NASA Technical Memorandum 112194

Measurement of Air Flow Characteristics Using Seven-Hole Cone Probes

Timothy T. Takahashi, Ames Research Center, Moffett Field, California

May 1997

National Aeronautics and
Space Administration

Ames Research Center
Moffett Field, California 94035-1000

Contents

Nomenclature

C_P	dimensionless pressure coefficient
C_{Ps}	static pressure sensitive dimensionless coefficient
C_{PT}	target pressure sensitive dimensionless coefficient
C_{Pt}	total pressure sensitive dimensionless coefficient
C_{Pta}	pressure coefficient
C_{Ptb}	pressure coefficient
C_{Ptc}	pressure coefficient
$C_{P\alpha}$	pitch sensitive dimensionless pressure coefficient
$C_{P\beta}$	yaw sensitive dimensionless pressure coefficient
d	diameter of circular cylinder (wake survey)
i	index variable (port number)
K	empirically determined coefficient
$\overline{P}$	instantaneous mean pressure of off-axis pressure ports
P_i	pressure at the *i*th port of the seven-hole probe
P_s	static pressure
$\overline{P}_s$	time-averaged static pressure
P_s'	unsteady static pressure
P_t	total pressure
$\overline{P}_t$	time-averaged total pressure
P_t'	unsteady total pressure
q	dynamic pressure
Re	Reynolds number
U	Downstream velocity
$\overline{U}$	time-averaged downstream velocity
U′	unsteady downstream velocity
x	wake survey downstream distance
α	flow angularity "pitch"
β	flow angularity "yaw"
γ	empirical coefficient (potential flow)
ε	empirical coefficient (potential flow)
Θ	included angle of incidence (potential flow)
λ	probe port "cone angle"
Π	dimensionless flow angularity coefficient
ϕ	probe port "clock angle"

Measurement of Air Flow Characteristics Using Seven-Hole Cone Probes

TIMOTHY T. TAKAHASHI*

Ames Research Center

Summary

The motivation for this work has been the development of a wake survey system. A seven-hole probe can measure the distribution of static pressure, total pressure, and flow angularity in a wind tunnel environment. The author describes the development of a simple, very efficient algorithm to compute flow properties from probe tip pressures. Its accuracy and applicability to unsteady, turbulent flow are discussed.

Introduction

For many years, multi-hole pressure probes have been used to measure flow angularity (refs. 1 and 2). The basic principle of operation is that a flow of given angularity and velocity at a given static pressure determines a unique pressure pattern across the various ports. The seven-hole probe has a specific construction advantage: six tubes of equal diameter fit exactly around a central tube of the same diameter. The geometry of the seven-hole probe is shown in figure 1. The resulting system is over-determined; there are more observations than states. The seven measured pressures determine the four output parameters: static pressure P_s, total pressure P_t, and flow angularity α and β.

In general, multi-hole pitot probes are used under the assumption that the flow of interest is steady. Great care is taken in the calibration process to ensure that the calibration flow is uniform and free from turbulence. However, real flows, particularly wake flows, are far from steady. The response of the probe to unsteady flow must be documented.

The decomposition algorithm, which reduces the seven pressures into the four air flow parameters, may be built using an ad hoc approach, an analytical flow model, or a mixture of theoretical and empirical rationale. This paper will describe the search for a procedure which combines good data quality with extreme processing efficiency.

* This work was performed while the author held a National Research Council–NASA Ames Research Associateship.

Thanks to Pat Moriarty, Stanford University, for the hot-film anemometry data and to Tony Whitmore, Dryden Flight Research Center, for inspiration in the "triples" derivation.

Application of Quasi-Steady Theory to Unsteady Flow

There is a need to understand the applicability of any steady-state analysis, calibration, or computational procedure to the unsteady, highly turbulent flow often found in wake flow. In the past, many authors addressed this topic. A brief survey of some relevant publications is given here. Together, they describe issues important in the design of seven-hole probes.

Goldstein (ref. 3) postulated that the response of a total head probe in a turbulent stream, P_t, is a measurement of mean total pressure $\overline{P_t}$ plus a steady state contribution due to the unsteady velocity field:

$$P_t = \overline{P_t} + P_t' = P_s \; 1/2\,\rho\overline{U}^2 + 1/2\,\rho\overline{U'^2}$$

In other words, Goldstein considered static pressure a mean flow property.

Jenkins (ref. 4) examined a free jet with a pitot tube equipped with a wide-bandwidth transducer. He correlated the unsteady pitot pressures with cross-wire hot-film velocimetry measurements. He found that the unsteady velocities as expressed by:

$$U' = \left(\overline{U}/2K\right)\left(P_t' - P_s'\right)/\left(\overline{P_t} - \overline{P_s}\right)$$

were entirely representative of the hot-film signal. K is an empirically determined parameter; Jenkins reports typical values of 1.114 to 1.127. In other words, Jenkins considered static pressure an instantaneous flow property.

The relationship between the shape of a pitot probe tip and its time-averaged response in a turbulent flow was studied by Becker and Brown (ref. 5). Different probe geometries (tapered, round and square nosed) were found to have markedly different response to turbulence. This behavior may be used to advantage; an estimate of root-mean-square turbulence may be made by comparing the

steady state stagnation pressures recorded by probes of differing geometry.

Walshe and Garner (ref. 6) studied the behavior of different probe configurations such as Pitot, Kiel, and five-hole probes in turbulent flow. They concluded that five-hole probes respond similarly to cowled pitot probes when measuring highly turbulent flow. Under these circumstances the mean dynamic pressure was judged to have up to 10% error. Flow angularity correlations were made comparing the five-hole probe used in a nulling and non-nulling configuration with the angularity of peak pressures obtained with an axial probe. Discrepancies were found between the three procedures when applied to turbulent flow.

Whitmore (refs. 7 and 8) has extended the concept of multi-hole pressure probes to the construction of flush-mounted airdata systems. The HI-FADS system utilizes 25 flush pressure ports mounted on an aircraft nosecone. It was calibrated for flight at both high angle of attack and large sideslip angles. At a 25 Hz computation rate, output from the HI-FADS system correlated well with flight data. One must note, however, that the flow over the aircraft nosecone is laminar.

In summary, the existing literature implies the following: (1) when measuring a flow with frequency content beneath the pneumatic attenuation limit of the probe, quasi-steady techniques can measure both mean and unsteady flow properties; (2) above this frequency limit, the probes tend to overestimate the mean dynamic pressure; and (3) uncertainty exists when mean flow angularity measurements are made in turbulent flow.

Despite these limitations, a probe calibrated under steady flow conditions remains a useful tool for measurement. Given sufficient frequency response, it can provide a metric of flow unsteadiness. A method of probe computations consistent with steady state theory will be developed below.

Potential Flow Model

Whitmore (ref. 8) has demonstrated the applicability of a simple, hemispherical potential flow model to derive expressions for determining flow angularity. The pressure coefficient at the surface of a hemisphere is:

$$C_P(\Theta) = 5/4 + 9/4\cos^2(\Theta) \qquad (1)$$

where Θ is the total flow incidence angle at the surface. Following Whitmore's derivation, the pressures on quasi-hemispherical shapes, such as the cone probe or an aircraft nose cone, may be approximated by:

$$C_P(\Theta) = \varepsilon + \gamma\cos^2(\Theta) \qquad (2)$$

where ε and γ are empirically determined for a given probe shape.

The port pressures are:

$$P_i = P_s + q\left[\cos^2(\Theta_i) + \varepsilon\sin^2(\Theta_i)\right] \qquad (3)$$

where

$$\Theta_i = \cos\alpha\cos\beta\cos\lambda_i + \sin\beta\sin\lambda_i\sin\phi_i + \sin\alpha\cos\beta\sin\lambda_i\cos\phi_i \qquad (4)$$

and λ_i and ϕ_i are coordinates of the pressure ports on the probe tip.

Figures 2(a) and 2(b) show the response of three meridional probe tip pressure ports (ports 1, 7, and 4) as a function of pitch angle, α. In figure 2(a), the pressures are computed using equation 3 where the empirical coefficient, ε, is chosen so that the computed pressures at $\alpha = 0$ are consistent with experiment. Figure 2(b) shows the actual probe pressures measured using a 45° cone probe. A comparison reveals that the empirical potential flow model provides qualitative, but not quantitative, prediction of the port pressures.

Method of Triples

The potential flow model may be used to further examine the possibility of contriving dimensionless pressure ratios which are solely functions of flow angularity. The simplest pressure ratios, or "triples," involve the pressures at three distinct ports:

$$\Pi_{ijk} = \left(P_i - P_j\right)/\left(P_j - P_k\right) \qquad i \neq j \neq k \qquad (5)$$

If one assumes that the pressures are governed by the potential flow model, equation 3, the pressures are:

$$\Pi_{ijk} = \left(\cos^2\Theta_i - \cos^2\Theta_j\right)/\left(\cos^2\Theta_j - \cos^2\Theta_k\right) \qquad (6)$$

Figures 3(a) and 3(b) demonstrate the differences between the actual probe response and simplified model given a pressure triple based upon three meridional pressure ports (ports 1, 7, and 4). Both the experimental data and the analytical model exhibit qualitative similarities including a singularity at a flow angularity of approximately one-half the probe cone angle.

The non-meridional triples are significantly less well behaved. As can be seen in figures 4(a) and 4(b), some of the non-meridional triples are multiply valued functions. $\Pi_{123} = 0.25$ may correspond to a flow angularity of –30°, –20°, –7°, and, possibly, +2°.

To reduce the system from an overdetermined state to one of one-to-one mapping, piecewise continuous pressure ratio functions must be used. Gallington (ref. 9) has found a complex, dimensionless pressure ratio is well behaved over a wide region of calibration space. A pair of pressure ratios, sensitive to pitch and yaw, are defined as functions of all seven pressures:

$$C_{P\alpha 7} = C_{Pta} + (C_{Ptb} - C_{Ptc})/2 \qquad (7a)$$

$$C_{P\beta 7} = (1/\sqrt{3})(C_{Ptb} + C_{Ptc}) \qquad (7b)$$

where

$$C_{Pta} = (P_4 - P_1)/(P_7 - \bar{P}) \qquad (8a)$$

$$C_{Ptb} = (P_3 - P_6)/(P_7 - \bar{P}) \qquad (8b)$$

$$C_{Ptc} = (P_2 - P_5)/(P_7 - \bar{P}) \qquad (8c)$$

$$\bar{P} = (1/6)(P_1 + P_2 + P_3 + P_4 + P_5 + P_6) \qquad (9)$$

The response of these coefficients to flow angularity is shown in figures 5 and 6. These figures show that Gallington's functions are well behaved and, more importantly, single valued over their entire calibration range. Provided that the flow over the entire probe tip is attached, all seven pressures are relevant. These coefficients directly define the flow angularity.

These calibration functions may be numerically inverted to define flow angularity, α and β, in terms of the two coefficients. A pair of transfer functions, f_α and f_β, may be computed:

$$\alpha = f_\alpha(C_{P\alpha i}, C_{P\beta i})$$

$$\beta = f_\beta(C_{P\alpha i}, C_{P\beta i})$$

They are tabulated in figures 7(a) and 7(b).

The inversion process was accomplished using the following procedure. First, a subroutine was written to interpolate values of $C_{P\alpha i}$ and $C_{P\beta i}$ from the calibration data set given any arbitrary values of α and β. This subroutine was embedded in a minimization algorithm designed to find an α and β such that the interpolated values of $C_{P\alpha i}$, $C_{P\beta i}$ match "target" values of these functions. In other words, to find the α and β corresponding to a given pair of values of the calibration functions:

Step 1: Choose a target value for the flow angularity coefficients $C_{P\alpha iT}$ and $C_{P\beta iT}$.

Step 2: Minimize: $f(\alpha,\beta)$

where

$$f(\alpha,\beta) = (C_{P\alpha i}(\alpha,\beta) - C_{P\alpha iT})^2 + (C_{P\beta i}(\alpha,\beta) - C_{P\beta iT})^2$$

Subject to: α,β bounded.

Step 3: At local minimum?

If YES, use derived α,β.

If NO, let α,β go out of bounds ($\alpha = -999°$, $\beta = -999°$).

Step 4: Insert values of α,β into the $f_{\alpha i}(C_{P\alpha i}, C_{P\beta i})$ and $f_{\beta i}(C_{P\alpha i}, C_{P\beta i})$ transfer function matrices.

Step 5: Choose new values of $C_{P\alpha iT}$ and $C_{P\beta i}$ and i.

Step 6: Go to **Step 2.**

High Flow Angularity Coefficients

For the high flow angularity situation, where the flow over the probe tip may have separated, unique pressure ratios must be formulated to avoid inclusion of ports on the lee side of the probe. Six pairs of coefficients may be formulated; each excludes specific neighboring ports. Because of the restricted domain of these functions, they need not be well defined about $\alpha = \beta = 0$.

The rule set divides the calibration set into specific sectors. In Zilliac (ref. 10), the choice of sector is determined from an a priori inspection of the dominant pressure. If $P_7 > P_i$ ($i = 1 \ldots 6$), then $C_{P\alpha 7}$ and $C_{P\beta 7}$ are utilized; if P_7 is not the dominant pressure, then the algorithm uses a separated flow pressure coefficient. P_i tends to exceed P_7 when the flow angularity reaches half of the cone angle (22.5° for a 45° cone probe). In reality, the flow will separate at much higher angles. This author concludes that Zilliac may have used separated flow coefficients prematurely.

A more lenient flow separation criterion may be developed. $C_{P\alpha 7}$ and $C_{P\beta 7}$ are extremely well behaved over the entire calibration range. Consequently, even at high flow angularity $C_{P\alpha}$ and $C_{P\beta}$ are a single valued function of flow angularity. The onset of flow separation may be identified by the angle found using $C_{P\alpha 7}$ and $C_{P\beta 7}$. For the case of a 45° cone probe, a cut-off point of 30° included angle was used.

Gallington (ref. 9) also developed a set of secondary flow angularity coefficients which have the property of being bounded, $|C_{P\alpha i}| < 2$ and $|C_{P\beta i}| < 2$, and free from singularities over their respective useful calibration ranges.

Sector 1 (P_1 dominant)

$$C_{P\alpha 1} = \Pi_{1267} = (P_1 - P_7)/[P_1 - (P_2 + P_6)/2] \quad (10a)$$

$$C_{P\beta 1} = \Pi_{126} = (P_6 - P_2)/[P_1 - (P_2 + P_6)/2] \quad (10b)$$

Sector 2 (P_2 dominant)

$$C_{P\alpha 2} = \Pi_{1237} = (P_2 - P_7)/[P_2 - (P_1 + P_3)/2] \quad (10c)$$

$$C_{P\beta 2} = \Pi_{123} = (P_1 - P_3)/[P_2 - (P_1 + P_3)/2] \quad (10d)$$

Sector 3 (P_3 dominant)

$$C_{P\alpha 3} = \Pi_{2347} = (P_3 - P_7)/[P_3 - (P_2 + P_4)/2] \quad (10e)$$

$$C_{P\beta 3} = \Pi_{234} = (P_2 - P_4)/[P_3 - (P_2 + P_4)/2] \quad (10f)$$

Sector 4 (P_4 dominant)

$$C_{P\alpha 4} = \Pi_{1237} = (P_4 - P_7)/[P_2 - (P_1 + P_3)/2] \quad (10g)$$

$$C_{P\beta 4} = \Pi_{345} = (P3 - P_5)/[P_4 - (P_3 + P_5)/2] \quad (10h)$$

Sector 5 (P_5 dominant)

$$C_{P\alpha 5} = \Pi_{4567} = (P_5 - P_7)/[P_5 - (P_4 + P_6)/2] \quad (10i)$$

$$C_{P\beta 5} = \Pi_{456} = (P_4 - P_6)/[P_5 - (P_4 + P_6)/2] \quad (10j)$$

Sector 6 (P_6 dominant)

$$C_{P\alpha 6} = \Pi_{1567} = (P_6 - P_7)/[P_6 - (P_5 + P_1)/2] \quad (10k)$$

$$C_{P\beta 6} = \Pi_{156} = (P_5 - P_1)/[P_6 - (P_5 + P_1)/2] \quad (10l)$$

Figures 8(a) and 8(b) demonstrate the behavior of $C_{P\alpha 1}$ for large positive α, and $C_{P\alpha 4}$ for large negative α. Each of these functions is singular about $\alpha = 0$ and shows strong asymptotic behavior for $\alpha > 30°$.

Algorithm—Part I: Flow Angularity

The processing algorithm may be implemented as follows:

Step 1: Compute the basic flow angularity coefficients: $C_{P\alpha 7}$ and $C_{P\beta 7}$.

Step 2: If they are outside the range of the sector 7 transfer function table, go to **Step 5**.

Step 3: Perform bilinear interpolation on the sector 7 tables to determine α and β.

Step 4: Are α and β indicative of separated flow over the probe tip?

If YES, continue on to **Step 5**.

If NO, then we are done.

Step 5: Determine which pressure port has the largest positive pressure (the "dominant hole")—this defines the sector, i.

Step 6: Compute the appropriate coefficients: $C_{P\alpha i}$, $C_{P\beta i}$.

Step 7: Are these coefficients within the range of the tables? If not, then we have BAD data.

Step 8: Perform bilinear interpolation on the separated flow tables to determine α and β.

Computation of Static and Total Pressure

With the flow angularity determined, the static and total pressure may be inferred from ratio of peak pressure to the mean of the pressures governed by unseparated flow. Static and total pressure coefficients are formulated in terms of the actual tunnel static pressure, P_s, and total pressure, P_t:

$$C_{Ps1} = [(P_2 + P_6)/2 - P_s]/[P_1 - (P_2 + P_6)/2] \quad (11a)$$

$$C_{Pt1} = (P_1 - P_t)/[P_1 - (P_2 + P_6)/2] \quad (11b)$$

$$C_{Ps2} = [(P_1 + P_3)/2 - P_s]/[P_2 - (P_1 + P_3)/2] \quad (11c)$$

$$C_{Pt2} = (P_2 - P_t)/[P_2 - (P_1 + P_3)/2] \quad (11d)$$

$$C_{Ps3} = [(P_2 + P_4)/2 - P_s]/[P_3 - (P_2 + P_4)/2] \quad (11e)$$

$$C_{Pt3} = (P_3 - P_t)/[P_3 - (P_2 + P_4)/2] \quad (11f)$$

$$C_{Ps4} = [(P_3 + P_5)/2 - P_s]/[P_4 - (P_3 + P_5)/2] \quad (11g)$$

$$C_{Pt4} = (P_4 - P_t)/[P_4 - (P_3 + P_5)/2] \quad (11h)$$

$$C_{Ps5} = [(P_4 + P_6)/2 - P_s]/[P_5 - (P_4 + P_6)/2] \quad (11i)$$

$$C_{Pt5} = (P_5 - P_t)/[P_5 - (P_4 + P_6)/2] \quad (11j)$$

$$C_{Ps6} = [(P_5 + P_1)/2 - P_s]/[P_6 - (P_5 + P_1)/2] \quad (11k)$$

$$C_{Pt6} = (P_6 - P_t)/[P_6 - (P_5 + P_1)/2] \quad (11l)$$

and

$$C_{Ps7} = (\bar{P} - P_s)/(P_7 - \bar{P}) \quad (11m)$$

$$C_{Pt7} = (\bar{P} - P_t)/(P_7 - \bar{P}) \quad (11n)$$

In theory, the static and total pressure can be reconstructed for any flow angularity. For example, in unseparated flow the static and total pressures are reconstructed from the probe pressures and estimated flow angularity as:

$$P_t = P_7 - C_{Pt7}(\alpha,\beta)(P_7 - \bar{P}) \quad (12)$$

and

$$P_s = P_7 - C_{Ps7}(\alpha,\beta)(P_7 - \bar{P}) \quad (13)$$

The reconstruction coefficients are well defined over their useful ranges; select matrices are tabulated in figures 9(a)–9(d).

Special care must be taken when recording the data used for calibration to obtain a correct estimate of static pressure. While total pressure remains invariant in an inviscid flow, an actual wind tunnel will exhibit an axial static pressure gradient. If an incorrect estimate of static pressure is made at calibration time, the probes will estimate an incorrect dynamic pressure in operation.

Algorithm—Part II: Static and Total Pressure

In terms of the computational algorithm, recall that the flow angularity computations have determined both the appropriate sector, i, and the flow angularity (α,β). These coefficients are used to perform a pair of bilinear interpolations upon the appropriate static and total pressure coefficient matrix.

Step 9: Interpolate C_{Psi} from α,β and i.

Step 10: Compute static pressure, P_s.

Step 11: Interpolate C_{PTi} from α,β and i.

Step 12: Compute total pressure, P_t.

Implementation of this algorithm is extremely efficient. On a 486 PC, a computational throughput of several hundred reductions per second is realized. A real-time computation of flow properties is possible.

Experimental Verification Turbulent Wake Flow

Results from two recent experiments will be shown. Both are studies of the downstream wakes of circular cylinders. Figures 10(a) and 10(b) derive from an experiment made in a small developmental wind tunnel at the Ames Fluid Mechanics Laboratory; figures 11 and 12 derive from an experiment made at the Ames 7- by 10-Foot Subsonic Wind Tunnel No. 1. The natural pneumatic dissipation of the probe tubing limits the frequency response to approximately 2 Hz. The Karman vortex street shed by the cylinders occurs at a much greater frequency: approximately 250 Hz for the Re = 40,000 experiment and approximately 60 Hz for the Re = 400,000 experiment. The probe will consider the vortex street a source of unsteady flow.

Figure 10(a) has the probe located 1.5 diameters downstream of the cylinder. The two traces correspond to the total pressure in the free stream and the total pressure with the probe immersed to the side of the near-field vortex street. The probe is sampled first at 200 Hz, then at 100 Hz; both in excess of the probe's low-pass pneumatic limit. It can be seen that the probe gives a uniform response to either a steady state or turbulent flow.

Figure 10(b) is of similar geometry, but with the probe moved farther back, to x/d = 5.5. The integration time is extended to 2.5 seconds, for an effective sample rate of 0.4 Hz. The values for total pressure remain constant. These two figures imply that the probe exhibits essentially steady state behavior when immersed in an unsteady flow with a frequency content far above the probe limit.

Additional data were taken to compare the performance of the seven-hole probe against a hot-film anemometer. For this experiment, the seven-hole probe was configured as it is to make two-dimensional wake surveys. The probe is slowly (0.5 inch per second) translated across the wake. Flow properties are measured in real time (three samples of 0.25 second integration time per second), and then gated to the desired spatial resolution.

Figure 11 shows that there is close agreement between the seven-hole probe and hot-film anemometer in the free-stream velocity measurements. However, the seven-hole probe tends to consistently overestimate the velocity when immersed in the vortex street. There is some scatter in the seven-hole data, attributable to the short-time integration interacting with buffeting of the probe boom (unavoidable in production testing). Figure 12 reveals that the root-mean-square unsteady velocity levels in the vortex street are as high as 25% of the local mean

velocity. The overprediction of dynamic pressure, however unwelcome, is consistent with Goldstein (ref. 3), Becker (ref. 5), and Walshe (ref. 6).

Error Analysis

A Monte Carlo simulation may be made to assess the sensitivity of the seven-hole probe algorithm to random error. A series of perturbations of increasing magnitude were statistically applied to basic pressure patterns corresponding to a flow angularity of $\alpha = 0°$, $\beta = 0°$ and $\alpha = 20°$, $\beta = -20°$, respectively. The results are shown in table 1. A 1% deviation in port pressures tends to lead to an uncertainty in flow angularity of approximately 2° at low flow angles, but tends toward unpredictably large excursions at high flow angularity; this is due to the shallow slopes of the separated-flow flow-angularity functions. Consequently, it is desirable to restrict the probe to as narrow a range of flow angularity as possible.

Concluding Remarks

Seven-hole cone probes are an accepted means to measure the essential mean properties of fluid flow. A rapid computational algorithm which incorporates as few as four bilinear interpolations and no more than six interpolations (16 to 24 array references and simple arithmetic) is presented. This has produced a computational algorithm efficient enough to allow the flow properties to be implemented in real time. Unsteady flow properties with frequency content below the pneumatic limit of the probe may be resolved. Higher frequency unsteady flow will tend to bias the probe into indicating a dynamic pressure in excess of the actual value. Over the region of calibration space where the flow has not separated, the flow angularity computations are robust. At high flow angularity, the separated flow coefficients become very sensitive to small perturbations. The computed flow directionality at high flow angularity is best left for qualitative, rather than quantitative, presentation.

Some issues require further investigation. There is a need to:

- Develop design guidelines to define the optimal balance between probe size and frequency response.
- Determine the limiting frequency where a quasi-steady calibration can be applied.
- Find well behaved flow angularity coefficients which are more tolerant of random error, at high flow angularity, than the Gallington set. (This author has tried many with little success.)
- Use seven-hole probes, in the absence of better flow angularity coefficients, in a semi-nulling configuration (where the peak flow angles are restricted).
- Address the meaning of mean flow properties, particularly static pressure, in a turbulent flow.
- Address the concept of unsteady, bandwidth limited static pressure, as reported by a real time multi-hole pitot probe, in turbulent flows.

References

1. Prandtl, L.; and Tietjens, O. G.: Applied Hydro- and Aeromechanics. McGraw-Hill, New York, 1934.
2. Schulze, W. M.; Ashby, G. C.; and Erwin, J. R.: Several Combination Probes for Surveying Static and Total Pressure and Flow Direction. NACA TN-2830, 1952.
3. Goldstein, S.: A Note on the Measurement of Total Head and Static Pressure in a Turbulent Stream. Proc. Roy. Soc. London A, vol. 155, 1936, pp. 570–575.
4. Jenkins, R. C.: Effects of Pitot Probe Shape on Measurement of Flow Turbulence. AIAA J., vol. 25, no. 6, 1987, pp. 889–892.
5. Becker, H. A.; and Brown, A. P. G.: Response of Pitot Probes in Turbulent Streams. J. Fluid Mech., vol 62, pt. 1, 1974, pp. 85–114.
6. Walsh, D. E.; and Garner, H. C.: Usefulness of Various Pressure Probes in Fluctuating Low-Speed Flow. British ARC Rpt. 21714, FM 2917, Feb. 1960.
7. Whitmore, S. A.; Moes, T. R.; and Larson, T. J.: Preliminary Results from a Subsonic High Angle-of-Attack Flush Airdata Sensing (HI-FADS) System: Design, Calibration and Flight-Test Evaluation. NASA TM-101713, 1990.
8. Whitmore, S. A.; and Moes, T. R.: Failure Detection and Fault Management Techniques for a Pneumatic High-Angle of Attack Flush Airdata Sensing System. NASA TM-4335, 1992.
9. Gallington, R. W.: Measurement of Very Large Flow Angles with Non-Nulling Seven-Hole Probes. USAFA-TR-80-17, 1980.
10. Zilliac, G. G.: Calibration of Seven Hole Pressure Probes for Use in Fluid Flows with Large Angularity. NASA TM-102200, 1990.

Table 1. Monte Carlo error analysis—seven-hole probe computational algorithm

Probe pressure port perturbation	$\alpha = 0°/\beta = 0°$ RMS error	$\alpha = 20°/\beta = -20°$ RMS error
±0%	±0°	±0°
±1/4%	±0.5°/±0.5°	±1.0°/±0.7°
±1/2%	±1°/±1°	±2°/±1.5°
±1%	±2°/±2°	±7.5°/±16°

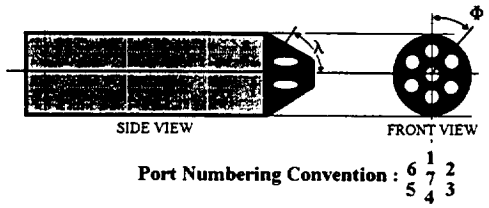

Figure 1. General schematic of a seven-hole probe.

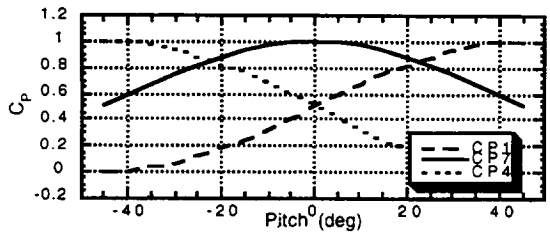

Figure 2(a). Pressure due to pitch. Empirical (potential flow) model. Meridional pressure ports (1, 7, and 4). ε = 0; chosen to match experimental data at 0° pitch angle.

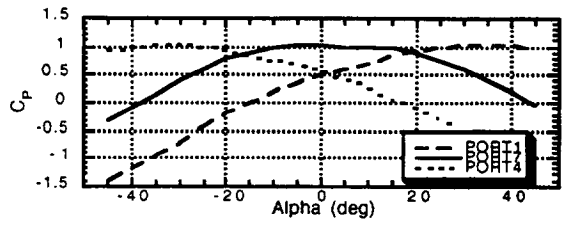

Figure 2(b). Pressure due to pitch. Experimental results. Meridional pressure ports (1, 7, and 4)). Pressures from 45° seven-hole cone probe calibration.

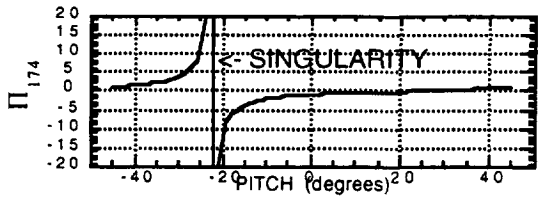

Figure 3(a). Simple, dimensionless pitch-sensitive pressure ratio as a function of pitch. Meridional "triple" ($\Pi_{174} = (P_1-P_7)/(P_7-P_4)$). Empirical (potential flow) model.

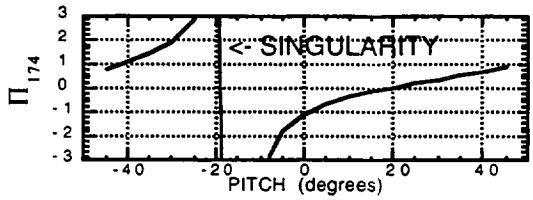

Figure 3(b). Simple, dimensionless pitch-sensitive pressure ratio as a function of pitch. Meridional "triple" ($\Pi_{174} = (P_1-P_7)/(P_7-P_4)$). Experimental results from 45° cone probe.

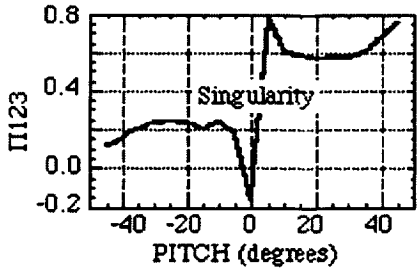

Figure 4(a). Simple, dimensionless pitch-sensitive pressure ratio as a function of pitch. Non-meridional "triple" (Π_{123}). Experimental data from 45° cone probe. Note singularity near 0° pitch angle.

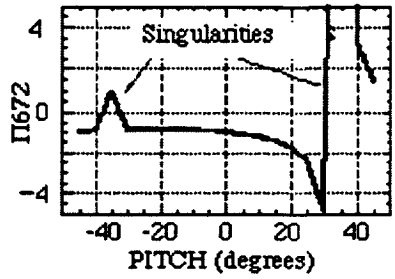

Figure 4(b). Simple, dimensionless pitch-sensitive pressure ratio as a function of pitch. Non-meridional "triple" (Π_{672}). Experimental data from 45° cone probe. Note singularities near –35° and +35° pitch angle.

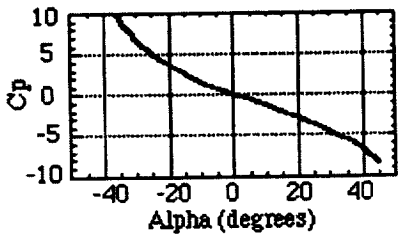

Figure 5. Complex, dimensionless pitch-sensitive pressure ratio, $C_{P\alpha 7}$, as a function of pitch ($\beta = 0$). Experimental results from 45° cone probe.

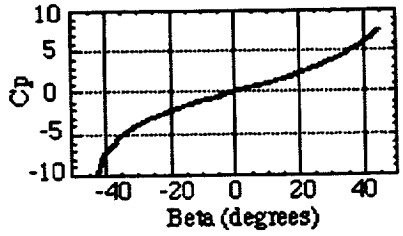

Figure 6. Complex, dimensionless yaw-sensitive pressure ratio, $C_{P\beta 7}$, as a function of yaw ($\alpha = 0$). Experimental results from 45° cone probe.

$C_{P\alpha 7}$

-5.0	+29	+30	+31	+31	+32	+32	+33	+32	+32	+32	+32
-4.0	+25	+25	+26	+26	+26	+27	+28	+27	+28	+28	+28
-3.0	+19	+20	+20	+20	+21	+21	+22	+22	+22	+22	+23
-2.0	+12	+13	+13	+14	+14	+15	+16	+16	+15	+16	+16
-1.0	+6	+6	+6	+7	+8	+8	+9	+9	+9	+9	+10
+0.0	-1	-1	-1	-0	+0	+1	+2	+2	+2	+3	+3
+1.0	-7	-7	-7	-7	-7	-6	-5	-5	-4	-4	-4
+2.0	-12	-12	-12	-13	-13	-12	-12	-11	-11	-11	-11
+3.0	-17	-17	-15	-17	-17	-17	-17	-16	-16	-16	-16
+4.0	-21	-21	-23	-21	-21	-21	-21	-20	-21	-21	-21
+5.0	-24	-25	-25	-24	-25	-25	-24	-24	-25	-25	-26

-5.0 -3.0 -1.0 +1.0 +3.0 +5.0

$C_{P\beta 7}$

Figure 7(a). Pitch transfer function for unseparated flow. Sector 7. $\alpha = f_\alpha (C_{P\alpha 7}, C_{P\beta 7})$.

$C_{P\alpha 7}$

-5.0	-34	-30	-25	-19	-12	-4	+5	+13	+20	+26	+31
-4.0	-35	-31	-26	-20	-12	-3	+6	+14	+22	+28	+33
-3.0	-36	-32	-27	-20	-12	-3	+6	+15	+23	+29	+34
-2.0	-35	-32	-27	-20	-12	-2	+8	+16	+24	+30	+35
-1.0	-34	-31	-26	-20	-11	-2	+8	+17	+24	+30	+35
+0.0	-33	-30	-25	-19	-10	-1	+9	+18	+25	+31	+36
+1.0	-32	-28	-23	-17	-9	+1	+10	+18	+25	+31	+36
+2.0	-31	-27	-22	-16	-8	+1	+10	+18	+25	+31	+36
+3.0	-30	-26	-21	-14	-6	+2	+10	+18	+24	+30	+35
+4.0	-28	-25	-15	-12	-5	+3	+10	+17	+23	+29	+34
+5.0	-26	-22	-16	-11	-4	+3	+10	+17	+23	+28	+33

-5.0 -3.0 -1.0 +1.0 +3.0 +5.0

$C_{P\beta 7}$

Figure 7(b). Yaw transfer function for unseparated flow. Sector 7. $\beta = f_\beta (C_{P\alpha 7}, C_{P\beta 7})$.

$C_{P\alpha 1}$

+0.00	+19	+19	+19	+19	+19	+19	+19	+19	+19
+0.25	+23	+23	+22	+22	+22	+22	+22	+21	+22
+0.50	+27	+27	+27	+26	+26	+26	+25	+25	+25
+0.75	+33	+33	+33	+33	+33	+33	+32	+31	+31
+1.00	-22	+44	+45	+45	+45	+45	+44	+42	+40
+1.25	-30	+45	+45	+45	+45	+45	+45	+45	+45
+1.50	-34	-45	+45	+45	+45	+45	+45	+45	-44
+1.75	-37	-45	+45	+45	+45	+45	+45	-45	-45
+2.00	-38	-45	-45	+45	+45	+45	+45	-45	-45

-2.0 -1.00 +0.0 +1.0 +2.0

$C_{P\beta 1}$

Figure 7(c). Pitch transfer function for separated flow. Sector 1. $\alpha = f_\alpha (C_{P\alpha 1}, C_{P\beta 1})$.

$C_{P\alpha 1}$

+0.00	+9	+6	+3	+0	-3	-5	-8	-11	-13
+0.25	+10	+7	+3	+0	-3	-6	-9	-12	-14
+0.50	+12	+8	+4	+0	-3	-7	-10	-13	-16
+0.75	+15	+10	+5	+1	-4	-8	-13	-16	-19
+1.00	-45	+14	+9	+2	-4	-11	-16	-20	-23
+1.25	-39	+15	+9	+2	-4	-11	-17	-21	-25
+1.50	-35	-36	+9	+2	-5	-11	-17	-21	+45
+1.75	-32	-33	+9	+2	-5	-11	-17	+45	+45
+2.00	-29	-30	-29	+2	-5	-11	-17	+43	+42

-2.0 -1.0 +0.0 +1.0 +2.0

$C_{P\beta 1}$

Figure 7(d). Yaw transfer function for separated flow. Sector 1. $\beta = f_\beta (C_{P\alpha 1}, C_{P\beta 1})$.

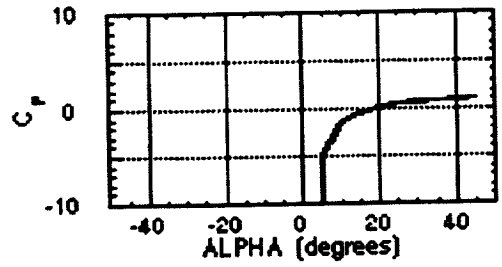

Figure 8(a). Complex, dimensionless pitch-sensitive separated flow pressure ratio, $C_{P\alpha 1}$, as a function of pitch ($\beta = 0$).

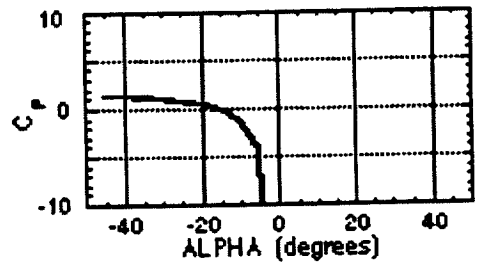

Figure 8(b). Complex, dimensionless pitch-sensitive, separated flow pressure ratio, $C_{P\alpha 4}$, as a function of pitch ($\beta = 0$).

β \ α	-45	-35	-25	-15	-5	+5	+15	+25	+35	+45
+5					-6.2	-5.9				
+15				-11.5	-1.4	-0.7	-0.8	-1.6	-8.0	
+25				-1.7	-0.3	-0.1	-0.1	-0.4	-1.7	
+35			-5.6	-0.8	-0.1	+0.0	+0.0	-0.1	-0.8	-5.4
+45	+8.9	-2.7	-0.6	-0.1	-0.0	-0.0	-0.1	-0.6	-2.8	+8.2

Figure 9(a). Static pressure coefficients for separated flow. Sector 1. C_{PS1}.

β \ α	-45	-35	-25	-15	-5	+5	+15	+25	+35	+45
+5					-6.2	-5.9				
+15				-11.5	-1.4	-0.7	-0.8	-1.6	-8.0	
+25				-1.7	-0.3	-0.1	-0.1	-0.4	-1.7	
+35			-5.6	-0.8	-0.1	+0.0	+0.0	-0.1	-0.8	-5.4
+45	+8.9	-2.7	-0.6	-0.1	-0.0	-0.0	-0.1	-0.6	-2.8	+8.2

Figure 9(b). Total pressure coefficients for separated flow. Sector 1. C_{Pt1}.

β \ α	-45	-35	-25	-15	-5	+5	+15	+25	+35	+45
-45										
-35		-8.7	-4.8	-3.4	-2.7	-2.6	-2.8	-3.4	-4.5	
-25		-4.4	-2.2	-1.4	-1.0	-0.9	-1.2	-1.8	-2.8	
-15		-2.9	-1.3	-0.6	-0.3	-0.3	-0.6	-1.1	-2.2	
-5		-2.3	-0.8	-0.2	-0.0	-0.0	-0.2	-0.8	-1.8	
+5		-2.1	-0.7	-0.2	-0.0	-0.0	-0.2	-0.8	-1.8	
+15		-2.0	-0.8	-0.3	-0.1	-0.1	-0.3	-0.9	-1.9	
+25		-2.7	-1.3	-0.8	-0.5	-0.5	-0.7	-1.3	-2.4	
+35		-4.1	-2.4	-1.6	-1.3	-1.3	-1.6	-2.2	-3.5	
+45										

Figure 9(c). Static pressure coefficients for unseparated flow. Sector 7. C_{PS7}.

β \ α	-45	-35	-25	-15	-5	+5	+15	+25	+35	+45
-45										
-35		-8.7	-4.8	-3.4	-2.7	-2.6	-2.8	-3.4	-4.5	
-25		-4.4	-2.2	-1.4	-1.0	-0.9	-1.2	-1.8	-2.8	
-15		-2.9	-1.3	-0.6	-0.3	-0.3	-0.6	-1.1	-2.2	
-5		-2.3	-0.8	-0.2	-0.0	-0.0	-0.2	-0.8	-1.8	
+5		-2.1	-0.7	-0.2	-0.0	-0.0	-0.2	-0.8	-1.8	
+15		-2.0	-0.8	-0.3	-0.1	-0.1	-0.3	-0.9	-1.9	
+25		-2.7	-1.3	-0.8	-0.5	-0.5	-0.7	-1.3	-2.4	
+35		-4.1	-2.4	-1.6	-1.3	-1.3	-1.6	-2.2	-3.5	
+45										

Figure 9(d). Total pressure coefficients for unseparated flow. Sector 7. C_{Pt7}.

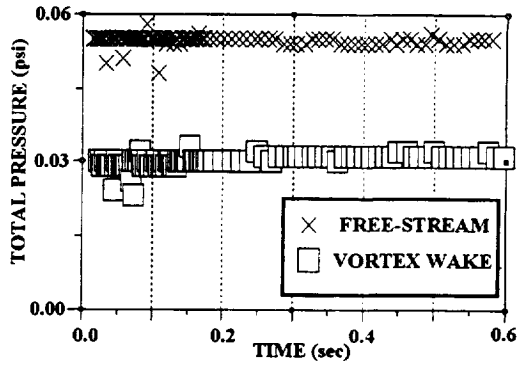

Figure 10(a). Time history of flow. Data from cylinder wake test (Re = 40,000, x/d = 1.5). Two spatial locations chosen—one in the free stream, the other in the vortex wake. 200 and 100 Hz sampling rates. Note minor "scatter" for 200 Hz sample rate (due to electrical noise) and minimal "scatter" for 100 Hz sample rate. There is no appreciable detection of the flow unsteadiness in the vortex wake.

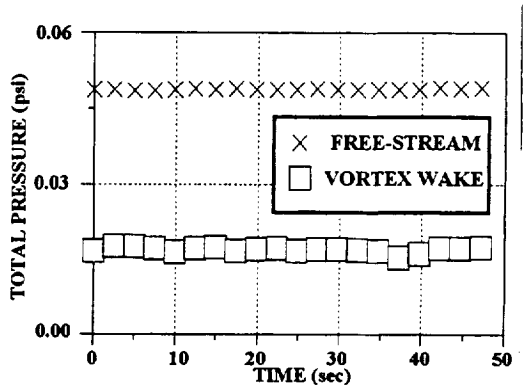

Figure 10(b). Time history in wake. Data from cylinder wake test (Re = 40,000, x/d = 5.5). Two spatial locations chosen—one in the free stream, the other in the vortex wake. 2.5 second integration time, 0.4 Hz sample rate.

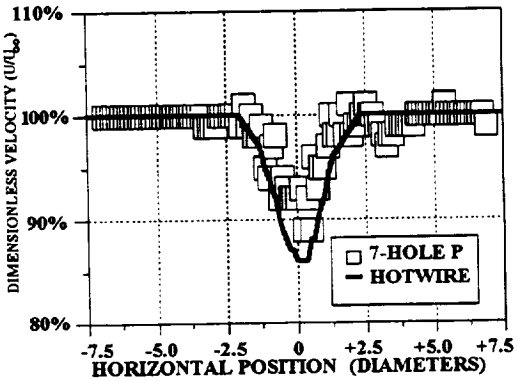

Figure 11. Axial velocity in cylinder wake (Re = 400,000, x/d = 7.5). Seven-hole and hot-film anemometry. 4 Hz data acquisition rate (0.1 second pressure integration time).

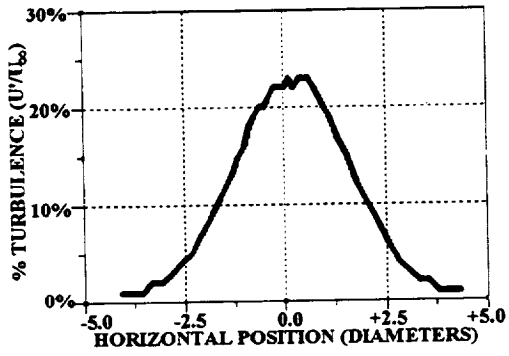

Figure 12. Turbulence in cylinder wake (Re = 400,000, x/d = 7.5) from hot-film anemometry.

Appendix A—Additional Figures

$C_{P\alpha 1}$

	-2.0	-1.00	+0.0	+1.0	+2.0
+0.00	+19	+19	+19	+19	+19
+0.25	+23	+23	+22	+22	+22
+0.50	+27	+27	+27	+26	+26
+0.75	+33	+33	+33	+33	+33
+1.00	-22	+44	+45	+45	+45
+1.25	-30	+45	+45	+45	+45
+1.50	-34	-45	+45	+45	+45
+1.75	-37	-45	+45	+45	+45
+2.00	-38	-45	-45	+45	+45

$C_{P\beta 1}$

Figure 13(a). Transfer function for separated flow. $\alpha = f_\alpha (C_{P\alpha 1}, C_{P\beta 1})$.

$C_{P\alpha 1}$

	-2.0	-1.0	+0.0	+1.0	+2.0
+0.00	+19	+18	+17	+15	+14
+0.25	+21	+20	+19	+18	+16
+0.50	+24	+23	+22	+20	+18
+0.75	+28	+27	+25	+24	+22
+1.00	+35	+33	+32	+30	+28
+1.25	-39	+45	+45	+45	+42
+1.50	-42	-45	+45	+45	+42
+1.75	-44	-45	+45	+45	+42
+2.00	-45	-45	+45	+45	+42

$C_{P\beta 1}$

Figure 13(d). Transfer function for separated flow. $\beta = f_\beta (C_{P\alpha 2}, C_{P\beta 2})$.

$C_{P\alpha 1}$

	-2.0	-1.0	+0.0	+1.0	+2.0
+0.00	+9	+6	+3	+0	-3
+0.25	+10	+7	+3	+0	-3
+0.50	+12	+8	+4	+0	-3
+0.75	+15	+10	+5	+1	-4
+1.00	-45	+14	+9	+2	-4
+1.25	-39	+15	+9	+2	-4
+1.50	-35	-36	+9	+2	-5
+1.75	-32	-33	+9	+2	-5
+2.00	-29	-30	-29	+2	-5

$C_{P\beta 1}$

Figure 13(b). Transfer function for separated flow. $\beta = f_\beta (C_{P\alpha 1}, C_{P\beta 1})$.

$C_{P\alpha 3}$

	-2.0	-1.00	+0.0	+1.0	+2.0
+0.00	-16	-14	-12	-9	-7
+0.25	-18	-16	-13	-11	-8
+0.50	-21	-18	-16	-13	-10
+0.75	-25	-23	-20	-17	-13
+1.00	-32	-30	-30	-25	-21
+1.25	-45	-45	-45	-45	-41
+1.50	-45	-45	-45	-45	-42
+1.75	+45	-45	-45	-45	-43
+2.00	+45	-45	-45	-45	-45

$C_{P\beta 3}$

Figure 13(e). Transfer function for separated flow. $\alpha = f_\alpha (C_{P\alpha 3}, C_{P\beta 3})$.

$C_{P\alpha 2}$

	-2.0	-1.00	+0.0	+1.0	+2.0
+0.00	+4	+6	+8	+10	+12
+0.25	+4	+7	+9	+11	+13
+0.50	+4	+7	+10	+13	+15
+0.75	+5	+8	+11	+15	+18
+1.00	+6	+11	+15	+20	+24
+1.25	+45	+18	+27	+38	+45
+1.50	+42	+45	+28	+39	+45
+1.75	+38	+44	+28	+39	+45
+2.00	+35	+36	+28	+40	+45

$C_{P\beta 2}$

Figure 13(c). Transfer function for separated flow. $\alpha = f_\alpha (C_{P\alpha 2}, C_{P\beta 2})$.

$C_{P\alpha 3}$

	-2.0	-1.0	+0.0	+1.0	+2.0
+0.00	+10	+11	+13	+13	+14
+0.25	+12	+13	+14	+15	+17
+0.50	+14	+15	+16	+18	+20
+0.75	+16	+18	+20	+22	+24
+1.00	+20	+23	+27	+29	+32
+1.25	+25	+29	+34	+40	+45
+1.50	+25	+29	+34	+40	+45
+1.75	+0	+29	+34	+40	+45
+2.00	-3	+29	+34	+40	+45

$C_{P\beta 3}$

Figure 13(f). Transfer function for separated flow. $\beta = f_\beta (C_{P\alpha 2}, C_{P\beta 2})$.

$C_{P\alpha 4}$

	-2.0	-1.00	+0.0	+1.0	+2.0
+0.00	-17 -16 -16 -16 -16 -16 -16 -16 -16				
+0.25	-18 -18 -18 -18 -18 -18 -18 -18 -18				
+0.50	-21 -21 -21 -21 -21 -21 -21 -20 -20				
+0.75	-25 -24 -25 -25 -24 -24 -24 -20 -24				
+1.00	-30 -31 -31 -30 -30 -30 -30 -30 -30				
+1.25	+45 -43 -42 -43 -43 -41 -43 -41 -40				
+1.50	+45 -45 -45 -45 -45 -45 -45 -45 +37				
+1.75	+45 -45 -45 -45 -45 -45 -45 +45 +39				
+2.00	+45 -45 -45 -45 -45 -45 -45 +45 +41				

$C_{P\beta 4}$

Figure 13(g). Transfer function for separated flow. $\alpha = f_\alpha (C_{P\alpha 4}, C_{P\beta 4})$.

$C_{P\alpha 5}$

	-2.0	-1.0	+0.0	+1.0	+2.0
+0.00	-19 -18 -17 -15 -14 -13 -11 -10 -8				
+0.25	-21 -20 -19 -17 -15 -14 -13 -11 -9				
+0.50	-24 -22 -21 -19 -19 -16 -14 -12 -10				
+0.75	-27 -25 -23 -22 -20 -18 -15 -13 -11				
+1.00	-31 -29 -28 -25 -24 -22 -18 -15 -13				
+1.25	-38 -37 -35 -32 -29 -27 -23 -20 -15				
+1.50	-45 -45 -45 -45 -41 -34 -29 -23 -18				
+1.75	-45 -45 -45 -45 -41 -35 -29 -23 -10				
+2.00	-45 -45 -45 -45 -42 -35 -29 -24 -7				

$C_{P\beta 5}$

Figure 13(j). Transfer function for separated flow. $\beta = f_\beta (C_{P\alpha 5}, C_{P\beta 5})$.

$C_{P\alpha 4}$

	-2.0	-1.0	+0.0	+1.0	+2.0
+0.00	-8 -6 -3 -1 +1 +3 +6 +8 +10				
+0.25	-9 -6 -4 -1 +2 +4 +7 +9 +11				
+0.50	-10 -7 -4 -1 +2 +5 +8 +11 +13				
+0.75	-11 -8 -4 -1 +3 +6 +9 -20 +15				
+1.00	-13 -9 -4 +0 +4 +8 +12 +15 +18				
+1.25	+45 -12 -6 -0 +5 +10 +16 +20 +23				
+1.50	+43 -13 -7 -0 +5 +11 +17 +22 -45				
+1.75	+41 -13 -7 -0 +5 +11 +17 -44 -44				
+2.00	+39 -13 -7 -0 +5 +11 +17 -42 -42				

$C_{P\beta 4}$

Figure 13(h). Transfer function for separated flow. $\beta = f_\beta (C_{P\alpha 4}, C_{P\beta 4})$.

$C_{P\alpha 6}$

	-2.0	-1.00	+0.0	+1.0	+2.0
+0.00	+19 +17 +14 +12 +9 +6 +3 +0 -3				
+0.25	+21 +19 +16 +13 +11 +7 +4 +0 -3				
+0.50	+25 +23 +19 +16 +13 +9 +5 +0 -4				
+0.75	+31 +28 +25 +22 +17 +12 +6 +0 -5				
+1.00	+39 +38 +36 +32 +27 +19 +10 +1 -7				
+1.25	+45 +45 +45 +45 +45 +26 +12 +0 -10				
+1.50	-45 +45 +45 +45 +45 +27 +12 -0 -10				
+1.75	-45 +45 +45 +45 +45 +27 +11 -1 +40				
+2.00	-45 -45 +45 +45 +45 +28 +10 -1 +38				

$C_{P\beta 6}$

Figure 13(k). Transfer function for separated flow. $\alpha = f_\alpha (C_{P\alpha 6}, C_{P\beta 6})$.

$C_{P\alpha 5}$

	-2.0	-1.00	+0.0	+1.0	+2.0
+0.00	-3 -5 -7 -8 -10 -12 -13 -15 -16				
+0.25	-3 -5 -8 -9 -11 -13 -15 -17 -18				
+0.50	-4 -6 -8 -10 -20 -15 -17 -19 -20				
+0.75	-4 -7 -9 -12 -16 -21 -20 -22 -24				
+1.00	-5 -8 -12 -15 -20 -24 -25 -26 -29				
+1.25	-7 -12 -17 -21 -25 -29 -33 -37 -37				
+1.50	-9 -18 -28 -41 -45 -45 -45 -45 -45				
+1.75	-9 -18 -28 -41 -45 -45 -45 -45 +45				
+2.00	-9 -18 -29 -42 -45 -45 -45 -45 +45				

$C_{P\beta 5}$

Figure 13(i). Transfer function for separated flow. $\alpha = f_\alpha (C_{P\alpha 5}, C_{P\beta 5})$.

$C_{P\alpha 6}$

	-2.0	-1.0	+0.0	+1.0	+2.0
+0.00	-13 -14 -15 -15 -16 -18 -18 -19 -19				
+0.25	-14 -15 -17 -18 -19 -20 -21 -22 -22				
+0.50	-16 -18 -19 -21 -22 -23 -24 -25 -26				
+0.75	-19 -21 -23 -25 -27 -28 -29 -30 -31				
+1.00	-23 -25 -29 -32 -36 -39 -39 -39 -38				
+1.25	-25 -28 -33 -40 -45 -45 -45 -45 -45				
+1.50	+5 -28 -33 -40 -45 -45 -45 -45 -45				
+1.75	+8 -28 -33 -40 -45 -45 -45 -45 +35				
+2.00	+12 +14 -33 -40 -45 -45 -45 -45 +38				

$C_{P\beta 6}$

Figure 13l(l). Transfer function for separated flow. $\beta = f_\beta (C_{P\alpha 6}, C_{P\beta 6})$.

β \ α	-45	-35	-25	-15	-5	+5	+15	+25	+35	+45
+5					-6.2	-5.9				
+15				-11.5	-1.4	-0.7	-0.8	-1.6	-8.0	
+25				-1.7	-0.3	-0.1	-0.1	-0.4	-1.7	
+35			-5.6	-0.8	-0.1	+0.0	+0.0	-0.1	-0.8	-5.4
+45	+8.9	-2.7	-0.6	-0.1	-0.0	-0.0	-0.1	-0.6	-2.8	+8.2

Figure 14(a). Static pressure coefficients, C_{PS1}.

β \ α	-45	-35	-25	-15	-5	+5	+15	+25	+35	+45
+5					-6.2	-5.9				
+15				-11.5	-1.4	-0.7	-0.8	-1.6	-8.0	
+25				-1.7	-0.3	-0.1	-0.1	-0.4	-1.7	
+35			-5.6	-0.8	-0.1	+0.0	+0.0	-0.1	-0.8	-5.4
+45	+8.9	-2.7	-0.6	-0.1	-0.0	-0.0	-0.1	-0.6	-2.8	+8.2

Figure 14(b). Total pressure coefficients, C_{Pt1}.

β \ α	+5	+15	+25	+35	+45
+5	-6.3	-1.0	-0.4	-0.3	-0.5
+15	-1.6	-0.2	-0.0	-0.1	-0.3
+25	-1.0	-0.1	+0.0	-0.0	-0.2
+35	-1.1	-0.2	-0.0	-0.0	-0.1
+45	-2.1	-0.5	-0.2	-0.1	-0.2

Figure 14(c). Static pressure coefficients, sector 2.

β \ α	+5	+15	+25	+35	+45
+5	-6.3	-1.0	-0.4	-0.3	-0.5
+15	-1.6	-0.2	-0.0	-0.1	-0.3
+25	-1.0	-0.1	+0.0	-0.0	-0.2
+35	-1.1	-0.2	-0.0	-0.0	-0.1
+45	-2.1	-0.5	-0.2	-0.1	-0.2

Figure 14(d). Total pressure coefficients, sector 3.

β \ α	+5	+15	+25	+35	+45
-45.0	-13.4	-1.4	-0.5	-0.2	-0.2
-35.0	-4.3	-0.8	-0.2	-0.1	-0.1
-25.0	-2.3	-0.4	-0.0	-0.0	-0.1
-15.0	-1.8	-0.2	-0.0	-0.0	-0.1
-5.0	-2.5	-0.4	-0.0	-0.0	-0.2

Figure 14(e). Static pressure coefficients, sector 3.

β \ α	+5.0	+15.0	+25.0	+35.0	+45.0
-45.0	-13.4	-1.4	-0.5	-0.2	-0.2
-35.0	-4.3	-0.8	-0.2	-0.1	-0.1
-25.0	-2.3	-0.4	-0.0	-0.0	-0.1
-15.0	-1.8	-0.2	-0.0	-0.0	-0.1
-5.0	-2.5	-0.4	-0.0	-0.0	-0.2

Figure 14(f). Total pressure coefficients, sector 3.

β \ α	-45	-35	-25	-15	-5	+5	+15	+25	+35	+45
-45	+19.4	-2.6	-0.7	-0.2	-0.1	-0.1	-0.1	-0.5	-1.8	+81.3
-35		-4.5	-0.9	-0.2	-0.0	-0.0	-0.1	-0.6	-3.1	
25			-1.7	-0.4	-0.1	-0.0	-0.3	-1.3	-13.7	
15			-6.3	-1.4	-0.7	-0.6	-1.3	-5.6		
-5				-14.5	-5.0	-4.7	-26.2			

Figure 14(g). Static pressure coefficients, sector 4.

β \ α	-45	-35	-25	-15	-5	+5	+15	+25	+35	+45
-45	+19.4	-2.6	-0.7	-0.2	-0.1	-0.1	-0.1	-0.5	-1.8	+81.3
-35		-4.5	-0.9	-0.2	-0.0	-0.0	-0.1	-0.6	-3.1	
-25		-29.0	-1.7	-0.4	-0.1	-0.0	-0.3	-1.3	-13.7	
-15			-6.3	-1.4	-0.7	-0.6	-1.3	-5.6		
-5				-14.5	-5.0	-4.7	-26.2			

Figure 14(h). Total pressure coefficients, sector 5.

β					
-45.0	-0.2	-0.1	-0.2	-0.5	-1.3
-35.0	-0.2	-0.1	-0.1	-0.2	-0.9
-25.0	-0.2	-0.0	-0.0	-0.1	-0.8
-15.0	-0.3	-0.1	-0.1	-0.3	-1.5
-5.0	-0.5	-0.3	-0.3	-0.9	-4.5
	-45.0	-35.0	-25.0	-15.0	-5.0
			α		

Figure 14(i). Static pressure coefficients, sector 5.

β					
-45.0	-0.2	-0.1	-0.2	-0.5	-1.3
-35.0	-0.2	-0.1	-0.1	-0.2	-0.9
-25.0	-0.2	-0.0	-0.0	-0.1	-0.8
-15.0	-0.3	-0.1	-0.1	-0.3	-1.5
-5.0	-0.5	-0.3	-0.3	-0.9	-4.5
	-45.0	-35.0	-25.0	-15.0	-5.0
			α		

Figure 14(j). Total pressure coefficients, sector 5.

β					
+5.0	-0.1	-0.0	-0.1	-0.6	-4.9
+15.0	-0.1	+0.0	-0.0	-0.4	-3.4
+25.0	-0.1	+0.0	-0.0	-0.5	-4.6
+35.0	-0.1	-0.1	-0.2	-1.0	-11.4
+45.0	-0.2	-0.2	-0.5	-1.9	+86.8
	-45.0	-35.0	-25.0	-15.0	-5.0
			α		

Figure 14(k). Static pressure coefficients, sector 6.

β					
+5.0	-0.1	-0.0	-0.1	-0.6	-4.9
+15.0	-0.1	+0.0	-0.0	-0.4	-3.4
+25.0	-0.1	+0.0	-0.0	-0.5	-4.6
+35.0	-0.1	-0.1	-0.2	-1.0	-11.4
+45.0	-0.2	-0.2	-0.5	-1.9	+86.8
	-45.0	-35.0	-25.0	-15.0	-5.0
			α		

Figure 14l(l). Total pressure coefficients, sector 6.

Appendix B—Calibration Code (Microsoft QuickBasic)

```
DECLARE FUNCTION min! (A!, B!, C!, d!, e!, F!, g!)

REM ************************************************************************
REM ************************************************************************
REM **                                                                    **
REM ** SHPCAL.BAS - program to build a calibration file for SHP.BAS       **
REM **                                                                    **
REM **  Program by Timothy Takahashi - National Research Council         **
REM **                                                                    **
REM ** Last Revision : October 5th, 1995                                  **
REM **                                                                    **
REM ** PC-Compatible / MS-DOS 6.2 / Microsoft Quick Basic v. 4.5          **
REM **                                                                    **
REM ************************************************************************
REM ************************************************************************

REM $DYNAMIC


REM >>> Dimension working arrays

OPTION BASE 1
DIM table1(7, 20, 20)
DIM table2(7, 20, 20)
DIM table3(7, 20, 20)
DIM table4(7, 20, 20)

DIM pres(400, 8)
DIM alpha(400), beta(400), NDpres(8)

REM >>> Open output files

OPEN "O", 2, "c:\qbasic\pcdas\shpcal.dat"

REM >>> Initialize working arrays

FOR probe% = 1 TO 3
PRINT "PROBE "; probe%
 PRINT "Initializing..."
 FOR i = 1 TO 20
  FOR j = 1 TO 20
   FOR k = 1 TO 7
   table1(k, j, i) = 99
   table2(k, j, i) = 99
   table3(k, j, i) = 99
   table4(k, j, i) = 99
   NEXT k
  NEXT j
 NEXT i

REM >>> Read calibration file into working arrays
```

```
PRINT "READING CALIBRATION PRESSURES"

OPEN "I", 1, "c:\qbasic\pcdas\calpr.dat"
i% = 1
110 :
IF EOF(1) THEN 120
INPUT #1, alpha(i%)
INPUT #1, beta(i%)

INPUT #1, pres(i%, 8)

FOR P% = 1 TO 3
 IF P% = probe% THEN
  INPUT #1, pres(i%, 1)
  INPUT #1, pres(i%, 2)
  INPUT #1, pres(i%, 3)
  INPUT #1, pres(i%, 4)
  INPUT #1, pres(i%, 5)
  INPUT #1, pres(i%, 6)
  INPUT #1, pres(i%, 7)
 ELSE
  INPUT #1, x
  INPUT #1, x
  INPUT #1, x
  INPUT #1, x
  INPUT #1, x
  INPUT #1, x
  INPUT #1, x
 END IF
NEXT P%
i% = i% + 1
GOTO 110

120 :  ncal = i% - 1
CLOSE 1


PRINT "Searching for Reference Total Pressure"

REM >>> Determine the reference total and static pressures.
REM >>> The reference total is p7/p8 at alpha=beta=0.

      FOR i = 1 TO ncal
          IF (ABS(alpha(i)) < 1 AND ABS(beta(i)) < 1) THEN
              IER = 1
              PTOT = pres(i, 7) / pres(i, 8)
              PDEN = pres(i, 8)
              PBARL = (pres(i, 1) + pres(i, 2) + pres(i, 3) + pres(i, 4) + pres(i,
5) + pres(i, 6)) / 6!
              GOTO 90
          END IF
      NEXT i
      PRINT "No reference total pressure found"
```

```
      STOP

90 :
      PRINT "REFERENCE TOTAL PRESSURE = "; PTOT * PDEN; "PSI"

      PST = 0
      PSTAT = PST / PDEN

REM >>> Compute Calibration Coefficients

      PRINT "Computing Coefficients"

      FOR i = 1 TO ncal

REM >>> non-dimensionalize pressures

          FOR j = 1 TO 8
              NDpres(j) = pres(i, j) / pres(i, 8)
          NEXT j

          ix% = (alpha(i)) / 5 + 10
          iy% = (beta(i)) / 5 + 10

REM >>>  Check to make sure pressure at hole 7 is not the lowest
REM >>>  pressure (Second check for separation at hole 7).

          PMIN = min(NDpres(1), NDpres(2), NDpres(3), NDpres(4), NDpres(5),
NDpres(6), NDpres(7))
          IF (NDpres(7) = PMIN) THEN 150

REM >>>  Compute the Coefficients for each Sector

REM >>>  Zone 1.

               PBAR1 = (NDpres(2) + NDpres(6)) / 2!
               PDIFF = NDpres(1) - (NDpres(2) + NDpres(6)) / 2!
               IF (ABS(PDIFF) <= .0000001#) THEN PDIFF = .0000001#
               CT1 = (NDpres(1) - NDpres(7)) / PDIFF
               CF1 = (NDpres(6) - NDpres(2)) / PDIFF
               CPTOT1 = (NDpres(1) - PTOT) / PDIFF
               CPSTA1 = (PBAR1 - PSTAT) / PDIFF
                   table1(1, ix%, iy%) = CT1
                   table2(1, ix%, iy%) = CF1
                   table3(1, ix%, iy%) = CPTOT1
                   table4(1, ix%, iy%) = CPSTA1

REM >>>  Zone 2.
               PDIFF = NDpres(2) - (NDpres(1) + NDpres(3)) / 2!
               PBAR2 = (NDpres(1) + NDpres(3)) / 2!
               IF (ABS(PDIFF) <= .0000001#) THEN PDIFF = .0000001#
               CT2 = (NDpres(2) - NDpres(7)) / PDIFF
               CF2 = (NDpres(1) - NDpres(3)) / PDIFF
               CPTOT2 = (NDpres(2) - PTOT) / PDIFF
               CPSTA2 = (PBAR2 - PSTAT) / PDIFF
                   table1(2, ix%, iy%) = CT2
                   table2(2, ix%, iy%) = CF2
```

```
                table3(2, ix%, iy%) = CPTOT2
                table4(2, ix%, iy%) = CPSTA2

REM >>>  Zone 3.
            PDIFF = NDpres(3) - (NDpres(2) + NDpres(4)) / 2!
            PBAR3 = (NDpres(2) + NDpres(4)) / 2!
            IF (ABS(PDIFF) <= .0000001#) THEN PDIFF = .0000001#
            CT3 = (NDpres(3) - NDpres(7)) / PDIFF
            CF3 = (NDpres(2) - NDpres(4)) / PDIFF
            CPTOT3 = (NDpres(3) - PTOT) / PDIFF
            CPSTA3 = (PBAR3 - PSTAT) / PDIFF
                table1(3, ix%, iy%) = CT3
                table2(3, ix%, iy%) = CF3
                table3(3, ix%, iy%) = CPTOT3
                table4(3, ix%, iy%) = CPSTA3

REM >>>  Zone 4.
            PDIFF = NDpres(4) - (NDpres(3) + NDpres(5)) / 2!
            PBAR4 = (NDpres(3) + NDpres(5)) / 2!
            IF (ABS(PDIFF) <= .0000001#) THEN PDIFF = .0000001#
            CT4 = (NDpres(4) - NDpres(7)) / PDIFF
            CF4 = (NDpres(3) - NDpres(5)) / PDIFF
            CPTOT4 = (NDpres(4) - PTOT) / PDIFF
            CPSTA4 = (PBAR4 - PSTAT) / PDIFF
            table1(4, ix%, iy%) = CT4
            table2(4, ix%, iy%) = CF4
            table3(4, ix%, iy%) = CPTOT4
            table4(4, ix%, iy%) = CPSTA4

REM >>>  Zone 5.
            PDIFF = NDpres(5) - (NDpres(4) + NDpres(6)) / 2!
            PBAR5 = (NDpres(4) + NDpres(6)) / 2!
            IF (ABS(PDIFF) <= .0000001#) THEN PDIFF = .0000001#
            CT5 = (NDpres(5) - NDpres(7)) / PDIFF
            CF5 = (NDpres(4) - NDpres(6)) / PDIFF
            CPTOT5 = (NDpres(5) - PTOT) / PDIFF
            CPSTA5 = (PBAR5 - PSTAT) / PDIFF
            table1(5, ix%, iy%) = CT5
            table2(5, ix%, iy%) = CF5
            table3(5, ix%, iy%) = CPTOT5
            table4(5, ix%, iy%) = CPSTA5

REM >>>  Zone 6.
            PDIFF = NDpres(6) - (NDpres(5) + NDpres(1)) / 2!
            PBAR6 = (NDpres(5) + NDpres(1)) / 2!
            IF (ABS(PDIFF) <= .0000001#) THEN PDIFF = .0000001#
            CT6 = (NDpres(6) - NDpres(7)) / PDIFF
            CF6 = (NDpres(5) - NDpres(1)) / PDIFF
            CPTOT6 = (NDpres(6) - PTOT) / PDIFF
            CPSTA6 = (PBAR6 - PSTAT) / PDIFF
            table1(6, ix%, iy%) = CT6
            table2(6, ix%, iy%) = CF6
            table3(6, ix%, iy%) = CPTOT6
            table4(6, ix%, iy%) = CPSTA6
```

```
REM >>>  Zone 7.
              PBARL = (NDpres(1) + NDpres(2) + NDpres(3) + NDpres(4) + NDpres(5) +
NDpres(6)) / 6!
              PDIFFL = NDpres(7) - PBARL
              CA1 = (NDpres(4) - NDpres(1)) / PDIFFL
              CA2 = (NDpres(3) - NDpres(6)) / PDIFFL
              CA3 = (NDpres(2) - NDpres(5)) / PDIFFL
              CBL = 1! / SQR(3!) * (CA2 + CA3)
              CAL = CA1 + (CA2 - CA3) / 2!
              PDIFF = PDIFFL
              PBAR7 = PBARL
              IF (ABS(PDIFF) <= .0000001#) THEN PDIFF = .0000001#
              CT7 = CAL
              CF7 = CBL
              cptot7 = (NDpres(7) - PTOT) / PDIFF
              cpsta7 = (PBAR7 - PSTAT) / PDIFF
              table1(7, ix%, iy%) = CT7
              table2(7, ix%, iy%) = CF7
              table3(7, ix%, iy%) = cptot7
              table4(7, ix%, iy%) = cpsta7

150   NEXT i

PRINT "EVALUATING STATIC AND TOTAL PRESSURE COEFFICIENTS"

REM >>> Write Static and Total Pressure Coefficient Arrays to Disk

FOR ix% = 1 TO 20
 FOR jx% = 1 TO 20
  FOR ndhol% = 1 TO 7
   WRITE #2, probe%, ndhol%, ix%, jx%, table3(ndhol%, ix%, jx%), table4(ndhol%, ix%,
jx%)
  NEXT ndhol%
 NEXT jx%
NEXT ix%


REM >>> Invert flow angularity matrix

FOR ndhol% = 7 TO 1 STEP -1
 PRINT "INVERTING CALIBRATION MATRIX FOR SECTOR "; ndhol%

REM >>> Different ranges for different sectors

 IF ndhol% = 7 THEN
   x0 = -4.5: x1 = 4.5: x2 = .5
   x3 = -4.5: x4 = 4.5: x5 = .5
  ELSE
   x0 = 0: x1 = 2: x2 = .25
   x3 = -2: x4 = 2: x5 = .5
  END IF

 FOR CpAlpha = x0 TO x1 STEP x2
  FOR CpBeta = x3 TO x4 STEP x5
```

```
      ix% = 0
      jx% = 0
      e0 = 9999999

REM >>> Search for minimum error in three phases

REM >>> Phase 1 - simple search w/o interpolation

      FOR i% = 1 TO 19
       FOR j% = 1 TO 19
        e = (table1(ndhol%, i%, j%) - CpAlpha) ^ 2 + (table2(ndhol%, i%, j%) -
CpBeta) ^ 2
        IF e < e0 THEN ix% = i%: jx% = j%: e0 = e
       NEXT j%
      NEXT i%

      IF (ix% = 0) OR (jx% = 0) THEN BEEP: alpha = -999: beta = -999: GOTO 1001

      IF (ix% = 1) THEN ix% = 2
      IF (ix% = 19) THEN ix% = 18
      IF (jx% = 1) THEN jx% = 2
      IF (jx% = 19) THEN jx% = 18

REM >>> Phase 2 - interpolate

      e0 = 99999

FOR i1 = ix% - 1 TO ix% + 1 STEP .25
 FOR j1 = jx% - 1 TO jx% + 1 STEP .25
  iy% = INT(i1): iz = i1 - iy%
  jy% = INT(j1): jz = j1 - jy%

  A = table1(ndhol%, iy% + 1, jy% + 1) * iz * jz + table1(ndhol%, iy%, jy% + 1) * (1
- iz) * jz + table1(ndhol%, iy%, jy%) * (1 - iz) * (1 - jz) + table1(ndhol%, iy% +
1, jy%) * iz * (1 - jz)
  B = table2(ndhol%, iy% + 1, jy% + 1) * iz * jz + table2(ndhol%, iy%, jy% + 1) * (1
- iz) * jz + table2(ndhol%, iy%, jy%) * (1 - iz) * (1 - jz) + table2(ndhol%, iy% +
1, jy%) * iz * (1 - jz)

  e = (A - CpAlpha) ^ 2 + (B - CpBeta) ^ 2
  IF e < e0 THEN i2 = i1: j2 = j1: e0 = e

 NEXT j1
NEXT i1

e0 = 99999

      IF (i2 < 1.25) THEN i2 = 1.25
      IF (i2 > 18.75) THEN i2 = 18.75
      IF (j2 < 1.25) THEN j2 = 1.25
      IF (j2 > 18.75) THEN j2 = 18.75

REM >>> Phase 3 - Interpolate over a finer grid

FOR i3 = i2 - .25 TO i2 + .25 STEP .0625
```

```
 FOR j3 = j2 - .25 TO j2 + .25 STEP .0625

  iy% = INT(i3): iz = i3 - iy%
  jy% = INT(j3): jz = j3 - jy%

  A = table1(ndhol%, iy% + 1, jy% + 1) * iz * jz + table1(ndhol%, iy%, jy% + 1) * (1
- iz) * jz + table1(ndhol%, iy%, jy%) * (1 - iz) * (1 - jz) + table1(ndhol%, iy% +
1, jy%) * iz * (1 - jz)
  B = table2(ndhol%, iy% + 1, jy% + 1) * iz * jz + table2(ndhol%, iy%, jy% + 1) * (1
- iz) * jz + table2(ndhol%, iy%, jy%) * (1 - iz) * (1 - jz) + table2(ndhol%, iy% +
1, jy%) * iz * (1 - jz)

  e = (A - CpAlpha) ^ 2 + (B - CpBeta) ^ 2
  IF e < e0 THEN i4 = i3: j4 = j3: e0 = e

 NEXT j3
NEXT i3

  alpha = (i4 - 10) * 5
  beta = (j4 - 10) * 5

  PRINT USING "SECTOR # | CPALPHA +#.### CPBETA +#.### |  ALPHA +##.# BETA +##.#";
ndhol%; CpAlpha; CpBeta; alpha; beta

  IF e0 > 1 THEN BEEP: alpha = -999: beta = -999: GOTO 1001

1001 :
  WRITE #2, probe%, ndhol%, CpAlpha, CpBeta, alpha, beta
  NEXT CpBeta
 NEXT CpAlpha


 NEXT ndhol%
PRINT
NEXT probe%

CLOSE

END

REM $STATIC
FUNCTION min (A, B, C, d, e, F, g)
n = 0
IF A < B THEN n = A ELSE n = B
IF C < n THEN n = C
IF d < n THEN n = d
IF e < n THEN n = e
IF F < n THEN n = F
IF g < n THEN n = g

min = n

END FUNCTION
```

Apendix C—Data Acquisition Code (Microsoft QuickBasic)

```
DECLARE SUB readanalog ()
DECLARE SUB readscales ()
DECLARE SUB analoginit ()
DECLARE SUB ReadCal ()
DECLARE SUB getdata ()
DECLARE SUB DebugProbes ()
DECLARE SUB crunch (P!(), O!(), probe%)
DECLARE FUNCTION max! (a!, b!, c!, d!, e!, f!, g!)
DECLARE FUNCTION min! (a!, b!, c!, d!, e!, f!, g!)
DECLARE SUB DecodeBuffer (buffer$)
DECLARE SUB Talk8400 (CMD$)
DECLARE SUB SloSurvey ()
DECLARE SUB survey ()
DECLARE SUB Main ()
DECLARE SUB PositionMenu ()
DECLARE SUB SmartMove ()
DECLARE SUB StillMoving ()
DECLARE SUB Init ()
DECLARE SUB SetUpCompumotor ()
DECLARE FUNCTION keystroke% (O$)
DECLARE SUB Menu (m$)
DECLARE SUB INIT1 ()
DECLARE SUB TalkCompumotor (Hstep&, Vstep&, Lstep&)
DECLARE SUB GetPosition ()
DECLARE SUB moveto (Hnew!, Vnew!, Lnew!)


' Common GPIB status variables

COMMON SHARED /NISTATBLK/ ibsta%, iberr%, IBCNT%, IBCNTL&

' GPIB status bit vector :
'       status variable ibsta and wait mask

CONST EERR = &H8000     ' Error detected
CONST TIMO = &H4000     ' Timeout
CONST RQS = &H800       ' Device requesting service


' GPIB Subroutine Declarations
  DECLARE SUB IBDEV (BYVAL BDID%, BYVAL PAD%, BYVAL SAD%, BYVAL TMO%, BYVAL EOS%,
BYVAL EOT%, ud%)
  DECLARE SUB IBCLR (BYVAL BD%)
  DECLARE SUB IBRDF (BYVAL BD%, FLNAME$)
  DECLARE SUB IBRD (BYVAL BD%, RD$)
  DECLARE SUB ibrsp (BYVAL BD%, spr%)
  DECLARE SUB ibwait (BYVAL BD%, BYVAL MASK%)
  DECLARE SUB IBWRT (BYVAL BD%, WRT$)
```

```
' ------------------------------------------------------
' Common block and Subroutine Declarations required
' by the National Instruments 488 interface card (PC2A)
' ------------------------------------------------------


DIM SHARED psi%, spx%, echo%
DIM SHARED Nvals, PktInfo(19), pktmsg$, pktXdat(4096)
DIM SHARED buffer3$, x$
DIM SHARED volts(16), scale(10)



REM >>> Sector 7 angularity matrix
DIM SHARED table1a(2, 19, 19)
DIM SHARED table2a(2, 19, 19)

REM >>> Sector 1-6 angularity matrix
DIM SHARED table1b(2, 6, 9, 9)
DIM SHARED table2b(2, 6, 9, 9)

REM >>> Total/Static Pressure matrix
DIM SHARED table3(2, 6, 20, 20)
DIM SHARED table4(2, 6, 20, 20)



buffer3$ = SPACE$(4096)

' ------------------------------------------------------
' Execuatable code begins here.
' ------------------------------------------------------


DIM SHARED xpos

xpos = -999


DIM SHARED pt$, test$
DIM SHARED tX, ty, Tz

CONST delay = 149

REM * SLOW-SURVEY.BAS - SURVEY RIG CONTROL MODULE w/ 7-HP DATA ACQ.

'
' Serial port
CONST port1$ = "COM2:2400,N,8,1,DS0,RS,CD0 "     ' compumotor 3000 link"

CONST motor% = 0              ' = 1 to disable motor control, = 0 to enable

'Set-up's for MetraByte Parallel Ports
```

```
'NOTE:&H90 = PA-Input,PB-Output,PC(0-3)-Output,PC(4-7)-Output

' Metrabyte to encoder wiring diagram:
' PA(0-7) input <------- D(0-7) output from each encoder
' PC0 output    -------> A0 (pin #4) of each encoder
' PC1 output    -------> A1 (pin #3) of each encoder
' PC2 output    -------> Strobe (pin #6) of each encoder
' PC(3-7)       -------> Device Select (pin #5) for encoder 1-5

CONST BASEADDR = &H2E0      'MetraByte Base Address (from switch settings
[00111111])
CONST PA = BASEADDR + 0
CONST PB = BASEADDR + 1
CONST PC = BASEADDR + 2
CONST CTRL = BASEADDR + 3   'MB Port Addresses
CONST CONTROL = &H90        'MetraByte Port-configuration Contol Word

'General Rig Parameters

CONST Hscale = 1000         'Scale factor to turn Inches into Compumotor 3000 counts
CONST VScale = 1000         'Scale factor ...
CONST LScale = 1 / 3 * 1000 'Scale factor ...


CONST VVELOC = 15
CONST LVELOC = 10

CONST HAccel = 10
CONST VAccel = 10
CONST LAccel = 10

' SURBEY RIG HOME POSITION
CONST HHome = 0          'Horizontal Home Position in inches
CONST VHome = 0          'Vertical Home Position in inches
CONST LHome = 0          'Longitudinal Home Position in inches

' SURVEY RIG MAXIMUM EXTENSION POSITION (inches)

CONST HLoLim = -32    'Horizontal Low Software Limit in Inches
CONST HHiLim = 32     'Horizontal High Software Limit in Inches
CONST VLoLim = -1     'Vertical Low Software Limit in Inches
CONST VHiLim = 60     'Vertical High Software Limit in Inches
CONST LLoLim = -16     'Longitudinal Low Software Limit in Inches
CONST LHiLim = 32     'Longitudinal High Software Limit in Inches

DIM SHARED closedloop%


CONST false = -1
CONST true = 1

DIM SHARED Vdir, ERP, hveloc
```

```
DIM SHARED P$(7), ACT(4), ESB$(4, 8)          ' encode control arrays

CALL Init
CALL ReadCal
CALL SetUpCompumotor
CALL Main



CHDIR "c:\qbasic\pcdas"

RUN "MENU.BAS"

SUB analoginit

DIM param%(60), DAT%(32000), gain%(15), nn(15), pp(15), offset(15)


 param%(0) = 0                    ' Board number
 param%(1) = &H300                ' Base I/O address
 param%(4) = 2                    ' IRQ level : IRQ2
 param%(6) = 20: param%(5) = 20: xmax = 2000 ' sample at 500hz

 param%(7) = 0                    ' Trigger mode, 0 : pacer trigger
 param%(8) = 0                    ' Non-cyclic
 param%(10) = VARPTR(DAT%(0))     ' Offset  of A/D data buffer A
 param%(11) = VARSEG(DAT%(0))     ' Segment of A/D data buffer A
 param%(12) = 0                   ' Data buffer B address, if not used,
 param%(13) = 0                   ' must set to 0.
 '
 param%(14) = 4096                  ' A/D conversion number
 param%(15) = 0                   ' A/D conversion start channel
 param%(16) = 15                  ' A/D conversion stop channel
 '
 param%(17) = &HFF ' gain table
 param%(18) = VARPTR(gain%(0))    ' Offset  of gain table buffer
 param%(19) = VARSEG(gain%(0))    ' Segment of gain table buffer

   gain%(0) = 0: pp(0) = 10: offset(0) = -5: ' +/- 5v external input
   gain%(1) = 0: pp(1) = 10: offset(1) = -5: ' Pa
   gain%(2) = 8: pp(2) = 20: offset(2) = -10: ' Pr-Pa
   gain%(3) = 8: pp(3) = 20: offset(3) = -10: ' Pr-Ps
   gain%(4) = 4: pp(4) = 10: offset(4) = 0: '  turntable
   gain%(5) = 0: pp(5) = 10: offset(5) = -5: ' inclinometer
   gain%(6) = 6: pp(6) = .1: offset(6) = 0: ' PTplenum barocel
   gain%(7) = 6: pp(7) = .1: offset(7) = 0: ' Tplenum thermocouple
   gain%(8) = 4: pp(8) = 10: offset(8) = 0: ' CFM plenum
   gain%(9) = 9: pp(9) = 2: offset(9) = -1:  ' PS forebody
   gain%(10) = 9: pp(10) = 2: offset(10) = -1
   gain%(11) = 9: pp(11) = 2: offset(11) = -1
   gain%(12) = 9: pp(12) = 2: offset(12) = -1
   gain%(13) = 9: pp(13) = 2: offset(13) = -1
   gain%(14) = 9: pp(14) = 2: offset(14) = -1
   gain%(15) = 9: pp(15) = 2: offset(15) = -1
```

```
 FUN% = 3                          ' FUNCTION 3
 CALL PCL818HG(FUN%, SEG param%(0))    ' Func 3 : Hardware initialization
 IF param%(45) <> 0 THEN PRINT "DRIVER INITIALIZATION FAILED !": STOP

 FUN% = 4                        ' FUNCTION 100
 CALL PCL818HG(FUN%, SEG param%(0))    ' Func 100  : A/D initialization
 IF param%(45) <> 0 THEN PRINT "A/D INITIALIZATION FAILED !": STOP

 FUN% = 12
 CALL PCL818HG(FUN%, SEG param%(0))    ' Func 100  : D/A initialization
 IF param%(45) <> 0 THEN PRINT "D/A INITIALIZATION FAILED !": STOP


END SUB

SUB crunch (P(), O(), probe%)

REM ***************************************************************
REM ** 7hole probe decomposition algorithm                       **
REM **    Input : p() array...                                    **
REM **         P(1)..P(7) are the pressures from the 7 hole probe **
REM **         P(8) is a normalizing pressure.. tunnel Q          **
REM **    Output : o() array...                                   **
REM **         O(1) = Pstatic                                     **
REM **         O(2) = Ptotal                                      **
REM **         O(3) = alpha (degrees)                             **
REM **         O(4) = beta (degrees)                              **
REM ***************************************************************

DIM NDpres(8), qc(2), pxt(2)

REM  Initialize

  alpha = -999
  beta = -999

REM  Normalize pressures.

  FOR j% = 1 TO 8
    NDpres(j%) = P(j%) / P(8)
  NEXT j%

REM  Compute the coefficients.

  PMAX = max(NDpres(1), NDpres(2), NDpres(3), NDpres(4), NDpres(5), NDpres(6),
NDpres(7))

REM  Default situation is zone 7

  ndhol% = 7
  PBAR = (NDpres(1) + NDpres(2) + NDpres(3) + NDpres(4) + NDpres(5) + NDpres(6)) /
6!
  PDIFF = NDpres(7) - PBAR
```

```
  CA1 = (NDpres(4) - NDpres(1)) / PDIFF
  CA2 = (NDpres(3) - NDpres(6)) / PDIFF
  CA3 = (NDpres(2) - NDpres(5)) / PDIFF
  cpbeta = 1! / SQR(3!) * (CA2 + CA3)
  cpAlpha = CA1 + (CA2 - CA3) / 2!


REM Check to see if flow angularity is outside zone 7 calibration range
  IF cpAlpha > 4.5 OR cpAlpha < -4.5 OR cpbeta > 4.5 OR cpbeta < -4.5 THEN 21

REM  look up alpha and beta from table
  ix = cpAlpha * 2 + 10: jx = cpbeta * 2 + 10
  iy% = INT(ix): jy% = INT(jx): iz = ix - iy%: jz = jx - jy%

  alpha = table1a(probe%, iy% + 1, jy% + 1) * iz * jz + table1a(probe%, iy%, jy% +
1) * (1 - iz) * jz + table1a(probe%, iy%, jy%) * (1 - iz) * (1 - jz) +
table1a(probe%, iy% + 1, jy%) * iz * (1 - jz)
  beta = table2a(probe%, iy% + 1, jy% + 1) * iz * jz + table2a(probe%, iy%, jy% + 1)
* (1 - iz) * jz + table2a(probe%, iy%, jy%) * (1 - iz) * (1 - jz) + table2a(probe%,
iy% + 1, jy%) * iz * (1 - jz)
  GOTO 25

REM  sectoring algorithm - use separated flow coefficients

21 :

REM  Zone 1.
      IF (NDpres(1) = PMAX) THEN
          ndhol% = 1
          PDIFF = NDpres(1) - (NDpres(2) + NDpres(6)) / 2!
          cpAlpha = (NDpres(1) - NDpres(7)) / PDIFF
          cpbeta = (NDpres(6) - NDpres(2)) / PDIFF
          PBAR = (NDpres(2) + NDpres(6)) / 2!
REM  Zone 2.
      ELSEIF (NDpres(2) = PMAX) THEN
          ndhol% = 2
          PDIFF = NDpres(2) - (NDpres(1) + NDpres(3)) / 2!
          cpAlpha = (NDpres(2) - NDpres(7)) / PDIFF
          cpbeta = (NDpres(1) - NDpres(3)) / PDIFF
          PBAR = (NDpres(1) + NDpres(3)) / 2!
REM  Zone 3.
      ELSEIF (NDpres(3) = PMAX) THEN
          ndhol% = 3
          PDIFF = NDpres(3) - (NDpres(2) + NDpres(4)) / 2!
          cpAlpha = (NDpres(3) - NDpres(7)) / PDIFF
          cpbeta = (NDpres(2) - NDpres(4)) / PDIFF
          PBAR = (NDpres(2) + NDpres(4)) / 2!
REM  Zone 4.
      ELSEIF (NDpres(4) = PMAX) THEN
          ndhol% = 4
          PDIFF = NDpres(4) - (NDpres(3) + NDpres(5)) / 2!
          cpAlpha = (NDpres(4) - NDpres(7)) / PDIFF
          cpbeta = (NDpres(3) - NDpres(5)) / PDIFF
          PBAR = (NDpres(3) + NDpres(5)) / 2!
```

```
REM  Zone 5.
      ELSEIF (NDpres(5) = PMAX) THEN
          ndhol% = 5
          PDIFF = NDpres(5) - (NDpres(4) + NDpres(6)) / 2!
          cpAlpha = (NDpres(5) - NDpres(7)) / PDIFF
          cpbeta = (NDpres(4) - NDpres(6)) / PDIFF
          PBAR = (NDpres(4) + NDpres(6)) / 2!
REM  Zone 6.
      ELSEIF (NDpres(6) = PMAX) THEN
          ndhol% = 6
          PDIFF = NDpres(6) - (NDpres(5) + NDpres(1)) / 2!
          cpAlpha = (NDpres(6) - NDpres(7)) / PDIFF
          cpbeta = (NDpres(5) - NDpres(1)) / PDIFF
          PBAR = (NDpres(5) + NDpres(1)) / 2!
REM  Zone 7.
      ELSE
       GOTO 25
      END IF

REM  inside calibration space?

'PRINT "SECTOR "; ndhol%, "CpAlpha ="; cpAlpha, "CpBeta ="; cpbeta;
'  LINE INPUT a$

  IF cpAlpha > 2 OR cpAlpha < 0 OR cpbeta > 2 OR cpbeta < -2 THEN 25


REM  look up alpha and beta from table - sectored coefficients

  ix = cpAlpha * 4 + 1: jx = cpbeta * 2 + 5
  iy% = INT(ix): jy% = INT(jx): iz = ix - iy%: jz = jx - jy%
  alpha = table1b(probe%, ndhol%, iy% + 1, jy% + 1) * iz * jz + table1b(probe%,
ndhol%, iy%, jy% + 1) * (1 - iz) * jz + table1b(probe%, ndhol%, iy%, jy%) * (1 - iz)
* (1 - jz) + table1b(probe%, ndhol%, iy% + 1, jy%) * iz * (1 - jz)
  beta = table2b(probe%, ndhol%, iy% + 1, jy% + 1) * iz * jz + table2b(probe%,
ndhol%, iy%, jy% + 1) * (1 - iz) * jz + table2b(probe%, ndhol%, iy%, jy%) * (1 - iz)
* (1 - jz) + table2b(probe%, ndhol%, iy% + 1, jy%) * iz * (1 - jz)

REM Have we determined flow angularity? If not error!
  IF ABS(alpha) > 90 OR ABS(beta) > 90 THEN 260


25 :


REM Null resiudal flow angularity
  IF probe% = 0 THEN alpha = alpha: beta = beta - 3
  IF probe% = 1 THEN alpha = alpha: beta = beta - 3
  IF probe% = 2 THEN alpha = alpha: beta = beta - 2


REM Determine Static and Total Pressure Coefficients
```

```
  IF alpha > 45 THEN alpha = 45
  IF alpha < -45 THEN alpha = -45
  IF beta > 45 THEN beta = 45
  IF beta < -45 THEN beta = -45


  ix = 10 + alpha / 5: jx = 10 + beta / 5
  iy% = INT(ix): iz = ix - iy%: jy% = INT(jx): jz = jx - jy%

  cptot = table3(probe%, ndhol% - 1, iy% + 1, jy% + 1) * iz * jz + table3(probe%,
ndhol% - 1, iy%, jy% + 1) * (1 - iz) * jz + table3(probe%, ndhol% - 1, iy%, jy%) *
(1 - iz) * (1 - jz) + table3(probe%, ndhol% - 1, iy% + 1, jy%) * iz * (1 - jz)
  cpsta = table4(probe%, ndhol% - 1, iy% + 1, jy% + 1) * iz * jz + table4(probe%,
ndhol% - 1, iy%, jy% + 1) * (1 - iz) * jz + table4(probe%, ndhol% - 1, iy%, jy%) *
(1 - iz) * (1 - jz) + table4(probe%, ndhol% - 1, iy% + 1, jy%) * iz * (1 - jz)

  PTOT = NDpres(ndhol%) * (1! - cptot) + PBAR * cptot
  PSTA = PBAR * (1 + cpsta) - NDpres(ndhol%) * cpsta

  tot = PTOT * P(8)
  sta = PSTA * P(8)
  Q = tot - sta

REM * make static presure correction

  qc(0) = .9
  qc(1) = .9
  qc(2) = .9
  pxt(0) = 1.09
  pxt(1) = 1.11
  pxt(2) = 1.12


  tot = tot * pxt(probe%)
  Q = Q / qc(probe%)

  sta = tot - Q

   O(1) = sta
   O(2) = tot
   O(3) = alpha
   O(4) = beta

   GOTO 270

REM   Notify user of errors.
260   PRINT "ERROR : Can't resolve flow angularity"
   BEEP
   O(1) = -9.9
   O(2) = -9.9
   O(3) = -9.9
   O(4) = -9.9
270   :
```

```
END SUB

SUB DebugProbes

SCREEN 0:  WIDTH 80, 43
CLS
PRINT "PROBE DEBUG ---------------------------------- PRESS ANY KEY TO STOP"
PRINT
PRINT "   PROBE       PRESSURE"
PRINT "-------------------------------------------------------------------------"
PRINT
VIEW PRINT 5 TO 43

 Talk8400 ("OD0")


1101 :

 SOUND 1000, 1

 Talk8400 ("AD1 1 1")

 tmx = TIMER + 1
11102 : IF TIMER < tmx THEN 11102


 Talk8400 ("OD1 1")

 LOCATE 5, 1
 FOR i% = 1 TO 26
  PRINT USING "   ###    +###.## psf  "; i%; pktXdat(i%) * 144;
  n% = pktXdat(i%) * 144 / 2
  IF n% > 40 THEN n% = 40
  IF n% < -40 THEN n% = 40

  IF n% > 0 THEN PRINT STRING$(n%, "#"); TAB(78); " "
  IF n% <= 0 THEN PRINT STRING$(-n%, "x"); TAB(78); " "

  IF i% = 1 OR i% = 2 OR i% = 9 OR i% = 16 OR i% = 23 THEN PRINT STRING$(70, "-")

 NEXT i%


IF INKEY$ = "" THEN 1101

CALL ibonl(psi%, 0): psi% = 0

LINE INPUT "PRESS ENTER TO RETURN TO MAIN MENU"; a$

VIEW PRINT 1 TO 43
SCREEN 0:  WIDTH 80, 43
CLS

END SUB
```

```
SUB DecodeBuffer (buffer$)

'-------------------------------------------------------------
'
' PktInfo - Packet Header Information array - contains
'
'     1) rc       = Response Code
'     2) rt       = Response Type
'     3) msglen   = Message Length
'     4) RetValue!    = Returned Value OR Error/Conf. Code
'
'     5) NRows    = number of rows
'     6) NCols    = number of columns
'
'     7) Msno     = Measurement Set Number
'     8) Nvals    = Number of values
'     9) CRS      = Cluster, Rack, Slot address
'    10) Utype    = Unit type
'    11) Tblno    = Table Number
'    12) Nframes  = Number of Frames per Data Point
'    13) yr       = year
'    14) mo       = month
'    15) day      = day
'    16) hr       = hour
'    17) mn       = minute
'    18) sec      = second
'    19) msec     = milli-seconds
'

IF LEN(buffer$) < 23 THEN PRINT "GARBLED BUFFER ERROR": GOTO exitsub


PKTfinished = 0
FOR i% = 1 TO 19: PktInfo(i%) = 0: NEXT i%
pktmsg$ = ""

' Get current response code, response type, and packet length.
rc = ASC(MID$(buffer$, 1, 1)): PktInfo(1) = rc
rt = ASC(MID$(buffer$, 2, 1)): PktInfo(2) = rt
msglen = ASC(MID$(buffer$, 3, 1)) * 256! + ASC(MID$(buffer$, 4, 1)): PktInfo(3) =
msglen          ' msg length

SELECT CASE rt
 CASE IS = 4
  GOSUB ACKpacket
 CASE IS = 9
  GOSUB SingleIEEEfloatPacket
 CASE IS = 19
  GOSUB StreamIEEEfloatPacket
 CASE IS = 33
  GOSUB ArrayIEEEfloatPacket
 CASE IS = 35
 CASE IS = 128
```

```
   GOSUB errorPacket
  CASE ELSE
   PRINT : PRINT : PRINT " Unknnown Packet type recieved : type = "; rt
   PRINT : LINE INPUT " Press ENTER to continue"; n$
 END SELECT


 GOTO exitsub



 '-----------------------------------------------------
 ' Error Packet:  rt = 128, and that indicates that
 ' some sort of error occured.  See PktMsg$ for the
 ' error message returned by the system.
 '-----------------------------------------------------

 errorPacket:
  GOSUB ACKpacket
  PRINT : PRINT " ERROR : "; pktmsg$: PRINT : PKTfinished = -1
 RETURN

 '-----------------------------------------------------
 ' Acknowledgement Packet:  rt = 4   everything O.K.
 '-----------------------------------------------------

 ACKpacket:
  ErConfCode& = CVL(MID$(buffer$, 8, 1) + MID$(buffer$, 7, 1) + MID$(buffer$, 6, 1) +
 MID$(buffer$, 5, 1))
  RetValue! = ErConfCode&
  PktInfo(4) = ErConfCode&
  pktmsg$ = MID$(buffer$, 9)
  PKTfinished = -1
 RETURN

 '-----------------------------------------------------
 ' Single IEEE Floating Point Number Packet: rt = 9
 '-----------------------------------------------------

 SingleIEEEfloatPacket:

 PktInfo(4) = CVS(MID$(buffer$, 8, 1) + MID$(buffer$, 7, 1) + MID$(buffer$, 6, 1) +
 MID$(buffer$, 5, 1))
 pktmsg$ = MID$(buffer$, 9)
 PKTfinished = -1
 RETURN

 '-----------------------------------------------------
 ' Stream Data, IEEE Single Precision: rt = 19
 '-----------------------------------------------------

 StreamIEEEfloatPacket:
 GOSUB DecodeStreamHeader
 j% = 25
```

```
FOR i% = 1 TO Nvals
  pktXdat(i%) = CVS(MID$(buffer$, j% + 3, 1) + MID$(buffer$, j% + 2, 1) +
MID$(buffer$, j% + 1, 1) + MID$(buffer$, j%, 1))
  j% = j% + 4
NEXT i%

PKTfinished = -1
RETURN


'-----------------------------------------------------
' Array Data, IEEE Float : rt = 33
'-----------------------------------------------------

ArrayIEEEfloatPacket: '

Nrows = ASC(MID$(buffer$, 5, 1)) * 256! + ASC(MID$(buffer$, 6, 1))
Ncols = ASC(MID$(buffer$, 7, 1)) * 256! + ASC(MID$(buffer$, 8, 1))

PktInfo(5) = Nrows
PktInfo(6) = Ncols

index% = 1
k% = 9

FOR i% = 1 TO Nrows
 FOR j% = 1 TO Ncols
    pktXdat(index%) = CVS(MID$(buffer$, k% + 3, 1) + MID$(buffer$, k% + 2, 1) +
MID$(buffer$, k% + 1, 1) + MID$(buffer$, k%, 1))
    k% = k% + 4
    index% = index% + 1
 NEXT j%
NEXT i%

PKTfinished = -1
RETURN


' ****************************************************
' decodearrayheader:
' ****************************************************

DecodeArrayHeader:

Nrows = ASC(MID$(buffer$, 5, 1)) * 256! + ASC(MID$(buffer$, 6, 1))
Ncols = ASC(MID$(buffer$, 7, 1)) * 256! + ASC(MID$(buffer$, 8, 1))

PktInfo(5) = Nrows
PktInfo(6) = Ncols

pktmsg$ = MID$(buffer$, 9, 40)

RETURN
```

```
' ******************************************************
' decodestreamheader:
' ******************************************************

DecodeStreamHeader:
'
' we have already decoded response code, response type,
' and response length
'

Msno = ASC(MID$(buffer$, 5, 1)) * 256! + ASC(MID$(buffer$, 6, 1))
Nvals = ASC(MID$(buffer$, 7, 1)) * 256! + ASC(MID$(buffer$, 8, 1))
crs = ASC(MID$(buffer$, 9, 1)) * 100 + ASC(MID$(buffer$, 10, 1)) * 10
crs = crs + ASC(MID$(buffer$, 11, 1))

Utype = ASC(MID$(buffer$, 12, 1))
Tblno = ASC(MID$(buffer$, 13, 1))

Nframes = ASC(MID$(buffer$, 14, 1))

yr = ASC(MID$(buffer$, 15, 1))
mo = ASC(MID$(buffer$, 16, 1))
dy = ASC(MID$(buffer$, 17, 1))
hr = ASC(MID$(buffer$, 18, 1))

mn = ASC(MID$(buffer$, 19, 1))
sc = ASC(MID$(buffer$, 20, 1))

msec = ASC(MID$(buffer$, 21, 1)) * 256! + ASC(MID$(buffer$, 22, 1))

PktInfo(7) = Msno
PktInfo(8) = Nvals
PktInfo(9) = crs
PktInfo(10) = Utype
PktInfo(11) = Tblno
PktInfo(12) = Nframes
PktInfo(13) = yr
PktInfo(14) = mo
PktInfo(15) = dy
PktInfo(16) = hr
PktInfo(17) = mn
PktInfo(18) = sc
PktInfo(19) = msec

RETURN

exitsub:


END SUB

SUB GetPosition
```

```
'-------------------------------------------------------------
'Parallel Read Encoder Position for MetraByte PIO-12 and AR Absolute
'encoders. The ACT array is used to determine which encoders will be
'Read.  The output will be in the arrays ERR and PLC.  The method
'used to reduce the parallel data will be determined by the status of
'the SP parameter for each encoder (either Binary or BCD).
'PARALLEL READ ENCODER POSITION

DIM plc(4)

'Set-up's for MetraByte Parallel Ports


'Converse in parallel
OUT CTRL, CONTROL'Set up MetraByte

ERP = false

 FOR R% = 0 TO 4
  IF ACT(R%) <> 0 THEN

   DEV = 2 ^ (R% + 3)'Select which encoder to poll for position and error data

   OUT PC, 4 + DEV + 3 'Strobe High (4), A0 and A1 High (3), Device Selected
   OUT PC, DEV + 3     'Strobe Low, etc.

   FOR i% = 0 TO delay: NEXT i%

   LSB = INP(PA)  'Get LSB when A0 and A1 high
   OUT PC, DEV + 1'Drop A1 Low

   FOR i% = 0 TO delay: NEXT i%

   LSB2 = INP(PA) 'Get LSB2 when A0 high and A1 low
   OUT PC, DEV    'Drop A0 Low as well

   FOR i% = 0 TO delay: NEXT i%

   MSB2 = INP(PA) 'Get MSB2 when A0 and A1 low
   OUT PC, DEV + 2'Raise A1 high

   FOR i% = 0 TO delay: NEXT i%

   MSB = INP(PA)  'Get MSB when A0 low and A1 high
   OUT PC, 4 + DEV + 3'Bring up all lines

   'Convert data to position and error information
   IF ESB$(R%, 6) = "0" THEN
    'Data interpreted as Binary
     SIGN = 1: IF (LSB AND 16) = 16 THEN LSB = LSB - 16: SIGN = -1
     IF (LSB AND 64) = 64 THEN ERP = true: LSB = LSB - 64
     IF (LSB AND 128) = 128 THEN ERP = true: LSB = LSB - 128
     plc(R%) = ((LSB / 65536!) + (LSB2 / 512) + MSB2 + (MSB * 256)) * SIGN
```

```
     ELSE

    'Data interpreted as Binary Coded Decimal
     SIGN = 1: IF (MSB AND 16) = 16 THEN MSB = MSB - 16: SIGN = -1
     IF (MSB AND 64) = 64 THEN ERP = true: MSB = MSB - 64
     IF (MSB AND 128) = 128 THEN ERP = true: MSB = MSB - 128
     plc(R%) = VAL(HEX$(MSB)) * 100 + VAL(HEX$(MSB2))
     plc(R%) = plc(R%) + VAL(HEX$(LSB2)) * .01 + VAL(HEX$(LSB)) * .0001
     plc(R%) = plc(R%) * SIGN

    END IF

   END IF

IF ABS(plc(R%)) > 69.999 THEN ERP = true: plc(R%) = -999

NEXT R%

IF ERP = true THEN SOUND 5000, 1

IF plc(1) = -999 THEN  ELSE tX = plc(1)  ' X (longitudinal)
IF plc(0) = -999 THEN  ELSE ty = plc(0)' Y (horizontal)
IF plc(2) = -999 THEN  ELSE Tz = plc(2) ' Z (vertical)

cx = POS(0): cy = CSRLIN:  VIEW PRINT 1 TO 43
COLOR 14, 1: LOCATE 3, 1: PRINT USING "TRAVERSE POSITION : HORZ +###.###in  VERT
+###.###in   LONG +###.###in"; ty; Tz; tX; TAB(80);
VIEW PRINT 4 TO 41: LOCATE cy, cx
COLOR 7, 0


END SUB

SUB Init

SCREEN 0
WIDTH 80, 43

PRINT
PRINT
PRINT "DONE."
PRINT
PRINT "SLOWSURVEY.BAS - WAKE SURVEY MODULE - 1995 - T.Takahashi"
PRINT
PRINT " Initializing...."

CHDIR "c:\qbasic\pcdas"

OPEN "I", 1, "pc-das.ini"
LINE INPUT #1, pt$
LINE INPUT #1, test$
CLOSE 1
```

```
IF RIGHT$(pt$, 1) = "\" AND RIGHT$(pt$, 2) <> ":\" THEN pt$ = LEFT$(pt$, LEN(pt$) -
1)

CHDIR pt$

IF RIGHT$(pt$, 1) <> "\" THEN pt$ = pt$ + "\"

PRINT " Reading 7HP Calibration Data"


END SUB

FUNCTION keystroke% (O$)


1000 :
 CALL GetPosition

 a$ = INKEY$
IF a$ = "" THEN 1000


 COLOR 7, 0: PRINT a$

 k% = INSTR(O$, a$)

 IF k% = 0 THEN BEEP: COLOR 4, 0: PRINT "KEYSTROKE ERROR! - VALID COMMANDS :"; O$:
COLOR 7!

 keystroke% = k%

END FUNCTION

SUB Main

1 :

IF motor = 1 THEN PRINT "      MOTOR CONTROL DISABLED"

Menu ("SURVEY  ALIGN  PROBE_DEBUG                                                    EXIT")

 COLOR 15
 PRINT "WAKE SURVEY CONTROL"
 COLOR 7
 PRINT "      ENTER COMMAND :";

 op = keystroke("SsAaPpEe")
 SELECT CASE op
   CASE 1, 2: CALL SloSurvey
   CASE 3, 4: CALL PositionMenu
   CASE 5, 6: CALL DebugProbes
   CASE 7, 8: GOTO done
 END SELECT
GOTO 1
```

```
 done:


 END SUB

 FUNCTION max (a, b, c, d, e, f, g)
 n = 0
 IF a > b THEN n = a ELSE n = b
 IF c > n THEN n = c
 IF d > n THEN n = d
 IF e > n THEN n = e
 IF f > n THEN n = f
 IF g > n THEN n = g
 max = n
 END FUNCTION

 SUB Menu (m$)

  cx = POS(0): cy = CSRLIN

      VIEW PRINT 1 TO 43

  LOCATE 1, 1: COLOR 7, 1: PRINT TAB(37); "PC-DAS"; TAB(80);
  LOCATE 2, 1:

  flag% = 32
  FOR i% = 1 TO LEN(m$)
   IF flag% = 32 THEN COLOR 15, 7 ELSE COLOR 0, 7
   flag% = ASC(MID$(m$, i%, 1))
   PRINT CHR$(flag%);
  NEXT i%
  PRINT TAB(80);
  LOCATE 42, 1: COLOR 0, 3
  PRINT "PATH: "; pt$; TAB(30); "TEST: "; test$; TAB(60); TIME$; " "; DATE$; TAB(79);
" ";

  VIEW PRINT 4 TO 41
  IF cy < 4 THEN cy = 4
  LOCATE cy, cx
  COLOR 7, 0

END SUB

FUNCTION min (a, b, c, d, e, f, g)
n = 0
IF a < b THEN n = a ELSE n = b
IF c < n THEN n = c
IF d < n THEN n = d
IF e < n THEN n = e
IF f < n THEN n = f
IF g < n THEN n = g

min = n
```

```
END FUNCTION

SUB moveto (Hnew, Vnew, Lnew)


'Variable Reference
'HPos,VPos,tX       = Current Position (Horizontal, Vertical, and Longitudinal)
'HOld,VOld,LOld       = Last (Previous) Position
'HNew,VNew,LNew       = New (Given/Expected) Position
'HHome,VHome,LHome    = Home Position
'HDist,VDist,LDist    = Distance from Current to New (ex. HD=HN-HorzC)
'HStep,VStep,LStep    = Compumotor Steps to achieve Distance
'-------------------------------------------------------------


PRINT USING "        Move to : +##.### +##.### +##.###"; Hnew; Vnew; Lnew

IF motor% = 1 THEN GOTO 12345

' make sure the traverse has stopped


t0 = TIMER

' Where are we? (we already know from the call to still moving)

loopback:

' How far do we need to go?
 HDist = Hnew - ty
 VDist = Vnew - Tz
 LDist = Lnew - tX

'Compute Motor Steps
 Hstep& = -HDist * Hscale
 Vstep& = VDist * VScale
 Lstep& = LDist * LScale

' PRINT Hstep&; Vstep&; Lstep&, ;
 'Which Way are we going?
 v1 = SGN(VDist)
 v2 = SGN(HDist)

 ' move specified distance minus last little bit
 IF ABS(Vstep&) >= 199 THEN Vstep& = Vstep& - v1 * 30
 IF ABS(Hstep&) >= 1299 THEN Hstep& = Hstep& + v2 * 10

 Vdir = v1

 CALL TalkCompumotor(Hstep&, Vstep&, Lstep&)

 IF ABS(Hstep&) <= 10 AND ABS(Vstep&) <= 40 THEN GOTO 12345
```

```
 REM * Wait one second for the compumotor action to begin

 t0 = TIMER + 1
 DO
 LOOP UNTIL (TIMER > t0)

 CALL StillMoving

GOTO loopback

12345 :

BEEP

END SUB

SUB PositionMenu

2 :

Menu ("MOVE_TO  UP  DOWN  LEFT  RIGHT                                    EXIT")

COLOR 15
PRINT "  ALIGN SURVEY RIG"
COLOR 7
PRINT "       ENTER COMMAND :";

op = keystroke("MmUuDdLlRrEe")
SELECT CASE op
 CASE 1, 2: GOTO SmartMove
 CASE 3, 4: GOTO Up
 CASE 5, 6: GOTO Down
 CASE 7, 8: GOTO Left
 CASE 9, 10: GOTO Right
 CASE 11, 12: GOTO 3
END SELECT

GOTO 2

H = ty: V = Tz: l = tX

Up:
 IF Tz < VHiLim THEN CALL moveto(ty, Tz + 1, tX)
 GOTO 2

Down:
 IF Tz > VLoLim THEN CALL moveto(ty, Tz - 1, tX)
 GOTO 2

Left:
 IF ty > HLoLim THEN CALL moveto(ty + 1, Tz, tX)
 GOTO 2

Right:
```

```
 IF ty < HHiLim THEN CALL moveto(ty - 1, Tz, tX)
 GOTO 2

SmartMove:

INPUT "       Enter New Position (H,V,L) : "; H, V, l

IF H < HLoLim OR H > HHiLim OR V < VLoLim OR V > VHiLim OR l < LLoLim OR l > LHiLim
THEN
 BEEP
 COLOR 4, 0
 PRINT "        ERROR! KEEP RIG WITHIN LIMITS!"
 PRINT "        HORZ from "; HLoLim; " to "; HHiLim
 PRINT "        VERT from "; VLoLim; " to "; VHiLim
 PRINT "        LONG from "; LLoLim; " to "; LHiLim
 PRINT
 GOTO 2
 END IF

PRINT "       Moving Traverse (press any key to interrupt)"

CALL moveto(H, V, l)
BEEP

GOTO 2


3 :

END SUB

SUB readanalog

DIM param%(60), DAT%(32000), gain%(15), nn(15), pp(15), offset(15)


 param%(0) = 0                    ' Board number
 param%(1) = &H300                ' Base I/O address
 param%(4) = 2                    ' IRQ level : IRQ2
 param%(6) = 20: param%(5) = 20: xmax = 2000 ' sample at 500hz

 param%(7) = 0                    ' Trigger mode, 0 : pacer trigger
 param%(8) = 0                    ' Non-cyclic
 param%(10) = VARPTR(DAT%(0))     ' Offset  of A/D data buffer A
 param%(11) = VARSEG(DAT%(0))     ' Segment of A/D data buffer A
 param%(12) = 0                   ' Data buffer B address, if not used,
 param%(13) = 0                   ' must set to 0.
 '
 param%(14) = 4096                  ' A/D conversion number
 param%(15) = 0                   ' A/D conversion start channel
 param%(16) = 15                  ' A/D conversion stop channel
 '
 param%(17) = &HFF ' gain table
 param%(18) = VARPTR(gain%(0))    ' Offset  of gain table buffer
```

```
 param%(19) = VARSEG(gain%(0))    ' Segment of gain table buffer

   gain%(0) = 0: pp(0) = 10: offset(0) = -5: ' +/- 5v external input
   gain%(1) = 0: pp(1) = 10: offset(1) = -5: ' Pa
   gain%(2) = 8: pp(2) = 20: offset(2) = -10: ' Pr-Pa
   gain%(3) = 8: pp(3) = 20: offset(3) = -10: ' Pr-Ps
   gain%(4) = 4: pp(4) = 10: offset(4) = 0: '  turntable
   gain%(5) = 0: pp(5) = 10: offset(5) = -5: ' inclinometer
   gain%(6) = 6: pp(6) = .1: offset(6) = 0: ' PTplenum barocel
   gain%(7) = 6: pp(7) = .1: offset(7) = 0: ' Tplenum thermocouple
   gain%(8) = 4: pp(8) = 10: offset(8) = 0: ' CFM plenum
   gain%(9) = 9: pp(9) = 2: offset(9) = -1:  ' PS forebody
   gain%(10) = 9: pp(10) = 2: offset(10) = -1
   gain%(11) = 9: pp(11) = 2: offset(11) = -1
   gain%(12) = 9: pp(12) = 2: offset(12) = -1
   gain%(13) = 9: pp(13) = 2: offset(13) = -1
   gain%(14) = 9: pp(14) = 2: offset(14) = -1
   gain%(15) = 9: pp(15) = 2: offset(15) = -1



 FUN% = 5                          ' FUNCTION 5
 CALL PCL818HG(FUN%, SEG param%(0))    ' Func 5 : Pacer trigger A/D
 IF param%(45) <> 0 THEN PRINT "A/D INTERRUPT DATA TRANSFER FAILED !": STOP

 FOR i% = 0 TO 15
  volts(i%) = 0: nn(i%) = 0
 NEXT i%

 FOR i% = 0 TO param%(14) - 1
   ix% = i% MOD 16
   volts(ix%) = volts(ix%) + (pp(ix%) * DAT%(i%) / 4096! + offset(ix%))
   nn(ix%) = nn(ix%) + 1
 NEXT i%

 FOR i% = 0 TO 15
  volts(i%) = volts(i%) / nn(i%)
 NEXT i%

END SUB

SUB ReadCal


PRINT "READING CALIBRATION MATRICIES"
OPEN "I", 1, "c:\qbasic\pcdas\shpcal.dat"
FOR probe% = 0 TO 2
PRINT " PROBE "; probe%
FOR ix% = 1 TO 20
 FOR jx% = 1 TO 20
  FOR ndhol% = 0 TO 6
   INPUT #1, a, b, c, d, e, f
   table3(probe%, ndhol%, ix%, jx%) = e
   table4(probe%, ndhol%, ix%, jx%) = f
```

```
  NEXT ndhol%
 NEXT jx%
NEXT ix%

 ndhol% = 7
 FOR cpAlpha = -4.5 TO 4.5 STEP .5
  FOR cpbeta = -4.5 TO 4.5 STEP .5
   INPUT #1, a, b, c, d, e, f
   x% = cpAlpha * 2 + 10
   y% = cpbeta * 2 + 10
   table1a(probe%, x%, y%) = e
   table2a(probe%, x%, y%) = f
  NEXT cpbeta
 NEXT cpAlpha

FOR ndhol% = 6 TO 1 STEP -1
 FOR cpAlpha = 0 TO 2 STEP .25
  FOR cpbeta = -2 TO 2 STEP .5
   INPUT #1, a, b, c, d, e, f
   x% = cpAlpha * 4 + 1
   y% = cpbeta * 2 + 5
   table1b(probe%, ndhol%, x%, y%) = e
   table2b(probe%, ndhol%, x%, y%) = f
  NEXT cpbeta
 NEXT cpAlpha
 NEXT ndhol%

NEXT probe%

CLOSE 1

END SUB

SUB readscales
DIM s(10)

REM * PIO 96 base address
basead2 = &H2D0

REM * reset
resetmux = &H4
inhibit1 = &H2
inhibit2 = &H1

REM * set up PIO 96 to talk to digital MUX
OUT basead2 + 3, &H9B
OUT basead2 + 7, &H90

OUT basead2 + 5, 0              ' reset scales
 REM * reset mux
 OUT basead2 + 5, resetmux
 FOR i = 0 TO 99: NEXT
 OUT basead2 + 5, 0
 REM * read data
```

```
 FOR n% = 0 TO 9
  OUT basead2 + 5, inhibit2
  OUT basead2 + 5, 0
  b4 = INP(basead2 + 4)
  b3 = INP(basead2 + 2)
  b2 = INP(basead2 + 1)
  b1 = INP(basead2)
  OUT basead2 + 5, inhibit1
  OUT basead2 + 5, 0
  IF (b4 AND 128) = 128 THEN b4 = b4 - 128: SIGN = -1 ELSE SIGN = 1
  IF (b1 > &H99) OR (b2 > &H99) OR (b3 > &H99) OR (b4 > &H99) THEN
   num = -999.99
  ELSE
   num = SIGN * (VAL(HEX$(b4)) * 100000 + VAL(HEX$(b3)) * 1000 + VAL(HEX$(b2)) * 10
+ VAL(HEX$(b1)) / 10)
  END IF
 s(n%) = num
 NEXT n%


 FOR n% = 0 TO 9
  IF s(n%) <> -999.99 THEN scale(n%) = s(n%)
 NEXT n%


END SUB

SUB SetUpCompumotor

'Set-up Encoder Status Block for 5 AR W/RS422 Serial Absolute Encoders.
'The "Status Block" is a series of array variables that contain (after
'initialization of the encoders) the current status of each encoder. By
'modifying these initial values, the encoders can be brought up in any
'state. The parameter strings (P$(n)) hold the command string used to
'communicate with the encoders. The ACT array holds either 0 or 1 for
'each encoder. If the encoder is to be used (ie. active) the ACT varaible
'will hold a 1 for that encoder. Otherwise ACT will hold a zero, causing
'the initialization section to ignore that encoder.

'ENCODER STATUS BLOCK
'Set-up's for Parameter command strings

P$(0) = "SN"'Set encoder Address Character (0-15)
P$(1) = "SE"'Set encoder Error Checking (0-Off,1-On)
P$(2) = "SD"'Set encoder Direction (0-inc. cclkwise,1-inc. clockwise)
P$(3) = "SM"'Set encoder Serial Duplex (0-Full,1-Half >see manual<)
P$(4) = "SO"'Set encoder Position Offset (0-cur. pos=0.000,A-pos=absolute)
P$(5) = "SF"'Set encoder Scale Factor (uffff-units digit & 4 dec. places)
P$(6) = "SP"'Set encoder Data Format (0-Bin/Hex,1-BCD/Dec [par/ser])
P$(7) = "AR"'Set encoder Auto Report (0-Off,1-On)

'Initialize each Encoder Block

'Encoder 0
```

```
ESB$(0, 0) = "0"   'SN Encoder 0 address is 0
ESB$(0, 1) = "1"   'SE Error checking OFF
ESB$(0, 2) = "1"   'SD Encoder counts increase as shaft turns clockwise
ESB$(0, 3) = "1"   'SM Serial HALF duplex
ESB$(0, 4) = "A"   'SO Position Offset OFF (ie. position is absolute)
ESB$(0, 5) = "10000"'SF Scale Factor is 1.0000 (ie. counts, not eng. units)
ESB$(0, 6) = "2"   'SP Parallel output in BCD, Serial output in Dec.
ESB$(0, 7) = "1"   'AR Auto Report Mode ON
ESB$(0, 8) = "Horizontal    "

'Encoder 1
ESB$(1, 0) = "1"   'SN Encoder 1 address is 1
ESB$(1, 1) = "1"   'SE Error checking ON
ESB$(1, 2) = "1"   'SD Encoder counts increase as shaft turns clockwise
ESB$(1, 3) = "1"   'SM Serial HALF duplex
ESB$(1, 4) = "A"   'SO Position Offset OFF (ie. position is absolute)
ESB$(1, 5) = "10000"'SF Scale Factor is 1.0000 (ie. counts, not eng. units)
ESB$(1, 6) = "2"   'SP Parallel output in BCD, Serial output in Dec.
ESB$(1, 7) = "1"   'AR Auto Report Mode ON
ESB$(1, 8) = "Longitudinal "

'Encoder 2
ESB$(2, 0) = "2"   'SN Encoder 2 address is 2
ESB$(2, 1) = "0"   'SE Error checking OFF
ESB$(2, 2) = "1"   'SD Encoder counts increase as shaft turns clockwise
ESB$(2, 3) = "1"   'SM Serial HALF duplex
ESB$(2, 4) = "A"   'SO Position Offset OFF (ie. position is absolute)
ESB$(2, 5) = "10000"'SF Scale Factor is 1.0000 (ie. counts, not eng. units)
ESB$(2, 6) = "1"   'SP Parallel output in BCD, Serial output in Decimal
ESB$(2, 7) = "1"   'AR Auto Report Mode ON
ESB$(2, 8) = "Vertical      "

'Encoder 3
ESB$(3, 0) = "3"   'SN Encoder 3 address is 3
ESB$(3, 1) = "0"   'SE Error checking OFF
ESB$(3, 2) = "1"   'SD Encoder counts increase as shaft turns clockwise
ESB$(3, 3) = "1"   'SM Serial HALF duplex
ESB$(3, 4) = "A"   'SO Position Offset OFF (ie. position is absolute)
ESB$(3, 5) = "10000"'SF Scale Factor is 1.0000 (ie. counts, not eng. units)
ESB$(3, 6) = "1"   'SP Parallel output in BCD, Serial output in Decimal
ESB$(3, 7) = "1"   'AR Auto Report Mode ON
ESB$(3, 8) = "Encoder 3     "

'Encoder 4
ESB$(4, 0) = "4"   'SN Encoder 4 address is 4
ESB$(4, 1) = "0"   'SE Error checking OFF
ESB$(4, 2) = "1"   'SD Encoder counts increase as shaft turns clockwise
ESB$(4, 3) = "1"   'SM Serial HALF duplex
ESB$(4, 4) = "A"   'SO Position Offset OFF (ie. position is absolute)
ESB$(4, 5) = "10000"'SF Scale Factor is 1.0000 (ie. counts, not eng. units)
ESB$(4, 6) = "1"   'SP Parallel output in BCD, Serial output in Decimal
ESB$(4, 7) = "1"   'AR Auto Report Mode ON
ESB$(4, 8) = "Encoder 4     "
```

```
'Set-up's for each encoder

 ACT(0) = 1   'Encoder 1 = on  (Horizontal)
 ACT(1) = 1   'Encoder 1 = on  (Longitudinal)
 ACT(2) = 1   'Encoder 2 = on  (Vertical)
 ACT(3) = 0   'Encoder 3 = off
 ACT(4) = 0   'Encoder 4 = off


hveloc = 12

END SUB

SUB SloSurvey

DIM P(8), O(4)

 CALL analoginit

 COLOR 15
 PRINT "     WAKE SURVEY"
 COLOR 7
 PRINT "        ALIGN SURVEY RIG TO DESIRED LONGITUDINAL POSITION"
 PRINT
 PRINT "        OK TO CONTINUE? [Y/N]"

 op = keystroke("YyNn")
 SELECT CASE op
  CASE 1, 2: GOTO 10000
  CASE 3, 4: GOTO 20000
 END SELECT



10000 :

 PRINT "        MODIFIED FOR FLAP EDGE TEST"
 INPUT "        ENTER STARTING COORDINATES (HORZ,VERT)    : "; h0, v0
 INPUT "        ENTER ENDING COORDINATES   (HORZ,VERT)    : "; h1, v1
 PRINT
 LINE INPUT "        ENTER COMMENTS : "; c$

 IF h1 < h0 THEN SWAP h0, h1
 IF v1 < v0 THEN SWAP v0, v1

 dv = v1 - v0: IF dv < 2 THEN dv = 2
 dh = h1 - h0: IF dh < 10 THEN dh = 10

 CALL GetPosition: 10 = tX
 PRINT "        MOVING TO INITIAL SURVEY LOCATION"
 CALL moveto(h0, v0, 10)

 LINE INPUT "ENTER FILESPEC FOR WAKE SURVEY DATA :"; f$
 IF f$ = "" THEN f$ = "c:\noname.dat"
```

```
 OPEN "O", 2, f$


 PRINT "        BEGINNING SLOW SURVEY"

 Talk8400 ("SC4 0")' disable front panel echo

 WRITE #2, test$
 WRITE #2, DATE$, TIME$
 WRITE #2, c$
 PRINT #2, ""
 WRITE #2, "X", "Y", "Z", "Q", "TT", "RH", "BARO", "P11", "P12", "P13", "P14",
"P15", "P16", "P17", "PT1", "PS1", "A1", "B1", "P21", "P22", "P23", "P24", "P25",
"P26", "P27", "PT2", "PS2", "A2", "B2", "P31", "P32", "P33", "P34", "P35", "P36",
"P37", _
 "PT3", "PS3", "A3", "B3"

 v0 = v0 - 1


 WHILE Tz < v1
  v0 = v0 + 1

REM * auto horz. domain for flap-edge test

  hveloc = 1: REM set a slow horizontal drift velocity

  h0 = 0: h1 = 15: dh = 15
  IF v0 > 20 THEN h0 = -25: h1 = 0: dh = 25
  IF v0 <= 20 THEN h0 = -10: h1 = 5: dh = 15
  IF INT(v0 / 4) * 4 = v0 THEN h0 = -30: h1 = 25: dh = 55

  hveloc = 10


  CALL GetPosition

  IF ABS(ty - h0) < .1 THEN
     ' at left boundary
     CALL moveto(h0, v0, 10)
     hx = h1
     direction = 1
    ELSE
     CALL moveto(h1, v0, 10)
     hx = h0
     direction = -1
    END IF

  REM * Here is a good place to acquire non-7hp telemetry
  REM *   Barocell, Tunnel Temp, Humidity

  CALL readanalog
  CALL readscales
```

```
  qscale = scale(6)
  IF scale(7) > -110 AND scale(7) < 0 THEN tt = -scale(7)
  rh = scale(8) / 10

  pBARO = volts(1) / .1001992: IF pBARO < 13 OR pBARO > 16 THEN pBARO = 14.7

  qcse = qscale ' converted to psf

' STATIC PLATE CORRECTION FUNCTION (after Wadcock, 7x10ist, 1996)

' aft static ring (#1) - nearest test section
'dqsp = -0.01197 - .01395 * qcse - .000145 * qcse ^ 2 - 3.491 E-07 * qcse ^

' mid static ring (#2)
'dqsp = .00773 - .1191 * qcse - 7.13e-5 * qcse ^ 2 - 3.67E-07 * qcse ^ 3

' fwd static ring (#3) - nearest settling chamber
 dqsp = .00011 - .1571 * qcse - .000148 * qcse ^ 2 - 2.603E-07 * qcse ^ 3

 Qps = qcse - dqsp ' q from pitot-static on c/l of tunnel - using static ring # 3


  HDist = h1 - h0: Hstep& = -HDist * Hscale * direction: REM CALC steps
  hveloc = 1: REM return to usual horizontal speed
  IF INT(v0 / 4) * 4 = v0 THEN hveloc = 2

  CALL TalkCompumotor(Hstep&, 0, 0)

  REM * wait a wee bit to let the motors spool up
  t0 = TIMER + 1
  DO
  LOOP UNTIL (TIMER > t0)

  PRINT "  Y      Z    | QPRBE| PT1   PS1   A1    B1 | PT2   PS2   A2    B2   |  PT3   PS3   A3
B3 "


  WHILE (ABS(ty - hx) > .1)

   REM * ACQUIRE PSI-DATA HERE
   Talk8400 ("AD1 1 1")



   REM * Get out position while the PSI is out doing its thing....
   CALL GetPosition

   REM * We can also start doing disk I/O as well
   PRINT #2, USING "+##.#!+##.#!+##.#!"; tX; ","; ty; ","; Tz; ",";
   PRINT #2, USING "+##.#!+##.#!+##.###!+##.#!"; tt; ","; rh; ","; pBARO; ","; Qps;
","";

   REM Wait 1/5 econd
```

```
   t0 = TIMER + .2
   DO
   LOOP UNTIL (TIMER > t0)

   REM * get the data from the PSIs
   Talk8400 ("OD1")

   t0 = PktInfo(18) + PktInfo(19) / 1000

   PRINT USING "+##.# +##.# | +### |"; ty; Tz; Qps;

 probe% = 0: P(1) = pktXdat(1): P(2) = pktXdat(2): P(3) = pktXdat(3)
            P(4) = pktXdat(4): P(5) = pktXdat(5): P(6) = pktXdat(6)
            P(7) = pktXdat(7): P(8) = Qps
  CALL crunch(P(), O(), probe%)
  PRINT USING "+## +## +## +## | "; O(1) * 144; O(2) * 144; O(3); O(4);
  PRINT #2, USING "+#.####!"; P(1); ","; P(2); ","; P(3); ","; P(4); ","; P(5); ",";
P(6); ","; P(7); ",";
  PRINT #2, USING "+#.####!"; O(1); ","; O(2); ",";
  PRINT #2, USING "+##.#!"; O(3); ","; O(4); ",";
  tp0 = O(1)

 probe% = 1: P(1) = pktXdat(8): P(2) = pktXdat(9): P(3) = pktXdat(10)
            P(4) = pktXdat(11): P(5) = pktXdat(12): P(6) = pktXdat(13)
            P(7) = pktXdat(14): P(8) = Qps
  CALL crunch(P(), O(), probe%)
  PRINT USING "+## +## +## +## | "; O(1) * 144; O(2) * 144; O(3); O(4);
  PRINT #2, USING "+#.####!"; P(1); ","; P(2); ","; P(3); ","; P(4); ","; P(5); ",";
P(6); ","; P(7); ",";
  PRINT #2, USING "+#.####!"; O(1); ","; O(2); ",";
  PRINT #2, USING "+##.#!"; O(3); ","; O(4); ",";
  tp1 = O(1)

 probe% = 2: P(1) = pktXdat(15): P(2) = pktXdat(16): P(3) = pktXdat(17)
            P(4) = pktXdat(18): P(5) = pktXdat(19): P(6) = pktXdat(20)
            P(7) = pktXdat(21): P(8) = Qps
  CALL crunch(P(), O(), probe%)
  PRINT USING "+## +## +## +##"; O(1) * 144; O(2) * 144; O(3); O(4)
  PRINT #2, USING "+#.####!"; P(1); ","; P(2); ","; P(3); ","; P(4); ","; P(5); ",";
P(6); ","; P(7); ",";
  PRINT #2, USING "+#.####!"; O(1); ","; O(2); ",";
  PRINT #2, USING "+##.#!"; O(3); ","; O(4); " "
  tp2 = O(2)

  WEND
  BEEP
  CALL ibonl(psi%, 0)
  psi% = 0
  IF INKEY$ = CHR$(27) THEN 30000

 WEND: REM * have finished a horizontal traverse


30000 :
```

```
 Talk8400 ("SC4 1")' front panel echo on
 CALL ibonl(psi%, 0): psi% = 0

 PRINT " SURVEY COMPLETE - RETURNING TO HOME POSITION"
 CALL moveto(h0, v0, l0)


 CLOSE 2


20000 :


 hveloc = 12
END SUB

SUB SmartMove

GetPosition


INPUT "Enter New Position (H,V,L) : "; ty, Tz, tX
H = ty: V = Tz: l = tX
CALL moveto(H, V, l)

BEEP

END SUB

SUB StillMoving

DO
GetPosition
LOOP UNTIL (ERP = false)

checkagain:
 HOld = ty: VOld = Tz: LOld = tX

 t0 = TIMER + .5: WHILE (TIMER < t0): WEND

 DO
  GetPosition
 LOOP UNTIL (ERP = false)


 IF INKEY$ <> "" THEN GOTO KeyBreak
 IF (ty < HLoLim) THEN GOTO OutOfBounds
 IF (ty > HHiLim) THEN GOTO OutOfBounds
 IF (Tz < VLoLim) THEN GOTO OutOfBounds
 IF (Tz > VHiLim) THEN GOTO OutOfBounds
 IF (tX < LLoLim) THEN GOTO OutOfBounds
 IF (tX > LHiLim) THEN GOTO OutOfBounds
```

```
 IF ((HOld = ty) AND (VOld = Tz) AND (LOld = tX)) THEN
   GOTO FINISHED
  END IF

GOTO checkagain


KeyBreak:
COLOR 14
PRINT "BREAK"
PRINT
PRINT "Interrupted by User"
GOTO StopMotor

OutOfBounds:
' check again
 GetPosition

 IF (ty < HLoLim) THEN GOTO Out2
 IF (ty > HHiLim) THEN GOTO Out2
 IF (Tz < VLoLim) THEN GOTO Out2
 IF (Tz > VHiLim) THEN GOTO Out2
 IF (tX < LLoLim) THEN GOTO Out2
 IF (tX > LHiLim) THEN GOTO Out2
 GOTO checkagain

Out2:
COLOR 12
PRINT "BREAK - RIG AT "; ty; Tz; tX
PRINT
PRINT "Survey Rig Out of Bounds!"

StopMotor:
 OPEN "O", 3, port1$
 PRINT #3, "STOP;"
 CLOSE 3

PRINT "PRESS [R] to RESUME (i.e. glitch on the encoders) any other key to stop"
a$ = ""
WHILE a$ = ""
 a$ = INKEY$
WEND
IF a$ = "R" OR a$ = "r" THEN GetPosition: GOTO checkagain ELSE GOTO FINISHED

FINISHED:


END SUB

SUB Talk8400 (CMD$)

'init psi
IF psi% = 0 THEN CALL IBDEV(0, 5, 0, 13, 1, 0, psi%): PRINT psi%
IF psi% = -1 THEN PRINT "IEEE488 INIT FALIURE": STOP
```

```
REM * send-wait-read to the PSI-8400

IF echo% = 1 THEN COLOR 10: PRINT , CMD$; : COLOR 7

IF icr% THEN CMD$ = CMD$ + CHR$(13)

CALL ibwait(psi%, &H8)
CALL IBWRT(psi%, CMD$)

x$ = buffer3$

DO
 CALL ibrsp(psi%, spr%)

LOOP WHILE (spr% AND &H40) = 0

CALL ibwait(psi%, &H4)

IF echo% = 1 THEN PRINT

CALL IBRD(psi%, x$)        ' IEEE Read routine
IF (ibsta% AND EERR) THEN PRINT "IBRD ERROR": STOP

buffer$ = LEFT$(x$, IBCNTL&)

CALL DecodeBuffer(buffer$)


END SUB

SUB TalkCompumotor (Hstep&, Vstep&, Lstep&)

'Set Up and run Utility Program in the Compumotor 3000.

PRINT USING "       STEP    : +###### +###### +######"; Hstep&; Vstep&; Lstep&
OPEN "O", 3, port1$
 PRINT #3, "STOP;"
 PRINT #3, "LOAD;"
 PRINT #3, "7000 VELC "; hveloc; " "; LVELOC; " "; VVELOC; ";"
 PRINT #3, "7001 ACEL "; HAccel; " "; LAccel; " "; VAccel; ";"
 PRINT #3, "7002 MOVE "; Hstep&; " "; Lstep&; " "; Vstep&; ";"
 PRINT #3, "7003 DONE;"
 PRINT #3, "*;"
 PRINT #3, "RUN FROM 7000;"
CLOSE 3

END SUB
```

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. AGENCY USE ONLY *(Leave blank)*	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	May 1997	Technical Memorandum

4. TITLE AND SUBTITLE
Measurement of Air Flow Characteristics Using Seven-Hole Cone Probes

5. FUNDING NUMBERS
505-59-53

6. AUTHOR(S)
Timothy T. Takahashi

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
Ames Research Center
Moffett Field, CA 94035-1000

8. PERFORMING ORGANIZATION REPORT NUMBER
A-976463A

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)
National Aeronautics and Space Administration
Washington, DC 20546-0001

10. SPONSORING/MONITORING AGENCY REPORT NUMBER
NASA TM-112194

11. SUPPLEMENTARY NOTES
Point of Contact: Timothy T. Takahashi, Ames Research Center, MS 247-2, Moffett Field, CA 94035-1000
(415) 604-4976

12a. DISTRIBUTION/AVAILABILITY STATEMENT
Unclassified — Unlimited
Subject Category 09, 35

12b. DISTRIBUTION CODE

13. ABSTRACT *(Maximum 200 words)*

The motivation for this work has been the development of a wake survey system. A seven-hole probe can measure the distribution of static pressure, total pressure, and flow angularity in a wind tunnel environment. The author describes the development of a simple, very efficient algorithm to compute flow properties from probe tip pressures. Its accuracy and applicability to unsteady, turbulent flow are discussed.

14. SUBJECT TERMS
Instrumentation, Pressure, Flow angle, Flow velocity

15. NUMBER OF PAGES
57

16. PRICE CODE
A04

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
Unclassified	Unclassified		