS,1)=LDC(PTR,1)
      TLDC(PTS,2)=LDC(PTR,2)
      TRDC(PTS,1)=RDC(PTR,1)
      TRDC(PTS,2)=RDC(PTR,2)
      DO 80 I8=1,9
        TEMP(PTS,I8)=DCLIP(PTR,I8)
80  CONTINUE
      IF(PTR.EQ.DPTS)THEN
        DO 90 I9=KNT,D
          PTS=PTS+1
          TLDC(PTS,1)=TCLIP(I9,1)
          TRDC(PTS,1)=TCLIP(I9,1)
          IF(TCLIP(I9,8).LT.LX)THEN
            TLDC(PTS,2)=LX
          ELSE
            TLDC(PTS,2)=TCLIP(I9,8)
          ENDIF
        ENDIF
      ENDIF
    ENDIF
  ENDIF
```

```

                IF(TCLIP(I9,9).GT.RX)THEN
                    TRDC(PTS,2)=RX
                ELSE
                    TRDC(PTS,2)=TCLIP(I9,9)
                ENDIF
                DO 85 I5=1,9
                    TEMP(PTS,I5)=TCLIP(I9,I5)
85             CONTINUE
90             CONTINUE
                ELSE
                    GOTO 30
                ENDIF
            ENDIF
        DO 100 I1=1,PTS
            LDC(I1,1)=TLDC(I1,1)
            LDC(I1,2)=TLDC(I1,2)
            RDC(I1,1)=TRDC(I1,1)
            RDC(I1,2)=TRDC(I1,2)
            DO 100 I2=1,9
                DCLIP(I1,I2)=TEMP(I1,I2)
100        CONTINUE
            DPTS=PTS
110        D=0
            JUMP=0
            RETURN
        END
*****
* THIS SUBROUTINE DETERMINES THE NEXT ACTION AT A SCAN *
*****
        SUBROUTINE NSTOP(TCLIP,D,DCLIP,DPTS,JUMP,I,ENDEF,BEGDEF,
        *ND,DNXT,MG)
        REAL TCLIP(20,9),DCLIP(20,9),BEGDEF,ENDEF,DNXT(10),MG
        INTEGER JUMP,I,D,DPTS,IB,ND
        I=0
        IB=0
        IF(D.NE.0)THEN
            BEGDEF=TCLIP(1,5)
            IB=1
            IF(D.GT.1)THEN
                DO 10 I1=2,D
                    IF(TCLIP(I1,5).LT.BEGDEF)THEN
                        BEGDEF=TCLIP(I1,5)
                        IB=I1
                    ENDIF
10             CONTINUE
            ENDIF
        ENDIF
        IF(DPTS.NE.0)THEN
            ENDEF=DCLIP(1,7)
            I=1
            IF(DPTS.GT.1)THEN

```



```

DO 20 I2=2,DPTS
  IF(DCLIP(I2,7).LT.ENDEF)THEN
    ENDEF=DCLIP(I2,7)
    I=I2
  ENDIF
20  CONTINUE

```

```

  ENDEF
ENDIF
IF(IB.NE.0)THEN
  IF(I.NE.0)THEN
    IF(BEGDEF.LT.ENDEF)THEN
      JUMP=1
    ELSE
      JUMP=-1
    ENDIF
  ELSE
    JUMP=1
  ENDIF
ELSEIF(I.NE.0)THEN
  IF(ND.GT.0)THEN
    IF((DNXT(5)-MG).LT.ENDEF)THEN
      JUMP=0
    ELSE
      JUMP=-1
    ENDIF
  ELSE
    JUMP=-1
  ENDIF
ELSE
  JUMP=0
ENDIF
RETURN
END

```

```

*****
* THIS SUBROUTINE DETERMINES WITH GRAIN DISTANCES *
* IF A DEFECT FALLS ON THE CURRENT SCAN LINE *
*****

```

```

SUBROUTINE WTHGRN(LMAR,RMAR,LDC,RDC,PFLAG,LFT,RFT,LX,RX,
*FT,MROW,DPTS,LMX,RMX,LMR,RMR,SWEEP,MG,LM,RM)
REAL LDC(20,2),RDC(20,2),LFT,RFT,LX,RX,FT,PFLAG,SWEEP,
*LMX,RMX,LM(10,3),RM(10,3),MG
INTEGER LMAR,RMAR,MROW,DPTS,LMR,RMR
IF(LMAR.EQ.0)THEN
  IF((LDC(1,2)-LX).GE.FT)THEN
    IF(ABS((LFT-LX)).LE..10)THEN
      PFLAG=2.0
    ELSEIF(MROW.EQ.5)THEN
      PFLAG=5.0
    ELSE
      PFLAG=4.0
    ENDIF
  ENDIF

```

```
LFT=LX
RFT=LDC(1,2)
IF(PFLAG.NE.4)THEN
  LMR=LMR+1
  LM(LMR,1)=PFLAG
  LM(LMR,2)=LMX
  LM(LMR,3)=SWEEP
ENDIF
GOTO 30
ENDIF
ENDIF
IF(RMAR.EQ.0)THEN
  IF((RX-RDC(DPTS,2)).GE.FT)THEN
    IF(ABS((RFT-RX)).LE..10)THEN
      PFLAG=3.0
    ELSEIF(MROW.EQ.5)THEN
      PFLAG=5.0
    ELSE
      PFLAG=4.0
    ENDIF
    LFT=RDC(DPTS,2)
    RFT=RX
    IF(PFLAG.NE.4)THEN
      RMR=RMR+1
      RM(RMR,1)=PFLAG
      RM(RMR,2)=RMX
      RM(RMR,3)=SWEEP
    ENDIF
    GOTO 30
  ENDIF
ENDIF
ENDIF
IF(MROW.EQ.5)THEN
  IF((LDC(1,2)-(LX+MG)).GE.FT)THEN
    PFLAG=5.0
    LFT=LX+MG
    RFT=LDC(1,2)
    LMR=LMR+1
    LM(LMR,1)=PFLAG
    LM(LMR,2)=LMX
    LM(LMR,3)=SWEEP
    GOTO 30
  ELSEIF(((RX-MG)-RDC(DPTS,2)).GE.FT)THEN
    PFLAG=5.0
    LFT=RDC(DPTS,2)
    RFT=RX-MG
    RMR=RMR+1
    RM(RMR,1)=PFLAG
    RM(RMR,2)=RMX
    RM(RMR,3)=SWEEP
    GOTO 30
  ELSEIF(DPTS.GT.1)THEN
```

```

      DO 20,I2=2,DPTS
        IF((LDC(I2,2)-RDC(I2-1,2)).GE.FT)THEN
          PFLAG=5.0
          LFT=RDC(I2-1,2)
          RFT=LDC(I2,2)
          LMR=LMR+1
          RMR=RMR+1
          LM(LMR,1)=PFLAG
          LM(LMR,2)=0.0
          LM(LMR,3)=SWEEP
          RM(RMR,1)=PFLAG
          RM(RMR,2)=0.0
          RM(RMR,3)=SWEEP
          GOTO 30
        ENDIF
20    CONTINUE
      PFLAG=4.0
    ENDIF
  ELSE
    PFLAG=4.0
  ENDIF
  LFT=LX+(RX-LX)*.5
  RFT=LFT+1.0
30  CONTINUE
  RETURN
END
*****
*   THIS SUBROUTINE READS SHAPE AND WANE FILES   *
*****
      SUBROUTINE READS(L1,L2,R1,R2,LSHP,RSHP,FIN,SWEEP)
      INTEGER FIN,RSHP,LSHP
      REAL L1,L2,R1,R2,SWEEP
      FIN=0
10  IF(L2.LE.SWEEP)THEN
      READ(LSHP,*)L1,L2
      IF(L1.EQ.-99.0)THEN
        FIN=LSHP
        GOTO 40
      ENDIF
      GOTO 10
    ENDIF
20  IF(R2.LE.SWEEP)THEN
      READ(RSHP,*)R1,R2
      IF(R1.EQ.-99.0)THEN
        FIN=RSHP
        GOTO 40
      ENDIF
      GOTO 20
    ENDIF
40  CONTINUE
  RETURN

```

```

      END
*****
*   THIS SUBROUTINE CLIPS FISHTAILS AND TRASH   *
*****
      SUBROUTINE FTCLIP(CLP,CKNT,SC,MS,CDE,TYPE,STRT,STP,PCS,
      *SEND,BEND)
      REAL CLP(500,4),STRT,STP,MS,SC,TYPE
      INTEGER CDE,CKNT,PCS,P,SEND,BEND
      IF(BEND.EQ.0)THEN
        IF(SEND.NE.0)THEN
          P=PCS-1
        ELSE
          P=PCS
        ENDIF
      ENDIF
10  IF((STP-STRT).GE.(SC+MS))THEN
      CKNT=CKNT+1
      CLP(CKNT,1)=TYPE
      CLP(CKNT,2)=STRT
      CLP(CKNT,3)=STRT+SC
      CLP(CKNT,4)=P
      STRT=CLP(CKNT,3)
      GOTO 10
    ELSEIF(CDE.EQ.2)THEN
      GOTO 20
    ELSE
      CKNT=CKNT+1
      CLP(CKNT,1)=TYPE
      CLP(CKNT,2)=STRT
      CLP(CKNT,4)=P
      IF((STP-STRT).GT.SC)THEN
        CLP(CKNT,3)=STRT+(STP-STRT)*.5
        STRT=CLP(CKNT,3)
        CKNT=CKNT+1
        CLP(CKNT,1)=TYPE
        CLP(CKNT,2)=STRT
        CLP(CKNT,4)=P
      ENDIF
    ENDIF
      CLP(CKNT,3)=STP
      STRT=CLP(CKNT,3)
      STP=STRT
20  CONTINUE
      RETURN
      END
*****
*   THIS SUBROUTINE CLIPS FULL WIDTH PIECES   *
*****
      SUBROUTINE FLCLIP(CLP,CKNT,FC,SC,MS,PANEL,CDE,STRT,STP,
      *FULLKNT,PCS,SEND,BEND)
      REAL CLP(500,4),FC,SC,MS,STRT,STP

```

```

      INTEGER CKNT,PANEL,CDE,FULLKNT,PCS,P,SEND,BEND
      IF(BEND.EQ.0)THEN
        IF(SEND.NE.0)THEN
          P=PCS-1
        ELSE
          P=PCS
        ENDIF
      ENDIF
10  IF((STP-STRT).GE.(FC+MS))THEN
      CKNT=CKNT+1
      CLP(CKNT,1)=1.0
      CLP(CKNT,2)=STRT
      CLP(CKNT,3)=STRT+FC
      CLP(CKNT,4)=P
      STRT=CLP(CKNT,3)
      FULLKNT=FULLKNT+1
      GOTO 10
    ELSEIF(CDE.EQ.2)THEN
      GOTO 20
    ELSE
      CKNT=CKNT+1
      CLP(CKNT,1)=1.0
      CLP(CKNT,2)=STRT
      CLP(CKNT,4)=P
      IF((STP-STRT).GT.(SC+MS))THEN
        CLP(CKNT,3)=STRT+SC
      ELSEIF((STP-STRT).GT.SC)THEN
        IF(PANEL.EQ.0)THEN
          CLP(CKNT,3)=STRT+(STP-STRT)*.5
        ELSE
          CLP(CKNT,3)=STRT+SC
        ENDIF
      ELSE
        GOTO 15
      ENDIF
      STRT=CLP(CKNT,3)
      CKNT=CKNT+1
      CLP(CKNT,1)=1.0
      CLP(CKNT,2)=STRT
      CLP(CKNT,4)=P
    ENDIF
15  CLP(CKNT,3)=STP
      STRT=CLP(CKNT,3)
      STP=STRT
20  CONTINUE
      RETURN
      END
*****
*   THIS SUBROUTINE CLOSES OUT SHAPE AND WANE FILES   *
*****
      SUBROUTINE REND(FIN,DC,BN,PCS,BEND,SWEEP,MG)

```

```
      INTEGER FIN,DC,BN,PCS,BEND,L(3,3),I1,I7,UN(3)
      REAL X,Y,SWEEP,MG,YY(3),LAST
      LAST=0.0
      YY(1)=0.0
      YY(2)=0.0
      YY(3)=0.0
      BEND=0
      GOTO(10,20,30,40)FIN
10    UN(1)=2
      UN(2)=3
      UN(3)=4
      GOTO 50
20    UN(1)=1
      UN(2)=3
      UN(3)=4
      GOTO 50
30    UN(1)=1
      UN(2)=2
      UN(3)=4
      GOTO 50
40    UN(1)=1
      UN(2)=2
      UN(3)=3
50    CONTINUE
      DO 70 I7=1,3
60      READ(UN(I7),*)X,Y
        IF(X.NE.-99.)THEN
          YY(I7)=Y
          GOTO 60
        ENDIF
70    CONTINUE
      LAST=MAX(YY(1),YY(2),YY(3))
      IF(LAST.NE.0.0)THEN
        SWEEP=LAST
      ENDIF
80    READ(FIN,*)I1,I2,I3
      IF(I1.EQ.-999)GOTO 80
      IF(I1.EQ.-111)THEN
        BEND=1
        GOTO 110
      ENDIF
      DC=I1
      BN=I2
      PCS=I3
      DO 100 I1=1,3
        READ(UN(I1),*)L(I1,1),L(I1,2),L(I1,3)
100    CONTINUE
      IF(((L(1,1)+L(2,1)+L(3,1))/3).NE.DC)THEN
        WRITE(8,*)'DIAMETER CLASSES DO NOT MATCH.'
        WRITE(8,105)L(1,1),L(2,1),L(3,1),DC
105    FORMAT(I3,2X,I3,2X,I3,4X,I3)
```

```

        STOP
    ENDIF
    IF(((L(1,2)+L(2,2)+L(3,2))/3).NE.BN)THEN
        WRITE(8,*)'BLOCK NUMBERS DO NOT MATCH.'
        WRITE(8,105)L(1,2),L(2,2),L(3,2),BN
        STOP
    ENDIF
    IF(((L(1,3)+L(2,3)+L(3,3))/3).NE.PCS)THEN
        WRITE(8,*)'PIECE NUMBERS DO NOT MATCH.'
        WRITE(8,105)L(1,3),L(2,3),L(3,3),PCS
        STOP
    ENDIF
110 CONTINUE
    RETURN
    END
*****
*   THIS SUBROUTINE SORTS/MERGES CLIPS FOR OUTPUT   *
*****
    SUBROUTINE CLPRITE(CLP,CKNT,MS,LFISH,RFISH,DFISH,RAND,
    *HALF,FULL,LFISH2,RFISH2,DFISH2,RAND2)
    INTEGER CKNT,KNT,PTR,W1,W2,LFISH,RFISH,DFISH,
    *RAND,HALF,FULL
    REAL CLP(500,4),TCLP(500,4),MS,CROSGRN,LFISH2,RFISH2,
    *DFISH2,RAND2
    KNT=0
    CROSGRN=0.0
    PTR=1
    DO 10 I1=1,CKNT
        IF((CLP(I1,3)-CLP(I1,2)).LT.MS)THEN
            CLP(I1,1)=4.0
            KNT=KNT+1
        ENDIF
10 CONTINUE
    IF(KNT.EQ.0)THEN
        DO 15 I5=1,CKNT
            DO 15 I6=1,4
                TCLP(I5,I6)=CLP(I5,I6)
15 CONTINUE
        KNT=1
        GOTO 43
    ENDIF
    KNT=0
    PTR=1
20 KNT=KNT+1
    IF(KNT.GT.999)THEN
        CALL PMDLOAD
        STOP
    ENDIF
    TCLP(KNT,1)=CLP(PTR,1)
    TCLP(KNT,2)=CLP(PTR,2)
    TCLP(KNT,4)=CLP(PTR,4)

```

```

      IF(PTR.EQ.CKNT)GOTO 40
      IF(CLP(PTR,1).EQ.4.0)THEN
30    PTR=PTR+1
      IF(CLP(PTR,1).EQ.4.0)THEN
        IF(PTR.EQ.CKNT)THEN
          IF(KNT.EQ.1)GOTO 40
          TCLP(KNT,2)=TCLP(KNT-1,3)
          GOTO 40
        ELSE
          GOTO 30
        ENDIF
      ELSE
        TCLP(KNT,3)=CLP(PTR,2)
        GOTO 20
      ENDIF
    ELSE
      TCLP(KNT,3)=CLP(PTR,3)
      PTR=PTR+1
      GOTO 20
    ENDIF
40  TCLP(KNT,3)=CLP(PTR,3)
43  CONTINUE
    DO 45 I5=1,KNT
      IF(TCLP(I5,1).NE.4.0)THEN
        CROSGRN=TCLP(I5,3)-TCLP(I5,2)
        IF(TCLP(I5,1).EQ.2.0)THEN
          LFISH=LFISH+1
          LFISH2=LFISH2+(CROSGRN*51.0)/144.0
        ELSEIF(TCLP(I5,1).EQ.3.0)THEN
          RFISH=RFISH+1
          RFISH2=RFISH2+(CROSGRN*51.0)/144.0
        ELSEIF(TCLP(I5,1).EQ.5.0)THEN
          DFISH=DFISH+1
          DFISH2=DFISH2+(CROSGRN*51.0)/144.0
        ELSEIF(CROSGRN.LT.26.59)THEN
          RAND=RAND+1
          RAND2=RAND2+(CROSGRN*100.0)/144.0
        ELSEIF(CROSGRN.GT.50.0)THEN
          FULL=FULL+1
        ELSE
          HALF=HALF+1
        ENDIF
      ENDIF
45  CONTINUE
      WRITE(7,50)((TCLP(W1,W2),W2=1,4),W1=1,KNT)
50  FORMAT(2X,F3.0,2X,F8.3,2X,F8.3,2X,F3.0,/)
      RETURN
      END

```



```
*****
* THIS SUBROUTINE ADJUSTS A CLIP FOR MARGIN SETTING *
*****
```

```
      SUBROUTINE MGNADJ(RECOV,INL,PFLAG,SWEEP,CDE,MG,M)
      REAL RECOV(5,3),INL,PFLAG,SWEEP,MG,M
      INTEGER CDE
```

C
C

```
      IF(INL.EQ.4.0)THEN
        SWEEP=SWEEP+M
        RECOV(4,3)=SWEEP
        CDE=1
      ELSEIF(PFLAG.EQ.4.0)THEN
        IF((RECOV(INL,3)-RECOV(INL,2)).GE.MG)THEN
          RECOV(INL,3)=RECOV(INL,3)-M
          RECOV(4,2)=RECOV(INL,3)
        ENDIF
        CDE=1
        GOTO 30
      ELSEIF(INL.EQ.1.0)THEN
        IF((RECOV(1,3)-RECOV(1,2)).GE.MG)THEN
          RECOV(1,3)=RECOV(1,3)-M
          RECOV(PFLAG,2)=RECOV(1,3)
        ENDIF
        GOTO 30
      ELSEIF(PFLAG.EQ.1.0)THEN
        SWEEP=SWEEP+M
        RECOV(INL,3)=SWEEP
      ELSEIF((RECOV(INL,3)-RECOV(INL,2)).GE.MG)THEN
        RECOV(INL,3)=RECOV(INL,3)-M
      ENDIF
      RECOV(PFLAG,2)=SWEEP
      RECOV(PFLAG,3)=SWEEP
30 CONTINUE
      RETURN
      END
```