General Disclaimer

One or more of the Following Statements may affect this Document

- This document has been reproduced from the best copy furnished by the organizational source. It is being released in the interest of making available as much information as possible.

- This document may contain data, which exceeds the sheet parameters. It was furnished in this condition by the organizational source and is the best copy available.

- This document may contain tone-on-tone or color graphs, charts and/or pictures, which have been reproduced in black and white.

- This document is paginated as submitted by the original source.

- Portions of this document are not fully legible due to the historical nature of some of the material. However, it is the best reproduction available from the original submission.

Produced by the NASA Center for Aerospace Information (CASI)

T. L. Olsen
W. F. Ng
W. F. O'Brien, Jr.

Mechanical Engineering
Virginia Polytechnic Institute and State University
Blacksburg, Virginia 24061

Interim Report #1
for the period June 16, 1984 to March 15, 1985

Prepared for:

National Aeronautics and Space Administration
NASA Grant Number NAG-1-486
James A. Schoenster, Technical Monitor
NASA-Langley Research Center

This is an informal interim report recommended at this time for internal use only. The contents will be included in a formal report suitable for public release or distribution at a later time.

April 1985
# TABLE OF CONTENTS

I. Introduction and Research Objectives ........................................... 3

II. Data Transfer from NASA Langley Research Center to VPI&SU .......... 6

III. Other Accomplishments .............................................................. 10

   A. Presentation and Publication ........................................... 10
   B. Improvement in Analytical Code ......................................... 10
   C. Data Reduction ............................................................... 10
   D. Interactions with Industry ............................................... 12

IV. Summary and Future Plan ........................................................... 23

V. References .................................................................................. 25

Appendix: Listing of programs used in data transfer ......................... 26
I. Introduction and Research Objectives

An important class of turbomachinery noise is produced by the interaction of the blades of a machine with flow perturbations within the machine. Such flow perturbations may be stationary in time and position, as would result from the potential field of a downstream strut. Disturbances originating from struts exist continually during operation of a machine, and will contribute to tone noise depending on the strength of the consequent flow response and pressure fluctuations.

The VPI&SU Turbomachinery Research Group had for some years carried on an investigation which was designed to provide insight into the fundamental aspects of fan rotor-downstream strut interaction (1,2,3). High response, miniature pressure transducers were embedded in the rotor blades of an experimental fan rig. Five downstream struts were placed at several downstream locations in the discharge flow annulus of the single-stage machine (Fig. 1). Significant interaction of the rotor blade surface pressures with the flow disturbance produced by the downstream struts was measured. Several numerical procedures for calculating the quasi-steady rotor response due to downstream flow obstructions were developed (3). A preliminary comparison of experimental and calculated fluctuating blade pressures on the rotor blades shows general agreement between the experimental and calculated values. Although progress has been made as a result of this effort, there are still areas in which further work is needed.
The objective of the program is to investigate the interaction of fan rotor flow with downstream struts. The results of this investigation will be an improved understanding of noise prediction with the goal of developing a design-for-noise capability and development of a data base for use in validating analytical prediction methods and noise-reduction concepts. The experimental program will focus on further data reduction of the existing data base from previous work.

The signal from the blade mounted pressure transducers will be used to calculate the unsteady lift and moment coefficients. In addition, power spectral analysis on the time-averaged data will be performed to retain the phase information. Both of these analyses will be used to validate the time-marching code, which was written to obtain the rotor blade flow response to a downstream strut row.

The present report gives a summary of the work for the period 6/16/84 – 3/15/85. Major topics to be covered in this report include the data transfer from NASA Langley Research Center to VPI&SU, further data reduction and the calculation of unsteady lift and moment. The report will conclude with a discussion of future plans.
Fig. 1 Cross section of compressor showing rotor-stator-strut spacing
II. Data Transfer NASA Langley Research Center to VPI&SU

This section describes in detail the transfer of analog data recorded using FM tape recorder to digital data that can be reduced using computers at VPI&SU. A listing of the programs is provided in the Appendix.

1. Digitization of Analog Data

The original analog data were recorded on half-inch tape running at a speed of 15 inches per second (ips). The data were digitized in the Central Data Transcription Facility at NASA Langley Research Center, Analysis and Computation Division, Computer Management Branch, Data Management Section. The data were digitized with an anti-aliasing filter set at 5 kHz flat. The digitization gave 360 points per cycle of rotor revolution. Since the rotor rotational speed fluctuates with time, the corresponding data sample rate will have to be adjusted to account for this. The time increment between data point varies between 50 and 70 microseconds. The once per revolution signal is used to determine the sampling rate to give 360 points per rotation. The digitized data were written on magnetic tape using the UNIVAX VARIAN Model 77-600 computer. The specifications on the format of the data written on the magnetic tape include the following: Nine tracks; Non-labeled; Edecie; 1600 BPI; Fixed block binary; 16 Bit Integer Words and Two's compliment (Integer *2). The characteristics of the digitized data file on magnetic tape include the following: Blocksize 3840, Record length 3840 and Record format F.
2. Transfer of digitized data to computer at VPI&SU

The digitized data is transferred to an IBM 3081 system. The system operates under the control of VM (Virtual Machine) and OS/VS 2.3 (MVS) and has twenty-four (24) megabytes of memory. A series of software programs were developed to facilitate the data transfer. Figure 2 shows the flow chart to read the data off the magnetic tape. The first program, TPP CNTL, executes a system program (TPPRINT) which checks the tape characteristics. The program TPPRINT scans a tape and tells the number of files, the block storage size, and the number of records per block (thus record length). Record length is equal to blocksize/(number of records per block). TPPRINT also gives a machine language translation of the stored data, which is not readily readable.

Next step in the data transfer is to copy NASA's tape to a VPI system tape. The VPI system tape is permanently active, easier to access and is in readable format. Program TAPE TAPE is written to create the system tape.

Two programs were written to read from the system tape. TAPE LISTE shows the first 20 elements of every line. This is not a necessary operation, but allows a visual look at a swath of the whole file from top to bottom. The second program, TAPE LISTL, shows the first few lines of the file in totality. This allows viewing the arrangement of data on each line. Some prior knowledge of the data expected aids in selecting the format used in TAPE LISTL. (See appendix for a sample output from the program TAPE LISTL).

As mentioned earlier, the time increment between data point varies between 50 and 70 microseconds. The rotor test speed of 2900 RPM fluc-
tuates. The analog data were digitized by 1) assigning a square wave pulse to match the once-per-rev signal, 2) dividing each pulse into 360 units, and 3) making a digital reading of each channel at each of 360 cycle positions in every revolution. Thus $\Delta \text{Time} = \Delta \text{Position} / \text{rotor speed}$. Since the rotor speed fluctuates, the time increment fluctuates and thus cannot correlate exactly with position. It is decided that the data will be analyzed in the position domain.

Next, the square wave must be checked to see that 360 readings occur in each cycle. The file POS CHECK searches for the drop in each square wave ($6000^+ \text{ followed by } 600^+$) and counts the number of entries since the preceding square wave. It then shows the number of cycle positions in each revolution; some truncation may occur and some cycles show 359 or 361 positions instead of 360. The approach taken here is to discard the bad cycles and the program CYCLE SELECT is written to do that. The program also ignores the first incomplete cycle and provides a printout of data points retained. A similar file, VOLT PLOT, performs the same data manipulation and then plots each channel, the square wave, and the once-per-rev signal versus cycle position for three good cycles.
TPP CNTL:
check tape
characteristics

TAPE TAPE:
transfer NASA's tape
to VPI's system tape

TAPE LISTE:
print first 20 elements
of every line

TAPE LISTL:
print first few lines of
the file in totality

POS CHECK:
check number of data
point in each cycle

CYCLE SELECT:
discard cycles where
N ≠ 360

VOLT PLOT:
plot each channel, the square
wave, and once-per-rev
signals against position

Fig. 2 Block diagram showing programs used in data transfer
III. Other Accomplishments

A. Presentation and Publication


B. Improvement in Analytical Code

Refinement in the time-marching code has led to improvement in the convergence of the solution. The calculation was carried out to 800 cycles with error in mass conservation less than 0.7% and error in total pressure conservation within 0.2%. The predicted pressure fluctuations on the blade surfaces compared very well with the measured data [4]. In addition, the Douglas-Newman program and the time-marching code are now better documented and readily available for future use.

C. Data Reduction

All analog data recorded on tapes had been digitized and transferred to the computers at VPI&SU for further data reduction. A computer program was written to take ensemble average on the data. Figure 3 shows some typical raw data and 200 cycles ensemble averaged data from pressure transducers mounted on the pressure and suction sides of the blade, with and without the downstream struts. The rotor is running at 2900 RPM and 15% surge margin. The transducers are at 50% span.
A procedure is also developed to calculate the unsteady lift and moment from the blade mounted transducers. The lift is calculated using

$$L = \sum_{i=1}^{6} (P_p - P_s) \Delta A_i$$

(1)

where $P_p$ and $P_s$ are the pressures on the pressure and suction surfaces of the blade, $\Delta A$ is the differential area covered by each transducer. The pressures can be written as two components:

$$P_p(t) = \overline{P_p} + P'_p(t)$$

and

$$P_s(t) = \overline{P_s} + P'_s(t)$$

where $\overline{P}$ is the steady-state pressure and $P'$ is the time-varying pressure. Thus

$$L = \sum_{i=1}^{6} (\overline{P_p} - \overline{P_s}) \Delta A_i + \sum_{i=1}^{6} (P'_p - P'_s) \Delta A_i$$

(2)

The first term is a constant and does not change with time. The second term can be calculated from the measurement using the blade mounted transducers. Figure 4.A shows the unsteady lift against rotor position for different downstream strut locations.

Similarly the unsteady moment is given by
\[ M = \sum_{i=1}^{6} d_i (F_p - F_s) \Delta A_i + \sum_{i=1}^{6} d_i (F_p' - F_s') \Delta A_i \]  

where \( d_i \) is the distance between the transducer and quarter chord from the leading edge of the blade.

Again the first term does not change with time and the second term can be calculated from the blade mounted transducers. Figure 4-3 shows the unsteady moment plotted against rotor position for different downstream strut locations.

D. Interactions with Industry

There is strong interest on the current work from the Aircraft Engine Business Group of General Electric Company at Cincinnati, Ohio. Discussions with their aeroacoustic design group have indicated that the problem is still of definite interest to them and to the aircraft manufacturer. They may use the analytical code for future investigations. Two specific areas of interest are: (1) Optimum positioning of the stators between the rotors and the struts to minimize unsteady interaction, (2) Effect of nonuniform circumferential spacing of the support struts on the fan-strut interaction (5). There will be continuous interactions between the Turbomachinery Research Group and aircraft engine industry.
Fig. 3.A.1. Unsteady pressure on pressure surface of rotor (raw data) with strut at 0.119 m from rotor trailing edge.
Fig. 3.A.2. Unsteady pressure on suction surface of rotor (raw data) with strut at 0.119 m from rotor trailing edge.
Fig. 3.B.1. Unsteady pressure on pressure surface of rotor (raw data) with strut removed.
Fig. 3.8.2. Unsteady pressure on suction surface of rotor (raw data) with strut removed.
Fig. 3.C.1. Unsteady pressure on pressure surface of rotor (200 ensemble averaged) with strut at 0.119 m from rotor trailing edge.
Fig. 3.C.2. Unsteady pressure on suction surface of rotor (200 ensemble averaged) with strut at 0.119 m from rotor trailing edge.
Fig. 3.D.1. Unsteady pressure on pressure surface of rotor (200 ensemble averaged) with strut removed.
Fig. 3.D.2. Unsteady pressure on suction surface of rotor (200 ensemble averaged) with strut removed.
Fig. 4.A. Unsteady lift as a function of strut positions.
IV. Summary and Future Plan

All analog data recorded on tapes have been digitized and successfully transferred to the computers at VPI&SU for further data reduction. In addition, a program was developed to calculate the unsteady lift and moment from the blade mounted transducers. The result of this analysis will be used to validate the numerical calculation. Work is in progress for implementing a digital Fast Fourier Transform (FFT) and Discrete Fourier Transform (DFT) subroutines for looking at spectral analysis of the data. A parametric study on the effect of different Fourier Transforms on the spectral analysis is underway.

In order to isolate the effect of downstream struts from other harmonic disturbances, it is necessary to subtract the amplitude of pressure fluctuation due to "background" disturbances from the amplitude of pressure fluctuation due to the downstream struts alone. In the previous data analysis, this was done by performing a power spectral analysis (FFT) on the raw data, then taking 200 averages of the power spectrum to give an overall spectral analysis. This was done for both cases; with and without the downstream struts. The difference in amplitude of the pressure fluctuation was then reported in Ref. (1). No phase information was retained in this data analysis.

Further data reduction will include an enhanced-signal spectral analysis (6) for the data set. The raw data will first be ensemble averaged (conditional averaging) and then a power spectral analysis will be performed on the averaged data while retaining the phase information. The vector difference in amplitude of the pressure fluctuation (with and without the downstream struts) can be obtained using a polar
plot (Nyquist plot). The result of this analysis will be compared to previous results as well as to the analytical model.

In addition, using the existing analytical code, an effort will be initiated to investigate the optimum positioning of the stators between the rotors and the struts to minimize unsteady interaction.
V. References


5. Gliebe, P. R. and Ho, F. Y., personal communications, June, October 1984.

Appendix: Listing of programs used in data transfer

1. Program TPP CNTL - execute a system program (TPPRINT) which checks the tape characteristics.

TPP CNTL

//50589TL0 JOB 529E5, 'TIM OLSEN', REGION=1024K, TIME=(,59)
//JOBPARM_LINES=99
//PRIORITY STANDARD
//LONGKEY WINGPAL
//STEP0001 EXEC PGM=TPPRINT SYSTEM PROGRAM WHICH CHECKS TAPE CHARACTERISTICS.
//SYSPRINT DD SYSOUT=* UNIT=TAPE,
//SYSUT1 DD DSN=VPI13,
//DISP=OLD,
//UNIT=TAPE,
//LABEL=(1,NL),
//VOL=SER=VPI13,
//DCB=BLKSIZE=32000 LARGEST POSSIBLE; TAPE WILL PROBABLY HAVE SMALLER BLOCKSIZE. DSN AND VOL
//SYSUT2 DD SYSOUT=* GIVE NAME OF NASA TAPE BEING CHECKED.
//SYSIN DD * FILES=99,STOPAFT=3
2. Program TAPE TAPE - transfer NASA's tape to VPI's system tape.

```
TAPE

//B0549TL0 JOB 529E5,'TIM OLSEN',REGION=1536K,TIME=(.59)
//CHOKEY WINGFAL
//ROUTE PRINT VPIVM1.OLSENTL
//ROUTE PUNCH VPIVM1.OLSENTL
//JOBPARM LINES=10
#PRIORITY IDLE
//STEP0001 EXEC PLICLNG
//PLI.SYSIN DD * PROCESS SOURCE, FLAG(I), GOSTMT;
READTP:
  PROCEDURE OPTIONS( MAIN );
  DECLARE SYSPRINT OUTPUT PRINT FILE;
  DECLARE SYSSIN INPUT RECORD FILE;
  DECLARE OUTFIL OUTPUT RECORD FILE;
  DECLARE DATA( 1920 ) FIXED II NARY( 15 )
  DECLARE COUNT FIXED BINARY( 31 )
  DECLARE END-OF-FILE FIXED BINARY( 31 ) INITIAL( 0 )
  ON ENUFILE( SYSIN ) END-OF-FILE = 1
  ON ERROR SNAP BEGIN
    ON ERROR SYSTEM
      PUT DATA
      STOP
    END
  DO WHILE( END OF FILE # 0 )
    READ FILE( SYSIN ) INTO( DATA )
    WRITE FILE( OUTFIL ) FROM( DATA )
  END;
END READTP;

// VOL SHOWS OLD TAPE NAME; LABEL & DSN SHOW FILE'S POSITION ON TAPE.
// GO.SYSIN DD DSN=FILE1,
  // DISP=OLD,
  // UNIT=TAPE,
  // LABEL=(1,NL),
  // VOL=SER=VP117,
  // DCB=(RECFM=F,LRECL=3840,BLKSIZE=3840)
// SYSIN REFERS TO OLD NASA TAPE; OUTFIL REFERS TO NEW SYSTEM TAPE.
// DSN SHOWS ASSIGNED FILE NAME ON NEW TAPE; LABEL SHOWS FILE'S
// POSITION ON TAPE; SER='SERIAL 9 1', ASSIGNED BY SYSTEM FIRST TIME
// AND SHOWN IN OUTPUT, SUBSEQUENTLY GIVEN BY USER.
// GO.OUTFIL DD DSN=A529E5.VPI17.FILE1,
  // DISP=(NEW,PASS),
  // UNIT=TAPE,
  // LABEL=(19,SL,EXPDT=88304),
  // VOL=(,RETAI.,SER=200109),
  // DCB=(RECFM=V8,LRECL=3844,BLKSIZE=16000,DEN=4)
// GO.DD1 DD DSN=c8&G00SET,DISP=(OLD,DELETE)
// THIS BLOCK MAY BE REPEATED FOR EACH FILE DESIRED WITH THE STEP *
// CHANGED AND THE FILE ATTRIBUTES ADJUSTED.
//STEP0002 EXEC PLICLNG
//PLI.SYSIN DD *
  // PROCESS SOURCE, FLAG(I), GOSTMT;
READTP:
  PROCEDURE OPTIONS( MAIN );
  DECLARE SYSPRINT OUTPUT PRINT FILE;
  DECLARE SYSSIN INPUT RECORD FILE;
```
Program TAPE TAPE continued

TAPE TAPE

DECLARE OUTFIL OUTPUT RECORD FILE;
DECLARE DATA (1920) FIXED BINARY(15);
DECLARE COUNT FIXED BINARY(31);
DECLARE END_OF_FILE FIXED BINARY(31) INITIAL(0);
ON ENDFILE(SYSIN) END_OF_FILE = 1;
ON ERROR SNAP BEGIN;
ON ERROR SYSTEM;
PUT DATA;
STOP;
END;
DO WHILE( END_OF_FILE = 0 );
READ FILE(SYSIN) INTO( DATA );
WRITE FILE(OUTFIL) FROM( DATA );
END;
END READTP;

//GO.SYSIN DD DSN=FILE2,
 // DISP=OLD,
 // UNIT=TAPE,
 // LABEL=(2,NL),
 // VOL=SER=VPI17,
 // DCB=(RECFM=F,LRECL=3840,BLKSIZE=3840)
//GO.OUTFIL DD DSN=A529E5.VPI17.FILE2,
 // DISP=(NEW,PASS),
 // UNIT=TAPE,
 // LABEL=(20,SL,EXPDT=88304),
 // VOL=,RETAIN,,REF=*,STEP0001.GO.OUTFIL),
 // DCB=(RECFM=VB,LRECL=3844,BLKSIZE=16000,DEN=4)
//GO.DDI DD DSN=RGOSET,DISP=(OLD,DELETE)
//STEP0003 EXEC PLIXCLO
//PII.SYSIN DD *
# PROCESS SOURCE, FLAG(1), GOSTMT;
READTP:
PROCEDURE OPTIONS( MAIN );
DECLARE SYSPRINT OUTPUT PRINT FILE;
DECLARE SYSIN INPUT RECORD FILE;
DECLARE OUTFIL OUTPUT RECORD FILE;
DECLARE DATA (1920) FIXED BINARY(15);
DECLARE COUNT FIXED BINARY(31);
DECLARE END_OF_FILE FIXED BINARY(31) INITIAL(0);
ON ENDFILE(SYSIN) END_OF_FILE = 1;
ON ERROR SNAP BEGIN;
ON ERROR SYSTEM;
PUT DATA;
STOP;
END;
DO WHILE( END_OF_FILE = 0 );
READ FILE(SYSIN) INTO( DATA );
WRITE FILE(OUTFIL) FROM( DATA );
END;
END READTP;

//GO.SYSIN DD DSN=FILE3,
 // DISP=OLD,
 // UNIT=TAPE,
Program.TAPE TAPE continued

TAPE TAPE

//
// LABEL=(3.NL),
// VOL=SER=VPI17,
// DBC=(RECFM=F,LRECL=3840,BLKSIZE=3840)
//GO.OUTFIL DD DSN=A529ES.VPI17.FILE3,
// DISP=(NEW,PASS),
// UNIT=TAPE,
// LABEL=(21.NL,EXPT=88304),
// VOL=(,RETAIN,,,SER=200109),
// DBC=(RECFM=VB,LRECL=3844,BLKSIZE=16000,DEN=4)
//GO.DDI DD DSN=8&GOSET,DISP=(OLD,DELETE)
//STEP0004 EXEC PLIXCLG
//PII.SYSIN DD *

* PROCESS SOURCE, FLAG(I), GOSTMT *
READTP:

PROCEDURE OPTIONS( MAIN ) ;
DECLARE SYSPRINT OUTPUT PRINT FILE ;
DECLARE SYSIN INPUT RECORD FILE ;
DECLARE OUTFIL OUTPUT RECORD FILE ;
DECLARE DATA( 1920 ) FIXED BINARY( 15 ) ;
DECLARE COUNT FIXED BINARY( 31 ) ;
DECLARE END_OF_FILE FIXED BINARY( 31 ) INITIAL( 0 ) ;
ON ERROR SNAP BEGIN
ON ERROR SYSTEM
PUT DATA
STOP
END
DO WHILE( END_OF_FILE = 0 ) ;
READ FILE( SYSIN ) INTO( DATA ) ;
WRITE FILE( OUTFIL ) FROM( DATA ) ;
END
END READTP

* GO.SYSIN DD DSN=FILE4,
* DISP=OLD.
* UNIT=TAPE,
* LABEL=(4.NL),
* VOL=SER=VPI17,
* DBC=(RECFM=F,LRECL=3840,BLKSIZE=3840)
*GO.OUTFIL DD DSN=A529ES.VPI17.FILE4,
* DISP=(NEW,PASS),
* UNIT=TAPE,
* LABEL=(22.NL,EXPT=87304),
* VOL=(,RETAIN,,,REF=8.STEP0001.GO.OUTFIL),
* DBC=(RECFM=VB,LRECL=3844,BLKSIZE=16000,DEN=4)
*GO.DDI DD DSN=8&GOSET,DISP=(OLD,DELETE)
3. Program TAPE LISTE - read from the system tape and print the first 20 elements of every line.

```
TAPE LISTE

//B0589TLO JOB 529E5,'TILDEN',REGION=1536K,TIME=(,59)
//JOBPARM LINES=40
//PRIORITY IDLE
//STEP0001 EXEC FORTVC0
//FORT.SYSIN DD *
INT0GER=2 DATA( 1920 )
10 CONTINUE
   READ( 5, END = 20 ) DATA
   WRITE( 6, 99 ) ( DATA( I ), I = 1, 20 )
99 FORMAT( ' ', 20I6 )
GOTO 10
20 CONTINUE
STOP
END

// DSN SHOWS NAME AND LABEL SHOWS POSITION OF FILE DESIRED.
//00.FT05F001 DD DSN=A529E5.VPI12.FILE1,
//      DISP=OLD,
//      UNIT=TAPE,
//      LABEL=(7,SL),
//      VOL=(,RETAIN,,,SER=200109) SERIAL 0 OF SYSTEM TAPE.
```
<table>
<thead>
<tr>
<th>File #</th>
<th>Counter</th>
<th>Hour</th>
<th>Second</th>
<th>Day</th>
<th>Min.</th>
<th>Channels</th>
<th>Once per Sq. rev Wave</th>
<th>Second</th>
<th>Channels</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1913</td>
<td>0</td>
<td>207</td>
<td>855</td>
<td>5700</td>
<td>4480</td>
<td>3299</td>
<td>3947</td>
<td>7340</td>
</tr>
<tr>
<td>1</td>
<td>1913</td>
<td>1</td>
<td>207</td>
<td>855</td>
<td>5700</td>
<td>4480</td>
<td>3099</td>
<td>3947</td>
<td>7340</td>
</tr>
<tr>
<td>1</td>
<td>1913</td>
<td>1</td>
<td>207</td>
<td>855</td>
<td>5700</td>
<td>4480</td>
<td>3099</td>
<td>3947</td>
<td>7340</td>
</tr>
<tr>
<td>1</td>
<td>1913</td>
<td>1</td>
<td>207</td>
<td>855</td>
<td>5700</td>
<td>4480</td>
<td>3099</td>
<td>3947</td>
<td>7340</td>
</tr>
<tr>
<td>1</td>
<td>1913</td>
<td>1</td>
<td>207</td>
<td>855</td>
<td>5700</td>
<td>4480</td>
<td>3099</td>
<td>3947</td>
<td>7340</td>
</tr>
<tr>
<td>1</td>
<td>1913</td>
<td>1</td>
<td>207</td>
<td>855</td>
<td>5700</td>
<td>4480</td>
<td>3099</td>
<td>3947</td>
<td>7340</td>
</tr>
<tr>
<td>1</td>
<td>1913</td>
<td>1</td>
<td>207</td>
<td>855</td>
<td>5700</td>
<td>4480</td>
<td>3099</td>
<td>3947</td>
<td>7340</td>
</tr>
<tr>
<td>1</td>
<td>1913</td>
<td>1</td>
<td>207</td>
<td>855</td>
<td>5700</td>
<td>4480</td>
<td>3099</td>
<td>3947</td>
<td>7340</td>
</tr>
<tr>
<td>1</td>
<td>1913</td>
<td>1</td>
<td>207</td>
<td>855</td>
<td>5700</td>
<td>4480</td>
<td>3099</td>
<td>3947</td>
<td>7340</td>
</tr>
<tr>
<td>1</td>
<td>1913</td>
<td>1</td>
<td>207</td>
<td>855</td>
<td>5700</td>
<td>4480</td>
<td>3099</td>
<td>3947</td>
<td>7340</td>
</tr>
</tbody>
</table>

Program TAPE LISTE continued - output

(truncated)
4. Program TAPE LISTL — read from the system tape and show the first few lines of the file in totality.

TAPE LISTL

//BO589TLO JOB 529E5,'TIM OLSEN',REGION=1536K,TIME=(,59)
//NJOBSPARM LINES=40
//NPRIORITY IDLE
//STEP0001 EXEC FORTVCO
//FORT.SYSIN DD *
INTEGER DATA( 1920 )
DO 10 I=1,3
READ(5) DATA
WRITE(6,99) DATA
99 FORMAT(1I10,5I6,191(10I6/,5I6)
10 CONTINUE
STOP
END
//00.FT05F001 DD DSN=A529E5.VPI12.FILE1,
// DISP=OLD,
// UNIT=TAPE,
// LABEL=(7,SL),
// VOL=(,RETAIN,,SER=200109)
Program TAPE LISTL continued -

output

<table>
<thead>
<tr>
<th>File#</th>
<th>Dummy</th>
<th>Hour</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 1915</td>
<td>2 7 1059</td>
<td></td>
</tr>
<tr>
<td>5996</td>
<td>7580</td>
<td>-5854</td>
</tr>
<tr>
<td>5996</td>
<td>7640</td>
<td>-6854</td>
</tr>
<tr>
<td>5996</td>
<td>7700</td>
<td>-6838</td>
</tr>
<tr>
<td>5996</td>
<td>7750</td>
<td>-5271</td>
</tr>
<tr>
<td>5996</td>
<td>7810</td>
<td>-9016</td>
</tr>
<tr>
<td>5996</td>
<td>7870</td>
<td>-6985</td>
</tr>
<tr>
<td>5996</td>
<td>7930</td>
<td>-5809</td>
</tr>
<tr>
<td>5996</td>
<td>7990</td>
<td>-3233</td>
</tr>
<tr>
<td>5996</td>
<td>8050</td>
<td>-5266</td>
</tr>
<tr>
<td>5996</td>
<td>8100</td>
<td>-6599</td>
</tr>
<tr>
<td>5996</td>
<td>8150</td>
<td>-5459</td>
</tr>
<tr>
<td>5996</td>
<td>8210</td>
<td>-6075</td>
</tr>
<tr>
<td>5996</td>
<td>8260</td>
<td>-7003</td>
</tr>
<tr>
<td>5996</td>
<td>8340</td>
<td>-6239</td>
</tr>
<tr>
<td>5996</td>
<td>8390</td>
<td>-5928</td>
</tr>
<tr>
<td>5996</td>
<td>8450</td>
<td>-5535</td>
</tr>
<tr>
<td>5996</td>
<td>8500</td>
<td>-4779</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Time</th>
<th>Time Channel</th>
<th>Once Square</th>
</tr>
</thead>
<tbody>
<tr>
<td>(in 1 second)</td>
<td>2 3 4 5 6 Rev</td>
<td></td>
</tr>
<tr>
<td>0 0 0 0 0 (5 zeroes indicate end of line)</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

One line of original data (1920 elements)
5. Program POS CHECK – check number of data points in each cycle.

POS CHECK

/* PRIORITY STANDARD */
/STEP0001 EXEC FORTVC

/*POS CHECK READS SYSTEM TAPE, CHECKS POSITION VS SQ WAVE.
I, J, K COUNTERS; TEST & ON FOR CONTROL, POS = ROTOR
ANLAR POSITION, OPR = ONCE PER REV, SQW = SQUARE WAVE*/

INTEGER*2 DATA(1920)
INTEGER I, J, K, POS, ON, TEST
REAL SQW(99500)

READ DATA FROM TAPE, ASSIGN SQUARE WAVE
DO 10 I=1,1500
READ(5) DATA
DO 20 J=1,191
SQW((I-1)*191+J) = DATA(10*J+5)
20 CONTINUE
10 CONTINUE

WRITE(6,98)
98 FORMAT(1H1,4(' POS SQWAVE'))

CHECK SQUARE WAVE, ASSIGN ROTOR ANGULAR POSITIONS
WRITE POSITIONS AND CHANNEL VALUES
POS = 0
ON = 0
TEST = 0
DO 30 K=1,99500
IF (TEST.EQ.1) THEN
TEST = 0
POS = 0
ON = 1
ENDIF
IF (SQW(K).GE.6000) THEN
IF (SQW(K+1).LE.1000) THEN
TEST = 1
WRITE(6,92) K, (POS+1), SQW(K), SQW(K+1)
92 Format(1H ,2I6, 2F7.0)
ENDIF
ENDIF
IF (ON.EQ.1) THEN
POS = POS+1
WRITE(6,99) K, POS, TIME(K), CH1(K), CH2(K), CH3(K),
CH4(K), CH5(K), CH6(K), OPR(K), SQW(K)
99 Format(1H ,2I5,F8.0,8F7.0)
30 CONTINUE
STOP
END

//GO.FT05F001 DD DSN=A529E5.VPI10.FILE1,
// DISP=OLD,
// UNIT=TAPE,
// LABEL=(1,5L),
// VOL=(,RETAIN,,SER=200109)
6. Program CYCLE SELECT - discard cycles where the number of points is not equal to 360.

CYCLE SELECT

//B0589TLO JOB 529E5,'TIM OLSen',REGION=1536K,TIME=(,59)
//LONGKEY WINGFAL
//J00PARM LINES=40, CARDS=100
//PRIORITRY STANDARD
//STEP0001 EXEC FORTVC
//FORT.SYSIN DD

CM** CYCLE SELECT READS SYSTEM TAPE, FINDS CYCLES WITH 360 UNITS.
CM** I, J, K, L, M COUNTERS; TEST & ON FOR CONTROL. POS = ROTOR ANGULAR POSITION, OPR = ONCE PER REV, SQW = SQUARE WAVE,
CM** CYCLE = CYCLE NUMBER (IF ACCEPTED)
CM** INTEGER DATA(1920)
CM** INTEGER I, J, K, L, M, MNEW, POS, CYCLE, N
CM** REAL APOS(1146), CH1(1146), CH2(1146), CH3(1146)
CM** REAL CH4(1146), CH5(1146), CH6(1146), OPR(1146), SQW(1146)
CM** READ DATA FROM TAPE, ASSIGN CHANNELS
CM** N = 720
CM** DO 10 I=1,6
CM** READ(5) DATA
CM** DO 20 J=1,191
CM** CH1((I-1)*191+J) = DATA(10*J-2)
CM** CH2((I-1)*191+J) = DATA(10*J-1)
CM** CH3((I-1)*191+J) = DATA(10*J)
CM** CH4((I-1)*191+J) = DATA(10*J+1)
CM** CH5((I-1)*191+J) = DATA(10*J+2)
CM** CH6((I-1)*191+J) = DATA(10*J+3)
CM** OPR((I-1)*191+J) = DATA(10*J+4)
CM** SQW((I-1)*191+J) = DATA(10*J+5)
CM** 20 CONTINUE
CM** 10 CONTINUE
CM** HEADING
CM** WRITE(6,98)
CM** 98 FORMAT('CYCLE POS ORIG APOS CHAN1 CHAN2 CHAN3 ',
CM** + 'CHAN4 CHAN5 CHAN6 1/REV SQWAVE')
CM** CHECK SQUARE WAVE, ASSIGN ROTOR ANGULAR POSITIONS,
CM** CHECK CYCLE FOR 360 POSITIONS, WRITE VALUES
CM** POS = 0
CM** CYCLE = 0
CM** DO 30 K=1,(N+360)
CM** POS = POS + 1
CM** IF (SQW(K).GE.6000) THEN
CM** IF (SQW(K+1).LE.1000) THEN
CM** IF (POS.EQ.360) THEN
CM** CYCLE = CYCLE + 1
CM** POS = 0
CM** DO 40 L=1,360
CM** M = K - 360 + L
CM** MNEW = (CYCLE-1)*360 + L

35
Program CYCLE SELECT continued

CYCLE SELECT

APOS(MNEW) = MNEW
CH1(MNEW) = CH1(M)
CH2(MNEW) = CH2(M)
CH3(MNEW) = CH3(M)
CH4(MNEW) = CH4(M)
CH5(MNEW) = CH5(M)
CH6(MNEW) = CH6(M)
OPR(MNEW) = OPR(M)
S0W(MNEW) = S0W(M)
WRITE(6,99) CYCLE, L, M, APOS(MNEW), CH1(MNEW),
  CH2(MNEW), CH3(MNEW), CH4(MNEW), CH5(MNEW),
  CH6(MNEW), OPR(MNEW), S0W(MNEW)

99  CONTINUE

ELSE
  POS = 0
ENDIF
ENDIF
ENDIF
30  CONTINUE
STOP
END

/GO.FT05F001 DD DSN=A529E5.VPI10.FILE1,
DISP=OLD,
UNIT=TAPE,
LABEL=(1,SL),
VOL=(,RETAIN,,,SER=200109)
7. Program VOLT PLOT - plot data from each channel (1-6), square wave, and once-per-rev signals against position

VOLT PLOT

//02589TLO JOB 529E5,'TIM OLSEN',REGION=1536K,TIME=(,59)
//PRIORITY STANDARD
//STEP0001 EXEC FORTVCOC

// VOLT PLOT READS SYSTEM TAPE, PLOTS CYCLES WITH 360 UNITS.
// I, J, K, L, M, N, POS, CYCT, CYCT
// ANGULAR POSITION, Opr = ONCE PER REV, SQW = SQUARE WAVE,
// M = ORIGINAL DATA COUNT, MNEW = DATA COUNT OF ACCEPTED CYCLES,
// CYCT = CYCLE COUNTER (IF ACCEPTED)

INTEGER DATA(1920)
INTEGER I, J, K, L, M, N, POS, CYCT
REAL APOS(1146), CH1(1146), CH2(1146), CH3(1146)
REAL CH4(1146), CH5(1146), CH6(1146), OPR(1146), SQW(1146)
N = 3

READ DATA FROM TAPE, ASSIGN CHANNELS

DO 10 I=1,(INT(360*(N+1)/191)+1)
  READ( 5 ) DATA
  DO 20 J=1,191
    CH1((I-1)*191+J) = DATA(I*10+J-2)
    CH2((I-1)*191+J) = DATA(I*10+J-1)
    CH3((I-1)*191+J) = DATA(I*10+J)
    CH4((I-1)*191+J) = DATA(I*10+J+1)
    CH5((I-1)*191+J) = DATA(I*10+J+2)
    CH6((I-1)*191+J) = DATA(I*10+J+3)
    OPR((I-1)*191+J) = DATA(I*10+J+4)
    SQW((I-1)*191+J) = DATA(I*10+J+5)
  CONTINUE
10 CONTINUE

WRITE(6,98)
98 FORMAT(1CYCLE POS ORIG APOS CHAN1 CHAN2 CHAN3 ',
         + 'CHAN4 CHAN5 CHAN6 1/REV SQWAVE')

CHECK SQUARE WAVE, ASSIGN ROTOR ANGULAR POSITIONS,
CHECK CYCLE FOR 360 POSITIONS, WRITE VALUES

POS = 0
CYCT = 0
DO 30 K=1,(N+1)*360
  POS = POS + 1
  IF (SQW(K).GE.6000) THEN
    IF (SQW(K+1).LE.1000) THEN
      IF (POS.EQ.360) THEN
        CYCT = CYCT + 1
        POS = 0
      DO 40 L=1,360
        M = K - 360 + L
        CONTINUE
30 CONTINUE
Program VOLT PLOT continued

```
VOLT PLOT

MNEW = (CYCT-1)*360 + L
APOS(MNEW) = MNEW
CH1(MNEW) = CH1(M)
CH2(MNEW) = CH2(M)
CH3(MNEW) = CH3(M)
CH4(MNEW) = CH4(M)
CH5(MNEW) = CH5(M)
CH6(MNEW) = CH6(M)
OPR(MNEW) = OPR(M)
SQW(MNEW) = SQW(M)
WRITE(6,99) CYCT, L, M, APOS(MNEW), CH1(MNEW),
            CH2(MNEW), CH3(MNEW), CH4(MNEW), CH5(MNEW),
            CH6(MNEW), OPR(MNEW), SQW(MNEW)

C CONTINUE
C axxa4.0XN
C XFIR a 0.0
C XINC a 90.0
C AYY a 7.0
C YFIR a 0.0
C YINC a 900.0
C APOS(N+1) a 0.0
C APOS(N+2) a 90.0
C SQW(N+1) a 0.0
C SQW(N+2) a 900.0
C ITYPE a 0
C ITEXT a 11
CALL PLOTS(0.0,0.0)
CALL PLOT(0.0,-3)
CALL VPISYM(0.75,7.0,0.12,'SQUARE WAVE',0..11)
CALL AXIS(0.0,0.0,'POSITION',-8,Axx,0.0,XFIR,XINC)
CALL AXIS(0.0,0.0,'VOLTAGE',7,AYY,90.0,YFIR,YINC)
CALL LINE(APOS,SQW,N,1,ITYPE,ITEXT)
CALL PLOT(6.,6.,-999)

PLOT ONCE PER REV

AXX a 4.0
XFIR a 0.0
XINC a 90.0
AYY a 6.0
YFIR a -15000.0
YINC a 5000.0
APOS(N+1) a -15000.0
APOS(N+2) a 90.0
OPR(N+1) = -15000.0
```

Program VOLT PLOT continued

VOLT PLOT

OPR(N+2) = 5000.0
ITYPE = 0
ITEXT = 11
CALL PLOT(0.0, 0.50)
CALL PLOT(1.1, -3)
CALL VPISYM(0.75, 6.0.0.12, 'ONCE PER REV', 0., 12)
CALL AXIS(0.0, 0.0, 'POSITION', -8, AXX, 0.0, XFIR, XINC)
CALL LINE(APOS, OPR, N+1, ITYPE, ITEXT)
CALL PLOT(6., 999)

CALL PLOT CHANNEL 1

CALL PLOT(1., 1., -3)
CALL VPISYM(0.75, 6.0.0.12, 'CHANNEL 1', 0., 9)
CALL AXIS(0.0, 0.0, 'POSITION', -8, AXX, 0.0, XFIR, XINC)
CALL LINE(APOS, CH1, N, ITYPE, ITEXT)
CALL PLOT(6., 999)

CALL PLOT CHANNEL 2

CALL PLOT(1., 1., -3)
CALL VPISYM(0.75, 6.0.0.12, 'CHANNEL 2', 0., 9)
CALL AXIS(0.0, 0.0, 'POSITION', -8, AXX, 0.0, XFIR, XINC)
CALL LINE(APOS, CH2, N, ITYPE, ITEXT)
CALL PLOT(6., 999)

39
Program VOLT PLOT continued

VOLT PLOT

Cxxx PLOT CHANNEL 3
Cxxx AXX = 4.0 * N
Cxxx XFIR = 0.0
Cxxx XINC = 90.0
Cxxx AYY = 6.0
Cxxx YFIR = -7000.0
Cxxx YINC = 2000.0
Cxxx APOS(N+1) = 0.0
Cxxx APOS(N+2) = 90.0
Cxxx CH3(N+1) = -7000.0
Cxxx CH3(N+2) = 2000.0
Cxxx ITYPE = 0
Cxxx ITEXT = 11
Cxxx CALL PLOTS(0,0,50)
Cxxx CALL PLOT(1.1,1.1,-3)
Cxxx CALL VPISYM(0.75,6.0,0.12,'CHANNEL 3',0.0)
Cxxx CALL AXIS(0.0,0.0,'VOLTAGE',7,AYY,90.0,YFIR,YINC)
Cxxx CALL LINE(APOS,CH3,N,1,ITYPE,ITEXT)
Cxxx CALL PLOT(6.6,6.6,-999)

Cxxx PLOT CHANNEL 4
Cxxx AXX = 4.0 * N
Cxxx XFIR = 0.0
Cxxx XINC = 90.0
Cxxx AYY = 7.0
Cxxx YFIR = -14000.0
Cxxx YINC = 2000.0
Cxxx APOS(N+1) = 0.0
Cxxx APOS(N+2) = 90.0
Cxxx CH4(N+1) = -14000.0
Cxxx CH4(N+2) = 2000.0
Cxxx ITYPE = 0
Cxxx ITEXT = 11
Cxxx CALL PLOTS(0,0,50)
Cxxx CALL PLOT(1.1,1.1,-3)
Cxxx CALL VPISYM(0.75,6.0,0.12,'CHANNEL 4',0.0)
Cxxx CALL AXIS(0.0,0.0,'VOLTAGE',7,AYY,90.0,YFIR,YINC)
Cxxx CALL LINE(APOS,CH4,N,1,ITYPE,ITEXT)
Cxxx CALL PLOT(6.6,6.6,-999)

Cxxx PLOT CHANNEL 5
Cxxx AXX = 4.0 * N
Cxxx XFIR = 0.0
Cxxx XINC = 90.0
Cxxx AYY = 5.0
Cxxx YFIR = -5000.0
Cxxx YINC = 2000.0
Cxxx APOS(N+1) = 0.0
Cxxx APOS(N+2) = 90.0
Cxxx CH5(N+1) = -5000.0
Cxxx CH5(N+2) = 90.0
Program VOLT PLOT  continued

VOLT    PLOT

CH5(N+2) = 2000.0
ITYPE = 0
ITEXT = 11
CALL PLOTS(0.0,0.50)
CALL PLOT(1.1,1.5)
CALL VPISYM(0.75,6.0,0.12,'CHANNEL 5',0.9)
CALL AXIS(0.0,0.0,'POSITION',-0.0,0.0,0.0,0.0)
CALL AXIS(0.0,0.0,'VOLTAGE',7.1,90.0,0.0,0.0)
CALL LINE(APOS,CH5,N+1,ITYPE,ITEXT)
CALL PLOT(6.,6.,999)

CH6(N+2) = 2000.0
CALL PLOT CHANNEL 6

A0 = 4.0 * N
AX = 0.0
XINC = 90.0
AY = 6.0
YFIR = 0.0
YANC = 2000.0
APOS(N+1) = 0.0
APOS(N+2) = 90.0
CH6(N+1) = -18000.0
CH6(N+2) = 2000.0
ITYPE = 0
ITEXT = 11
CALL PLOTS(0.0,0.50)
CALL PLOT(1.1,1.5)
CALL VPISYM(0.75,6.0,0.12,'CHANNEL 6',0.9)
CALL AXIS(0.0,0.0,'POSITION',-0.0,0.0,0.0,0.0)
CALL AXIS(0.0,0.0,'VOLTAGE',7.1,90.0,0.0,0.0)
CALL LINE(APOS,CH6,N+1,ITYPE,ITEXT)
CALL PLOT(6.,6.,999)
STOP
END

//00.FT0SF001 DD DSN=A529ES.VPI10.FILE1,
//DISP=OLD,
//UNIT=TAPIE,
//LABEL=(1,5L),
//VOL=(,RETI,9999,SER=200109)