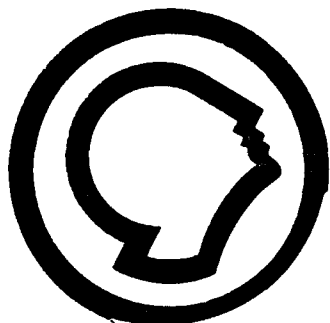


AD-A249 034



FOG-M FIBER OPTICS WINDING PROCESS
SIMULATION AND VALIDATION

(5-31823)

DTIC
ELECTE
APR 08 1992
S D

This document has been approved
for public release and sale; its
distribution is unlimited.

92 4 07 008

92-08929



Research Institute
The University of Alabama in Huntsville

Unclassified

SECURITY CLASSIFICATION OF THIS PAGE

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188
Exp. Date: Jun 30, 1986

1. REPORT SECURITY CLASSIFICATION UNCLASSIFIED			1b. RESTRICTIVE MARKINGS	
2. SECURITY CLASSIFICATION AUTHORITY			3. DISTRIBUTION / AVAILABILITY OF REPORT Distribution A - Distribution A is Unlimited	
4. DECLASSIFICATION / DOWNGRADING SCHEDULE			5. MONITORING ORGANIZATION REPORT NUMBER(S)	
6. PERFORMING ORGANIZATION REPORT NUMBER(S) 5-31823			7a. NAME OF MONITORING ORGANIZATION SRS Technologies	
7. NAME OF PERFORMING ORGANIZATION DAH Research Institute		8b. OFFICE SYMBOL (if applicable) RRP	7b. ADDRESS (City, State, and ZIP Code) 990 Explorer Blvd., Cummings Research Pk. Huntsville, AL 35806	
8. ADDRESS (City, State, and ZIP Code) The University of Alabama in Huntsville Research Institute Room E-47 Huntsville, AL 35899		9. PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER DAAH01-87-D-0179 (D. O. 7)		
9. NAME OF FUNDING / SPONSORING ORGANIZATION		10. SOURCE OF FUNDING NUMBERS		
8b. OFFICE SYMBOL (if applicable)		PROGRAM ELEMENT NO.	PROJECT NO.	TASK NO.
10. ADDRESS (City, State, and ZIP Code)		WORK UNIT ACCESSION NO.		
11. TITLE (Include Security Classification) FOG-M Fiber Optics Winding Process Simulation and Validation Unclassified				
12. PERSONAL AUTHOR(S) Timothy D. Morgan and Roy A. Kesmodel				
13a. TYPE OF REPORT Final	13b. TIME COVERED FROM 12/30/87 TO 6/18/88	14. DATE OF REPORT (Year, Month, Day) 88 / 06 / 30	15. PAGE COUNT 67	
16. SUPPLEMENTARY NOTATION				
17. COSATI CODES			18. SUBJECT TERMS (Continue on reverse if necessary and identify by block number)	
FIELD	GROUP	SUB-GROUP		
			FOG-M Simulation Fiber Optics Simulation Fiber Winding Simulation	
19. ABSTRACT (Continue on reverse if necessary and identify by block number)				
A software package was developed to simulate the operation of winding fiber optic cable on a bobbin for the FOG-M fiber optic guided missile; the simulation allows engineers to predict the effect of various winding unit parameter changes. The operation of the fiber winder and the simulation are described.				
20. DISTRIBUTION / AVAILABILITY OF ABSTRACT ☒ UNCLASSIFIED/UNLIMITED ☐ SAME AS RPT. ☐ DTIC USERS			21. ABSTRACT SECURITY CLASSIFICATION Unclassified	
22a. NAME OF RESPONSIBLE INDIVIDUAL Mr. Leo M. Collyar			22b. TELEPHONE (Include Area Code) (205)-895-2900	22c. OFFICE SYMBOL

Technical Report 5-31823
Contract No. DAAH01-87-D-0179
Delivery Order No. 7
UAH Report 694

FOG-M FIBER OPTICS WINDING PROCESS
SIMULATION AND VALIDATION

(5-31823)

Final Technical Report for period
30 December 1987 through 18 June 1988

(June)

Prepared by
Timothy D. Morgan
and
Roy A. Kesmodel

Research Institute
University of Alabama in Huntsville
Huntsville, Alabama 35899

Prepared for
SRS Technologies
990 Explorer Blvd. N.W.
Cummings Research Park West
Huntsville, AL 35806

Attn: Mr. Leo M. Collyar

Accession For	
NTIS CRA&I	☒
DTIC TAB	☐
Unannounced	☐
Justification	
By	
Distribution /	
Availability Codes	
Dist	Avail and/or Special
A-1	

Preface

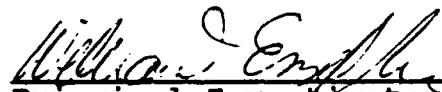
This technical report was prepared by the Resident Research Program Office of the University of Alabama in Huntsville Research Institute. This report is to serve as documentation of technical work performed under contract number DAAH01-87-D-0179-0007. Mr. Timothy D. Morgan was principal investigator and Mr. Roy A. Kesmodel was the principal software engineer. Mr. Leo Collyar of SRS Technologies provided technical coordination.

The views, opinions, and/or findings contained in this report are those of the author(s) and should not be construed as an official Department of the Army position, policy or decision unless so designated by other official documenta-tion.

Except as provided by the Contract Data Requirements List DD form 1423, hereof, the distribution of any contract report in any stage of development or completion is prohibited without the approval of the Contracting Officer.

Prepared for: Commander
 U.S. Army Missile Command
 Redstone Arsenal, AL 35898

I have reviewed this report, dated 30-JUNE-1988
and the report contains no classified information.


Principal Investigator

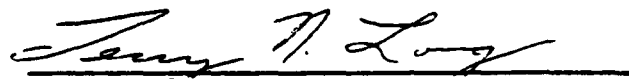

Approval

TABLE OF CONTENTS

	<u>Page</u>
1.0 Introduction	1
2.0 The FOG-M Winding Cell	4
3.0 Simulation Development	14
3.1 Introduction	14
3.2 Translation of INTEL FORTRAN-86 to VAX FORTRAN-77	15
3.3 Physical Modules of the FOG-M WC	16
3.3.1 The Motors, Encoders and Bobbin Table	16
3.3.2 The Lag Angle Detector (LAD)	33
3.3.3 Fiber Tension and the Tension Sensor	35
3.4 Linking the Controller Software and the Physical Subroutines	42
3.5 Output of Winding Parameters	42
4.0 Testing and Evaluation	43
5.0 Conclusion	44
Appendices	

LIST OF ILLUSTRATIONS

Figure Number	Title	<u>Page</u>
1	Path of Fiber Through Winding Cell	2
2	Simplified Diagram of Lag Angle	3
3	The FOG-M Winding Cell	5
4	FOG-M WC Modules	9
5	Mandrel and Lead Screw Motor Hex Commands vs. Output RPM	18
6	Mandrel DAC Output Voltage vs. Decimal Equivalents of Hex Commands	25
7	Mandrel Motor Input Voltage vs. Output RPM, Mandrel Voltage Increasing	26
8	Mandrel Motor Input Voltage vs. Output RPM, Mandrel Voltage Decreasing	27
9	Lead Screw DAC Output Voltage vs. Decimal Equivalents of Hex Commands	28
10	Lead Screw Input Voltage vs. Output RPM, Lead Screw Voltage Increasing	29
11	Lead Screw Input Voltage vs. Output RPM, Lead Screw Voltage Decreasing	30

1.0 INTRODUCTION

The objective of this task was to develop a software simulation of the operations of the FOG-M Winding Cell (WC). The simulation was written in FORTRAN-77 to run on a VAX 11/750. The FOG-M WC is designed to accurately wind fiberglass optical fiber on a bobbin. This bobbin is then installed in the FOG-M missile. The purpose of the Winding Cell is to wrap multiple layers of fiber on the bobbin so that it can be rapidly paid-out during the flight of the FOG-M missile. The tension of the fiber and the angle that the fiber makes with the surface of the bobbin (lag angle) are two of the most important factors in achieving the necessary tight, even layers of fiber. For an illustration of the fiber path through the winding cell, see Figure 1. For an illustration of the lag angle, see Figure 2.

The goal of this task was to develop a software model of the FOG-M WC which would allow monitoring of the tension and lag angle under differing conditions, such as various fiber diameters and winding speeds. Completion of the task accomplished these objectives and this report provides a detailed account of the research and development of the FOG-M WC simulation package.

PATH OF FIBER THROUGH WINDING CELL

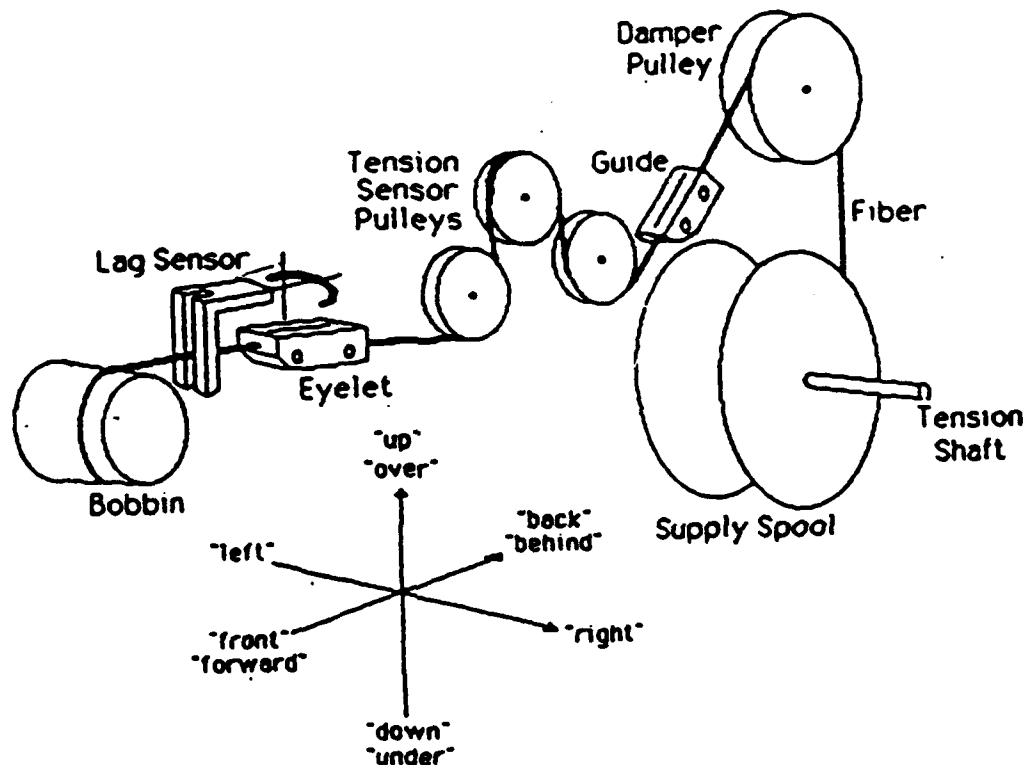


Figure 1

Reprinted from:

Automatic Fiber Winder Operation Manual

U.S. Army Missile Command
Research, Development and Engineering Center
Guidance and Control Directorate
1986

SIMPLIFIED DIAGRAM OF LAG ANGLE

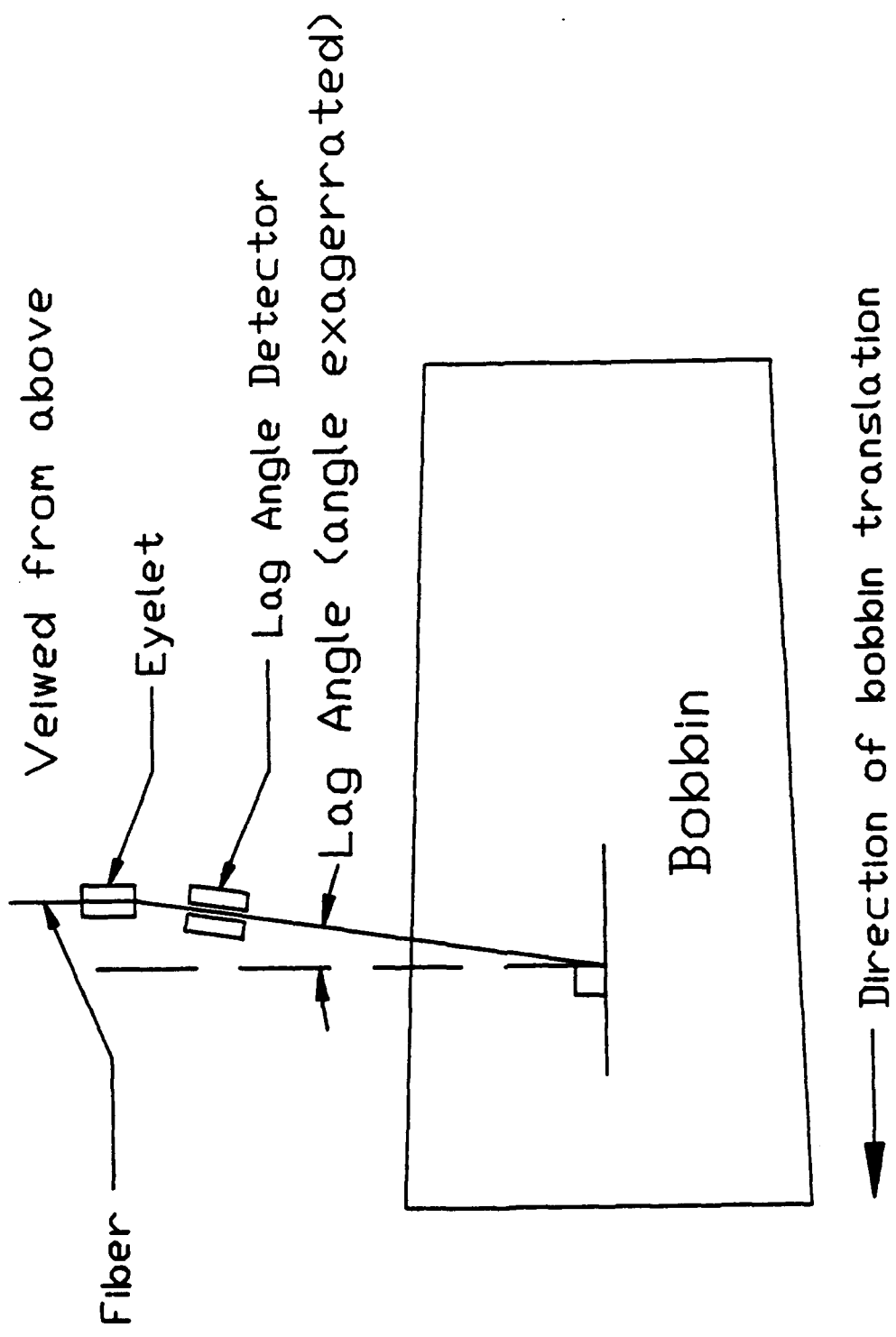


Figure 2

2.0 THE FOG-M WINDING CELL

The FOG-M WC is designed to take fiberglass optical fiber from a supply spool and wind it in very tight, even layers onto the bobbin, which is then installed into the FOG-M missile. In order to facilitate writing the software, the FOG-M WC was defined in terms of two basic modules: the physical components of the FOG-M WC and the INTEL controller system. Figure 3 illustrates the complete FOG-M WC.

The physical components of the FOG-M WC were divided into several smaller modules: the mandrel assembly, the lead screw assembly, the tension motor and supply spool, the tension control assembly, and the lag angle detector (LAD) assembly.

The mandrel assembly consists of the mandrel motor, the mandrel encoder, the mandrel, and the fiber bobbin. The mandrel motor is coupled to, and rotates the mandrel. The mandrel encoder monitors the mandrel motor rotation and reports the angular position of the mandrel to the controller hardware. The bobbin is mounted on, and rotated by, the mandrel.

The FOG-M Winding Cell

(Simplified)

- 1 Lead Screw Assembly
- 2 Bobbin Table Assembly
- 3 Mandrel Assembly
- 4 Tension Motor
- 5 Tension Sensor Assembly
- 6 Lag Angle Detector Assembly

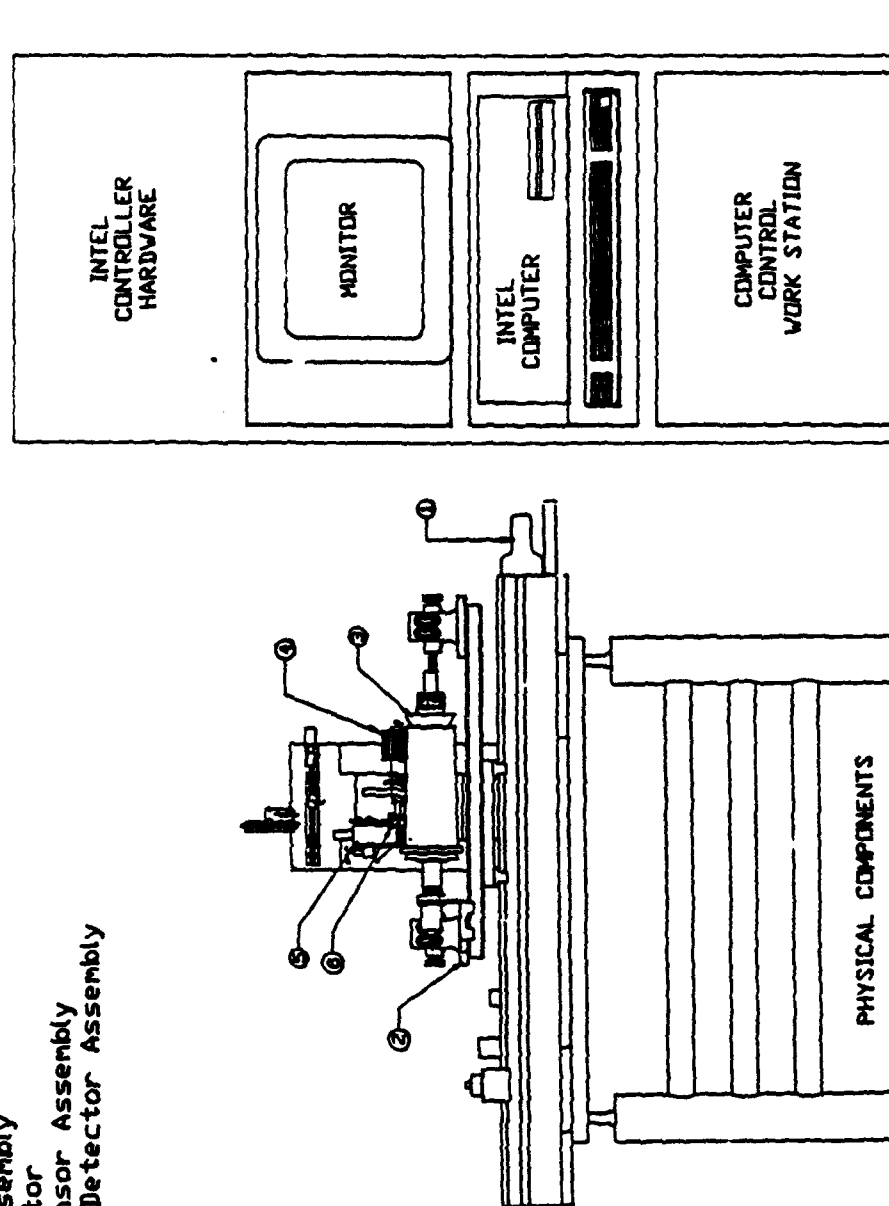


Figure 3.

The lead screw assembly consists of the lead screw motor, the lead screw encoder, the lead screw, and the mandrel assembly table. The lead screw motor is coupled to the lead screw and rotates it. The lead screw encoder monitors the lead screw motor rotation and reports the angular position of the lead screw to the controller hardware. The mandrel assembly is mounted on a table which slides laterally on a set of rails. The rotation of the lead screw causes the mandrel assembly table to move along these rails.

The fiber supply spool is mounted on a shaft which is coupled to the tension motor. The tension motor provides a reverse torque to the fiber supply spool, thereby causing tension in the fiber.

The tension control assembly consists of the tension sensor and the analog tension control circuit. The tension sensor reports the tension in the fiber. The analog tension control circuit monitors this tension. The analog tension circuit controls the tension motor to maintain the tension at the reference level, based upon a comparison between the monitored tension and a reference tension supplied by the controller hardware.

The lag angle detector (LAD) assembly consists of an eyelet and the lag angle detector. The eyelet is the final stationary point through which the fiber passes before it reaches the bobbin. After passing through the eyelet, the fiber advances through a moveable guide. The guide swings in the horizontal plane as the lag angle changes (see Figure 1). The LAD monitors the motions of this moveable guide and reports them to the controller hardware.

The INTEL controller system consists of three basic parts; an INTEL PC computer, controller software written in INTEL FORTRAN-86, and input/output hardware. The purpose of the INTEL computer is to provide an interface between the controlling software and the human operator of the winding cell. The computer accepts inputs for winding parameters and, if necessary, commands to halt the system (panic button) from the operator. The controller software calculates and issues commands to the physical system. These commands are calculated from the initial winding parameters input by the operator and from calculations based on values obtained while monitoring the winding cell. The hardware of the INTEL computer provides an interface between the controlling software and the winding cell. This interface issues winding commands and automatically monitors the winding

process.

The INTEL FORTRAN-86 controller software is the program which controls most of the operations of the FOG-M WC. The program first prompts the operator to provide the winding parameters by reading them either from a data file or from operator keyboard input. It then initiates the winding sequence and controls the entire winding process from start to finish. The control sequence consists of ramping the tension, lead screw, and mandrel motors; monitoring the mandrel and lead screw encoders and maintaining a set ratio (based mainly on average fiber diameter and the number of threads per inch in the lead screw) between their speeds; monitoring the lag angle detector and adjusting lead screw speed to compensate for variances in fiber diameter; testing for encoder errors (program is terminated if too many errors occur); and maintaining a reference voltage for the analog tension control circuit. For a block diagram of the main control sequence, refer to Figure 4. The initiation sequence consists of several steps. In the first step, the tension motor is ramped to the preset tension voltage. Prior to the initiation of a wind, there is usually a small amount of slack in the fiber. Ramping is the process by which the voltage to a motor is raised to its full winding value at a

FDG-M WC MODULES

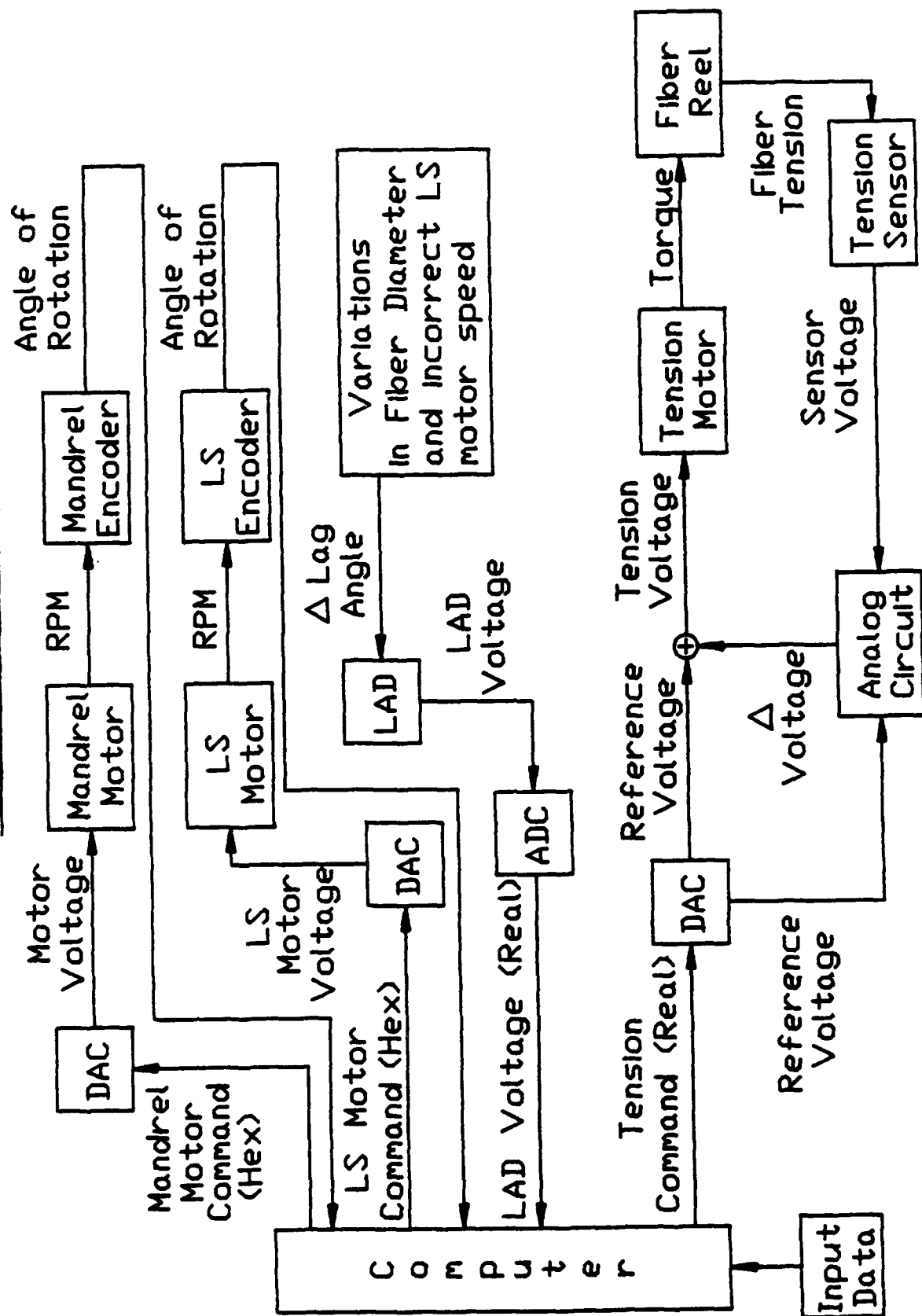


FIGURE 4.

constant slope rather than simply turning it on at that value. Ramping the tension motor voltage provides an even increase in the tension in the fiber. This will gradually remove any slack in the fiber (abrupt removal of this slack could cause fiber breakage). The computer then determines the reference (perpendicular) voltage for the lag angle detector by reading the detector several times. The initial angle for the perpendicular set point is predetermined by the operator. The fiber is connected to the spool at a spot well off of perpendicular (as much as an inch off). The operator winds fiber onto the bobbin by turning the bobbin by hand. As he turns the bobbin, each succeeding turn of fiber will approach the previous turn. As the lag angle approaches perpendicular, the distance between the turns decreases. When the present turn of fiber just barely touches the preceding turn, the lag angle is considered to be perpendicular. These steps have been performed prior to starting the wind. Therefore, when the computer reads the LAD, the perpendicular has already been set. The software then causes a delay of about 10 seconds to allow the system to settle. The controller next commands the mandrel to rotate about 1/8th of a turn, this is to establish the correct lag angle for winding. After another delay of about 1-1/2 seconds, the lead screw motor is ramped to full winding

speed and, almost immediately afterwards, the mandrel motor is ramped to full winding speed. The lead screw and mandrel motors voltages are ramped to avoid excessive stresses to the coupling mechanisms. The mandrel is controlled in 'open loop' fashion. That is, after ramping, an initial winding speed command is issued and the mandrel maintains that speed throughout each layer. The controller issues no commands to the mandrel other than to ramp it to the full winding voltage or to turn it off. As a full wind progresses, the computer constantly monitors the mandrel and lead screw encoders to make sure that the motors are operating properly. The encoders are electronic devices connected to the mandrel and lead screw shafts. The encoders constantly monitor the shafts and report their absolute angular positions. As the encoders are capable of detecting 256 positions in a rotation, the angle can be reported in one 8-bit word. The controller software converts this 8-bit word to an angle in degrees. If an excessive number of errors are detected, the wind is terminated. An error occurs when either of the motors turns a great deal more than the other. Breakage of or gross slippage in the coupling between a motor and a shaft are the only times when an error could occur. The lag angle detector is also monitored to insure that the lead screw is traversing the bobbin table at the proper rate. The bobbin

table should move laterally at a rate equal to one fiber diameter per rotation of the mandrel. The lag angle detector is an electronic device that the fiber passes through just prior to contacting the bobbin. It is capable of detecting small changes in the angle that the fiber makes with the surface of the bobbin. This angle is reported to the controller as a voltage. The analog voltage from the lag angle detector is input to an analog to digital converter (ADC) circuit. The ADC converts this voltage into two 8-bit words. These words are passed to the controlling software. The software converts the two words into a real number and compares this value to the reference voltage. If the lead screw is rotating either too rapidly or too slowly, the controller detects this error as a difference between the reported voltage from the lag angle detector and its reference voltage. The controller then issues a new command to the lead screw motor to correct the error. The controller commands are calculated by the software. They are converted into hexadecimal values and sent to a digital to analog converter (DAC) circuit. The DAC converts these hexadecimal values into analog voltages. These analog voltages are sent as controlling voltages to the lead screw motor.

The program keeps track of the number of turns in each layer

and the total number of layers in a wind. At the end of each layer, the program can automatically handle the step back maneuver. While winding, first turns of one layer must not lie directly on top of the last turns of the preceding layer. During payout, this would result in the last turns of the top layer pulling off the first turns of the lower layer, resulting in a tangling of the fiber and probable fiber breakage. To avoid this problem, the first turn of each succeeding layer is wound beginning several turns back from the last turn of the preceding layer. This is known as 'stepback'. During the stepback maneuver, the mandrel and lead screw motors are halted. Then the lead screw motor is reversed and run for a short time to pull the fiber back over the preceding layer. The lead screw motor is again halted, reversed, and run for a very short period until the lag angle reaches the perpendicular point for the next layer. The lead screw motor is again halted and reversed to prepare it for the next layer. The stepback maneuver is completed and the control sequence loops back to the beginning. When the controlling software determines that all layers have been wound, the winding sequence terminates and a menu appears on the terminal screen. For a more detailed description of the physical system, controller functions, and menu options of the FOG-M WC, refer to Appendix G (Automatic Fiber Winder

Operations Manual), under separate cover.

3.0 SIMULATION DEVELOPMENT

3.1 Introduction

Due to the anticipated complexity and size of the simulation software code, it was decided that simulation development should be started on the existing Army VAX 11/750 CAD/CAM system. Permission was obtained from Army personnel to utilize their VAX 11/750 and FORTRAN-77 compiler in design of the simulation. With this change, it was also determined that development of the integrated "real time" IBM PC based workstation was not feasible for the phase one simulation effort.

The physical system of the FOG-M WC was defined in terms of modules. The motions of these modules were then modelled with preliminary subroutines based on data obtained from the winding cell lab. The INTEL controller software was obtained from the Army and translated into VAX FORTRAN-77 for use in the simulation. Experimental data relating to controller commands vs. module motions was obtained from the Army and was used to create accurate subroutines to simulate the

modules of the physical system. The simulation subroutines were linked to the translated controller software. Finally, operator control of winding parameters was provided.

3.2 Translation of INTEL FORTRAN-86 to VAX FORTRAN-77

To more accurately emulate actual operations of the Winding Cell, the existing INTEL FORTRAN-86 software was converted into VAX FORTRAN-77. This has provided a level of accuracy to the simulation which would be extremely difficult to achieve by attempting to simulate the controlling software.

The conversion consisted of analyzing many of the FORTRAN-86 commands to ascertain their functions and converting them to similar FORTRAN-77 commands (in some cases, writing functions or subroutines to perform the command) and converting some of the FORTRAN-86 variable types to compatible FORTRAN-77 types. The software was extensively analyzed to determine the functions of all of the subroutines. For a complete listing of the converted INTEL source code, refer to Appendix E, under separate cover. All of the variables were carefully examined to determine the exact formats of the inputs and outputs expected from the physical system by the software. For a complete list of all

of the variables in the INTEL software, with brief descriptions of their functions, refer to Appendix B (The INTEL Variables). Appendix C provides a list of the INTEL subroutines with brief descriptions of each. Appendix D provides a chart which shows all of the INTEL variables and the subroutines each variable appears in. The full controller software now runs on the VAX and functions the same as the INTEL system. The program gives the same prompts to the operator, accepts the initial values, and operates on them in the same manner as the INTEL system.

3.3 Physical Modules of the FOG-M WC

3.3.1 The Motors, Encoders, and Bobbin Table

The modules of the physical portions of the winding cell were modeled as subroutines. The initial models of the mandrel and lead screw motors were derived from data appearing on a graph which was obtained from engineers working with the winding cell. The graph provides the relationship between the decimal equivalents to the hexadecimal controller commands, the motor voltages, and the rotational speeds of the motors in revolutions per minute (RPM). For a copy of this graph, refer to Figure 5. The formulas for the motors

were as follows:

For the mandrel motor:

$$\text{BRPM} = -51.4 \times \text{BMV} - 30.2$$

Where:

BRPM = Rate of bobbin rotation in RPM

MNMV = Mandrel motor voltage in volts

For the lead screw motor:

$$\text{LSRPM} = -12.2 \times \text{LSMV} - 9.6$$

Where:

LSRPM = Rate of lead screw rotation in RPM

LSMV = Lead screw motor voltage in volts

The formula for the translation of the bobbin table was derived based on the fact that the lead screw has a pitch of 10 threads per inch. As the table must move at a rate equal to one fiber diameter per bobbin rotation (which is the same

MANDREL AND LEAD SCREW MOTOR HEX COMMANDS VS. OUTPUT RPM

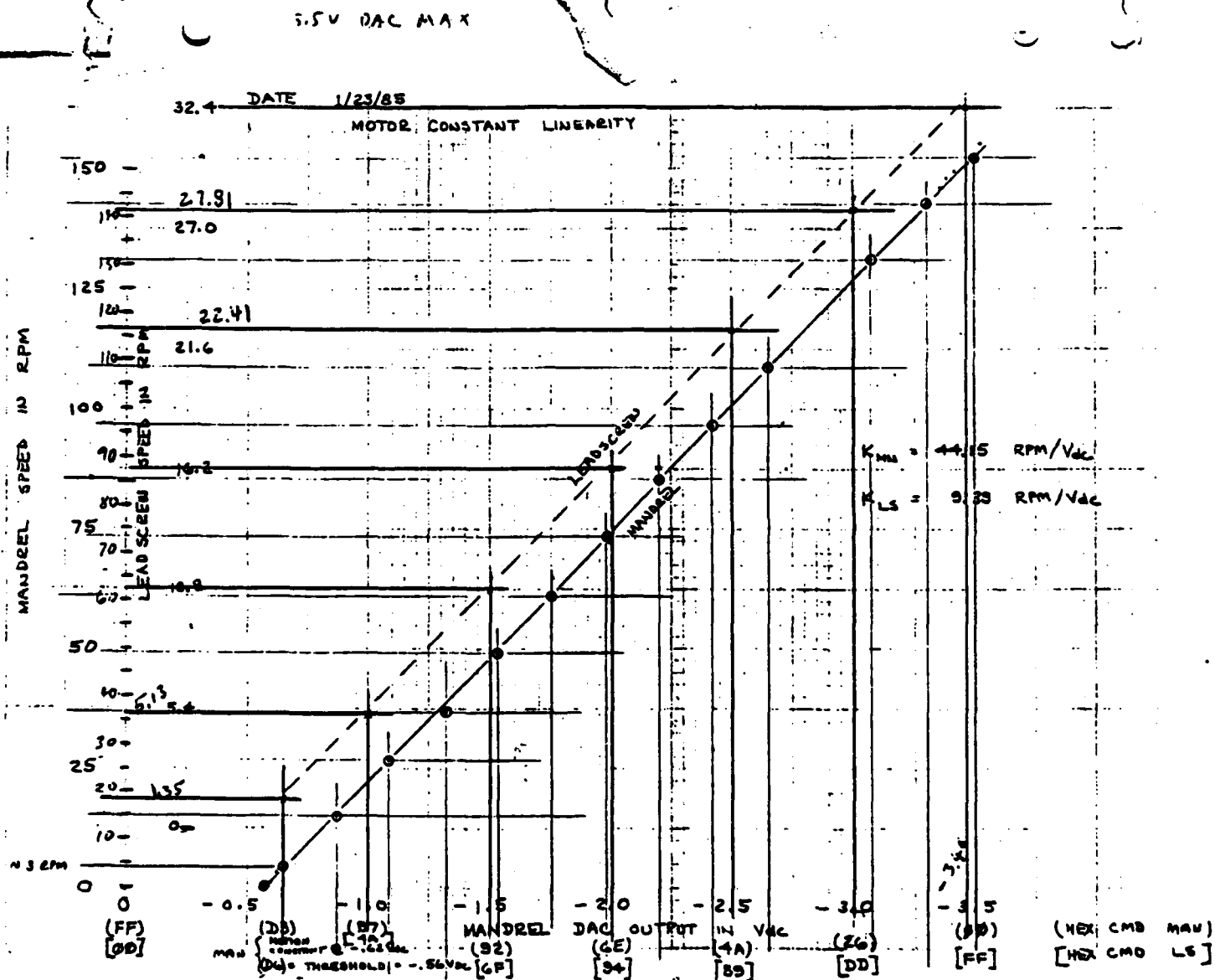


FIGURE 5.

Reprinted from:

Chart provided by FOG-M WC Lab Engineers

U. S. Army Missile Command
 Research, Development and Engineering Center
 Guidance and Control Directorate
 1985

as one mandrel rotation), a formula can be deduced based on the rotational speed of the mandrel and the average fiber, diameter as follows:

$$TTRIS = MNRDS \times FDI / 360$$

Where:

TTRIS = Table translation in inches / second

MNRDS = Mandrel rotation in degrees / second

FDI = Average fiber diameter in inches

At this point, a very simple simulation for the motions of the mandrel, the lead screw, and the bobbin table was written. A driver program was written which used the rotational speed of the mandrel, an arbitrarily chosen fiber diameter, and a time step function to simulate the movements of the mandrel, lead screw, and bobbin table.

Additional experimental data on the mandrel and lead screw motors was obtained from Army sources. The data was generated as described in the following paragraph.

Hexadecimal commands were sent from the computer to the DAC. The DAC converted these commands into analog voltages. The voltages were sent to the motors. The resulting motor RPM

were recorded. For the mandrel motor, the hex commands began at hex FF (dead stop) and gradually increased to hex 00 (full speed). Full speed for the mandrel motor was about 157 RPM. The mandrel motor was then gradually brought back to stop. For the lead screw motor the commands were exactly reversed; hex 00 was dead stop and hex FF was full speed. Full speed for the lead screw motor was about 28.6 RPM. The lead screw motor was also gradually taken from dead stop to full speed and back to dead stop. As the motors were ramped up and down, the voltage from the DAC and the RPM of the motors were monitored. See Tables 3.1 through 3.4 for reproductions of this data.

The data was analyzed using a commercial plotting package. The analysis yielded highly accurate formulas for the relationships between computer hex command and DAC analog voltage output and the relationships between motor analog voltage input and output RPM. These formulas were used to fine-tune the preliminary simulation subroutines. The same formulas were used for the subroutines in the completed simulation. Plots produced by the plotting package and the formulas used can be found in Figures 6 through 11. As the formulas found for the increasing modes varied slightly from those found for the decreasing modes, the values were

TABLE 3.1

EXPERIMENTAL DATA FOR INCREASING MANDREL VOLTAGE

Hex Command	DAC Voltage Out	Mandrel Motor RPM
D2	.62	2.30
D1	.64	2.90
D0	.65	3.70
CE	.67	5.02
C8	.75	8.90
C0	.85	14.00
B8	.95	18.30
B0	1.05	23.90
A8	1.20	29.50
A0	1.30	35.00
98	1.40	40.80
90	1.50	46.90
88	1.60	52.00
80	1.73	57.70
78	1.85	63.30
70	1.97	68.90
68	2.10	74.50
60	2.20	80.00
58	2.30	85.78
50	2.43	96.00
48	2.60	103.60
40	2.80	109.20
38	2.80	115.30
30	2.95	121.40
28	3.00	127.00
20	3.20	133.60
18	3.30	139.00
10	3.40	145.00
08	3.50	151.00
00	3.60	157.00

TABLE 3.2
EXPERIMENTAL DATA FOR DECREASING MANDREL VOLTAGE

Hex Command	DAC Voltage Out	Mandrel Motor RPM
00	3.60	157.00
10	3.40	145.00
20	3.20	134.00
30	3.00	122.00
40	2.70	110.00
50	2.50	98.00
60	2.20	81.00
70	1.95	70.00
80	1.75	59.00
90	1.50	48.00
A0	1.30	37.00
B0	1.07	25.00
C0	.85	15.00
D0	.65	4.00
D2	.62	2.70

TABLE 3.3

EXPERIMENTAL DATA FOR INCREASING LEAD SCREW VOLTAGE

Hex Command	DAC Voltage Out	LS Motor RPM
63	1.15	1.87
66	1.17	3.00
69	1.19	3.28
6C	1.21	4.45
70	1.30	6.60
78	1.35	7.50
80	1.45	8.90
88	1.50	9.50
90	1.65	10.80
98	1.70	11.25
A0	1.85	11.60
A8	1.90	11.80
B0	2.05	15.00
B8	2.10	15.90
C0	2.25	18.20
C8	2.35	18.80

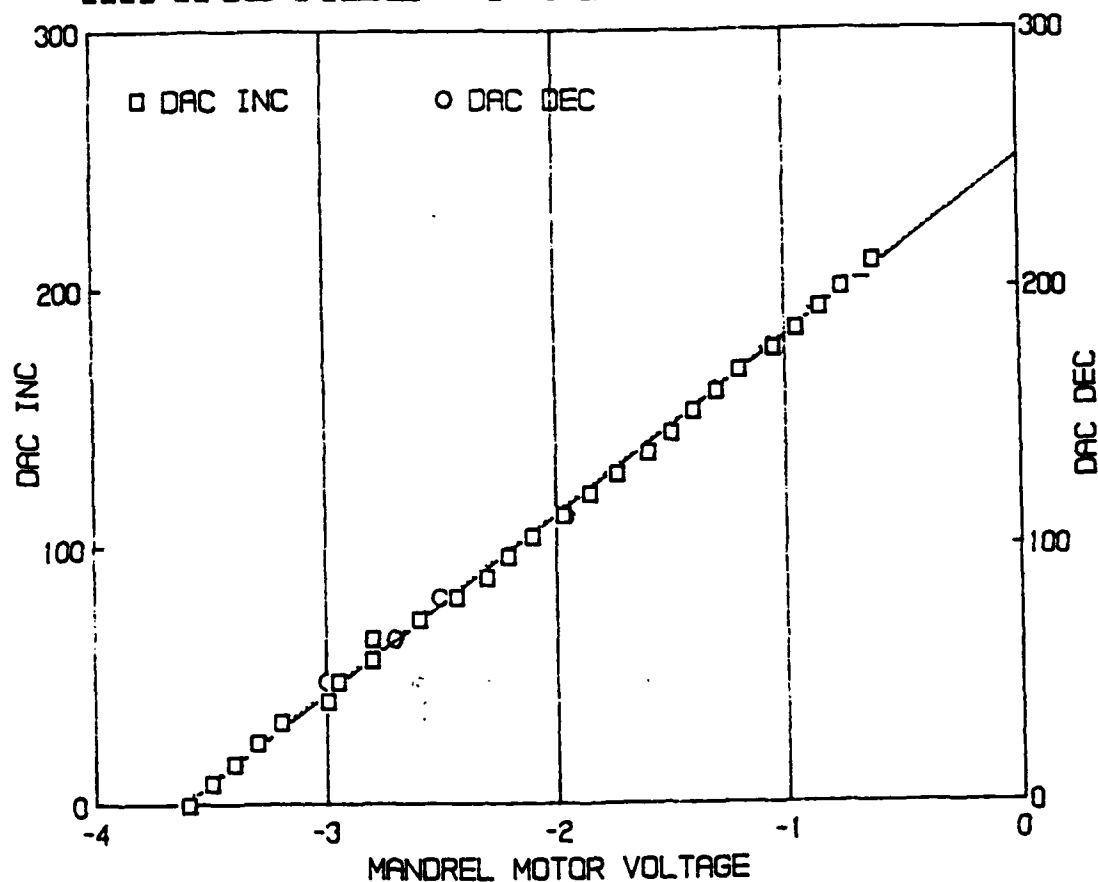
TABLE 3.4

EXPERIMENTAL DATA FOR DECREASING LEAD SCREW VOLTAGE

Hex Command	DAC Voltage Out	LS Motor RPM
FF	3.30	28.60
E0	2.80	23.90
D0	2.60	22.00
C0	2.40	19.70
B0	2.20	17.10
A0	2.00	14.70
90	1.65	11.50
80	1.45	8.50
78	1.35	7.03
70	1.30	6.66
68	1.23	5.40
60	1.10	4.80
58	1.00	3.60
50	.93	3.05
48	.80	1.51
40	.75	0.90

MANDREL DAC
OUTPUT VOLTAGE
VS.
DECIMAL EQUIVALENTS OF HEX COMMANDS

MANDREL VOLT VS DAC

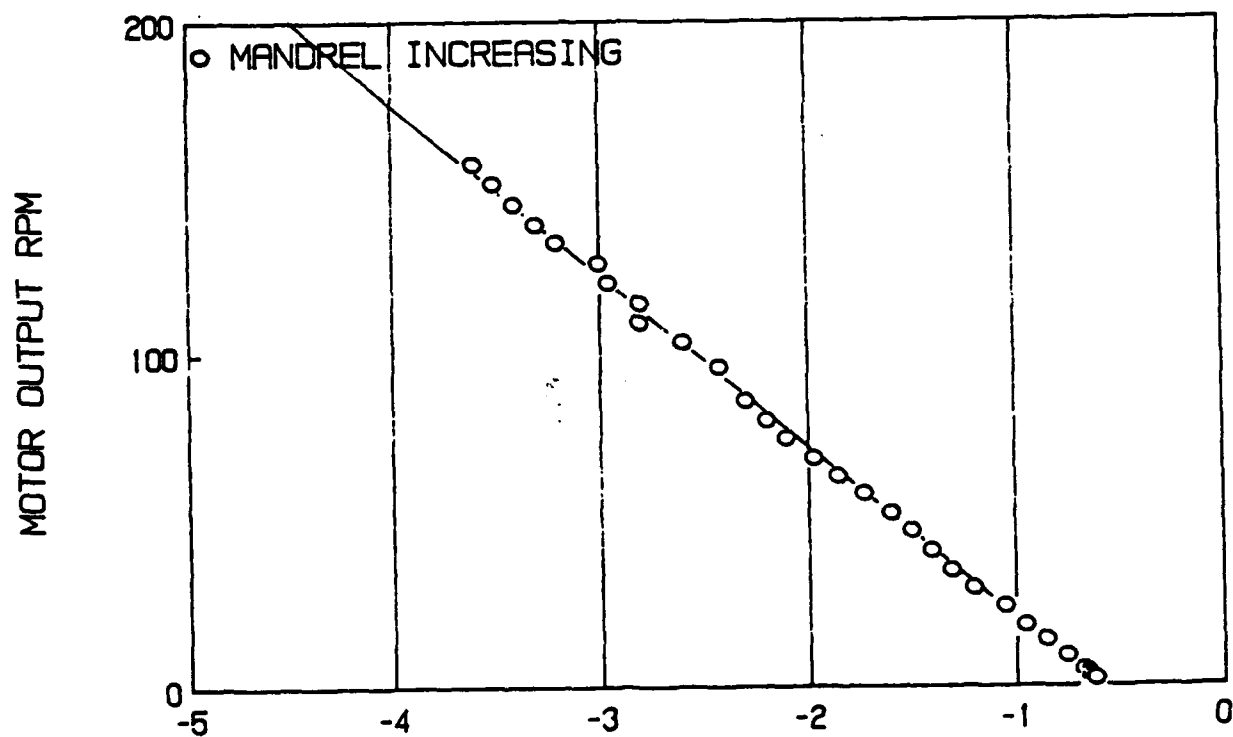


$$\text{VOLTS} = 1.44732238 \times \text{HEX} - 3.623469818$$

FIGURE 6.

MANDREL MOTOR
INPUT VOLTAGE
VS.
OUTPUT RPM
MANDREL VOLTAGE INCREASING

MANDREL INCREASING



$Y = -51.54191883X - 30.82779968$

FIGURE 7.

MANDREL MOTOR
INPUT VOLTAGE
VS.
OUTPUT RPM
MANDREL VOLTAGE DECREASING

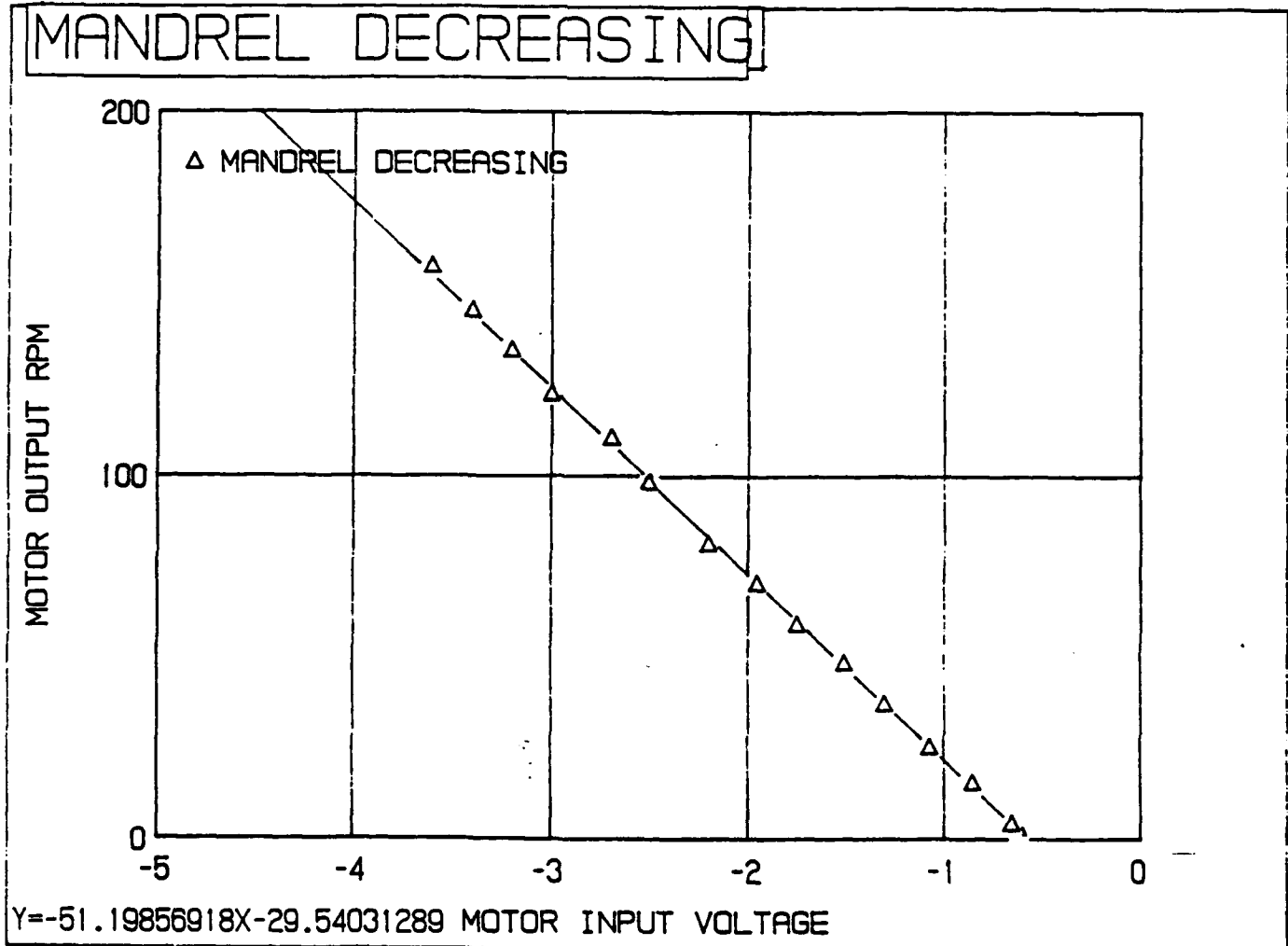
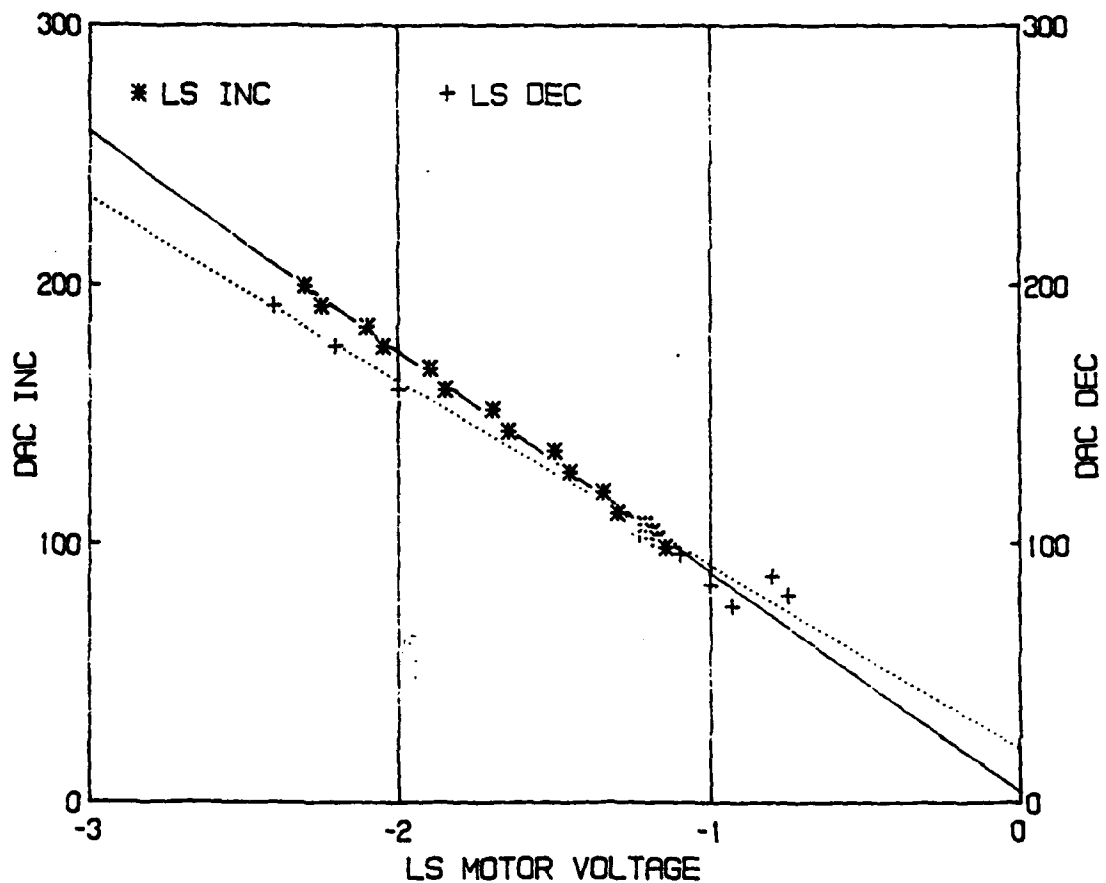


FIGURE 8.

LEAD SCREW DAC
OUTPUT VOLTAGE
VS.
DECIMAL EQUIVALENTS OF HEX COMMANDS

LS VOLTS vs DAC



$$\text{VOLTS} = -1.410443734\text{E-}2 \times \text{HEX} + 2.956092605\text{E-}1$$

FIGURE 9.

LEAD SCREW MOTOR
INPUT VOLTAGE
VS.
OUTPUT RPM
LEAD SCREW VOLTAGE INCREASING

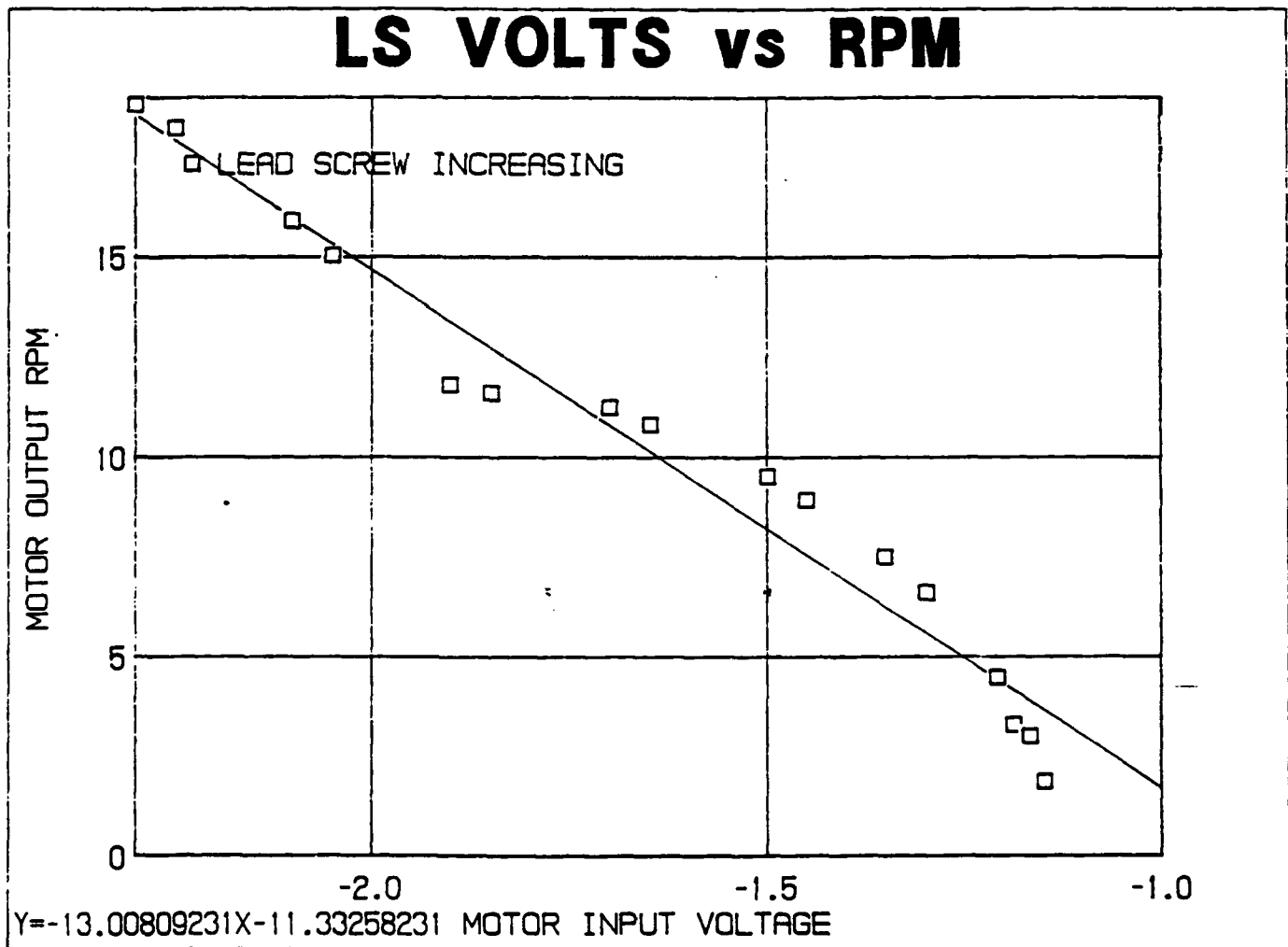


FIGURE 10.

LEAD SCREW MOTOR
INPUT VOLTAGE
VS.
OUTPUT RPM
LEAD SCREW VOLTAGE DECREASING

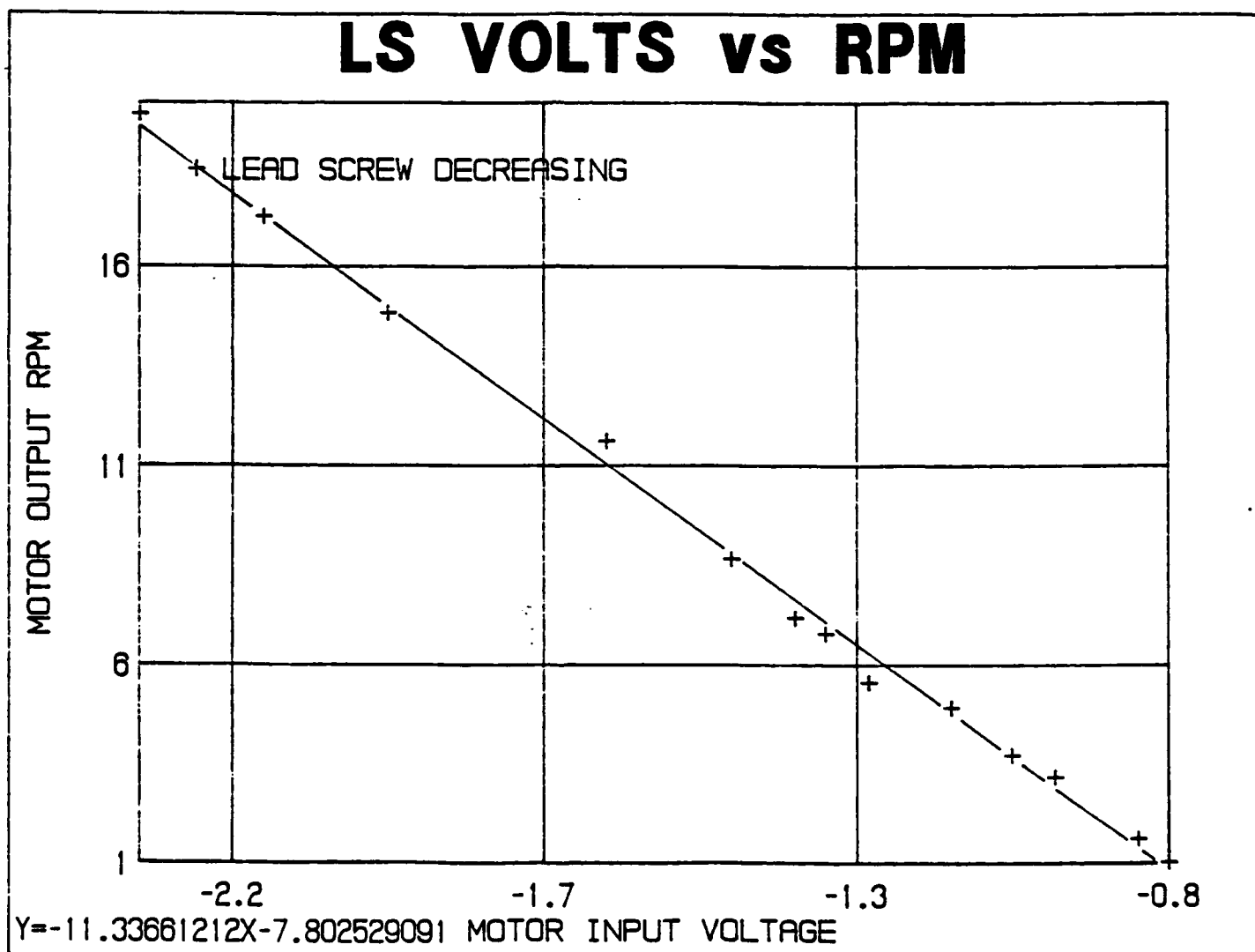


FIGURE 11.

averaged to yield the formulas used in the simulation.

The input voltages to the motors result in outputs of RPM, which are reported by the encoders. Therefore it was not necessary to have separate subroutines for the motors and the encoders. The simulation subroutines for these physical elements of the winding cell are as follows:

Physical Element	Subroutine
Mandrel DAC	MNDAC
Mandrel Encoder	MNENC
LS DAC	LSDAC
LS Encoder	LSENC

The formulas used to model these elements were as follows:

For the mandrel DAC:

$$VMN = 1.45 \times 10^{-2} \times MHEX - 3.62$$

Where:

VMN = Mandrel motor voltage in volts

MHEX = Decimal equivalent of the hexadecimal
mandrel motor command

For the mandrel encoder:

$$\text{RMRPM} = -51.37 \times \text{VMN} - 30.18$$

Where:

RMRPM = Rate of mandrel rotation in RPM

For the lead screw DAC:

$$\text{VLS} = -1.41 \times 10^{-2} \times \text{LHEX} + 2.96 \times 10^{-1}$$

Where:

VLS = Lead screw motor voltage in volts

LHEX = Decimal equivalent of the hexadecimal
lead screw motor command

For the lead screw encoder:

$$\text{RLRPM} = -12.17 \times \text{VLS} - 9.57$$

Where:

RLRPM = Rate of lead screw rotation in RPM

3.3.2 The Lag Angle Detector (LAD)

In order to simulate the lag angle detector (LAD) it was necessary to model variations in fiber diameter and LAD scale factor. The LAD scale factor is a ratio between the distance the bobbin moves laterally and the output voltage of the LAD. Or more accurately, it is a ratio between the LAD output voltage and the amount of lateral movement in the end of the fiber which contacts the bobbin. This value is predetermined in the controller software. The LAD scale factor has units of volts/inch. Therefore, the LAD output voltage is affected by the current fiber diameter. To simulate variances in the fiber diameter, a SIN function about the average diameter was used. The amplitude of the sine wave was equal to the variance in the fiber diameter. The sine wave had a variable period to give some randomness to the variations in fiber diameter. The formulas used to vary the fiber diameter was as follows:

$$AMP = (MXFD - MNFD) / 2$$

$$AVG = (MXFD + MNFD) / 2$$

$$FD = AMP \times \sin (TFUNC) + AVG$$

Where:

MXFD = The maximum fiber diameter

MNFD = The minimum fiber diameter

AMP = The amplitude of the variance function

AVG = The average fiber diameter (or mid point of
the function)

TFUNC = Angular position based on the speed of the
wind

FD = The current fiber diameter.

As the LAD scale factor is in volts/inch, the current value for the factor can be calculated by multiplying the detector gain by the current fiber diameter, where the wire wound detector output voltage is 0.57 volts/inch. The formulas used to simulate the LAD were as follows:

$$dD = DE - DA$$

$$dV = LDI \times dD \times SF$$

$$LADV = RV + dV$$

Where:

DE = The distance the bobbin should have traveled

DA = The distance the bobbin actually traveled

dD = The difference between DE and DA

LDI = The Lead Screw direction indicator (+1 or -1)
 SF = The LAD scale factor
 dV = The change in LAD voltage
 RV = The LAD reference voltage
 LADV = LAD output voltage

The simulation assumed that the slot in the LAD had a width which was very close to the average diameter of the fiber. Due to wear, the width was somewhat larger than the diameter of the fiber. This may cause a delay in the response of the LAD resulting in differences between simulation response and the actual response. This delay was not included in the simulation as periodic replacement of the worn LAD returns the system to its intended operating condition. The simulation subroutines for these physical elements of the winding cell are as follows:

Physical Element	Subroutine
Fiber diameter variances	FIBER
Lag Angle Detector	LAD
Scale Factor	FACTOR

3.3.3 Fiber Tension and the Tension Sensor

The tension in the fiber is primarily a function of winding

speed. As each layer of fiber has already been wound when adhesive is applied, it was determined that there is an insignificant effect on the tension in the fiber during a wind due to adhesive application. Any effects from adhesive application will occur during temperature curing of the bobbin and during payout. The analog tension control circuit will maintain the tension at a set level based on a reference voltage supplied by the controller software. As the response time of the analog tension control circuit is very rapid and momentary forces due to the mass of the supply spool are small compared to the amount of the torque supplied by the tension motor, it was determined that the effects of the supply spool mass could be disregarded for the purposes of this simulation. During development of this simulation, it was also determined that the fiber coefficient of elasticity would have an insignificant effect on the tension. Formulas were found which calculate the maximum allowable tension, given the coefficient of elasticity of a pure material. However, due to impurities and flaws in an actual spool of fiber, this limit was discovered to have of no practical value. The actual breaking tension could be as much as one, or possibly two, orders of magnitude less than the limit calculated. Therefore, the effects of the fiber coefficient of elasticity was also disregarded for the purposes of this

simulation.

Due to the voltage vs. torque properties of the tension motor, the tension in the fiber at any time may be found by:

$$FT = TMV \times TSF$$

Where:

FT = Fiber tension in grams

TMV = Tension motor voltage in volts

TSF = Tension scale factor in volts / gram

The tension scale factor varies depending on whether the system is in its static (ie: tension motor ramped, but mandrel stationary) or dynamic (ie: tension motor ramped and mandrel winding) mode. In its static mode, the tension is a function of the tension motor voltage alone. The formula for the tension in this mode is:

$$FT = TMV \times 500 \text{ grams/volt}$$

This formula comes from the Automatic Fiber Winder Operations Manual, page 26 (refer to Appendix G under separate cover). At the time of the development of this simulation, fiber was being wound at 133 RPM. The desired fiber tension at that

time was 115 grams. In order to achieve this tension, a tension voltage of 0.1848 volts was being used. This yielded the following formula for the dynamic tension:

$$FT = TMV \times 622 \text{ grams/volt}$$

The difference in the tension scale factor is due to back EMF in the tension motor. This variance in the tension scale factor is a function of the winding speed. Using the two data points of a scale factor equaling 500 for a winding speed of 0 RPM and a scale factor equaling 622 for a winding speed of 133 RPM (the only data available) the following formula for the scale factor was used:

$$TSF = 0.9173 \times \text{RPM} + 500 \text{ grams/volt}$$

The tension in the fiber is affected by the winding speed as seen by the fiber, ie: the speed of fiber travel through the tension system. As the bobbin rotates at a constant speed, any change in the apparent diameter of the bobbin will alter the speed of fiber travel. Two factors which can change the apparent diameter of the bobbin are the buildup of layers of fiber on the bobbin and the taper of the bobbin. The maximum amount of change in apparent bobbin diameter due to layer

buildup is very small, on the order of 1×10^{-4} to 1×10^{-5} of a bobbin diameter. Therefore, the effects of fiber diameter were disregarded in this simulation. The effects of the taper of the bobbin are considerable. As previously shown, fiber tension in the dynamic mode is directly related to bobbin rpm (winding speed). The bobbin rpm, as seen by the fiber, varies depending on the point of contact between the fiber and the bobbin. If the winding speed is set at the large end of the bobbin (the usual case) the apparent rpm will decrease as the fiber travels towards the small end of the bobbin. As the circumference of the bobbin decreases, less fiber is pulled from the supply reel per revolution. The variance in the bobbin diameter is due to the taper angle of the bobbin. This variance is therefore constant. The formulas for the bobbin diameter at any time during the wind are:

$$CR = \sin (TA) \times LW$$

$$ND = LD - 2 \times CR$$

Where:

CR = The change in bobbin radius

TA = The taper angle of the bobbin

LW = Present position of fiber along bobbin

ND = New bobbin diameter

LD = Diameter of the large end of the bobbin

As the taper of the bobbin is known, and the position of the fiber on the bobbin can be calculated; the apparent rpm can also be calculated. The formula for the rpm as seen by the fiber follows:

$$\text{ARPM} = \text{ORPM} \times (\text{LD} / \text{ND})$$

Where:

ARPM = Apparent bobbin rpm as seen by the fiber

ORPM = Original bobbin rpm set by winding
parameters (constant)

With the above calculations, the dynamic tension in the fiber can be calculated. The formula used for the dynamic tension follows:

$$\text{FTD} = \text{TMV} \times (0.9173 \times \text{ARPM} + 500)$$

Where:

FTD = The fiber tension, dynamic

The analog tension control circuit maintains a set tension on the fiber. Therefore, the analog circuit must vary the tension motor voltage to maintain this set tension. The analog circuit accomplishes this by either adding or subtracting a correction voltage to the reference voltage. This correction voltage can be calculated based upon the desired tension and the dynamic tension. The formula for the correction voltage follows:

$$CV = (DFT - FTD) \times TSF$$

Where:

DFT = The desired fiber tension as set by winding parameters

With the above calculations, the tension sensor and analog tension control circuit were modeled. The formula used for the analog tension control circuit follows:

$$NTMV = RTMV + CV$$

Where:

NTMV = The new tension motor voltage

RTMV = The reference tension motor voltage as specified in the initialization parameters

3.4 Linking Controller Software and Physical Subroutines

At this point in development, all of the modules of the physical system of the FOG-M WC had been simulated as subroutines to be called by the controller software. In the INTEL software, the actual values sent to, or received from, the physical system are passed by function calls to a special input/output card in the INTEL hardware. Each module of the physical system has its own memory location in that card. The controller software sends data to, or receives data from, any particular module by referencing its memory location in the function call. To simulate the I/O card, two subroutines were written (called INPUT and OUTPUT) which compare the values passed to them against a list of the locations assigned to the physical modules. When a match is found between a passed value and an assigned value, the I/O subroutines call the appropriate simulation subroutine.

3.5 Output of Winding Parameters

Finally, a subroutine was written which allows the operator to specify which of the winding parameters should be printed. The subroutine also allows the operator to specify how often, in terms of simulation time, he wishes these parameters to be

reported. This final step of connecting the controller software to the simulation routines and providing for parameter output completed development of the simulation of the FOG-M WC.

4.0 TESTING AND EVALUATION

The simulation accepts inputs from, and interacts with, the operator in the same fashion as the FOG-M WC. Diagnostic print statements were embedded in the simulation software to allow the developer to monitor the winding parameters. Actual winding data obtained from Army sources was compared against data produced by the simulation. The controlling software manipulates the inputs and calculates controlling values which are the same as those produced by the INTEL controlling software. After the development of the simulation software was evaluated, these diagnostic print statements were removed from the code. The subroutines accept inputs from the controlling software, model the actual physical operations of the winding cell, and send values back to the controlling software which are the same as if the software was monitoring the real winding cell. The controller software interfaces with the simulation subroutines in the same manner as the INTEL system interfaces

with the physical system. That is, the controller software sends commands and accepts inputs from the simulated winding cell in the same way as it would perform with the physical winding cell. Operator control of the reporting of winding parameters is provided.

5.0 CONCLUSION

In conclusion, the purpose of the task was fulfilled in that the simulation accurately modeled the functions and operations of the FOG-M WC and allowed the operator to monitor the winding parameters. As mentioned previously in section 3.0 (Simulation Development), it was determined that the integrated, "real time", IBM PC based workstation was not feasible for the phase one simulation effort.

APPENDIX A

FOG-M AUTOMATIC FIBER WINDER SIMULATION OPERATIONS MANUAL

1.0 INTRODUCTION

The purpose of this manual is to provide detailed instructions for the operation of the FOG-M Automatic Fiber Winder Simulation. The simulation operates the same as the actual FOG-M Automatic Fiber Winder. For detailed instructions on the operations of the FOG-M Automatic Fiber Winder, refer to the Automatic Fiber Winder Operation Manual, U. S. Army Missile Command, Research, Development and Engineering Center, Guidance and Control Directorate, 1986, which appears in Appendix G, under separate cover. Added input of initial winding parameters and the ability to output certain winding parameters during simulation operation are the only differences between the operations of the simulation and those of the fiber winder.

2.0 SIMULATION OPERATIONS

The simulation program is on the Army VAX 11/750 CAD/CAM

System. This manual assumes that the operator is familiar with logon procedures, running programs, and extracting data from files in the VAX 11/750. This manual also assumes that the operator is familiar with all of the steps necessary to operate the actual automatic fiber winder.

The operation of the FOG-M Automatic Fiber Winder Simulation can be broken down into three steps: initializing the simulation, running the simulation, and extracting the simulation data.

2.1 Simulation Initialization

Step 1 - Log on to the VAX 11/750 under the proper directory.

Step 2 - Type @FOGSIM

The program will perform some of the initialization steps and the Main System Menu will appear. This menu is the same as the one which appears on the terminal of the fiber winder.

Step 3 - If an initialization parameters file has not been created, select menu choice 3 ("Coefficient check/change/enter"). This is also the easiest way to create an initialization file. The default name for the initialization parameters file is SIMIN.DAT.

The Coefficient subroutine will operate the same as on the actual fiber winder, with the following exceptions: the program asks if the output should be sent to the screen or to a file, and several parameters have been added. If the output is to be sent to a file, the program prompts for an output file name, the default is **SIMOUT.DAT**. The inputs for the initialization parameters have changed as follows:

Parameters 1 through 3 remain unchanged.

Parameters 4 through 38 have become 11 through 45, but are otherwise unchanged.

Parameters 4 through 10 are now as follows:

- 4 fib-dia-max-r: the maximum fiber diameter, in inches, as it is given on the manufacturer's specification sheet. (REAL)

- 5 fib-dia-min-r: the minimum fiber diameter, in inches, as it is given on the manufacturer's specification sheet. (REAL)

- 6 fib-dia-o/s end-r: the diameter of the outside
 end of the fiber, in inches, as it is given
 on the manufacturer's specification sheet.
 (REAL)
- 7 spool-dia-large-end: the diameter of the large
 end of the bobbin, in inches. (REAL)
- 8 spool-taper angle: the taper angle of the
 bobbin, in inches.
- 9 mean-fib-dia-r: the average fiber diameter,
 in inches, as it is given on the
 manufacturer's specification sheet. (REAL)
- 10 spool-wind area len: the length of the area on
 the spool, in inches, onto which the fiber
 will be wound. (REAL)

Step 3 - If a new initialization file is being created, be sure to save it when prompted. The operator can then either supply his own name to the initialization parameters file or use the default name (SIMIN.DAT).

The initialization phase is now complete.

2.2 Running the Simulation

After the initialization phase is completed, the Main System Menu will reappear. Any choice on the menu may now be selected. To simulate an automatic wind, select choice 6 ("Automatic Wind"). If screen output was chosen in the initialization phase, simulation data will begin appearing on the screen. At this point termination of the program can only be achieved by typing CTRL-C. This will terminate the entire simulation and all steps must be performed again, except for the creation of an initialization file. If output to a file was chosen the only outputs to the screen will be those normally seen during an automatic winding session. The simulation will terminate at the end of the wind and return to the Main System Menu. If another simulation is not desired, select menu choice 8 ("Exit Program"). and the program will terminate.

2.3 Extracting Simulation Data

The output parameters will now be in the file specified, the default is SIMOUT.DAT. The data may be viewed on the screen

or sent to a printer through the normal VAX 11/750 DCL commands.

APPENDIX B

VARIABLES OF THE TRANSLATED INTEL CODE

VARIABLE	DESCRIPTION
kcaw	Status byte of DAC
meaw	Address for Mandrel Encoder data
leaw	Address for LS Encoder data
ipaw	DIGITAL INPUT
illaaw	ADC Analog input: Lag > Voltage
i28aaw	Address for Tension Control Command
ldacaw	Address for LS Motor Voltage Command
mdaw	Address for Mandrel Motor Voltage Command
ldiraw	Address for LS Direction Flag
kopaw	DIGITAL OUTPUT
adjfac	Turns ratio adjustment factor(CNTRL2) (errsig*zlcgr)
afr	Average fiber diameter(inches)[P3]
ahalf	{float(ihalf)} (I28DP)
alaref	Supervised stepback flag 1=yes, 0=no
alhr	Average fiber diameter high limit(in)[limit(in)[P1]
allr	Average fiber diameter low limit(in)[Plimit(in)[P2]
avr	Average of lag > voltages
colale	Left(?) end ref voltage for crossover(pullback)
colare	Rt(?) end ref voltage for crossover(pullback)
curlag	Current lag > voltage[P32]
delvsb	Delta lag sensor voltage. Ref to drive LS to expected perpendicular position after stepback
dlar	LS > of rotation
dmar	Mandrel > of rotation
dum1	Temp storage for {zlar} (ladsf)
dum2	Temp storage for {zltr} (ladsf)

VARIABLE	DESCRIPTION
dum3	Temp storage for {zmar} (ladsf)
dum4	Temp storage for {zmrar} (ladsf)
dum5	Temp storage for {zmtr} (ladsf)
error	Lag distance error((zldvr-zldrvr)*1000.)(cntrl1)
errsig	Lag error voltage {(zldvr-zldrvr)*zldr}(cntrl2,3)
fddr	Fiber diameter delta value for cntrl1&2
fdrnr	Fiber diameter resolution # [P5]
filnam	Filnam for default program parameters
flr	{zlrvr} (ziap)
head1	Header for printout
head2	Header for printout
head3	Header for printout
i	General loop counter
ibythi	Hi byte for transfer to DAC {int2(iword/16)}
ibytlo	Lo byte for transfer to DAC {int2(iword.and.'1111'B)*16}
icb	Pass value for cntrl1(int(lbb)){lcb}
icount	Exit control if problems w/ziap
icur	Initialization parameters
icyc	Process cycle counter [P30]
icycle	Process cycle counter
ideflt	Default initialization parameters
idel	Pass value for delay timer
idelay	Pass value for delay timer
idum	Pass value for ilbyte
ierre	Count of apparent encoder errors so far [P27]
ierren	Count of encoder errors (ierre)
ifci	Pass Value (lfci)
ifile	Flag if o/p file to be written
ifmi	Loop limiter (int(lffi))
ifull	('fff'X)
ih	Character array used for o/p

VARIABLE	DESCRIPTION
ihalf	{'800'X} {'01'X} Lower limit check?
ii	Used in printing tension profile {i+20}
iili	{1} Upper limit check?
iin	1 byte integer used for ADC reads
ildvb	Initial LS DAC value [P6]
ili	Loop control
imdvb	Initial Mandrel DAC value [P7]
imod	Flag to indicate user interrupt 0=no [P28]
imode	Menu choice {imod} interrupt flag
inbyte	O/P of ilbyte ({int2(idum)})
intout	O/P block value holder
iout	1 byte integer used for DAC writes
iovrld	Initialization parameters
ip	Counter (module ipci) of process cycle [P29]
ipci	Cycle # indicator for updating process control w/lag sensor reading [P8]
ipi	Loop controller
irhflg	Flags array
irowe	Loop control for printing e=end
irows	Loop control for printing s=start
istat	Status of DAC
item	Print parameter
itype	Type of control method {lscont}
iwrd	I/P word for tension voltage
ilyn	'Yes/No' answer holder
jdum	I/P variable for jbytil
jtem	Print parameter
kai	Kaw w/ status bits shifted out(int(kaw/16))
kaw	Hi & lo bytes of ADC read merged
kdtw	Process cycle delay time for high speed wind[P4]
kdummy	Dummy read to set up ADC

VARIABLE	DESCRIPTION
kemb	Used in test to see if conversion done {1}
khb	Hi byte of ADC read
kkdtw	Timer amount {int(kdtw)}
kkrtw	Timer amount {krtw}
klay	Counter, for current layer of wind [P25]
knterr	Count of encoder errors
kntlat	# of layers completed
koi	{'800'X}
krtw	Ramp time delay [P23]
lb	Stores value {ilbyte(iin)}
lbb	Pass value
lcb	Lag-ch-b [P9]
ldb	Command word indicating LS direction [P15] ldb2 {int2(ldb)}
ldbsv	Temporary holder {ldb}
ldi	Flag for direction of LS travel for 1st layer -1=toward motor +1=toward encoder [P16]
ldvb	{int(lnom-ldi*lout)}
ldvbsb	LS speed command used to position LS during stepback
ldvbsv	Last LS speed command
leob	Holder for LS encoder data
lfci	# of consecutive lag sensor readings to take to get average Lag filter Cycles [P10]
lffi	I/P variable
lnom	Nominal LS command based on man command & turns ratio
lout	Rounded filter O/P {int(zoutr)}
lscont	Type of LS control [P33]
mdvb	Mandrel command during ramp up
mdvbs	Temporary holder for mdvb

VARIABLE	DESCRIPTION
mdvbsb	Mandrel speed during stepback
meob	Holder for mandrel encoder data
mmdvb	Nominal mandrel command [P20]
mmdvbs	Temporary holder for value of mmdvb
nbrlay	# of layers desired for this wind
nbrpar	# of parameters (38)
nbrrow	Loop limiter for print
nbrstp	Loop limiter. # of steps for tension ramp up
nlay	# of desired layers for wind [P26]
noparm	# of parameter to change
plear	Current LS < (float(leob)*360./256.)
pmear	Current MAN< (float(meob)*360./256.)
ratio	Check for encoder errors (abs(dmar)+1.1)/(abs(dlar)+1.)
rcur	Initialization parameters
rdeflt	Initialization parameters
reflag	Lag-det-ref-val [P31]
relout	Array for O/P block
rltr	# base ref-layer-turns [P24]
rovrid	Initialization parameters
sbcont	Type stepback control 0.=no pause [P34]
sfld	LAD-scale-factor [P38] (abs((vend-vstrt)/tnsman))
sfls	Lag sensor0-to-lag turns scale factor
splear	Temporary holder for plear
spmear	Temporary holder for pmear
svr	Sum of lag angle voltages
tnscur	Default tension voltage per layer
tnsinc	Default tension voltage increment per layer
tnslay	Array containing tension for EACH layer
tnsmac	Observed mandrel turns

VARIABLE	DESCRIPTION
tnsman	Desired # of mandrel turns for calibration of lag angle sensor
tnsmax	Maximum tension voltage
tnsmer	Desired # of encoder read mandrel turns for calibration
tnstem	'Old' tension voltage for ramping
tnsvol	Tension voltage during ramp up. I/P to i28dp tr0 {afdr/0.1}
trail1	Footer for print
trail2	Footer for print
trnco	# of mandrel turns for crossover
trnlag	# lag turns desired [P35]
trnlr	# lag turns for Manrev
trnpb	# pullback turns
trnpul	# pullback turns desired [P36]
trnref	{zmtr}
trnsb	# of stepback turns
trnstp	# stepback turns desired [P37]
trr	Turns ratio (trr-adjfac)
turns	I/P to manrev (trnlr)
type	Type of # (int, real, hex)
v0	Perpen voltage point for lag sensor
varnam	Array of variable names
vend	Ending voltage in LAD calibration
vpbel	Lag sensor pull back voltage for even layers
vpbol	Lag sensor pull back voltage for odd layers
vperp	Target lag sensor voltage for lead of 'trnsb' turns
vr	Current lag < voltage for summing
vr1	Ref lag voltage for odd layers
vr2	Ref lag voltage for even layers
vsb	Target lag > voltage for stepback

VARIABLE	DESCRIPTION
vstrt	Starting voltage in LAD calibration
x	Dummy call to ziap
zilar	Current LS < of rot {float(leob)*360./256.}
zimar	Current MAN < of rot {float(meob)*360./256.}
zlar	Total LS > of rotation [P13]
zlcgr	LS computer gain [P14]
zldr	LS direction {float(ldi)}
zldrvr	Reference lag > voltage for present layer
zldvr	Present lag angle voltage
zlear	Current LS < of rot {float(leob)*360./256.}
zlesr	1-LS error{zlrrar-zltr} 2-LS command {zlear*3600.} (error*gain) 3-LS error {(zlrrar*znr-zltr)*3600.}
zlr	Limit [P12]
zlrrar	Cumulative > of rotation for LS
zlrvr	Real # for status checking of # of turns in wind layer [P11]
zltr	Current # of LS turns made [P17]
zmar	Total man < of rot [P18]
zmear	Current Man < of rotation {float(meob)*360./256.}
zmeec	Man encoder correction value {zmeecr}
zmeecr	Correction factor between mandrel encoder revs & true mandrel revs man-enc-err-corr {tnsmac/tnsmer} [P19]
zmr	
zmrar	Corrected cumulative > of rotation of mandrel for current layer [P21]
zmtr	Current # of motor/encoder turns for mandrel [P22]

VARIABLE**DESCRIPTION****znr****zoutr****zr****ztim****LS direction (zldr)**

APPENDIX C

SUBROUTINES OF THE TRANSLATED INTEL CODE

SUBROUTINE	DESCRIPTION
AFW003	Main controller program module
CALIB	Calibrates LAD
CALSYS	Allows calibration of scale & correlation factors
CNTRL1	Type 1 LS control routine
CNTRL2	Type 2 LS control routine
CNTRL3	Type 3 LS control routine
FILRED	Reads I/P file of initial variable values
FILRIT	Writes O/P file of initial variable values
I1BYTE	Converts INT1 to INT2 w/ same bit pattern
I28DP	Converts tension voltage O/P
I28P	Initializes tension DAC
INITIN	Gets initial variable values
JBYT11	Converts INT2 to INT1 w/ same bit pattern
LADSF	Calculates LAD scale factor
LMSPD	Not used
MANREV	Revolves Mandrel user specified amount
MEMO	Not used
MENU	Gives process choices
MERR	Calibrates Mandrel Error Correction value
PEP	Converts Mandrel & LS encoder data
REVLS	Sets parameters to reverse the LS
STPBAK	Performs Stepback maneuver
STRUP	Calls starting routines
SYSINI	Initializes variables
TENCAL	Not used
TENCOR	Not used
TENMOT	Allows "offline" control of tension, man, & LS
TNSPRO	Establishes tension-layer profile
ZIAP	Converts LAD I/P

APPENDIX D

Specific Variables in Each INTEL Subroutine

VARIABLE NAME	SUBROUTINE																									
	S	M	Z	P	C	I	R	C	C	C	I	I	M	S	T	F	A	C	L	F	T	I	J	M		
	Y	E	I	E	A	N	E	N	N	N	2	2	A	T	N	I	F	A	A	I	E	1	B	E		
	S	N	A	P	L	I	V	T	T	T	8	8	N	P	S	L	W	L	D	L	N	B	Y	R		
	I	U	P		I	T	L	R	R	R	P	D	R	B	P	R	O	S	S	R	M	Y	T	R		
	N				B	I	S	L	L	L		P	E	A	R	E	O	Y	F	I	O	T	I			
	I				N		1	2	3				V	K	O	D	3	S		T	T	E	1			
kaw	X																X									
meaw	X			X									X													
leaw	X			X									X													
ipaw	X																	X								
illaaw	X		X															X								
i28aaw	X											X	X					X								
ldacaw	X						X	X	X					X				X				X				
mdaw	X												X	X				X				X				
ldiraw	X					X												X				X				
kopaw	X																	X								
adjfac							X																			
afdr																	X	X				X				
ahalf												X														
alaref														X			X									
alhr	X																X					X				
allr																	X					X				
avr			X																							
colale					X									X			X									
colare					X									X			X									
curlag	X																X					X				
delvsb					X												X									
dlar				X																						
dmar				X																						
dum1																										
dum2																										
dum3																										
dum4																										
dum5																										

	SUBROUTINE																										
	S	M	Z	P	C	I	R	C	C	C	I	I	M	S	T	F	A	C	L	F	T	I	J	M			
	Y	E	I	E	A	N	E	N	N	N	2	2	A	T	N	I	F	A	A	I	E	1	B	E			
	S	N	A	P	L	I	V	T	T	T	8	8	N	P	S	L	W	L	D	L	N	B	Y	R			
	I	U	P		I	T	L	R	R	R	P	D	R	B	P	R	O	S	S	R	M	Y	T	R			
	N				B	I	S	L	L	L		P	E	A	R	E	O	Y	F	I	O	T	I				
	I				N		1	2	3				V	K	O	D	3	S		T	T	E	1				
VAR NAME																											
error																											
errsig	X																										
fddr	X X																										
fdrnr	X X X																										
filnam	X X																										
flr	X																										
head1	X																										
head2	X																										
head3	X																										
i	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	
ibythi													X														
ibytlo													X														
icb	X																										
icount	X																										
icur	X																										
icyc	X	X X X																									
icycle	X																										
ideflt	X																										
idel	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	?	
idelay	X																										
idum	X																										
ierre	X X																										
ierren	X X X																										
ifci	X																										
ifile	X																										
ifmi	X																										
ifull	X																										
ih	X																										
ihalf	X																										
ii	X X																										
iili	X																										
iin	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	?	

VAR NAME	SUBROUTINE																									
	S	M	Z	P	C	I	R	C	C	C	I	I	M	S	T	F	A	C	L	F	T	I	J	M		
	Y	E	I	E	A	N	E	N	N	N	2	2	A	T	N	I	F	A	A	I	E	1	B	E		
	S	N	A	P	L	V	T	T	T	8	8	N	P	S	L	W	L	D	L	N	B	Y	R			
	I	U	P		I	T	L	R	R	R	P	D	R	B	P	R	O	S	S	R	M	Y	T	R		
	N				B	I	S	L	L	L		P	E	A	R	E	O	Y	F	I	O	T	I			
	I					N		1	2	3			V	K	O	D	3	S		T	T	E	1			
ildvb																X	X			X	X					
ili				X																						
imdvb																X	X			X	X					
imod						X										X	X			X						
imode		X	X	X		X										X	X	X		X						
inbyte											X											X				
intout																	X									
iout	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	?
iovrid						X																				
ip						X										X	X			X						
ipci									X							X				X						
ipi						X			X								X									
irowe						X																				
irows						X																				
istat										X	X															
item				X																						
itype																	X									
iword											X															
iyndum				X	X								X	X		X	X	X		X			X			
jtem						X																				
kai				X																						
kaw				X																						
kdtw													X	X		X	X			X						
kdummy	X																									
kemb				X																						
knb				X																						
kkdtw												X	X			X										
kkrtw																X				X						
klay						X										X	X			X						
knterr				X																						
kntlay																	X									

VAR NAME	SUBROUTINE																							
	S	M	Z	P	C	I	R	C	C	C	I	I	M	S	T	F	A	C	L	F	T	I	J	M
	Y	E	I	E	A	N	E	N	N	N	2	2	A	T	N	I	F	A	A	I	E	1	B	E
	S	N	A	P	L	I	V	T	T	T	8	8	N	P	S	L	W	L	D	L	N	B	Y	R
	I	U	P		I	T	L	R	R	R	P	D	R	B	P	R	O	S	S	R	M	Y	T	R
	N				B	I	S	L	L	L		P	E	A	R	E	O	Y	F	I	O	T	I	
	I				N		1	2	3				V	K	O	D	3	S		T	T	E	1	
koi				X																				
krtw																	X	X			X			
lb				X																				
lbb				X																				
lcb					X									X		X	X			X				
ldb						X										X	X			X	X			
ldb2							X										X				X			
ldbsv							X																	
ldi						X	X	X								X	X			X				
ldvb								X	X	X							X							
ldvbsb				X				X					X		X					X				
ldvbsv								X	X	X														
leob				X									X				X							
lfci					X									X		X	X			X				
lffi				X																				
lhex																								
lnom								X																
lout								X									X							
lscont						X											X	X			X			
madr																								
madres																								
madrind																								
mdvb				X			X					X	X				X							
mdvbs					X								X	X										
mdvbsb														X		X					X			
meob				X									X	X			X							
mhex																								
mmdvb				X									X	X		X	X			X				
mmdvbs													X	X										
nbrlay						X										X		X						
nbrpar						X																		
nbrrow																X								

VAR NAME	SUBROUTINE																									
	S	M	Z	P	C	I	R	C	C	C	I	I	M	S	T	F	A	C	L	F	T	I	J	M		
	Y	E	I	E	A	N	E	N	N	N	2	2	A	T	N	I	F	A	A	I	E	1	B	E		
	S	N	A	P	L	I	V	T	T	T	8	8	N	P	S	L	W	L	D	L	N	B	Y	R		
	I	U	P		I	T	L	R	R	R	P	D	R	B	P	R	O	S	S	R	M	Y	T	R		
	N				B	I	S	L	L	L		P	E	A	R	E	O	Y	F	I	O	T	I			
	I				N		1	2	3				V	K	O	D	3	S		T	T	E	1			
nbrstp	X														X					X				X		
nlay						X										X	X			X						
noparm						X																				
plear				X								X					X									
pmear				X								X	X				X									
ratio				X																						
rcur						X																				
rdeflt						X																				
reflag					X											X	X			X						
relout																	X									
rltr					X											X	X			X						
rovrid					X																					
sbcont					X											X	X			X						
sfld																X	X	X	X	X						
sfls					X							X				X	X									
shmv																										
shrt																										
shrtrpm																										
splear					X																					
spmear					X																					
spmv																										
sprt																										
sprtrpm																										
svr																										
tidel																										
tnscur	X															X										
tnsinc	X															X										
tnslay	X															X										
tnsmac																								X		
tnsman																			X							
tnsmax	X															X										
tnsmer																									X	

VAR NAME	SUBROUTINE																									
	S	M	Z	P	C	I	R	C	C	C	I	I	M	S	T	F	A	C	L	F	T	I	J	M		
	Y	E	I	E	A	N	E	N	N	N	2	2	A	T	N	I	F	A	A	I	E	1	B	E		
	S	N	A	P	L	I	V	T	T	T	8	8	N	P	S	L	W	L	D	L	N	B	Y	R		
	I	U	P		I	T	L	R	R	R	P	D	R	B	P	R	O	S	S	R	M	Y	T	R		
	N				B	I	S	L	L	L		P	E	A	R	E	O	Y	F	T	O	T	I			
	I				N		1	2	3			V	K	O	D	3	S		T	T	E	1				
tnstem																					X					
tnsvol	X											X		X		X		X		X				X		
tr0																	X									
trail1						X																				
trail2						X																				
trnco					X												X				X					
trnlag					X												X	X			X					
trnlr					X									X			X									
trnpb					X												X									
trnpul					X												X	X			X					
trnref													X													
trnsb					X								X				X									
trnstp						X											X	X			X					
trr								X	X	X							X									
tshrt																										
tsprt																										
ttshrt																										
ttsprt																										
turns					X								X													
type						X																				
v0					X									X		X										
value																										
varnam						X																				
vend																								X		
volts1																										
voltsm																										
vpbel					X														X							
vpbol					X														X							
vperp														X												
vr				X																						
vr1					X															X						
vr2					X															X						

VAR NAME	SUBROUTINE																														
	S	M	Z	P	C	I	R	C	C	C	I	I	M	S	T	F	A	C	L	F	T	I	J	M							
	Y	E	I	E	A	N	E	N	N	N	2	2	A	T	N	I	F	A	A	I	E	1	B	E							
	S	N	A	P	L	I	V	T	T	T	8	8	N	P	S	L	W	L	D	L	N	B	Y	R							
	I	U	P		I	T	L	R	R	R	P	D	R	B	P	R	O	S	S	R	M	Y	T	R							
	N				B	I	S	L	L	L		P	E	A	R	E	O	Y	F	I	O	T	I								
	I				N		1	2	3			V	K	O	D	3	S		T	T	E	1									
vsb														X																	
vstrt															X																
x															X																
zilar															X																
zimar														X	X																
zlar			X		X												X	X		X	X										
zlcgr				X		X	X	X	X							X				X											
zldr			X	X		X	X	X	X						X		X														
zldrvr														X	X	X		X													
zldvr														X	X	X		X													
zlear			X												X																
zlesr															X	X															
zlr															X											X					
zlrrar														X	X																
zlrvr		X												X											X						
zltr			X		X		X	X	X							X	X		X	X											
zmar			X		X												X	X		X	X										
zmear			X																												
zmeec															X																
zmeecr			X												X		X						X								
zmr														X	X											X					
zmrar		X		X												X	X	X		X											
zmtr		X		X		X	X	X						X	X		X	X		X	X										
znr														X	X											X					
zoutr														X																	
zr															X																
ztim																															

APPENDIX E

SUBROUTINES OF THE SIMULATION CODE

SUBROUTINE	DESCRIPTION
ANCIRC	Simulates Analog tension control circuit
DATOUT	Outputs desired winding parameters
DTIMER	Simulates process times
FACTOR	Calculates actual LAD scale factor
FIBER	Calculates current fiber diameter
FIBTEN	Simulates tension in the fiber
INPUT	Simulates digital I/P channels
INT4	Dummy routine
LAD	Simulates Lag Angle Detector
LSDAC	Simulates Lead Screw Digital to Analog Card
LSENC	Simulates Lead Screw Encoder
MNDAC	Simulates Mandrel Digital to Analog Card
MNENC	Simulates Mandrel Encoder
OUTPUT	Simulates digital O/P channels
SPOOLDIA	Calculates current spool diameter
TNSMOT	Simulates tension motor
TIMER	Provides delay times