MEASUREMENT OF VORTEX FLOW FIELDS

T. Kevin McDevitt, Todd A. Ambur, Gary M. Orngard, F. Kevin Owen

COMPLERE INC.
Palo Alto, CA

Contract NAS1-18667
January 1992
Abstract

The objective of Phase II was to design, build and demonstrate a three dimensional laser fluorescence anemometer for use in the Langley 16- by 24- Inch Water Tunnel. Innovative optical design flexibility combined with compact and portable data acquisition and control systems have been incorporated into the instrument. This will allow its use by NASA in other test facilities. The final instrument and support systems differ in several significant aspects from the design envisaged during our Phase II proposal preparations. Our original mirror traverse alignment concept has been replaced by a more versatile fiber optic system. This facilitates normal and off-axis beam alignment, removes mirror losses and improves laser safety. This added optical flexibility will also enable simple adaptation for use in the adjacent jet facility. New proprietary concepts in transmitting color separation, light collection and novel prism separation of the scattered light have also been designed and built into the system. Off-axis beam traverse and alignment proved much more complex than initially conceived. This led to the requirement for a specialized, programmable traverse controller and the inclusion of an additional traverse for the off-axis arm. To meet this challenge, an “in-house” prototype unit was designed and built and traverse control software developed specifically for the water tunnel traverse applications. A specialized data acquisition interface was also required. This was designed and built for the Laser Fluorescence Anemometer system.

Introduction

At present, significant efforts are being made to effect design changes which will improve aircraft agility, maneuverability and performance. But, although significant progress has been made in computational aerodynamics, reliable design changes still cannot be made without recourse to experiment. Attempts to extend tactical flight envelopes still require extensive preflight ground based model testing. Unfortunately, conventional wind tunnel testing is expensive and time consuming and most facilities were built before present-day optical methods for quantitative flow field measurements were envisaged. Consequently, the non-intrusive detailed documentations of lee-side vortex flow-fields which are often required to support design evaluation and optimization are few.

However, in the past, qualitative water tunnel simulations have guided many practical designs and, since most of these facilities have been built with excellent optical access, they are ideally suited for use in advanced flow field diagnostics. Since the performance of most lifting and maneuvering bodies is governed by extensive transitional and turbulent viscous wakes and vortical lee-side flows, non-intrusive optical measurement techniques are required. Consequently, water tunnels offer the opportunity to obtain inexpensive, detailed flow field measurements to support "cut and try" designs and basic research.

To realize this capability, a two dimensional laser fluorescence anemometer was built and tested in the Ames-Dryden Water Tunnel during Phase I. The instrument was used in an experimental study of vortex flow fields designed to determine the mechanisms and feasibility of controlling vortex breakdown by introducing relatively low rates of jet blowing along the vortex core. The objective of Phase II was to build a three dimensional instrument for studies in the
Background

When a slender delta wing is at an angle of attack to an oncoming stream, the upper and lower surface boundary layers flow outward and separate from the leading edges to form two free shear layers that roll up into a pair of vortices above the wing. Increasing angle of attack strengthens the vortices until the induced wing pressure field and associated adverse streamwise pressure gradients cause vortex breakdown. The flow is further complicated as the leading edge vortices mix with the wake from the trailing edge downstream of the wing. The phenomenon of vortex breakdown (or vortex bursting) can have a significant influence on control surface performance and unsteady loading. The inherent unsteadiness of the breakdown process compounds the problem as it continually moves the breakdown region back and forth along the vortex axis. This creates serious time dependent flow problems and asymmetrically disposed breakdown positions above the wing that are aggravated with side-slip.

Wide variations of breakdown patterns have been observed, and with increasing swirl the patterns change from spiral to near axisymmetric (Ref. 1). Spiral breakdown most commonly occurs over delta wings. In this breakdown process, the filament of fluid along the axis does not spread out symmetrically from a fixed stagnation point but, instead, takes on a spiral form around an unsteady "stagnation point" which varies in both space and time. Axisymmetric breakdown over delta wings, although rare, can also occur (Ref. 2). In this case, the vortex has a roughly axisymmetric breakdown pattern with a characteristic bubble which can have single or multiple tails (Ref. 3).

Unfortunately, the parameters and conditions that result in vortex breakdown are poorly understood, because reliable quantitative experimental data are difficult to obtain. With limited experimental information to guide flow field modeling, numerical studies of vortex breakdown and control have met with only limited success (Ref. 4). There have been two principal reasons for this. In the first place, flow field unsteadiness associated with breakdown produces directional intermittency. This leads to large uncertainties in mean and unsteady flow measurements obtained with conventional pitot and hot wire probes. Secondly, and perhaps more important, is the fact that vortex breakdown is known to be extremely sensitive to any form of introduced disturbance. Probes, due to their blockage, may drastically alter the breakdown position. For these reasons, almost complete reliance has been placed on flow visualization techniques to determine flow field characteristics. In the past, air or hydrogen bubbles have been used as tracers to visualize flow patterns in water tunnels. For steady flows, streak lines can be identified with streamline patterns. However, in more complex flows of practical interest, the use of bubbles for flow visualization has distinct drawbacks. First of all, their introduction acts as a fluid lubricant which alters the apparent fluid viscosity and so its turbulent structure. Secondly, light refraction at the gas/water interfaces will destroy laser beam coherence and make it impossible to obtain laser velocimeter measurements in the regions where the tracer is present. But, with the advent of the laser fluorescence anemometer, there are now opportunities to determine accurate quantitative flow field velocity measurements of the vortex bursting process.
The Laser Fluorescence Anemometer

The principle of operation of the laser fluorescence anemometer is shown schematically in Fig. 1. The mean velocity and turbulence measurements are made with a dual-beam velocimeter utilizing a Bragg cell that enables moving interference fringes to be generated in the focal volume so that instantaneous velocity magnitude and direction measurements can be achieved from the frequency shift ($f_D$) around the incident and modulated laser beam interference frequency ($f_0$).

\[ i.e. \ U = \frac{\lambda (f_D - f_0)}{2 \sin \left( \frac{\theta}{2} \right)} \]

where $\lambda$ is the wavelength of the incident laser light. Mean and fluctuating concentration measurements are achieved by observing the intensity of fluorescent light emitted from the focal volume at a different wavelength ($\lambda_c$). At correct levels, tracer fluorescence is linearly proportional to the trace material concentration and, therefore, fluorescent intensity is directly proportional to the concentration of fluid from the seeded flow in the focal volume. The cathode current from a second photomultiplier tube is coupled to a high-gain current to voltage converter to produce a continuous voltage proportional to the instantaneous concentration.

Since fluorescence is such a complex phenomenon dependent upon many parameters, only those of particular importance to the present application have been considered. In this context, the principal requirements were linearity combined with adequate sensitivity (i.e., signal/noise ratio), frequency response, and spatial resolution. Since the fluorescent intensity of a particular organic dye can be a strong function of the solvent, which in this case was room temperature water, other dye-solvent-temperature combinations would produce different (and possibly increased) fluorescence. However, the fluorescent output was more than adequate for the present tests. In addition, the relationship between fluorescent intensity and concentration is of course exponential but, at the extremely low dye concentrations used in these experiments, a linear approximation could be made without introducing significant errors.

The data in Fig. 2 show this linearity of the present technique, which employed dysodium fluorescein dye (a sodium salt of fluorescein which has been used for flow visualization for many years) as the trace material in water. Since, for a given dye concentration, the measured fluorescence is a function of laser beam intensity and collection optics, the ordinate of Fig. 2 has been plotted in arbitrary (voltage) units. The high signal-to-noise ratio that can be obtained for dye concentrations as low as 0.04 ppm (by weight) is illustrated in Fig. 3. The rise time ($\sim 50 \mu s$), which corresponds to the time taken to chop the laser beam, shows the adequate frequency response of the system.

Since there is usually an overlap of the absorption and emission curves for most dye-solvent combinations it is possible that fluorescent photons emitted from the probe volume could be reabsorbed by the dye molecules that are between the probe volume and the collection optics. This effect could produce a range dependent signal. However, fluorescent intensity measurements previously obtained across an entire test section when filled with stagnant water at maximum dye concentration showed that these effects and those of possible beam attenuation were negligible. Thus, unlike absorption techniques which measure integrated ("line of sight") properties, the receiving optical arrangement primarily governs the spatial resolution of the fluorescence technique. In the present experiments, off-axis light collection and multimode fiber aperture size resulted in a maximum focal volume length dimension of approximately 0.5 mm although smaller
spatial resolution can be achieved by appropriate choice of collection optics without affecting the LDV, since the velocimeter interference fringes are moving.

The instrument delivered to NASA LaRC comprises three primary elements namely: the optical, traverse control, and data acquisition systems. The Laser Optical system (Fig. 4) uses a 6 watt Argon-ion laser. The transmitting optical arrangement (Fig. 5) is straightforward with a few unique features addressing the common problem of beam distortion or thermal blooming at higher laser powers. Frequency shifting is done before the color separation prisms using a single Acousto-Optic modulator made of a selected flint glass which can handle the full laser power with minimal distortion. (For significantly greater laser powers we also found a water cooled Bragg cell which was more than required for this application.) This is followed by the color separation prisms, the first of which are fused silica for power handling capacity. A final prism of dense flint provides maximum dispersion once the laser beam has been split into at least eight beams. Final color selection is made using right angle prisms. The lines used for this experiment were 514.5nm, 488nm, 476.5nm. Although the emission spectrum of the fluorescence is centered about 515nm, there was sufficient higher wavelength emission for the edge filter to select wavelengths above 525nm. Other laser lines could have been selected if needed for fluorescence excitation or for separation from emission lines.

Pure fused-silica core single-mode polarization-preserving fibers are used for light transmission; two fibers per color. The use of optical fibers not only avoids the tedium of mirror-traverse alignment, but also greatly simplifies the transmission of light to the third axis. The pure fused silica core fibers seem immune from the progressive transmission losses which are found in other fibers. Polarization preserving fibers provide greater modal stability when the fibers are flexed or manipulated. For mechanical protection the fibers are armored and contained within a conduit. Upon exiting the fibers the beams are collimated at 4.4 mm dia. with a separation of 60mm.

Forward-scattered light is collected with a single 80mm diameter lens and focused into a 200 μm multi-mode optical fiber which conducts it to the color separation and signal detection box (Fig. 6). The fluorescence signal is split off with an edge filter. For maximum efficiency, a prism separation scheme is used rather than di-chroic filter and interference filters for the LDV signals.

Experience has shown that accurate positioning is vital to a successful test program. Accurate positioning is complicated in an air-window-water environment due to the difference in refractive indices on either side of the water tunnel window. Refractive index problems are particularly acute for the third component beams which are transmitted at 45 degrees to the normal incidence of the four beam axis. Fig. 7 illustrates the problem, which may not be intuitively obvious. In order to traverse the measurement volume horizontally some distance in water, the two orthogonal optical components, axial(x) and model vertical(z), must be traversed some lesser amount in air. In order for the third or off-axis pair of beams to intersect the measurement volume at the required angle of thirty degrees, the beams must strike the window at an angle of approximately forty-five degrees. In addition, linear horizontal movement of the focal point of the third pair of beams requires lens movement on a sloping line. As the lens gets closer to the window, it will have to rise. The length of movement on this sloping path will be only about half
of the movement of the focal point in water. Thus, two lens systems must move on different paths at different speeds in order to maintain a coincident focal point. To sort this all out, two three-axis traverse tables were installed for computer controlled, algorithm driven positioning of the LDV probe volume and forward scatter collecting optics. A fourth traversing axis supports the off-axis transmitted beams. Position is maintained by a custom designed eight axis traverse controller with micro-stepping drives, optical encoder feedback, and limit switch safety stops. Details of the Traverse Control System are given in Appendix A.

A Laser Velocimeter Data Acquisition System (LVDAS) has been designed. This instrument processes one to three channels of LDV data and digitizes up to four channels of analog data, one of which represents the concentration of dye. The instrument ensures coincidence and multiplexes the data to the computer. Velocities and analog channel values are displayed as well as data rates. Details are given in Appendix B.

Fig. 8 shows the modified data handling system for the 3D laser fluorescence anemometer. The continuous though not necessarily non-zero output from the high-gain current to voltage converter is fed directly into an analog to digital converter to provide 12 bits of digital information at 50 kHz. In water flows, this was more than sufficient to provide essentially real time point concentration data in digital form. But data from the three component LV system were not continuous wave since particle arrival times in the focal volume were random. However, whenever valid and essentially coincident times data were received on all LV channels, a necessary requirement for shear stress measurement, these velocities, along with the instantaneous concentration voltage, was recorded. From a series of such readings, mean and turbulent velocity and concentration profiles were determined along with the turbulent shear stress and velocity/concentration cross correlations. These latter cross products provide new information on turbulent mixing rates in complex flows. In addition, we are able to determine details of the concentration/turbulent intermittencies from ensemble averages generated for selected instantaneous concentration levels. This will shed quantitative light on the turbulent structure and entrainment of fluid originating from different points in the flow field.

**Experimental Details**

**Test Facility**

The NASA Langley 16- by 24- Inch Water Tunnel is shown in Fig. 9. The tunnel has a vertical test section with an effective working length of about 4.5 ft. The velocity in the test section can be varied from 0 to 0.75 ft/sec., resulting in unit Reynolds numbers from 0 to 7.73 x 10^4 ft\(^{-1}\) based on a water temperature of 78°F. The normal test velocity yielding smooth flow is 0.25 ft/sec, resulting in unit Reynolds numbers of 2.58 x 10^4 ft\(^{-1}\) at 68°F. The model support system has deflection ranges of ±33° and ±15° in two planes of rotation. Rotation is accomplished via electronic remote control, and visual indicators allow the user to set angles within about ±0.25°.

The fluorescence seeding method for this investigation used fluorescein dye injected into the jet flow field from inside the model. Naturally-occurring particles in the water were used for seeding for the LDV part of the instrument. A representative size distribution provided by NASA is shown in Fig. 10.
Model

The test model selected by NASA was a non-axisymmetric afterbody propulsion model and is shown in Fig. 11. This model is a scaled-down version of a model to be tested in the NASA Langley 16-foot Transonic Wind Tunnel as part of a computational fluid dynamics code validation study. It can be used to simulate nozzle exit velocity ratios typical of those in the wind tunnel study. It consists of a generic forebody with a non-axisymmetric boattail and nozzle. Water is injected into the interior of the model and exhausted through flow-conditioning foam ahead of the throat and exits through the nozzle. Fluorescein dye is introduced upstream of the model in the water supply tube for the jet.

Sample Test Results

Unfortunately, optical beam refraction problems caused by complex test section wall deformations (=0.10") under hydrostatic loading impeded laser beam alignment although limited data were obtained at zero angle of attack and a nominal jet exit to free stream velocity ratio of 1.5. For these test conditions, the cross flow velocity components (v,w) were negligible as expected. However, the model did provide a flowfield in which the basic instrument concepts could be verified. Fig. 12 shows the measured axial velocity distributions. It shows the extent of the jet and the location of the velocity defects in the model wall wakes. Clearly, the jet is highly skewed, but this has since been corrected by subsequent modification of the internal model flow treatment devices. The mean concentration profile (F g. 13), which is much more symmetric, defines the extent of the jet at this axial station. Some insight into the unsteady features of the flow can be obtained from the fluctuation measurements. Figs. 14 and 15 show there is significant mixing in the outer shear layers between the jet and freestream flows. Peak fluctuation levels in the regions of maximum mean gradients indicate that small scale mixing is dominant. A measure of the level of streamwise mixing in the jet can be determined from Fig. 16. As expected the u'c' cross correlation function is positive as faster moving jet fluid is associated with higher concentration whereas fluid originating from the slower moving freestream has lower or zero concentration.

Concluding Remarks

A Laser Fluorescence Anemometer which comprises a three component laser doppler velocimeter system with a fourth channel to measure fluorescent dye concentration has been installed in the NASA Langley Water Tunnel. The system includes custom designed optics, data acquisition, and traverse control instruments and a custom software package.

Feasibility studies clearly demonstrate how water tunnels can be used in conjunction with advanced optical techniques to provide non-intrusive detailed flow field measurements of complex fluid flows with a minimum of expense. The measurements show that the laser fluorescence anemometer will provide new insight into the structure, entrainment and mixing of vortical and shear layer flows.
References


![Fig. 1 Laser Fluorescence Anemometer](image-url)
5.0

30 m²

0.0

DYE CONCENTRATION (ppm by wt.)

0.0

0.2

0.4

1.0

2.0

3.0

4.0

5.0

FLUORESCENT INTENSITY
(ARBITRARY UNITS)

Fig. 2 Relationship Between Dye Concentration and Fluorescent Output

0.5 V

0.2 ms

Fig. 3 Fluorescence Sensitivity and Frequency Response
Fig. 4a Schematic of Plan View LFA System

Fig. 4b Schematic of Side View LFA System

Fiber Optic Cables/Conduit
Fig. 5 Sending Optics
Fig. 6 Receiving Optics
Beam spacing = 60mm
Focal length = 19.413"
Off axis angle = 45°
Auxiliary slide angle = 30.95°
Lucite thickness = 1.25"
Water - n = 1.333
Lucite - n = 1.43
Air - n = 1.00

Fig. 7 The Refraction Problem
Fig. 8 Data Acquisition and Traverse Control Systems
Water pump (800 gal/min)

Water filter system

Water storage tank

Observation window

Traversing probe mechanism location

Work platform

Flow direction

16- by 24-in. test section

Fig. 9 Langley 16- by 24- Inch Water Tunnel
Fig. 10 Size Distribution Particle Concentration
Fig. 11 Propulsion Model
Fig. 12 Axial Velocity Profile

Fig. 13 Concentration Profile
Fig. 14 Velocity Fluctuations

Fig. 15 Concentration Fluctuations
Fig. 16 Velocity-Concentration Cross-Correlation
Appendix A. The Traverse Control System

The traverse control system is made up of four subsystems, see Fig. A1. The first subsystem is the main data taking computer, the HP 9000-330. The second subsystem, the TCS8 (Traverse Control System 8 Axis), receives high level traverse commands from the HP 9000. The full duplex serial communications that links these two subsystems allows the HP 9000 to monitor the position and status of each axis in the system, see Appendix A.3 TCS8’s Serial Interface Command Descriptions. The TCS8 can also function as a stand alone traverse controller.

Through the use of the TCS8’s front panel, an operator can execute all of the commands that the HP 9000 can, plus the operator can control all axes in jog mode, see Appendix A.1 TCS8’s Front Panel Descriptions and Appendix A.2 TCS8’s Local Command Descriptions. The third subsystem, the MDS (Motor Drive System), is controlled solely by the TCS8. The TCS8 translates the high level commands from the HP 9000 and its front panel into low level indexer commands, see The Compumotor AX Drive User Manual previously delivered. The TCS8 also receives encoder pulses from the traverses via the MDS. This allows the TCS8 to display realtime position information on its front panel. The fourth and final subsystem of the traverse control system is the slide, motor, encoder, and limit switches that make up each axis. A drawing of each cable that is used to connect the traverse control system is included in Appendix A.4 Traverse Control System Cables.

The TCS8

The TCS8 is a microprocessor controlled system designed to interface an operator with a traverse system. The operator can utilize the TCS8 through the front panel, see Appendix A.1 TCS8’s Front Panel Descriptions and Appendix A.2 TCS8’s Local Command Descriptions, and/or with one or two host computers over serial interfaces, see Appendix A.3 TCS8’s Serial Interface Command Descriptions. The TCS8 stores all the critical parameters of motion, for each of the eight axes that it controls, in non-volatile memory. The critical parameters of motion being: position, encoder counts per unit travel, encoder counts per motor revolution, velocity, and acceleration. All of these parameters may be viewed, set, and saved. The TCS8 has three modes of motion. They are absolute, relative, and jogged. With absolute movements, the operator specifies the final location; with relative movements, a distance is specified; and with jogged movements the operator presses a jog key on the front panel of the TCS8 until the desired location is obtained.

The Motor Drive System

Each of the two MDS’s have 4 indexer/drivers contained within them. The TCS8 communicates with the indexers in the MDS’s over a closed loop serial daisy chain. When two MDS’s are used, as in this setup, the first MDS in the chain must be set to 8 and the second set to 4. By setting the first MDS to 8, the operator is opening the closed loop serial daisy chain allowing the second MDS to be included in the chain. The 4/8 switch is located on the back panel of each MDS, see Fig. A2. This figure also shows the location of all the motor, limit, and encoder...
connections. Channels X1, X2, Y1, and Y2 of the TCS8 control axis 1 through 4 on the first MDS and channels Z1, Z2, A1, and A2 control axis 1 through 4 on the second MDS. The TCS8 Encoders connector on the back of each MDS has a corresponding connector of the back of the TCS8, see Fig. A3 Schematic of TCS8 Back Panel. The interconnecting cable is detailed in Appendix A.4 Traverse Control System Cables.

**Positioning Resolution**

The indexer/drivers that are used in the MDS can drive the motors at 12,800 steps/revolution. The encoder used on each axis are 1000 pulses/revolution with quadrature encoding. Quadrature encoding adds a factor of 4 to the number of pulses/revolution to make this number 4000 pulses/revolution. This number, 4000 pulses/revolution, is well within the limit of 12,800 steps/revolution set by the indexer. The final factor in the product of the resolution of an axis is the number of threads/inch of the lead screw. All of the axes of the traverse system, except the auxiliary axis, have lead screws of 5 threads/inch, the auxiliary axis has a lead screw of 10 threads/inch. So the positioning resolution of the axes with a 5 threads/inch lead screw is 0.00005 inches and the auxiliary axis has a resolution of 0.000025 inches.
Fig. A1 Langley Traverse Control System
Fig. A2 Schematic of Motor Drive System Back Panel
Fig. A3 Schematic of TCS8 Back Panel
1. POSITION DISPLAY WINDOWS
There are eight windows corresponding to the eight axes that the TCS8 is capable of controlling. The position of each axis is continuously updated, by monitoring its encoder, and displayed in a fixed format of a sign, two digits, a decimal point, and four digits.

2. POWER KEY
The power key is used to store the current configuration to nonvolatile memory before turning off power to the TCS8. Pressing the power key turns the displays off and saves the current configuration. Pressing it again turns the displays back on. This key can be used to implement a screen saver function.

3. JOG CONTROL KEYS
These keys are used to control up to eight axes in a jog mode. The mode (slaved, one’s only, or two’s only) can be set through the jog menu. When the operator presses a jog key, the respective axis will begin to move. The direction that the axis moves is determined by the operator pressing either a plus or minus jog key. A plus jog key will turn the lead screw in a clockwise direction (away from the motor), a minus jog key will turn it in the counter-clockwise direction (towards the motor). By releasing the jog key the operator stops motion on that axis. Motion will also stop, if the axis reaches the limit for the direction it is moving, or if the indexer determines that the axis has stalled.
4. **SCROLL KEYS**
These keys are used to scroll items through the MENU, COMMAND, and CHANNEL windows. All of the menus, their commands, and channel variations will be detailed in another appendix.

5. **COMMAND WINDOWS**
These three windows (MENU, COMMAND, and CHANNEL) are used, in tandem with their respective scroll keys, to formulate a command to be executed by the TCS8.

6. **EXECUTE KEY**
This key is used to execute the command currently formulated in the MENU, COMMAND, and CHANNEL windows.

7. **DATA WINDOW**
Many of the TCS8’s commands require some added data, e.g. the distance to move or an axis’ encoder counts per unit. Data for these commands is entered from the numeric key pad on the lower right of the TCS8 into the DATA window. Only a valid real number can be entered into the DATA window. If the operator enters an invalid real number the character that is invalid will flash until the operator presses backspace or a valid character.

8. **STOP KEY**
The stop key, when pressed, will stop motion on all axes. The TCS8 will not lose track of the position of any axis. A move command started by the host computer and stopped by the stop key will finish normally with the position being reported. The position reported is the instantaneous position when the stop key was pressed. The final position of the axis being moved could be different than what was reported thus the host computer should read the position again after a panic stop.

9. **STATUS WINDOW**
The STATUS window reflects the result of all commands. For commands that are not instantaneous, this window displays a busy status and then when the command completes it displays a ready status. The results of all view commands are displayed in the STATUS window. The STATUS window also displays the activity over the COM interfaces. For example, when the command for viewing position is sent over the COM1 interface, the STATUS window will display “COM1 VP” and when the command is completed the window will display “COM1 vp”.

10. **NUMERIC KEY PAD**
The numeric key pad is used to enter a number into the data window. The user may backspace in the window or clear (shift-backspace) the window. Only a valid real number can be entered into the data window. If the operator enters an invalid real number the character that is invalid will flash until the operator presses backspace or a valid character.
Appendix A.2  TCS8’s Local Command Descriptions

This appendix describes the command set that can be executed from the front panel of the TCS8. Using the up and down keys under the MENU, COMMAND, and CHANNEL windows, the operator can formulate a command and then execute it by pressing the EXECUTE key. Some commands require extra information to be entered into the DATA window through the use of the numeric key pad. Each description includes a list of related commands that should be referred to to enhance the operator’s understanding of the command. Also where applicable, the default setting is given.

MENU: MOVE

COMMAND: TO ZERO

CHANNELS: ALL, ONE’S, TWO’S, X1&X2, Y1&Y2, Z1&Z2, A1&A2, X1, X2, Y1, Y2, Z1, Z2, A1, A2

DESCRIPTION: The MOVE TO ZERO command is an easy way to move some or all of the axes to the zero position. This command can also be accomplished with the MOVE ABSOLUTE command and a zero in the DATA window. Before using this command, selected axes must be initialized with the INIT DRIVE ON command. This command can be canceled by pressing the STOP key. When the STOP key is pressed, all axes will stop immediately. If an axis encounters a limit before reaching zero, the rest of its movement is aborted.

RELATED COMMANDS: MOVE ABSOLUTE, MOVE RELATIVE, INIT Drive ON

MENU: MOVE

COMMAND: ABSOLUTE

CHANNELS: X1&X2, Y1&Y2, Z1&Z2, A1&A2, X1, X2, Y1, Y2, Z1, Z2, A1, A2

DESCRIPTION: The MOVE ABSOLUTE command requires a position to be entered in the DATA window. This position and the current position of the axis is used to calculate the relative distance the axis must move. Before using this command, selected axes must be initialized with the INIT DRIVE ON command. This command can be canceled by pressing the STOP key. When the STOP key is pressed, all axes will stop immediately. If an axis encounters a limit before reaching the position entered in the DATA window, the rest of its movement is aborted.

RELATED COMMANDS: MOVE TO ZERO, MOVE RELATIVE, INIT Drive ON
MENU: MOVE

COMMAND: RELATIVE

CHANNELS:  X1&X2, Y1&Y2, Z1&Z2, A1&A2, X1, X2, Y1, Y2, Z1, Z2, A1, A2

DESCRIPTION: The MOVE RELATIVE command requires a distance to be entered in the DATA window. This position is used to calculate the relative distance the axis must move. Before using this command, selected axes must be initialized with the INIT DRIVE ON command. This command can be canceled by pressing the STOP key. When the STOP key is pressed, all axes will stop immediately. If an axis encounters a limit before moving the distance entered in the DATA window, the rest of its movement is aborted.

RELATED COMMANDS: MOVE TO ZERO, MOVE ABSOLUTE, INIT Drive ON

MENU: JOG

COMMAND: MODE

CHANNELS:  SLAVED, ONE'S, TWO'S

DESCRIPTION: The JOG MODE command sets the way the JOG keys operate. When SLAVED is the setting, both the one and two axis of the X, Y, Z, or A coordinate will move the same amount. When ONE'S is the setting, only the one axes of the X, Y, Z, or A coordinate will move. And finally, when TWO'S is the setting, only the two axes of the X, Y, Z, or A coordinate will move. The current mode is marked with an asterisk.

RELATED COMMANDS: none

DEFAULT: SLAVED
DESCRIPTION: The SET CPU command allows the user to change the counts per unit travel. The CPU for an axis is determined by multiplying the encoder resolution (counts/revolution) by the lead screws resolution (revolutions/unit of travel). A units conversion can be added here to change for example from inches to centimeters. When the CPU for an axis is changed, the position is automatically converted. This command requires a value to be entered in the DATA window.

RELATED COMMANDS: SET CPR, SET POSITION

DEFAULT: X1 20000
X2 20000
Y1 20000
Y2 20000
Z1 20000
Z2 20000
A1 40000
A2 40000

DESCRIPTION: The SET CPR command allows the user to change the encoder counts per motor revolution. The CPR for an axis is determined by dividing the encoder resolution (counts/revolution) by the lead screws resolution (revolutions/unit of travel). The encoder counts per motor revolution, that is entered in the DATA window, must be a positive integer.

RELATED COMMANDS: SET CPU

DEFAULT: X1 4000
X2 4000
Y1 4000
Y2 4000
Z1 4000
Z2 4000
A1 4000
A2 4000
MENU: SET

COMMAND: POSITION

CHANNELS: ALL, X1&X2, Y1&Y2, Z1&Z2, A1&A2, X1, X2, Y1, Y2, Z1, Z2, A1, A2

DESCRIPTION: The SET POSITION command allows the user to change the current position of an axis. The new position must be entered in the DATA window before executing the command.

RELATED COMMANDS: SET CPU

MENU: SET

COMMAND: VELOCITY

CHANNELS: ALL, X1&X2, Y1&Y2, Z1&Z2, A1&A2, X1, X2, Y1, Y2, Z1, Z2, A1, A2

DESCRIPTION: The SET VELOCITY command allows the user to change the maximum speed at which an axis will travel. The range of valid velocities is 0.002 to 50.000 revolutions per second. The default is 5 revs/sec. An axis may stall at velocities higher than the default. The new velocity must be entered in the DATA window before executing the command.

RELATED COMMANDS: SET ACCEL.

DEFAULT: X1 5.000
X2 5.000
Y1 5.000
Y2 5.000
Z1 5.000
Z2 5.000
A1 5.000
A2 5.000
MENU: SET

COMMAND: ACCEL.

CHANNELS: ALL, X1&X2, Y1&Y2, Z1&Z2, A1&A2, X1, X2, Y1, Y2, Z1, Z2, A1, A2

DESCRIPTION: The SET ACCEL. command allows the user to change the maximum acceleration for an axis. The range of valid accelerations is 0.01 to 999.99 revolutions per second per second. The default is 5 revs/sec/sec. The new acceleration must be entered in the DATA window before executing the command.

RELATED COMMANDS: SET VELOCITY

DEFAULT: X1 5.00
X2 5.00
Y1 5.00
Y2 5.00
Z1 5.00
Z2 5.00
A1 5.00
A2 5.00

MENU: SET

COMMAND: CmtrsOn

CHANNELS: ALL, ONE'S, TWO'S, X1&X2, Y1&Y2, Z1&Z2, A1&A2, X1, X2, Y1, Y2, Z1, Z2, A1, A2

DESCRIPTION: The SET CmtrsOn command allows the user to turn the motor currents on. The motor current must be on for an axis to be moved. The information in the DATA window is ignored.

RELATED COMMANDS: SET CmtrsOff
MENU: SET

COMMAND: CntsOff

CHANNELS: ALL, ONE'S, TWO'S, X1&X2, Y1&Y2, Z1&Z2, A1&A2, X1, X2, Y1, Y2, Z1, Z2, A1, A2

DESCRIPTION: The SET CntsOff command allows the user to power down motors when they will not be used for long periods of time. The information in the DATA window is ignored.

RELATED COMMANDS: SET CntsOn

---

MENU: SET

COMMAND: INITS ON

CHANNELS: ALL, ONE'S, TWO'S, X1&X2, Y1&Y2, Z1&Z2, A1&A2, X1, X2, Y1, Y2, Z1, Z2, A1, A2

DESCRIPTION: The SET INITS ON command allows the user to initialize the indexers without turning on the power to the motors. The information in the DATA window is ignored.

RELATED COMMANDS: INIT Drive ON

---

MENU: VIEW

COMMAND: Cnt/Unit

CHANNELS: X1, X2, Y1, Y2, Z1, Z2, A1, A2

DESCRIPTION: The VIEW Cnt/Unit command displays the current setting of the encoder counts per unit travel parameter for the selected axis in the STATUS window. The information in the DATA window is ignored.

RELATED COMMANDS: SET CPU
MENU: VIEW

COMMAND: Cnt/MRev

CHANNELS: X1, X2, Y1, Y2, Z1, Z2, A1, A2

DESCRIPTION: The VIEW Cnt/MRev command displays the current setting of the encoder counts per motor revolution parameter for the selected axis in the STATUS window. The information in the DATA window is ignored.

RELATED COMMANDS: SET CPR

---

MENU: VIEW

COMMAND: VELOCITY

CHANNELS: X1, X2, Y1, Y2, Z1, Z2, A1, A2

DESCRIPTION: The VIEW VELOCITY command displays the current setting of the velocity parameter for the selected axis in the STATUS window. The information in the DATA window is ignored.

RELATED COMMANDS: SET VELOCITY

---

MENU: VIEW

COMMAND: ACCEL.

CHANNELS: X1, X2, Y1, Y2, Z1, Z2, A1, A2

DESCRIPTION: The VIEW ACCEL. command displays the current setting of the acceleration parameter for the selected axis in the STATUS window. The information in the DATA window is ignored.

RELATED COMMANDS: SET ACCEL.
MENU: VIEW

COMMAND: INIT

CHANNELS: none

DESCRIPTION: The VIEW INIT command uses the STATUS window to display a one(initialized) or a zero(uninitialized) for each axis. The STATUS window has eight characters; left to right respectively reflecting the status of: X1, X2 ..., A1, A2. The information in the DATA window is ignored.

RELATED COMMANDS: SET INITS, INIT Drive ON

MENU: VIEW

COMMAND: CURRENTS

CHANNELS: none

DESCRIPTION: The VIEW CURRENTS command uses the STATUS window to display a one(current on) or a zero(current off) for each axis. The STATUS window has eight characters; left to right respectively reflecting the status of: X1, X2 ..., A1, A2. The information in the DATA window is ignored.

RELATED COMMANDS: SET CntsOn, SET CntsOff, INIT Drive ON, INIT Drive OFF

MENU: VIEW

COMMAND: Plus LMT

CHANNELS: none

DESCRIPTION: The VIEW Plus LMT command uses the STATUS window to display a one(on limit) or a zero(not on limit) for each axis. The STATUS window has eight characters; left to right respectively reflecting the status of: X1, X2 ..., A1, A2. The information in the DATA window is ignored.

RELATED COMMANDS: none
MENU: VIEW

COMMAND: Minus LMT

CHANNELS: none

DESCRIPTION: The VIEW Minus LMT command uses the STATUS window to display a one(on limit) or a zero(not on limit) for each axis. The STATUS window has eight characters; left to right respectively reflecting the status of: X1, X2 ..., A1, A2. The information in the DATA window is ignored.

RELATED COMMANDS: none

MENU: VIEW

COMMAND: HOME

CHANNELS: none

DESCRIPTION: The VIEW HOME command uses the STATUS window to display a one(on limit) or a zero(not on limit) for each axis. The STATUS window has eight characters; left to right respectively reflecting the status of: X1, X2 ..., A1, A2. The information in the DATA window is ignored.

RELATED COMMANDS: none

MENU: VIEW

COMMAND: STALL

CHANNELS: none

DESCRIPTION: The VIEW STALL command uses the STATUS window to display a one(stalled) or a zero(not stalled) for each axis. The STATUS window has eight characters; left to right respectively reflecting the status of: X1, X2 ..., A1, A2. The information in the DATA window is ignored.

RELATED COMMANDS: none
**MENU:** INIT

**COMMAND:** DEFAULT

**CHANNELS:** none

**DESCRIPTION:** The INIT DEFAULT command restores the initial factory defaults (CPU, CPR, VELOCITY, ACCELERATION, BAUD RATE, BITS/CHAR, PARITY, STOP BITS, HANDSHAKE) of the TCS8. After executing this command, execute the command INIT Drive ON to initialize the indexers. The information in the DATA window is ignored.

**RELATED COMMANDS:** SET CPU, SET CPR, SET VELOCITY, SET ACCEL.

---

**MENU:** INIT

**COMMAND:** Drive ON

**CHANNELS:** ALL, ONE'S, TWO'S, X1&X2, Y1&Y2, Z1&Z2, A1&A2, X1, X2, Y1, Y2, Z1, Z2, A1, A2

**DESCRIPTION:** The INIT Drive ON command initializes the selected axes for movement. After executing this command the currents are on to the motors. The information in the DATA window is ignored.

**RELATED COMMANDS:** SET CPU, SET CPR, SET VELOCITY, SET ACCEL., SET CnrtsOn, SET CnrtsOff, INIT DEFAULT

---

**MENU:** INIT

**COMMAND:** Drive OFF

**CHANNELS:** ALL, ONE'S, TWO'S, X1&X2, Y1&Y2, Z1&Z2, A1&A2, X1, X2, Y1, Y2, Z1, Z2, A1, A2

**DESCRIPTION:** The INIT Drive OFF command is an alias for SET CnrtsOff.

**RELATED COMMANDS:** SET CnrtsOff
MENU: COM1/COM2

COMMAND: BaudRate

CHANNELS: 19.2K, 9600, 4800, 2400, 1200, 300, 110

DESCRIPTION: The COM1/COM2 BaudRate command set the baud rate for the selected communication channel. The information in the DATA window is ignored. The current baud rate is marked with an asterisk.

RELATED COMMANDS: none

DEFAULT: 9600

———

MENU: COM1/COM2

COMMAND: Bit/Char

CHANNELS: SEVEN, EIGHT

DESCRIPTION: The COM1/COM2 Bit/Char command set the bits per character for the selected communication channel. The information in the DATA window is ignored. The current number of bits per character is marked with an asterisk.

RELATED COMMANDS: none

DEFAULT: EIGHT

———

MENU: COM1/COM2

COMMAND: Parity

CHANNELS: NONE, EVEN, ODD

DESCRIPTION: The COM1/COM2 Parity command set the parity for the selected communication channel. The information in the DATA window is ignored. The current parity is marked with an asterisk.

RELATED COMMANDS: none

DEFAULT: EVEN
MENU: COM1/COM2

COMMAND: StopBits

CHANNELS: 1, 1.5, 2

DESCRIPTION: The COM1/COM2 StopBits command set the stop bits for the selected communication channel. The information in the DATA window is ignored. The current number of stop bits is marked with an asterisk.

RELATED COMMANDS: none

DEFAULT: 1

---

MENU: COM1/COM2

COMMAND: HandShak

CHANNELS: NO, YES

DESCRIPTION: The COM1/COM2 HandShak command set the handshake for the selected communication channel. The information in the DATA window is ignored. An asterisk marks whether there is handshaking or not.

RELATED COMMANDS: none

DEFAULT: YES
Appendix A.3 TCS8’s Serial Interface Command Descriptions

This appendix describes the command set that can be executed through the serial interfaces of the TCS8. Each description includes a code section that outlines the characters that must be sent to execute the command. The vertical bar in this section is used as a separator and is not sent as part of the command code. The symbol “CRLF” stands for the two characters carriage return and line feed. Also where applicable, the default setting is given.

**COMMAND:** CHANGE SERIAL CONFIGURATION

**CODE:** CS COM;CATEGORY;ATTRIBUTE;

**PARAMETERS:**

<table>
<thead>
<tr>
<th>COM:</th>
<th>1/COM1</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>2/COM2</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>CATEGORY:</th>
<th>0/BAUDRATE</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>ATTRIBUTE:</th>
<th>0/19.2K</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1/9600</td>
</tr>
<tr>
<td></td>
<td>2/4800</td>
</tr>
<tr>
<td></td>
<td>3/2400</td>
</tr>
<tr>
<td></td>
<td>4/1200</td>
</tr>
<tr>
<td></td>
<td>5/300</td>
</tr>
<tr>
<td></td>
<td>6/110</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>CATEGORY:</th>
<th>1/(BITS PER CHARACTER)</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>ATTRIBUTE:</th>
<th>0/SEVEN</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>CATEGORY:</th>
<th>2/(PARITY)</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>ATTRIBUTE:</th>
<th>0/NONE</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>CATEGORY:</th>
<th>3/(STOP BITS)</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>ATTRIBUTE:</th>
<th>0/ONE</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>CATEGORY:</th>
<th>4/(HANDSHAKE)</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>ATTRIBUTE:</th>
<th>0/NO</th>
</tr>
</thead>
</table>

|         | 1/YES |

**DESCRIPTION:** This command must be executed with extreme caution and thought. If the user changes an attribute of the same COM port that he is sending the command, he must change to that attribute on the host computer before sending the next command. The best way to change the serial configuration of a COM port is to utilize the front panel commands.

**DEFAULT:** 9600 baud, EIGHT bits/char, EVEN parity, ONE stop bit, handshaking YES

**EXAMPLE:** To change the baudrate of COM1 to 2400 the user must send CS1;0;3;
COMMAND: MOVE TO ABSOLUTE POSITION AND REPORT FINAL POSITION

CODE: MA CHANNEL:POSITION,CHANNEL:POSITION,...\CRLF

PARAMETERS: CHANNEL: 0/ALL CHANNELS
1/X1
2/X2
3/Y1
4/Y2
5/Z1
6/Z2
7/A1
8/A2

POSITION: Real number free format

DESCRIPTION: This command moves selected channels to absolute positions.

EXAMPLES:
To move all channels to zero the user may send MA0:0,CRLF or MA12345678:0,CRLF
To move channel X1 to zero the user must send MA 1:0,CRLF
To move channels X1 and X2 to zero the user may send MA12:0,CRLF or MA1:0,2:0,CRLF or MA1:0,CRLF and MA2:0,CRLF

COMMAND: MOVE TO RELATIVE DISTANCE AND REPORT FINAL POSITION

CODE: MR CHANNEL:DISTANCE,CHANNEL:DISTANCE,...\CRLF

PARAMETERS: CHANNEL: 0/ALL CHANNELS
1/X1
2/X2
3/Y1
4/Y2
5/Z1
6/Z2
7/A1
8/A2

POSITION: Real number free format

DESCRIPTION: This command moves selected channels relative distances.

EXAMPLES:
To move all channels one unit the user may send MR0:1,CRLF or MR12345678:1,CRLF
To move channel X1 one unit the user must send MR1:1,CRLF
To move channels X1 and X2 one unit the user may send MR12:1,CRLF or MR1:1,2:1,CRLF or MR1:1,CRLF and MR2:1,CRLF
COMMAND: SET ACCELERATION

CODE: SA CHANNEL:ACCELERATION,CHANNEL:ACCELERATION,...|CRLF

PARAMETERS: CHANNEL: 0/ALL CHANNELS
1/X1
2/X2
3/Y1
4/Y2
5/Z1
6/Z2
7/A1
8/A2

ACCELERATION: Real number free format between 0.01 and 99.99 inclusive.

DESCRIPTION: This command sets the acceleration for selected channels.

DEFAULT: All channels 5.00 revolutions/second/second

EXAMPLES: All channels 5.00 revolutions/second/second

To set the acceleration for all channels to 4.00 revolutions/second/second the user may send
SA0:4.00,CRLF or SA12345678:4.00,CRLF
To set the acceleration for channel X1 to 4.00 revolutions/second/second the user must send
SA1:4.00,CRLF
To set the acceleration for channels X1 and X2 to 4.00 revolutions/second/second the user may send
SA12:4.00,CRLF or SA1:4.00,2:4.00,CRLF or SA1:4.00,CRLF and SA2:4.00,CRLF

COMMAND: VIEW ACCELERATION

CODE: VA CHANNEL:CHANNEL...|CRLF

PARAMETERS: CHANNEL: 0/ALL CHANNELS
1/X1
2/X2
3/Y1
4/Y2
5/Z1
6/Z2
7/A1
8/A2

DESCRIPTION: This command views the acceleration for selected channels. The TCS8
transmits each of the accelerations requested back to the host computer separated by carriage return
line feeds.

EXAMPLES: All channels 5.00 revolutions/second/second

To view the acceleration for all channels the user may send VA0CRLF or VA12345678CRLF
To view the acceleration for channel X1 the user must send VA1CRLF
To view the acceleration for channels X1 and X2 the user may send VA12CRLF or VA1CRLF and VA2CRLF
COMMAND: SET VELOCITY

CODE: SV CHANNEL:VELOCITY,CHANNEL:VELOCITY,...!CRLF

PARAMETERS: CHANNEL: 0/ALL CHANNELS
1/X1
2/X2
3/Y1
4/Y2
5/Z1
6/Z2
7/A1
8/A2

VELOCITY: Real number free format between 0.001 and 50.000 inclusive.

DESCRIPTION: This command sets the velocity for selected channels.

DEFAULT: All channels 5.000 revolutions/second

EXAMPLES:
To set the velocity for all channels to 4.00 revolutions/second the user may send SV0:4.00,CRLF or SV12345678:4.00,CRLF
To set the velocity for channel X1 to 4.00 revolutions/second the user must send SV1:4.00,CRLF
To set the velocity for channels X1 and X2 to 4.00 revolutions/second the user may send SV12:4.00,CRLF or SV1:4.00,2:4.00,CRLF or SV1:4.00,CRLF and SV2:4.00,CRLF

COMMAND: VIEW VELOCITY

CODE: VV CHANNEL|CHANNEL...!CRLF

PARAMETERS: CHANNEL: 0/ALL CHANNELS
1/X1
2/X2
3/Y1
4/Y2
5/Z1
6/Z2
7/A1
8/A2

DESCRIPTION: This command views the velocity for selected channels. The TCS8 transmits each of the velocities requested back to the host computer separated by carriage return line feeds.

EXAMPLES:
To view the velocity for all channels the user may send VV0CRLF or VV12345678CRLF
To view the velocity for channel X1 the user must send VV1CRLF
To view the velocity for channels X1 and X2 the user may send VV12CRLF or VV1CRLF and VV2CRLF
COMMAND: SET ENCODER COUNTS PER UNIT TRAVEL

CODE: SU CHANNEL:CPU, CHANNEL:CPU,...CRLF

PARAMETERS: CHANNEL: 0/ALL CHANNELS
1/X1
2/X2
3/Y1
4/Y2
5/Z1
6/Z2
7/A1
8/A2

CPU: Non-zero real number free format.

DESCRIPTION: This command sets the encoder counts per unit travel for selected channels.

DEFAULT: X1,2,Y1,Y2,Z1,Z2 20000 counts/inch and A1,A2 40000 counts/inch

EXAMPLES:
To set the encoder counts per unit travel for all channels to 5000 the user may send
SU0:5000,CRLF or SU12345678:5000,CRLF
To set the encoder counts per unit travel for channel X1 to 5000 the user must send
SU1:5000,CRLF
To set the encoder counts per unit travel for channels X1 and X2 to 5000 the user may send
SU12:5000,CRLF or SU1:5000,2:5000,CRLF or SU1:5000,CRLF and SU2:5000,CRLF

COMMAND: VIEW ENCODER COUNTS PER UNIT TRAVEL

CODE: VU CHANNEL|CHANNEL...CRLF

PARAMETERS: CHANNEL: 0/ALL CHANNELS
1/X1
2/X2
3/Y1
4/Y2
5/Z1
6/Z2
7/A1
8/A2

DESCRIPTION: This command views the encoder counts per unit travel for selected channels.
The TCS8 transmits each of the encoder counts per unit travel requested back to the host computer
separated by carriage return line feeds.

EXAMPLES:
To view the encoder counts per unit travel for all channels the user may send VU0CRLF or
VU12345678CRLF
To view the encoder counts per unit travel for channel X1 the user must send VU1CRLF
To view the encoder counts per unit travel for channels X1 and X2 the user may send VU1CRLF
or VU1CRLF and VU2CRLF
COMMAND: SET ENCODER COUNTS PER MOTOR REVOLUTION

CODE: SR CHANNEL:CPR,CHANNEL:CPR,...!CRLF

PARAMETERS: CHANNEL: 0/ALL CHANNELS
1/X1
2/X2
3/Y1
4/Y2
5/Z1
6/Z2
7/A1
8/A2

CPU: Non-zero integer free format.

DESCRIPTION: This command sets the encoder counts per motor revolution for selected channels.

DEFAULT: X1,2,Y1,Y2,Z1,Z2 and A1,A2 4000 counts/inch

EXAMPLES:
To set the encoder counts per motor revolution for all channels to 500 the user may send
SR0:500,CRLF or SR12345678:500,CRLF
To set the encoder counts per motor revolution for channel X1 to 500 the user must send
SR1:500,CRLF
To set the encoder counts per motor revolution for channels X1 and X2 to 500 the user may send
SR12:500,CRLF or SR1:500,2:500,CRLF or SR1:500,CRLF and SR2:500,CRLF

COMMAND: VIEW ENCODER COUNTS PER MOTOR REVOLUTION

CODE: VR CHANNEL:CHANNEL...!CRLF

PARAMETERS: CHANNEL: 0/ALL CHANNELS
1/X1
2/X2
3/Y1
4/Y2
5/Z1
6/Z2
7/A1
8/A2

DESCRIPTION: This command views the encoder counts per motor revolution for selected channels. The TCS8 transmits each of the encoder counts per motor revolution requested back to the host computer separated by carriage return line feeds.

EXAMPLES:
To view the encoder counts per motor revolution for all channels the user may send VR0CRLF or VR12345678CRLF
To view the encoder counts per motor revolution for channel X1 the user must send VR1CRLF
To view the encoder counts per motor revolution for channels X1 and X2 the user may send
VR12CRLF or VR1CRLF and VR2CRLF
COMMAND: SET POSITION

CODE: SP CHANNEL:POSITION,CHANNEL:POSITION,...|CRLF

PARAMETERS: CHANNEL: 0/ALL CHANNELS
1/X1
2/X2
3/Y1
4/Y2
5/Z1
6/Z2
7/A1
8/A2

POSITION: real number.

DESCRIPTION: This command sets the position for selected channels.

EXAMPLES:
To set the position for all channels to 1.5 the user may send SP0:1.5,CRLF or SP12345678:1.5,CRLF
To set the position for channel X1 to 1.5 the user must send SPI:1.5,CRLF
To set the position for channels X1 and X2 to 1.5 the user may send SP12:I.5,CRLF or SPI:1.5,2:1.5,CRLF or SPI:1.5,CRLF and SP2:1.5,CRLF

COMMAND: VIEW POSITION

CODE: VP CHANNEL|CHANNEL...|CRLF

PARAMETERS: CHANNEL: 0/ALL CHANNELS
1/X1
2/X2
3/Y1
4/Y2
5/Z1
6/Z2
7/A1
8/A2

DESCRIPTION: This command views the position for selected channels. The TCS8 transmits each of the positions requested back to the host computer separated by carriage return line feeds.

EXAMPLES:
To view the position for all channels the user may send VP0CRLF or VP12345678CRLF
To view the position for channel X1 the user must send VP1CRLF
To view the position for channels X1 and X2 the user may send VP12CRLF or VP1CRLF and VP2CRLF
**COMMAND:** SET CURRENT TO MOTOR WINDINGS

**CODE:** SC CHANNEL:ON/OFF,CHANNEL:ON/OFF,...\r\n
**PARAMETERS:**

Channel: 0/ALL CHANNELS
1/X1
2/X2
3/Y1
4/Y2
5/Z1
6/Z2
7/A1
8/A2

On/Off: 1/ON
0/OFF

**DESCRIPTION:** This command sets the current to the motor windings for selected channels on or off.

**EXAMPLES:**
To set the current to the motor windings for all channels on the user may send SC0:1,\r\n
**COMMAND:** VIEW CURRENT TO MOTOR WINDINGS

**CODE:** VC CHANNEL|CHANNEL...\r\n
**PARAMETERS:**

Channel: 0/ALL CHANNELS
1/X1
2/X2
3/Y1
4/Y2
5/Z1
6/Z2
7/A1
8/A2

**DESCRIPTION:** This command views the current to the motor windings for selected channels. The TCS8 transmits each response of on/off (1/0) back to the host computer separated by carriage return line feeds.

**EXAMPLES:**
To view the current to the motor windings for all channels the user may send VC0\r\n
To view the current to the motor windings for channel X1 the user must send VC1\r\n
To view the current to the motor windings for channels X1 and X2 the user may send VC1\r\n
46
**COMMAND:** SET INITIALIZATION OF INDEXER/DRIVERS

**CODE:** SI CHANNEL|CHANNEL...|CRLF

**PARAMETERS:**

- **CHANNEL:**
  - 0/ALL CHANNELS
  - 1/X1
  - 2/X2
  - 3/Y1
  - 4/Y2
  - 5/Z1
  - 6/Z2
  - 7/A1
  - 8/A2

**DESCRIPTION:** This command sends the current value of the acceleration, velocity, and the encoder counts per motor revolution to the indexer/driver for the selected channels. This command must be sent before any move commands may be sent.

**EXAMPLES:**
- To initialize all channels the user may send SI0CRLF or SI12345678CRLF
- To initialize channel X1 the user must send SI1CRLF
- To initialize channels X1 and X2 the user may send SI12CRLF or SI1CRLF and SI2CRLF

**COMMAND:** VIEW INITIALIZATION OF INDEXER/DRIVERS

**CODE:** VI CHANNEL|CHANNEL...|CRLF

**PARAMETERS:**

- **CHANNEL:**
  - 0/ALL CHANNELS
  - 1/X1
  - 2/X2
  - 3/Y1
  - 4/Y2
  - 5/Z1
  - 6/Z2
  - 7/A1
  - 8/A2

**DESCRIPTION:** This command returns "1" if the indexer/driver has been initialized since the TCS8 was turned on and "0" if it has not. The TCS8 transmits each of the responses back to the host computer separated by carriage return line feeds.

**EXAMPLES:**
- To check the initialization of all channels the user may send VI0CRLF or VI12345678CRLF
- To check the initialization of channel X1 the user must send VI1CRLF
- To check the initialization of channels X1 and X2 the user may send VI12CRLF or VI1CRLF and VI2CRLF
Appendix A.4

Traverse Control System Cables
MOTOR DRIVE SYSTEM

ENCODER SIGNALS

HOST COMPUTER

TCS8

SERIAL INTERFACES

MOTOR DRIVE SYSTEM

DWG# LFA-TAA-002
DWG# LFA-TAA-003
DWG# LFA-TAA-004
DWG# LFA-TAA-005
DWG# LFA-TAA-006
DWG# LFA-TAA-007

DWG# LFA-TAA-008

COMPLERE INC.

NASA LaRC LASER FLUORESCENCE ANEMOMETER
LFA TRAVERSE SYSTEM
TRAVERSE SYSTEM CABLE ROUTING DIAGRAM

DRAWING DATE
JULY 8, 1991

FILE NAME
LANGLEY TRAVERSE DIAGRAM

DESIGN ENGINEER
TODD A. AMBUR

DRAWING NUMBER
LFA-TAA-001
COMPLERE INC.
MOTOR DRIVE SYSTEM

<table>
<thead>
<tr>
<th>SOCKET</th>
<th>SIGNAL</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>NC</td>
</tr>
<tr>
<td>2</td>
<td>A+</td>
</tr>
<tr>
<td>3</td>
<td>NC</td>
</tr>
<tr>
<td>4</td>
<td>A-</td>
</tr>
<tr>
<td>5</td>
<td>NC</td>
</tr>
<tr>
<td>6</td>
<td>SHIELD</td>
</tr>
<tr>
<td>7</td>
<td>NC</td>
</tr>
<tr>
<td>8</td>
<td>B+</td>
</tr>
<tr>
<td>9</td>
<td>NC</td>
</tr>
<tr>
<td>10</td>
<td>B-</td>
</tr>
<tr>
<td>11</td>
<td>NC</td>
</tr>
<tr>
<td>12</td>
<td>NC</td>
</tr>
<tr>
<td>13</td>
<td>NC</td>
</tr>
<tr>
<td>14</td>
<td>NC</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>PIN</th>
<th>SIGNAL</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>NC</td>
</tr>
<tr>
<td>2</td>
<td>A+</td>
</tr>
<tr>
<td>3</td>
<td>NC</td>
</tr>
<tr>
<td>4</td>
<td>A-</td>
</tr>
<tr>
<td>5</td>
<td>NC</td>
</tr>
<tr>
<td>6</td>
<td>SHIELD</td>
</tr>
<tr>
<td>7</td>
<td>NC</td>
</tr>
<tr>
<td>8</td>
<td>B+</td>
</tr>
<tr>
<td>9</td>
<td>NC</td>
</tr>
<tr>
<td>10</td>
<td>B-</td>
</tr>
<tr>
<td>11</td>
<td>NC</td>
</tr>
<tr>
<td>12</td>
<td>NC</td>
</tr>
<tr>
<td>13</td>
<td>NC</td>
</tr>
<tr>
<td>14</td>
<td>NC</td>
</tr>
</tbody>
</table>

AMPHENOL CONNECTOR 206043-1
AMPHENOL SOCKETS 66360-2

AMPHENOL CONNECTOR 206044-1
AMPHENOL CABLE CLAMP 206070-1
AMPHENOL PINS 66361-2

BELDEN CABLE 9418

AMPHENOL CONNECTOR 206044-1
AMPHENOL CABLE CLAMP 206070-1
AMPHENOL PINS 66361-2

AMPHENOL CONNECTOR 206043-3
AMPHENOL CABLE CLAMP 66360-2
AMPHENOL SOCKETS 66360-2

SIGNAL DESCRIPTION
A+ Motor Winding
A- Motor Winding
B+ Motor Winding
B- Motor Winding
SHIELD Motor Case Ground
NC No Connection

COMPLERE INC.
NASA LaRC LASER FLUORESCENCE ANEMOMETER
LFA TRAVERSE SYSTEM
MOTOR DRIVE SYSTEM TO COMPUMOTOR STEPPER MOTOR

DRAWING DATE: JULY 3, 1991
FILE NAME: LANGLEY MOTOR
DESIGN ENGINEER: TODD A. AMBUR
DRAWING NUMBER: LFA-TAA-002
<table>
<thead>
<tr>
<th>PIN</th>
<th>SIGNAL</th>
<th>SOCKET SIGNAL</th>
<th>CONNECTOR</th>
<th>AMPLONE PINS</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>A+</td>
<td>A+</td>
<td>AMPHENOL CONNECTOR 206705-1</td>
<td>66103-2</td>
<td>Quadrature Encoder Signal</td>
</tr>
<tr>
<td>2</td>
<td>A-</td>
<td>A-</td>
<td>AMPHENOL CONNECTOR 206708-1</td>
<td>66105-2</td>
<td>Logical Complement of A+</td>
</tr>
<tr>
<td>3</td>
<td>B+</td>
<td>B+</td>
<td>AMPHENOL CABLE CLAMP 206966-1</td>
<td>66105-2</td>
<td>Quadrature Encoder Signal</td>
</tr>
<tr>
<td>4</td>
<td>B-</td>
<td>B-</td>
<td>AMPHENOL SOCKETS 66106-1</td>
<td>66105-2</td>
<td>Logical Complement of B+</td>
</tr>
<tr>
<td>5</td>
<td>SHIELD</td>
<td>SHIELD</td>
<td>AMPHENOL CABLE CLAMP 206966-1</td>
<td>66105-2</td>
<td>Case Ground</td>
</tr>
<tr>
<td>6</td>
<td>Z+</td>
<td>Z+</td>
<td>AMPHENOL CONNECTOR 206708-1</td>
<td>66105-2</td>
<td>Logical Complement of Z+</td>
</tr>
<tr>
<td>7</td>
<td>Z-</td>
<td>Z-</td>
<td>AMPHENOL CONNECTOR 206705-2</td>
<td>66103-2</td>
<td>Case Ground</td>
</tr>
<tr>
<td>8</td>
<td>+5VDC</td>
<td>+5VDC</td>
<td>Belden Cable 9504</td>
<td></td>
<td>Logical Ground</td>
</tr>
<tr>
<td>9</td>
<td>GROUND</td>
<td>GROUND</td>
<td>AMPHENOL CONNECTOR 206705-1</td>
<td>66103-2</td>
<td>No Connection</td>
</tr>
</tbody>
</table>

**DYNAMICS RESEARCH ENCODER**

<table>
<thead>
<tr>
<th>PIN</th>
<th>SIGNAL</th>
<th>SOCKET SIGNAL</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>A+</td>
<td>A+</td>
</tr>
<tr>
<td>2</td>
<td>A-</td>
<td>A-</td>
</tr>
<tr>
<td>3</td>
<td>B+</td>
<td>B+</td>
</tr>
<tr>
<td>4</td>
<td>B-</td>
<td>B-</td>
</tr>
<tr>
<td>5</td>
<td>SHIELD</td>
<td>SHIELD</td>
</tr>
<tr>
<td>6</td>
<td>Z+</td>
<td>Z+</td>
</tr>
<tr>
<td>7</td>
<td>Z-</td>
<td>Z-</td>
</tr>
<tr>
<td>8</td>
<td>+5VDC</td>
<td>+5VDC</td>
</tr>
<tr>
<td>9</td>
<td>GROUND</td>
<td>GROUND</td>
</tr>
</tbody>
</table>
## COMPLERE INC.

### MOTOR DRIVE SYSTEM

<table>
<thead>
<tr>
<th>SOCKET</th>
<th>SIGNAL</th>
<th>PIN</th>
<th>SIGNAL</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>C1 +5V</td>
<td>1</td>
<td>C1 +5V</td>
</tr>
<tr>
<td>2</td>
<td>C1 B+</td>
<td>2</td>
<td>C1 B+</td>
</tr>
<tr>
<td>3</td>
<td>C1 SHIELD</td>
<td>3</td>
<td>C1 SHIELD</td>
</tr>
<tr>
<td>4</td>
<td>C2 A+</td>
<td>4</td>
<td>C2 A+</td>
</tr>
<tr>
<td>5</td>
<td>C2 GND</td>
<td>5</td>
<td>C2 GND</td>
</tr>
<tr>
<td>6</td>
<td>C3 +5V</td>
<td>6</td>
<td>C3 +5V</td>
</tr>
<tr>
<td>7</td>
<td>C3 B+</td>
<td>7</td>
<td>C3 B+</td>
</tr>
<tr>
<td>8</td>
<td>C3 SHIELD</td>
<td>8</td>
<td>C3 SHIELD</td>
</tr>
<tr>
<td>9</td>
<td>C4 A+</td>
<td>9</td>
<td>C4 A+</td>
</tr>
<tr>
<td>10</td>
<td>C4 GND</td>
<td>10</td>
<td>C4 GND</td>
</tr>
<tr>
<td>11</td>
<td>NC</td>
<td>11</td>
<td>NC</td>
</tr>
<tr>
<td>12</td>
<td>NC</td>
<td>12</td>
<td>NC</td>
</tr>
<tr>
<td>13</td>
<td>NC</td>
<td>13</td>
<td>NC</td>
</tr>
<tr>
<td>14</td>
<td>C1 A+</td>
<td>14</td>
<td>C1 A+</td>
</tr>
<tr>
<td>15</td>
<td>C1 GND</td>
<td>15</td>
<td>C1 GND</td>
</tr>
<tr>
<td>16</td>
<td>C2 +5V</td>
<td>16</td>
<td>C2 +5V</td>
</tr>
<tr>
<td>17</td>
<td>C2 B+</td>
<td>17</td>
<td>C2 B+</td>
</tr>
<tr>
<td>18</td>
<td>C2 SHIELD</td>
<td>18</td>
<td>C2 SHIELD</td>
</tr>
<tr>
<td>19</td>
<td>C3 A+</td>
<td>19</td>
<td>C3 A+</td>
</tr>
<tr>
<td>20</td>
<td>C3 GND</td>
<td>20</td>
<td>C3 GND</td>
</tr>
<tr>
<td>21</td>
<td>C4 +5V</td>
<td>21</td>
<td>C4 +5V</td>
</tr>
<tr>
<td>22</td>
<td>C4 B+</td>
<td>22</td>
<td>C4 B+</td>
</tr>
<tr>
<td>23</td>
<td>C4 SHIELD</td>
<td>23</td>
<td>C4 SHIELD</td>
</tr>
<tr>
<td>24</td>
<td>NC</td>
<td>24</td>
<td>NC</td>
</tr>
<tr>
<td>25</td>
<td>NC</td>
<td>25</td>
<td>NC</td>
</tr>
</tbody>
</table>

**DB-25 FEMALE**

**6 FT. DB-25 STRAIGHT THROUGH EXTENSION CABLE**

**DB-25 MALE**

### TCS8

<table>
<thead>
<tr>
<th>PIN</th>
<th>SIGNAL</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>NC</td>
</tr>
<tr>
<td>2</td>
<td>C1 B+</td>
</tr>
<tr>
<td>3</td>
<td>C1 SHIELD</td>
</tr>
<tr>
<td>4</td>
<td>C2 A+</td>
</tr>
<tr>
<td>5</td>
<td>C2 GND</td>
</tr>
<tr>
<td>6</td>
<td>C3 +5V</td>
</tr>
<tr>
<td>7</td>
<td>C3 B+</td>
</tr>
<tr>
<td>8</td>
<td>C3 SHIELD</td>
</tr>
<tr>
<td>9</td>
<td>C4 A+</td>
</tr>
<tr>
<td>10</td>
<td>C4 GND</td>
</tr>
<tr>
<td>11</td>
<td>NC</td>
</tr>
<tr>
<td>12</td>
<td>NC</td>
</tr>
<tr>
<td>13</td>
<td>NC</td>
</tr>
<tr>
<td>14</td>
<td>C1 A+</td>
</tr>
<tr>
<td>15</td>
<td>C1 GND</td>
</tr>
<tr>
<td>16</td>
<td>C2 +5V</td>
</tr>
<tr>
<td>17</td>
<td>C2 B+</td>
</tr>
<tr>
<td>18</td>
<td>C2 SHIELD</td>
</tr>
<tr>
<td>19</td>
<td>C3 A+</td>
</tr>
<tr>
<td>20</td>
<td>C3 GND</td>
</tr>
<tr>
<td>21</td>
<td>C4 +5V</td>
</tr>
<tr>
<td>22</td>
<td>C4 B+</td>
</tr>
<tr>
<td>23</td>
<td>C4 SHIELD</td>
</tr>
<tr>
<td>24</td>
<td>NC</td>
</tr>
<tr>
<td>25</td>
<td>NC</td>
</tr>
</tbody>
</table>

**DB-25 FEMALE**

**DB-25 MALE**

### COMPLERE INC.

- NASA LaRC LASER FLUORESCENCE ANEMOMETER
- LFA TRAVERSE SYSTEM
- MDS TO TCS8 ENCODER SIGNALS

**DRAWING DATE**

**FILE NAME**

**DESIGN ENGINEER**

**DRAWING NUMBER**

**JULY 3, 1991**

**ENCODER SIGNALS**

**TODD A. AMBUR**

**LFA-TAA-005**
Appendix B  Laser Velocimeter Data Acquisition System

The LVDAS acquires simultaneous digital data, analog data, and time information data. The data are sampled, multiplexed, buffered, and then transferred to the facility’s host computer for further data reduction, analysis, and presentation.

Four 16 bit word parallel input ports are provided to accept the digital output of LV counter processors and/or other instrumentation.

New applications in laser velocimetry have brought about the need for a more advanced laser velocimeter data acquisition system. These new applications require high data rates that are not hindered by on-line time dependent data sorting and real time graphic data presentation. The new Laser Velocimeter Data Acquisition System (LVDAS) was designed specifically to meet these advanced requirements.

High data acquisition rates are achieved by providing a separate latched input for each laser velocimeter digital input and a separate converter for each laser fluorescence analog input. The system will allow for a data acquisition rates of approximately 100,000 samples per second simultaneously on each of the laser velocimeter and laser fluorescence inputs.

A 32 bit time of day (TOD) 10MHz counter is used to tag arrival times to acquired digital LDV data as they become available on each of four digital inputs. When a data valid sync pulse is sensed for a particular channel, the LVDAS latches the current TOD into a 32 bit time of arrival register (TOA). A separate TOA register is available for each digital input so that particle arrival times of measured velocity information U, V, W can be monitored for coincidence. The latched times of arrivals have a resolution of 100 ns and maximum time of over 7 minutes.

All of the acquired digital velocity data with corresponding time of arrival data can be processed and stored. However, if coincident data is required, then the arrival time of the various channels can be conditionally accepted if they all occur within a finite window of time. These coincident events can then be assigned interarrival times which represent elapsed time since the previous event.

The coincidence control logic allows for 3 channel coincidence. The coincidence time is adjustable from 0.1 μs to 1 s. In addition to the laser velocimeter inputs, three additional data words are generated internally. They are the inter arrival time, the coincidence time, and status words. The inter arrival and coincidence time is provided by a clock whose resolution and maximum time is 100 ns and 500 seconds respectively. The status word contains information about coincidence and the order in which the laser velocimeter data arrived.

During data acquisition, it is important that the user obtain some visual feedback about the data being acquired. This is necessary so that the user can make informed decisions about both the quality and quantity of data received. The user is either reassured about the quality of the data or can make alterations and improvements in technique while on line. To help achieve this, the instantaneous velocities are used to generate real time histograms from which probability density distributions are determined for all velocity components.

Additionally, the laser velocimeter data acquisition system has the capability of reducing the raw laser velocimeter data. Each laser velocimeter output contains the information required to
calculate the instantaneous velocities. From the instantaneous velocity determinations, the average velocities, turbulence levels, and the turbulence cross correlations are all be calculated.

The coincidence control logic will allow for up to 4 channel coincidence. The coincidence time can be adjustable to any resolution or duration within the capability of the time of arrival registers. When coincident criteria are met, the analog inputs can be sampled and converted to provide concurrent data with the digital data. A single time of arrival is latched for all four of the analog to digital inputs since they are all sampled and converted simultaneously. A final time of arrival is latched for external events. These might be derived from such sources as oscillating models or model surfaces, rotating helicopter blades, rotating engine fans, or flow sensors.

All digital Macrodyne data, optional digital data, analog to digital data, and time of arrival data can be sent by the LVDAS to other computers via two serial and two parallel input/output ports. One parallel port will be used for the HP series 9000 model 330 computer while the other can be used by the facility host computer. The serial ports can be used by PC type computers such as IBM's or MACs. Software has been developed for on-line data acquisition and display. A program listing is enclosed.
PROGRAM DESCRIPTION:

This program provides the capability to acquire simultaneous Laser Doppler Velocimeter (LDV), Laser Fluorescence Anemometer (LFA), and Analog Voltage Data at user selectable traverse controlled probe volume positions within the tunnel flow. The LVDAS (Laser Velocimeter Data Acquisition System) is used to sample the LDV, LFA, and Analog Voltage data simultaneously with a coincidence criterion being applied to LDV incoming data. The LVDAS also generates interarrival times and coincidence time.

The measured LDV data provides the necessary frequency information from which three components of flow velocities can be determined. These velocities are measured directly in "LASER" coordinates. Coordinate system transformations are applied to these measured velocities to obtain velocities in "TUNNEL" and "MODEL" coordinates.

The TCS8 (Traverse Control System) is used to precisely move the LDV probe volume within the tunnel and about the model. The TCS8 provides three axes plus one auxiliary axis of traverse capability for both the transmitting and receiving side optical packages. The Tx and Rx side traverses can be moved independently to achieve laser alignment or they can be moved together to maintain laser alignment.

The TCS8 will give the traverse positions in TCS8 coordinates where one inch of commanded movement will yield one inch of movement on the traverse slides. However, this will not yield one inch of movement on the probe volume crossover point within the water filled tunnel because of the differences of refraction in air, glass, and water. Therefore, coordinate system transformations are applied to TCS8 positions to obtain positions in "TUNNEL" and "MODEL" coordinates.

During data acquisition, real time histograms will be displayed of the LDV and analog data. After the data has been acquired, the averages, standard deviations, and shear stresses will be calculated and displayed in profile plots where the data is plotted versus traverse position. The reduced data is also sent to the printer in tabular form. The reduced data as well as the raw data are stored along with the tunnel conditions on the hard disc for archival purposes and also to allow for further data reduction, data plotting, or data transfer to other computers.

PROGRAM OPERATION:

The following power up sequences should be completed before this program is run:

1. Turn on the "Motor Drive System" boxes.
2. Turn on the "TCS8" traverse control system.
3. Turn on the "LVDAS" Laser Velocimeter Data Acquisition System.
4. Turn on the HP series 9000 model 330 computer.

This program will automatically be loaded and run when the computer is turned on. If it is not loaded then you can type in the following commands to load and then run it.

```
LOAD "LDVWT,1400,0,0"
RUN
```

When the program is ready for user operation, it will display three things on the CRT. These are the main menu, TCS8 traverse positions, and new sets of histogram & profile graphs. If they do not appear on the CRT then you should perform the following actions to reinitialize the systems.

1. Press shift reset on the HP series 9000 model 330 computers keyboard.
2. Press reset on the back of the TCS8.
3. Press reset on the front (or back) of the LVDAS.
4. LOAD "LDVWT,1400,0,0"
5. RUN

PROGRAM VARIABLES:

Mass Storage Variables:

- System$:
  - Tells the program where to read/store system data related files.
- Data$:
  - Tells the program where to read/store raw & reduced data related files.
- Title$:
  - File name for tunnel conditions data or raw & reduced data.
- Menu Variables:
  - Menu$(*):
    - String array where each element describes its corresponding menu subroutine's function.
  - Menu:
    - Used as an index to the string array Menu$(*). Indicates which of the menus has been selected as the current menu.
  - Key:
    - Used as an index to the string array Menu$(*). Indicates which one of eight menu subroutines in the menu is to be executed.
  - Busy:
    - Tells the Menu Status subprogram to display the current menu selection in inverse video.
  - Ready:
    - Tells the Menu Status subprogram to display the current menu selection in normal text.
- Traverse Position Variables:
  - Tcs1(**):
    - TCS8 transmitting side traverse positions (X1,Y1,Z1,A1) in TCS coordinates.
  - Tcs2(**):
    - TCS8 receiving side traverse positions (X2,Y2,Z2,A2) in TCS coordinates.
  - Tun(*):
    - Traverse positions (X,Y,Z) in TUNNEL coordinates.
Auto Move Traverse Position Variables:

- **Pos(***)**: Array of preprogrammed auto move positions.
- **PnameS(***)**: Names for the variables in **Pos(***)**.
- **Pimges(***)**: Image formats for the variables in **Pos(***)**.
- **PunitsS(***)**: Units for the variables in **Pos(***)**.
- **Npos**: Number of preprogrammed auto move positions in **Pos(***)**.
- **Faxis**: Specifies which axis is to be traversed for the profile. Also defines axis for plots.

Traverse Coordinate System Transformation Variables:

- **Index(***)**: Array of indexes of refraction for air, glass, and water.
- **Index(1)**: Index of refraction for Air.
- **Index(2)**: Index of refraction for Glass.
- **Index(3)**: Index of refraction for Water.
- **Theta**: Tx Side Off Axis Angle.
- **Fs**: Focal length for sending side on axis and off axis lenses.
- **Fr**: Focal length for receiving side off axis lens.
- **Bs**: Beam spacings for sending side on axis and off axis beam pairs.
- **Br**: Beam spacing for receiving side off axis.
- **Ts**: Angle of off axis sending side beam pair.
- **Tr**: Angle of off axis receiving side beam pair.
- **Ta**: Sending side off axis auxiliary traverse angle.
- **Tcs2tun1(***)**: Sending side coordinate system transformation matrix for converting **Tos1(***) to **Tun(***)
- **Tun2tcs1(***)**: Receiving side coordinate system transformation matrix for converting **Tos1(***) to **Tun(***)
- **Tcs2tun2(***)**: Receiving side coordinate system transformation matrix for converting **Tos2(***) to **Tun(***)
- **Tun2tcs2(***)**: Receiving side coordinate system transformation matrix for converting **Tos2(***) to **Tun(***)
- **Tcs2mod(***)**: Coordinate system transformation matrix for converting **Tos2(***) to **Mod(***)
- **Mod2tun(***)**: Coordinate system transformation matrix for converting **Mod(***) to **Tun(***)

Velocity Coordinate System Transformation Variables:

- **Index(***)**: Array of indexes of refraction for air, glass, and water.
- **Index(1)**: Index of refraction for Air.
- **Index(2)**: Index of refraction for Glass.
- **Index(3)**: Index of refraction for Water.
- **Theta1(***)**: Angles between LASER & TUNNEL UVW laser beams in Air (N=Index1).
- **Theta3(***)**: Angles between LASER & TUNNEL UVW laser beams in Water (N=Index3).
- **Tun2ldv(***)**: Coordinate system transformation matrix for converting from **TUNNEL** to **LASER**
- **Ldv2tun(***)**: Coordinate system transformation matrix for converting from **LASER** to **TUNNEL**

Traverse & Velocity Coordinate System Transformation Variables:

- **Alpha(***)**: Angles of attack, yaw, and roll.
- **Alpha(1)**: Angle of Attack.
- **Alpha(2)**: Angle of Yaw.
- **Alpha(3)**: Angle of Roll.
- **Mod2tun(***)**: Coordinate system transformation matrix for converting positions & velocities from **MODEL** to **TUNNEL**
- **Tun2pmod(***)**: Coordinate system transformation matrix for converting positions & velocities from **TUNNEL** to **MODEL**

Tunnel Condition Variables:

- **Array(***)**: Array of tunnel conditions, laser parameters, graph scales, etc.
- **NameS(***)**: Names for the variables in **Array(***)**.
- **UnitsS(***)**: Units for the variables in **Array(***)**.

Misc. Tunnel Condition Variables:

- **Date**: Date.
- **Time**: Time.
- **Run**: Run Number.
- **File**: File Number.
- **Mach**: Mach Number.
- **Temp**: Room Temperature (deg. F).
- **Uedge**: Freestream Velocity.
- **Ujet_u**: Jet exit velocity normalized by Uedge.

LVDAS Variables:

- **Table(***)**: Lookup table of frequencies.
- **Atime**: The maximum desired acquisition time (seconds).
- **Ctime**: The maximum desired coincidence time (seconds).
- **At_exp**: Exponent for interarrival times.
- **Ct_exp**: Exponent for coincidence times.
- **Nreads**: Number of desired samples.
- **Nsam**: Number of acquired samples.
- **Coin(***)**: Coincidence criteria.
1700 Cmask Coincidence mask for U, V, W selection.
1710 Raw(*) Array of raw data acquired from the LVDAS.

1730 Instantaneous Velocity and Voltage Variables:
1740 UI(*) Read from LVDAS as the instantaneous U frequency data, then converted into U velocities.
1750 VI(*) Read from LVDAS as the instantaneous V frequency data, then converted into V velocities.
1760 WI(*) Read from LVDAS as the instantaneous W frequency data, then converted into W velocities.
1770 BI(*) Read from LVDAS as the instantaneous B voltage data.
1780 II(*) Read from LVDAS as the raw interarrival time data, then converted into interarrival times.
1790 CI(*) Read from LVDAS as the raw coincidence time data, then converted into coincidence times.
1800 Valid(*) Validation words. Initially all ones, then some set to zero during histogram clipping.

1840 Histogram Clipping Variables:
1850 Umin The minimum acceptable U frequency (MHz).
1860 Umax The maximum acceptable U frequency (MHz).
1870 Vmin The minimum acceptable V frequency (MHz).
1880 Vmax The maximum acceptable V frequency (MHz).
1890 Wmin The minimum acceptable W frequency (MHz).
1900 Wmax The maximum acceptable W frequency (MHz).
1910 Clip Clip: 1 turn histogram clipping on; 0 turns it off.

1940 Frequency to Velocity Conversion Variables:
1950 Beam_spc(*) Beam spacing at lens.
1960 Pool_len(*) Focal length.
1970 Beam_sep(*) Beam separation angle in degrees (full angle).
1980 Wave_len(*) Wave length.
2000 Frng_spc(*) Fringe spacings.
2010 Brng_freq(*) Bragg Frequencies.
2020 Mix_freq(*) Mixing Frequencies.
2030 Mea_sgn(*) Measured Frequencies' Signs.
2040 Brng_sgn(*) Bragg Frequencies' Signs.
2050 Mix_sgn(*) Mixing Frequencies' Signs.

2070 Summation Variables:
2080 Sum(1,1) Summation of all of the valid UI.
2090 Sum(2,1) Summation of all of the valid VI.
2100 Sum(3,1) Summation of all of the valid WI.
2110 Sum(4,1) Summation of all of the valid BI.
2120 Sum(5,1) Summation of all of the valid WI.
2130 Sum(6,1) Summation of all of the valid II.
2140 Sum(7,1) Summation of all of the valid CI.
2150 Sum(1,2) Summation of all of the valid U*I*U.
2160 Sum(2,2) Summation of all of the valid V*V.
2170 Sum(3,2) Summation of all of the valid W*W.
2180 Sum(4,2) Summation of all of the valid B*B.
2190 Sum(5,2) Summation of all of the valid B*U.
2200 Sum(6,2) Summation of all of the valid B*I.
2210 Sum(7,2) Summation of all of the valid C*I.
2220 Sum(1,3) Summation of all of the valid U*I*U.
2230 Sum(2,3) Summation of all of the valid V*V.
2240 Sum(3,3) Summation of all of the valid W*W.
2250 Sum(4,3) Summation of all of the valid B*B.
2260 Sum(5,3) Summation of all of the valid B*U.
2270 Sum(6,3) Summation of all of the valid B*I.
2280 Sum(7,3) Summation of all of the valid C*I.
2290 Sum(l,2) Summation of all of the valid U*I*U.
2300 Sum(2,1) Summation of all of the valid U*I*U.
2310 N(*) Number of valid samples for the above summations.

2330 Reduced Data Variables:
2340 U Average U frequency or velocity.
2350 V Average V frequency or velocity.
2360 W Average W frequency or velocity.
2370 A Average A voltage.
2380 B Average B voltage.
2390 I Average interarrival time.
2400 C Average coincidence time.
2410 U1 Standard deviation for U frequency or velocity.
2420 V1 Standard deviation for V frequency or velocity.
2430 W1 Standard deviation for W frequency or velocity.
2440 A1 Standard deviation for A voltage.
2450 B1 Standard deviation for B voltage.
2460 I1 Standard deviation for interarrival time.
2470 C1 Standard deviation for coincidence time.
2480 U1v1 Velocity:Velocity Shear Stress.
2490 V1w1 Velocity:Velocity Shear Stress.
2890 DIM Tc
2870
2850 COM
2830 COM
2810 COM
2790 SymbolS(*) Array of symbol arrays. Each symbol array contains a distinct geometric symbol.
2780 Symbol(*) Array of coordinates which when connected produce a distinct geometric symbol.
2770 Dot(*) Array of coordinates which produce a dot. The dot symbol is added to all symbols.
2760 Noc(*) The number of coordinates in a symbol.
2750 S(*) Used to index the symbols array.

2800 Data Plotting Symbol Variables:
2780 Symbols(*) Array of symbol arrays. Each symbol array contains a distinct geometric symbol.
2770 Symbol(*) Array of coordinates which when connected produce a distinct geometric symbol.
2760 Dot(*) Array of coordinates which produce a dot. The dot symbol is added to all symbols.
2750 Noc(*) The number of coordinates in a symbol.
2740 S(*) Used to index the symbols array.

2850 Histogram and Profile Graph Variables:
2840 Wndw(*) Array containing the plot's scales.
2830 Vpwt(*) Array containing the plot's CRT position.
2820 Xdiv(*) Array containing the number of X divisions for the plot's X axis.
2810 Ydiv(*) Array containing the number of Y divisions for the plot's Y axis.
2800 Xlabel$(*)) String array containing labels for the X axis.
2790 Ylabel$(*)) String array containing labels for the Y axis.
2780 TitleS(*) String array containing labels for the Plots.
2770 Ximage$(*)) String array containing image formats for the X axis labeling.
2760 Yimage$(*)) String array containing image formats for the Y axis labeling.
2750 LegendS(*) String array containing labels for each symbol in a profile plot.
2740 G(*) Used as an index to the above arrays. Specifies one of nine plots.
2730 Gsave(*) Used to save the entire graphics contents of the CRT.

2800 OPTION BASE 1
2800 COM /Data/ INTEGER Raw(1000,10),Valid(1000),REAL Table(0;32766),U1(1000),V1(1000),W1(1000),A1(1000),
2790 BI(1000),CI(1000),C1(1000)
2780 COM /Array/ NameS(100,4)[10],ImageS(100,4)[10],UnitsS(100,4)[10],REAL Array(100,4)
2770 COM /Ps: PnameS(25,1)[10],Images(25,1)[10],Points(25,1)[10],REAL Pos(25,1),Bpos
2760 COM /Graph/ Wndw(9,4),Vpwt(9,4),Xdiv(9),Ydiv(9),Xlabel$(9),Ylabel$(9),TitleS(9)[80],
2750 XimageS(9)[80],YimageS(9)[80],LegendS(9,5)[80]
2740 COM Run,File,Palette
2730 DIM MenusS(5,8)[80],SystemS[20],DataS[20],FilesS[50],LS[160]
2720 INTEGER Gsave(1020,1024),At_exp,Ct_exp,Cmask,NumX,N(10)
2710 REAL Atime,Ctime,Sum(10,3),SymbolsS(5,0;2,3),Apos,Bpos
2700 DIM Tcs2tun(4,4),T2c2cs(4,4),Tun2mod(3,3),Tun2dv(3,3),Tun(4),Tcs1(4)
2690 DIM Tcs2tun(4,4),T2c2cs(4,4),Mod2tun(3,3),Mod2dv(3,3),Mod(4),Tcs2(4)
2680 DIM Beam_sep(3),Wave_len(3),Brq_fq(3),Brq_qn(3),Index(3),Coln(3),ThetaS(3,3),Alpha(3)
2670 ! Perform trigonometric operations in degrees.
2660 DEG
2650 ! Clear the CRT and direct printed output to it.
2640 CLEAR SCREEN
2630 GCLEAR
2620 PRINTER IS CRT
2610 ! Perform any necessary setup and initialization routines.
2600 GOSUB Setup up ! Initialize the HP to LVDAS interface.
2590 GOSUB File_set_up ! Select mass storage device for system & data files.
2580 GOSUB Tcs8_set_up ! Initialize the HP to TCS8 interface.
2570 GOSUB Menu_set_up ! Initialize the user driven menus and display the main menu.
2560 GOSUB Graph_set_up ! Initialize the CRT and plot the nine empty plots for profiles and histograms.
2550 ! The main program, while continually displaying the time of day, will wait for menu key selection.
2540 Here:
2530 Date=TIMEODATE
2520 Time=Date
2510 DISP TIMES(Time),DATES(Date)
2500 GOTO Here
2490 On_key:
2480 ON KEY 1 GOSUB Key1 ! If the user function key #1 is ever pressed then execute the "Key1" subroutine.
2470 ON KEY 2 GOSUB Key2 ! If the user function key #2 is ever pressed then execute the "Key2" subroutine.
2460 ON KEY 3 GOSUB Key3 ! If the user function key #3 is ever pressed then execute the "Key3" subroutine.
2450 ON KEY 4 GOSUB Key4 ! If the user function key #4 is ever pressed then execute the "Key4" subroutine.
2440 ON KEY 5 GOSUB Key5 ! If the user function key #5 is ever pressed then execute the "Key5" subroutine.
2430 ON KEY 6 GOSUB Key6 ! If the user function key #6 is ever pressed then execute the "Key6" subroutine.
2420 ON KEY 7 GOSUB Key7 ! If the user function key #7 is ever pressed then execute the "Key7" subroutine.
2410 ON KEY 8 GOSUB Key8 ! If the user function key #8 is ever pressed then execute the "Key8" subroutine.
2400 RETURN
2390 Keys:
2380 ! Subroutine Key1,Key2,Key3,Key4,Key5,Key6,Key7,Key8 descriptions:
2370 ! When one of the special user function keys is pressed, the main program will execute one the following eight subroutines. Each of these subroutines performs essentially the same basic
2360 ! function in that it subsequently executes one of the menu subroutines. The particular menu
2350 ! subroutine to be executed will depend on the current menu selected and the current key pressed.
2340 ! Before the selected menu subroutine is executed, the corresponding menu entry at the top of
2330 ! the CRT is redisplayed in inverse video. This indicates that the menu selection has been
2320 ! acknowledged and that any resultant actions are still in progress. When the highlighted menu
2310 ! subroutine has completed the current TCS8 traverse positions will be read and updated on the CRT
2300 ! display. The corresponding menu entry displayed at the top of the CRT is redisplayed in normal

62
Variables:

Menu indicates which of the menus has been selected as the current menu.
Key indicates which one of eight menu subroutines in the menu is to be executed.
Menu(*) string array where each element describes its corresponding menu subroutine's function.
Busy tells the Menu Status subroutine to display the current menu selection in inverse video.
Ready tells the Menu Status subroutine to display the current menu selection in normal text.

Key1:
CALL Menu_status(Menu, Key, Busy, Menu(*))
ON Menu GOSUB M1k1, M2k1, M3k1, M4k1, M5k1, M6k1, M7k1
CALL Menu_status(Menu, Key, Ready, Menu(*))
CALL Tcs8Read(@Tcs8, Mod(*), Tun(*), Tcs1(*), Tcs2(*), Tcs2tun1(*), Tcs2tun2(*), Tun2mod(*))
RETURN

Key2:
CALL Menu_status(Menu, Key, Busy, Menu(*))
ON Menu GOSUB M1k2, M2k2, M3k2, M4k2, M5k2, M6k2, M7k2
CALL Menu_status(Menu, Key, Ready, Menu(*))
CALL Tcs8Read(@Tcs8, Mod(*), Tun(*), Tcs1(*), Tcs2(*), Tcs2tun1(*), Tcs2tun2(*), Tun2mod(*))
RETURN

Key3:
CALL Menu_status(Menu, Key, Busy, Menu(*))
ON Menu GOSUB M1k3, M2k3, M3k3, M4k3, M5k3, M6k3, M7k3
CALL Menu_status(Menu, Key, Ready, Menu(*))
CALL Tcs8Read(@Tcs8, Mod(*), Tun(*), Tcs1(*), Tcs2(*), Tcs2tun1(*), Tcs2tun2(*), Tun2mod(*))
RETURN

Key4:
CALL Menu_status(Menu, Key, Busy, Menu(*))
ON Menu GOSUB M1k4, M2k4, M3k4, M4k4, M5k4, M6k4, M7k4
CALL Menu_status(Menu, Key, Ready, Menu(*))
CALL Tcs8Read(@Tcs8, Mod(*), Tun(*), Tcs1(*), Tcs2(*), Tcs2tun1(*), Tcs2tun2(*), Tun2mod(*))
RETURN

Key5:
CALL Menu_status(Menu, Key, Busy, Menu(*))
ON Menu GOSUB M1k5, M2k5, M3k5, M4k5, M5k5, M6k5, M7k5
CALL Menu_status(Menu, Key, Ready, Menu(*))
CALL Tcs8Read(@Tcs8, Mod(*), Tun(*), Tcs1(*), Tcs2(*), Tcs2tun1(*), Tcs2tun2(*), Tun2mod(*))
RETURN

Key6:
CALL Menu_status(Menu, Key, Busy, Menu(*))
ON Menu GOSUB M1k6, M2k6, M3k6, M4k6, M5k6, M6k6, M7k6
CALL Menu_status(Menu, Key, Ready, Menu(*))
CALL Tcs8Read(@Tcs8, Mod(*), Tun(*), Tcs1(*), Tcs2(*), Tcs2tun1(*), Tcs2tun2(*), Tun2mod(*))
RETURN

Key7:
CALL Menu_status(Menu, Key, Busy, Menu(*))
ON Menu GOSUB M1k7, M2k7, M3k7, M4k7, M5k7, M6k7, M7k7
CALL Menu_status(Menu, Key, Ready, Menu(*))
CALL Tcs8Read(@Tcs8, Mod(*), Tun(*), Tcs1(*), Tcs2(*), Tcs2tun1(*), Tcs2tun2(*), Tun2mod(*))
RETURN

Key8:
CALL Menu_status(Menu, Key, Busy, Menu(*))
ON Menu GOSUB M1k8, M2k8, M3k8, M4k8, M5k8, M6k8, M7k8
CALL Menu_status(Menu, Key, Ready, Menu(*))
CALL Tcs8Read(@Tcs8, Mod(*), Tun(*), Tcs1(*), Tcs2(*), Tcs2tun1(*), Tcs2tun2(*), Tun2mod(*))
RETURN

Menu:

Descriptions of the "Main Menu" subroutines M1k1,...,M1k8:

The eight subroutines M1k1,...,M1k8 together implement the "Main Menu". The following will be displayed at the top left of the CRT display when the "Main Menu" is selected:

- M1k1: Laser Alignment
- M1k2: Pre Run
- M1k3: Post Run (Dum Graphic Graph)
- M1k4: Set Auto Move Positions
- M1k5: Move traverse
- M1k6: Make data
- M1k7: Auto move and take
- M1k8: Display Histogram

M1k1 will change the current active menu from the "Main Menu" to the "Laser Alignment Menu".
M1k2 will change the current active menu from the "Main Menu" to the "Pre Run Menu". M1k3 will transfer the graphics contents of the CRT to the printer. This provides a hard copy of the profile plots. M1k4 has the user enter predefined traverse positions for a profile plot. M1k5 moves the traverse to a user selectable position. M1k6 acquires LVDA data at the current TCSB traverse position. M1k7 acquires LVDA data at each of the preprogrammed TCSB traverse positions set up by M1k4. M1k8 repeatedly displays five channels of real time histograms until the user presses any key on the keyboard.

M1k1: Change the current active menu from the "Main Menu" to the "Laser Alignment Menu".
Menu=2
CALL Menu_disp(Menu,Menu$(*))

| I Change the current active menu from the "Main Menu" to the "Pre Run Menu". |
| RETURN |

| 4400 Mlk2: |
| I Transfer the graphics contents of the CRT to the printer. This provides a hard copy of the plots. |
| 4410 KEY LABELS OFF |
| I Turn off the key labels so that they won't be printed. |
| 4420 CALL Menu_disp(Menu,Menu$(*)) |
| RETURN |

| 4440 Mlk3: |
| I Move the traverse to a user selectable position. |
| 4450 CALL Read_calc_fill |
| 4460 CALL TcsRead(Tcs8,Mod(*),Tun(*),Tcs1(*),Tcs2(*),Tcs2tun1(*),Tcs2tun2(*),Tun2mod(*)) |
| 4470 CALL Enter_value(CHRS(NUM("X")+Paxis-1),Mod(Paxis),"K") |

| 4480 Mlk4: |
| I Have the user enter predefined traverse positions for a profile plot. |
| 4490 CALL Enter_value("Number of traverse positions",Npos,"K") |
| 4500 REDIM Pos(Npos,1),Pname$(Npos,1),Pimage$(Npos,1),PunitsS(Npos,1) |
| 4510 MAT Pimage$= "MD,4D"

4520 MAT Punits$= "in"

4530 FOR K=1 TO Npos

4540 PnameS(K,1)="Pos #&VALS(K)"

4550 NEXT K

4560 GSTORE Gsave(*)

4570 CALL Change("VALUES",Pos(*),PnameS(*),Pimage$(*),PunitsS(*))

4580 GLOAD Gsave(*)

4590 CALL Menu_disp(Menu,Menu$(*))

| RETURN |

| 4600 Mlk5: |
| I Acquire LVDAS data at the current TCS8 traverse position. |
| 4610 INPUT "Press any key to TAKE DATA" |
| 4620 CALL Rt_histo(0,Lvdas,Symbols(*),1) |
| 4630 Cmask=Coln(1)*1+Coln(2)*2+Coln(3)*4 |
| 4640 Neam=MIN(Nreads,1000) |
| 4650 Date=TIMEDATE |
| 4660 TIME=Date |
| 4670 Time=Date |
| 4680 CALL Lvdas_take(0,Lvdas,Atime,Ctime,At_exp,Ct_exp,Cmask,Neam) |
| 4690 IF Neam=1 THEN |
| 4700 SELECT Paxis |
| 4710 CASE 3 |
| DS="X" |
| FS="Y" |
| Apos=Mod(1) |
| Bpos=Mod(2) |
| 4720 CASE 2 |
| DS="X" |
| FS="Z" |
| Apos=Mod(1) |
| Bpos=Mod(3) |
| 4730 CASE 1 |
| DS="Y" |
| FS="Z" |
| Apos=Mod(2) |
| Bpos=Mod(3) |
| 4740 END SELECT |
| 4750 OUTPUT PRT USING "K",CHR$(27),"&K25\"&CHR$(27),"19D",RPT$="",140 |
| 4760 CALL Rt_histo(0,Lvdas,Symbols(*),1) |
| 4770 Cmask=Coln(1)*1+Coln(2)*2+Coln(3)*4 |
| 4780 Neam=MIN(Nreads,1000) |
| 4790 Date=TIMEDATE |
| 4800 TIME=Date |
| 4810 CALL Lvdas_take(0,Lvdas,Atime,Ctime,At_exp,Ct_exp,Cmask,Neam) |
| 4820 IF Neam=1 THEN |
| 4830 SELECT Paxis |
| 4840 CASE 3 |
| DS="X" |
| FS="Y" |
| Apos=Mod(1) |
| Bpos=Mod(2) |
| 4850 CASE 2 |
| DS="X" |
| FS="Z" |
| Apos=Mod(1) |
| Bpos=Mod(3) |
| 4860 END SELECT |
| 4870 OUTPUT PRT USING "K",CHR$(27),"&K25\"&CHR$(27),"19D",RPT$="",140 |
| 4880 CALL Rt_histo(0,Lvdas,Symbols(*),1) |
| 4890 Cmask=Coln(1)*1+Coln(2)*2+Coln(3)*4 |
| 4900 Neam=MIN(Nreads,1000) |
| 4910 Date=TIMEDATE |
| 4920 TIME=Date |
| 4930 CALL Lvdas_take(0,Lvdas,Atime,Ctime,At_exp,Ct_exp,Cmask,Neam) |
| 4940 IF Neam=1 THEN |
| 4950 SELECT Paxis |
| 4960 CASE 3 |
| DS="X" |
| FS="Y" |
| Apos=Mod(1) |
| Bpos=Mod(2) |
| 4970 CASE 2 |
| DS="X" |
| FS="Z" |
| Apos=Mod(1) |
| Bpos=Mod(3) |
| 4980 END SELECT |
| 4990 OUTPUT PRT USING "K",CHR$(27),"&K25\"&CHR$(27),"19D",RPT$="",140 |
| 5000 CALL Rt_histo(0,Lvdas,Symbols(*),1) |
| 5010 Cmask=Coln(1)*1+Coln(2)*2+Coln(3)*4 |
| 5020 Neam=MIN(Nreads,1000) |
| 5030 Date=TIMEDATE |
| 5040 TIME=Date |
| 5050 CALL Lvdas_take(0,Lvdas,Atime,Ctime,At_exp,Ct_exp,Cmask,Neam) |
| 5060 IF Neam=1 THEN |
| 5070 SELECT Paxis |
| 5080 CASE 3 |
| DS="X" |
| FS="Y" |
| Apos=Mod(1) |
| Bpos=Mod(2) |
| 5090 CASE 2 |
| DS="X" |
| FS="Z" |
| Apos=Mod(1) |
| Bpos=Mod(3) |
| 5100 END SELECT |
| 5110 OUTPUT PRT USING "K",CHR$(27),"&K25\"&CHR$(27),"19D",RPT$="",140 |
| 5120 CALL Rt_histo(0,Lvdas,Symbols(*),1) |
| 5130 Cmask=Coln(1)*1+Coln(2)*2+Coln(3)*4 |
| 5140 Neam=MIN(Nreads,1000) |
| 5150 Date=TIMEDATE |
| 5160 TIME=Date |
| 5170 CALL Lvdas_take(0,Lvdas,Atime,Ctime,At_exp,Ct_exp,Cmask,Neam) |
| 5180 IF Neam=1 THEN |
| 5190 SELECT Paxis |
| 5200 CASE 3 |
| DS="X" |
| FS="Y" |
| Apos=Mod(1) |
| Bpos=Mod(2) |
| 5210 CASE 2 |
| DS="X" |
| FS="Z" |
| Apos=Mod(1) |
| Bpos=Mod(3) |
| 5220 END SELECT |
| 5230 OUTPUT PRT USING "K",CHR$(27),"&K25\"&CHR$(27),"19D",RPT$="",140 |
| 5240 CALL Rt_histo(0,Lvdas,Symbols(*),1) |
| 5250 Cmask=Coln(1)*1+Coln(2)*2+Coln(3)*4 |
| 5260 Neam=MIN(Nreads,1000) |
| 5270 Date=TIMEDATE |
| 5280 TIME=Date |
| 5290 CALL Lvdas_take(0,Lvdas,Atime,Ctime,At_exp,Ct_exp,Cmask,Neam) |
| 5300 IF Neam=1 THEN |
CALL Data_transf(Ldv2tun(*),U,V,W)

! user enter a movement for the X axis

specifies movements to be relative to the current position

M2K3 selects the TCS
M2K2 selects whether the M2KI will change the current active menu
M2K1: Coordinates: MODEL
M2K4: Mode: ABSOLUTE

the second analog channel B is not being used
following will be displayed at the top left of the CRT display when the "Laser Alignment Menu" is selected:
M2K1: Return to main menu
M2K2: Sides: Tx & Rx
M2K3: Coordinates: MODEL
M2K4: Mode: ABSOLUTE
M2K5: Move X
M2K6: Move Y
M2K7: Move Z
M2K8: Move A

M2K1 will change the current active menu from the "Laser Alignment Menu" to the "Main Menu".
M2K2 selects whether the transmitting, receiving, or both sides of the traverse are to be moved.
M2K3 selects the TCS, TUNNEL, or MODEL coordinate systems for traverse movements.
M2K4 sets the movement types to be relative to the current position or to absolute positions.
M2K5 has the user enter a movement for the X axis and then the movement is performed. M2K6 has the user enter
M2K1: Change the current active menu from the "Laser Alignment Menu" to the "Main Menu".

CALL Menu disp(Menu,Menu$(*))
RETURN

M2K2: Select whether the transmitting, receiving, or both sides of the traverse are to be moved.

SELECT TRIMS(Menu$ (Menu,Key) [20])
CASE "Tx & Rx"
   Menu$ (Menu,Key) [20] = "Tx"
   Menu$ (Menu,Key) [20] = "Rx"
CASE "Tx"
   Menu$ (Menu,Key) [20] = "Tx & Rx"
END SELECT

CALL Menu disp(Menu,Menu$(*))
RETURN

M2K3: Select the TCS, TUNNEL, or MODEL coordinate systems for traverse movements.

SELECT TRIMS(Menu$ (Menu,Key) [20])
CASE "MODEL"
   Menu$ (Menu,Key) [20] = "TUNNEL"
CASE "TUNNEL"
   Menu$ (Menu,Key) [20] = "TCS"
CASE "TCS"
   Menu$ (Menu,Key) [20] = "MODEL"
END SELECT

CALL Menu disp(Menu,Menu$(*))
RETURN

M2K4: Specifies movements to be relative to the current profile position or to absolute positions.

SELECT TRIMS(Menu$ (Menu,Key) [20])
CASE "ABSOLUTE"
   Menu$ (Menu,Key) [20] = "RELATIVE"
CASE "RELATIVE"
   Menu$ (Menu,Key) [20] = "ABSOLUTE"
END SELECT

CALL Menu disp(Menu,Menu$(*))
RETURN

M2K5: The subroutines M2K5 thru M2K8 all execute the same code. The code will have the user enter a movement for the X,Y,Z, or A depending on what the value of "Key" is. The user specified movement will then be performed.

CALL Menu disp(Menu,Menu$(*))
RETURN

M2K6: Side$=TRIMS(Menu$ (Menu,2) [20])
Coor$=TRIMS(Menu$ (Menu,3) [20])
Mode$=TRIMS(Menu$ (Menu,4) [20])
CALL Enter value(Menu$ "Movement",Movement,"4D.5D")
ON NO CALL Do nothing
ON KBD CALL Do something

CALL Tcsread((Tcs8,Mod,*,Tun,*,Tcs2,*,Tcs2tun,*,Tcs2tun2,*,Tun2mod,*)
CALL Tcsmove((Tcs8,Mod,*,Tun,*,Tcs2,*,Tcs2tun,*,Tun2tcs,*,Tun2tcs2,*,Side$,Coor$,Mode$,
Key=4,Movement)
CALL Tcsread(Tcs8,Mod,*,Tun,*,Tcs2,*,Tcs2tun,*,Tcs2tun2,*,Tun2mod,*)
DISP ""
OFF KBD
RETURN

Menu3: Descriptions of the "Pre Run Menu" subroutines M3K1,....M3K8:

M3K1: Return to MAIN menu
M3K2: Enter Run & File Numbers
M3K3: Enter Number of Samples
M3K4: Select Traverse Axis for Profile
M3K5: Print Coordinate Transformation Matrices
M3K6: Setup Graphics
M3K7: Tunnel Conditions
M3K8: Traverse

M3K1 will change the current active menu from the "Pre Run Menu" to the "Main Menu". M3K2 has the user enter the Run and File numbers. A new run number should be assigned to each profile while a new file number is assigned to each set of data. M3K3 has the user enter the desired number of samples. M3K4 has the user select which axis to traverse in for the profiles. M3K5 prints the coordinate system transformation matrices for both traverse positions and velocities. M3K6 creates a new set of empty plots for new profiles. M3K7 will change the current active menu from the "Pre Run Menu" to the "Tunnel Conditions Menu". M3K8 will change the current active menu from the "Pre Run Menu" to the "Traverse Menu".

M3K1: Change the current active menu from the "Pre Run Menu" to the "Main Menu".

Menu=1
CALL Menu_disp(Menu,Menu$(*))
RETURN
M3k2: ! Have the user enter the Run and File numbers.
CALL Enter_value("Run",Run,"3D,2D")
CALL Enter_value("File",File,"3D")
RETURN
M3k3: ! Have the user enter the desired number of samples.
CALL Enter_value("Number of Samples ",Nreads,"K")
RETURN
M3k4: ! Have the user select which axis to traverse in for the profiles.
CALL Enter_string("Traverse Axis for Profiles ",Paxis,$)
M4k2: ! Load the old tunnel conditions from a file on the disk. This loads the default values.
GOSUB Read_cal.fill
RETURN
M4k3: ! Save the current tunnel conditions to a file on the disk. This updates the default values on the disk.
GOSUB Read_cal.fill
GOSUB Save_array
RETURN
M4k4: ! Print the current tunnel conditions.
GOSUB Read_cal.fill
GOSUB Print_header
RETURN
M4k5: ! Have the user enter values for the tunnel condition variables.
GSTORE Gsave(*)
GOSUB Read_cal.fill
CALL Change("VALUES",Array(*),Name$(*),Image$(*),Units$(*))
GOSUB Read_cal.fill
GLOAD Gsave(*)
RETURN
M4k6: ! Have the user enter names for the tunnel condition variables.
GSTORE Gsave(*)
GOSUB Read_cal.fill
CALL Change("NAMES",Array(*),Name$(*),Image$(*),Units$(*))
GOSUB Read_cal.fill
GLOAD Gsave(*)
RETURN
M4k7: ! Have the user enter units for the tunnel condition variables.
GSTORE Gsave(*)
GOSUB Read_cal.fill
CALL Change("UNITS",Array(*),Name$(*),Image$(*),Units$(*))
GOSUB Read_cal.fill
GLOAD Gsave(*)
RETURN
M4k8: ! Have the user enter image formats for the tunnel condition variables.
GSTORE Gsave(*)
GOSUB Read_cal.fill
CALL Change("IMAGES",Array(*),Name$(*),Image$(*),Units$(*))
GOSUB Read_cal.fill
GLOAD Gsave(*)
RETURN

Menu5: Descriptions of the "Traverse Menu" subroutines MSK1,...,MSK8:
The eight subroutines MSK1,...,MSK8 together implement the "Traverse Menu". The following will be displayed at the top left of the CRT display when the "Traverse Menu" is selected:

MSK1: Return to PRE RUN menu
MSK2: View & Set TCS8 Positions
MSK3: View & Set TCS8 Units
MSK4: View & Set TCS8 Revolution
MSK5: View & Set TCS8 Velocity
MSK6: View & Set TCS8 Acceleration
MSK7:
MSK8:

MSK1 will change the current active menu from the "Traverse Menu" to the "Pre Run Menu". MSK2 reads from the TCS8 the current positions and lets the user change them. The new positions are then sent to the TCS8. MSK3 reads from the TCS8 the current counts per unit length (inches) and lets the user change them. The new counts per unit length are then sent to the TCS8. MSK4 reads from the TCS8 the current counts per revolution and lets the user change them. The new counts per revolution are then sent to the TCS8. MSK5 reads from the TCS8 the current velocities and lets the user change them. The new velocities are then sent to the TCS8. MSK6 reads from the TCS8 the current accelerations and lets the user change them. The new accelerations are then sent to the TCS8. MSK7 does nothing. MSK8 does nothing.

Menu=3
CALL Menu_disp(Menu,Menu$(*))
RETURN

Menu3
CALL Menu_disp(Menu,Menu$(*))
RETURN

Menu2: ! Read current TCS8 positions, have the user update them, & then send the new values to the TCS8.
CALL Tcs8set("P",@Tcs8)
! View and set TCS8 Positions.
GRAPHICS ON
CALL Menu_disp(Menu,Menu$(*))
RETURN

Menu3
CALL Tcs8set("U",@Tcs8)
! View and set TCS8 counts per Unit length.
GRAPHICS ON
CALL Menu_disp(Menu,Menu$(*))
RETURN

Menu4: ! Read current TCS8 counts per revolution, have the user update them, & then send new values to the TCS8.
CALL Tcs8set("R",@Tcs8)
! View and set TCS8 counts per Revolution.
GRAPHICS ON
68
820 CALL Menu disp(Menu,Menu$(*))
830 RETURN
840 M5k5: ! Read current TCS8 velocities, have the user update them, & then send the new values to the TCS8.
845 CALL Tcs8Set("V",@Tcs8) ! View and set TCS8 Velocities.
850 GRAPHICS ON
855 CALL Menu disp(Menu,Menu$(*))
860 RETURN
865 M5k6: ! Read current TCS8 accelerations, have the user update them, & then send the new values to the TCS8.
870 CALL Tcs8Set("A",@Tcs8) ! View and set TCS8 Accelerations.
875 GRAPHICS ON
880 CALL Menu disp(Menu,Menu$(*))
890 RETURN
895 M5k7: RETURN ! This subroutine does nothing.
900 M5k8: RETURN ! This subroutine does nothing.
905 Quit: Quit=1 ! Quit will be set during a multiple position scan (see MK7) if any key is pressed on the keyboard during the scan. This indicates that the scan should be terminated.
910 Lvdas_set_up: ! This subroutine initializes the HP to LVDAS high speed parallel interface. A communications path named "&Lvdas" is opened. Also, this subroutine creates the raw data to frequency conversion look up table.
915 CALL Lvdas_init(&Lvdas)
920 CALL Table(Table(*))
925 RETURN
930 File_set_up: ! This subroutine reads the initialization files from the disk. System$ tells the program where to read a system related files while Data$=":
935 System$="t,1400,0,0"
940 Data$="t,1400,0,1"
945 LOAD KEY "KEYS$"System$
950 GOSUB Read_array
955 GOSUB Read_calc_fill
960 GOSUB Save_array
965 CLEAR SCREEN
970 RETURN
975 Tcs8_set_up: ! This subroutine initialized the HP to TCS8 serial interface. The communications path "&Tcs8" is opened.
980 CALL Tcs8Init(&Tcs8)
985 CALL Tcs8Read(&Tcs8,Mod(*)\),Tun(*),Tcs1(*),Tcs2(*),Tcs2tun1(*),Tcs2tun2(*),Tun2mod(*))
990 GOSUB Calc
995 GOSUB Fill
1000 RETURN
1005 Grph_set_up: ! This subroutine defines the graphics symbols for plotting data points, clears and initializes the CRT, and displays a new empty set of graphs for histogram and profile plotting.
1010 CALL Read_symbols(Symbols(*))
1015 CALL Crt init
1020 CALL Setup_graph(Array(*),Image$(*),Paxls,Symbols(*))
1025 RETURN
1030 Menu_set_up: ! This subroutine defines the menu descriptors for all of the menus. The current menu is set to the "Main" Menu and its menu is displayed at the top of the screen.
1035 CALL Menu read(Menu$(*))
1040 CALL Menu disp(Menu,Menu$(*))
1045 GOSUB On key
1050 Busy=0
1055 Ready=1
1060 RETURN
1065 Print_header: ! This subroutine prints a header on the printers paper. The "header" is a formatted list of all of the tunnel conditions, laser parameters, and graph scales.
1070 PRINT USING "#,8,5(K)";CHR$(27);"*",625*;CHR$(27);"*",190"
1075 CALL Array print(Array(*),Name$(*),Image$(*),Units$(*))
1080 PRINT USING "#,8,5(K)";CHR$(27);"*";
1085 PRINT USING "#,8,5(K)";CHR$(27);"*"
1090 PRINTER IS CRT
1095 RETURN
1100 Read_calc_fill: ! This subroutine extracts (reads) the tunnel conditions from the Array(*). These values can be used to calculate other tunnel conditions. The original tunnel conditions along with any calculated tunnel conditions are then packed back (filled) into the Array(*).
1105 GOSUB Read
1110 GOSUB Calc
1115 GOSUB Fill
1120 RETURN
1125 Store_header: ! This subroutine stores the header Array(*) and other arrays onto the disk. There will be one header file for each run number. For example, if the run number equals 1, then the data will be stored in a file named "R1". This file will include an extensive list of tunnel conditions, laser parameters, graph scales, traverse positions, coordinate system transformation matrices, etc.
1130 ! Set Files equal to the file name for the header file. Each run number will have a different file name.
1135 File$="R";EVALS(Run)&Data$
1140 PRINT USING "#",Run;EVALS(Run)&Data$
1145 ! Check if the file already exists. If it does then, ask the user if he wants to overwrite the old file.
1150 OFF ERROR
1155 FOR K=1 TO 10
1160 WAIT .2
1165 BEEP
1170 ERROR
CREATE BDAT File$,Ysize
Calculate the data's file size.

OPEN Err$="EVAL$(Run)&"data",File$ OPEN
Open the data's file.

READ_data: ! Th...
GOSUM Save_array
RETURN
Save_array:

This subroutine saves the header Array(*) onto the disk in a file named “ARRAY”. The file will then have

* default values for the tunnel conditions, laser parameters, graph scales, etc. This file is not meant to
* be attached to any run number or profile scan. It is used to provide default values for the program so
* that the user will not have to enter a rather lengthy list of tunnel conditions.

ASSIGN #file TO “ARRAY4$System$”
OUTPUT #file:Array(*), NameS(*), ImageS(*), UnitsS(*)
OUTPUT #file:Tun2Tcs2(*), Tun2Tcs8(*), Mod2tun(*), Tun2lv(*)
OUTPUT #file:Tcs2Tun(*), Tcs2Tun2(*), Tun2mod(*), Ldv2tun(*)
ASSIGN #file TO *
RETURN

Fill:

This subroutine fills the Array(*) with the current tunnel conditions, laser parameters, and histogram

& profile scales.

Array(1,1)=Date  ! Date.
Array(1,2)=Mach  ! Mach Number.
Array(1,4)=Alpha(1)  ! Angle of Attack.
Array(2,1)=Time  ! Time.
Array(2,2)=Temp  ! Room Temperature (deg. F).
Array(2,4)=Alpha(2)  ! Angle of Yaw.
Array(3,1)=Run  ! Run Number.
Array(3,2)=Uedge  ! Freestream Velocity.
Array(3,4)=Alpha(3)  ! Angle of Roll.
Array(4,1)=File  ! File Number.
Array(4,2)=u_max  ! Jet exit velocity normalized by Uedge.
Array(4,4)=Theta  ! Tx Side Off Axis Angle.
MAT Array(11:14,1)= Mod  ! Frobe volume positions in MODEL coordinates.
MAT Array(11:14,2)= Tun  ! Frobe volume positions in TUNNEL coordinates.
MAT Array(11:14,3)= Tcs8  ! Rx side traverse positions in Tcs8 coordinates.
MAT Array(11:14,4)= Tcs2  ! Rx side traverse positions in Tcs2 coordinates.
MAT Array(21,1:3)= Index  ! Index of refraction of for laser light (e.g. NaI,Nglass,Nwater).
MAT Array(22:24,1:3)= Theta1  ! Angles between LASER & TUNNEL UVW laser beams in Air (N=Index1).
MAT Array(25:27,1:3)= Theta2  ! Angles between LASER & TUNNEL UVW laser beams in Water (N=Index3).
MAT Array(31,1:3)= Beam_spec  ! Beam spacing at lens.
MAT Array(32,1:3)= Focu_len  ! Focal length.
MAT Array(33,1:3)= Beam_sep  ! Beam separation angle in degrees (full angle).
MAT Array(34,1:3)= Wave_len  ! Wave length.
MAT Array(35,1:3)= Freq_freq  ! Fringe spacing.
MAT Array(36:13)= Bgr_freq  ! Bragg frequency.
MAT Array(37,1:3)= Mix_freq  ! Mixing frequency.
MAT Array(38,1:3)= Mea_sgn  ! Sign of measured frequency in velocity equation.
MAT Array(39,1:3)= Bgr_sgn  ! Sign of bragg frequency in velocity equation.
MAT Array(40,1:3)= Mix_sgn  ! Sign of mixing frequency in velocity equation.
MAT Array(41,1:3)= Coinc  ! Coincidence criteria.
MAT Array(42,1)=Umin  ! Frequency minimum for U calculation.
MAT Array(42,2)=Vmin  ! Frequency minimum for V calculation.
MAT Array(42,3)=Wmin  ! Frequency minimum for W calculation.
MAT Array(43,1)=u_max  ! Frequency maximum for U calculation.
MAT Array(43,2)=v_max  ! Frequency maximum for V calculation.
MAT Array(43,3)=w_max  ! Frequency maximum for W calculation.
MAT Array(51,1)=reads  ! Number of desired samples.
MAT Array(52,1)=nacx  ! Number of acquired samples.
MAT Array(52,2)=atime  ! Acquisition time.
MAT Array(52,3)=ctime  ! Coincidence time.
MAT Array(51,3)=At_exp  ! Acquisition time exponent.
MAT Array(52,3)=Cc_exp  ! Coincidence time exponent.
MAT Array(51,4)=Pasix  ! Axis for plots.
MAT Array(52,4)=Clip  ! Clip: 1 turn histogram clipping on; 0 turns it off.
RETURN
Read:

This subroutine extracts (reads) the current tunnel conditions, laser parameters, and histogram

& profile scales from the Array(*).

Date=TIMEDATE  ! Date.
Mach=Array(1,2)  ! Mach Number.
Alpha(1)=Array(1,4)  ! Angle of Attack.
Time=Date  ! Time.
Temp=Array(2,2)  ! Room Temperature (deg. F).
Alpha(2)=Array(2,4)  ! Angle of Yaw.
Uedge=Array(3,2)  ! Freestream Velocity.
Alpha(3)=Array(3,4)  ! Angle of Roll.
Uexit=Array(4,2)  ! Jet exit velocity normalized by Uedge.
Theta=Array(4,4)  ! Tx Side Off Axis Angle.
MAT Mod= Array(11:14,1)  ! Frobe volume positions in MODEL coordinates.
MAT Tun= Array(11:14,2)  ! Frobe volume positions in TUNNEL coordinates.
MAT Tcs1= Array(11:14,3)  ! Rx side traverse positions in Tcs1 coordinates.
MAT Tcs2= Array(11:14,4)  ! Rx side traverse positions in Tcs2 coordinates.
MAT Index= Array(21,1:3)  ! Index of refraction of for laser light (e.g. NaI,Nglass,Nwater).
MAT Theta1= Array(22:24,1:3)  ! Angles between LASER & TUNNEL UVW laser beams in Air (N=Index1).
MAT Theta2= Array(25:27,1:3)  ! Angles between LASER & TUNNEL UVW laser beams in Water (N=Index3).
MAT Beam_spec= Array(31,1:3)  ! Beam spacing at lens.
MAT Focu_len= Array(32,1:3)  ! Focal length.
### Menu _ nonothing_ SUB Do_nothing

**Description:**

This subprogram is called when the keys on the keyboard are pressed during TCS8 traverse movements. This is done so that any STOP, PAUSE, or RESET keys will be ignored. This prevents stopping the program while the HP and TCS8 are communicating with each other. Otherwise, they might get out of sync while communicating resulting in system hang ups.

**Variables:**

- **KS** String used to flush the keyboard buffer.

### Menu _ nothing_ SUBEND

### Menu_read_ SUB Menu_read(MenuS(*))

**Description:**

This subprogram reads in the menu descriptors for each entry of the five menus.

**Variables:**

- **Menu** Used as an index to the string array MenuS(*).
- **Key** Used as an index to the string array MenuS(*).
- **MenuS(*)** String array where each element describes its corresponding menu subroutine's function.
- **L$** String use to read in the menu descriptor from the data statements.

### Menu Disp_ SUBMenu disp(Menu,Menu$(*))

**Description:**

This subprogram displays the current menu at the top of the CRT.

**Variables:**

- **Menu** Used as an index to the string array MenuS(*).
Variables:
- Parameters, graph scales, traverse positions, and coordinate system transformation matrices.

Description:
- **Array**: Array of tunnel conditions, laser parameters, graph scales, etc.
- **NameS**: Names for the variables in Array(*).
- **ImageS**: Image formats for the variables in Array(*)
- **UnitsS**: Units for the variables in Array(*)
- **X**: Used as an index to the above arrays and string arrays
- **Y**: Used as an index to the above arrays and string arrays
- **Before**: Number of digits before the decimal point in the image format
- **After**: Number of digits after the decimal point in the image format

Subroutines:
- **Enter_string**
- **Enter_value**
- **Menu_status**
- **Disp_chrs**
- **Menu**
- **Key**
- **Pen**
- **ValueS**
- **ImageS**
- **NameS**
- **UnitsS**

Sample Subroutine:
```plaintext
SUBEND
```

Main Program:
```plaintext
Variables:
- NameS, ValueS, ImageS

Description:
- This program reads in default data for each of the variable's names, values, image formats, and units. These variables include, but are not limited to, the tunnel conditions, laser parameters, graph scales, traverse positions, and coordinate system transformation matrices.
```

74
Variables:

- Array(*) Array of tunnel conditions, laser parameters, graph scales, etc.
- NameS(*) Names for the variables in Array(*).
- ImageS(*) Image formats for the variables in Array(*).
- UnitsS(*) Units for the variables in Array(*).
- X Used as in index to the above arrays and string arrays.
- Y Used as in index to the above arrays and string arrays.

13800 IF NameS(Y2,X)<>"" THEN 13830
13810 FOR X=I TO SIZE(Array,2)
13820 FOR Y2=Y1 TO SIZE(Array,1)
13830 SUBEXIT
13840 NEXT X
13850 NEXT Y
13860 PRINT USING "#,8A,A, 9A, X,3A, 6X";TRIMS(NameS(Y,X))
13870 PRINT USING "#,8A,A, 9A, X,3A, 6X";TRIMS(NameS(Y,X)),"&ImageS(Y,X)="
13880 PRINT USING "#,8A,A, 9A, X,3A, 6X";TRIMS(NameS(Y,X)),"&UnitsS(Y,X)="
13890 CASE ELSE
13900 IF ImageS(Y,X)="9D" THEN ImageS(Y,X)="9D"
13910 ON ERROR GOTO 13930
13920 PRINT USING "#,8A,A, "ImageS(Y,X)"",X,3A, 6X";TRIMS(NameS(Y,X))="",Array(Y,X),UnitsS(Y,X)
13930 GOTO 13950
13940 PRINT USING "#,8A,A, X,3A, 6X";TRIMS(NameS(Y,X)),"&Array(Y,X)="
13950 END SELECT
13960 NEXT Y
13970 PRINT IN
13980 NEXT Y
13990 SUBEND
14000 Change: START of Subprogram
14010 Change: START of Subprogram
14020 Description:
14030 This subprogram displays on the CRT the values of each of the variables with their names, image formats, and units. The user can select one of the variables and enter a new value, name, image format, or units. The user selects the particular variable by using the left, right, up, and down cursor keys. The selected variable will appear in inverse video. When it is not selected, it will appear in normal text. When the user has selected the appropriate variable he should then press the "Select" key on the keyboard. Then, depending on the value of Type$ he will be asked to enter a new value, name, image format, or units. To exit the change variables mode press the "Escape" key.
14040 There are three types of data that are passed to the subprogram. The first type of data includes, but is not limited to, the tunnel conditions, laser parameters, and graph scales. With this first type the user is allowed to enter new variable values, names, image formats, and units. The second type of data is the "Auto Move and Take" data. These data are for the programed traverse positions used in a profile scan. The third type of data is the "View and ICU" parameters data acquired from and then sent back to the TCS8.
14050 Variables:

- Array(*) Array whose values, names, image formats, or units are to be modified.
- NameS(*) Names for the variables in Array(*).
- ImageS(*) Image formats for the variables in Array(*).
- UnitsS(*) Units for the variables in Array(*).
- Type$ Indicates which type of data is to be entered.
  - Type$="VALUES" has the user enter a new value for the selected variable.
  - Type$="NAME$" has the user enter a new name for the selected variable.
  - Type$="IMAGES" has the user enter a new image format for the selected variable.
  - Type$="UNITS" has the user enter a new units for the selected variable.
- X Used as in index to the above arrays.
- Y Used as in index to the above arrays.

14290 Printer is CRT
14300 FOR Y=1 TO SIZE(Array,1)
14310 NEXT Y
14320 PRINT USING "#,8A,A, 9A, X,3A, 6X";TRIMS(NameS(Y1,X))="" THEN 14380
14330 NEXT X
14340 NEXT Y
14350 CLEAR SCREEN
14360 SUBEXIT
14370 FOR Y2=Y1 TO SIZE(Array,1)
14380 NEXT Y
14390 FOR X=1 TO SIZE(Array,2)
14400 IF NameS(Y2,X)="" THEN 14430
14410 NEXT X
14420 GOTO 14440
14430 NEXT Y2
14440 FOR Y2=2 TO SIZE(Array,1)
14450 FOR X=1 TO SIZE(Array,2)
14460 IF NameS(Y2,X)<>"" THEN 14490
14470 NEXT X
14480 NEXT Y2
14490 CLEAR SCREEN
14500 CALL Display(TypeS,Y1,Y2,Array(*),NameS(*),ImageS(*),UnitsS(*))
14520 Done=0
14530 X=1
14540 Y=Y1
14550 ON KBD ALL,15 GOSUB Kbd
14560 Wait:
14570 IF NOT Done THEN Wait
14580 OFF KBD
14590 CLEAR SCREEN
14600 NEXT Y
14610 SUBEXIT
14620 Kbd:
14630 CALL Update(TypeS,X,Y1,Y2,Done,Array(*),NameS(*),ImageS(*),UnitsS(*))
14630 RETURN
14640 SUBEND
14650 Display: SUB Display(TypeS,Y1,Y2,Array(*),NameS(*),ImageS(*),UnitsS(*))
14660 ! Description:
14670 ! This program displays on the CRT the values of each of variables with their names, image
14680 ! formats, and units.
14690 ! Variables:
14700 ! Array(*) Array whose values, names, image formats, or units are to be modified.
14710 ! NameS(*) Names for the variables in Array(*).
14720 ! ImageS(*) Image formats for the variables in Array(*).
14730 ! UnitsS(*) Units for the variables in Array(*).
14740 ! TypeS Indicates which type of data is to be entered.
14750 ! TypeS="VALUES" has the user enter a new value for the selected variable.
14760 ! TypeS="NAMES" has the user enter a new name for the selected variable.
14770 ! TypeS="IMAGES" has the user enter a new image format for the selected variable.
14780 ! TypeS="UNITS" has the user enter a new units for the selected variable.
14790 ! X Used as in index to the above arrays and string arrays.
14800 ! Y Used as in index to the above arrays and string arrays.
14810 FOR Y=Y1 TO Y2
14820 FOR X=1 TO SIZE(Array,2)
14830 CALL Select(TypeS,X,Y1,Y2,0,Array(*),NameS(*),ImageS(*),UnitsS(*))
14840 NEXT X
14850 NEXT Y
14860 CALL Select(TypeS,1,Y1,Y2,1,Array(*),NameS(*),ImageS(*),UnitsS(*))
14870 SUBEND
14880 Select: SUB Select(TypeS,X,Y1,Y2,C,Array(*),NameS(*),ImageS(*),UnitsS(*))
14890 ! Description:
14900 ! This program displays on the CRT the value of one variable along with its names, image
14910 ! format, and units.
14920 ! Variables:
14930 ! Array(*) Array whose values, names, image formats, or units are to be modified.
14940 ! NameS(*) Names for the variables in Array(*)
14950 ! ImageS(*) Image formats for the variables in Array(*)
14960 ! UnitsS(*) Units for the variables in Array(*)
14970 ! TypeS Indicates which type of data is to be entered.
14980 ! TypeS="VALUES" has the user enter a new value for the selected variable.
14990 ! TypeS="NAMES" has the user enter a new name for the selected variable.
15000 ! TypeS="IMAGES" has the user enter a new image format for the selected variable.
15010 ! TypeS="UNITS" has the user enter a new units for the selected variable.
15020 ! X Used as in index to the above arrays and string arrays.
15030 ! Y Used as in index to the above arrays and string arrays.
15040 PRINT CHR$(128+c);TABXY(26*X-24,15+Y-Y1+1);
15050 PRINT RPTS(" ",23);TABXY(26*X-24,15+Y-Y1+1);
15060 IF NameS(Y,X)="" AND Array(Y,X)=0 THEN 15260
15070 ImgS=ImageS(Y,X)
15080 UntS=UnitsS(Y1,X)
15090 IF ImgS=Y,X)="" THEN ImgS="X"
15100 IF UntS=Y,X)="" THEN UntS=""
15110 SELECT Types
15120 CASE "VALUES"
15130 SELECT NameS(Y,X)
15140 CASE "Date"
15150 CASE "Time"
15160 CASE ELSE
15170 PRINT USING ",10A,A,";NameS(Y,X)",";Array(Y,X),UntS
15180 END SELECT
15190 CASE "NAMES"
15200 PRINT USING ",10A,A,8A";NameS(Y,X)",";NameS(Y,X)
15210 CASE "UNITS"

77
SUBEND
SUBEND

DESCRIPTION:
This subroutine scrolls through the variables displayed on the CRT and has the user enter updated values. The user can select one of the variables and enter a new value, name, image, format, or units. The user selects the particular variable by using the left, right, up, down cursor keys. This subroutine will only have been called after a keyboard key has been pressed. If a cursor key has been pressed then the previously selected variable will be redisplayed in normal text and the new selected variable will appear in inverse video text. When the user has selected the appropriate variable he will have pressed the "Select" key on the keyboard. Then, depending on the value of the Type$ he will be asked to enter a new value, name, image format, or units. To exit the change variables mode the user will have pressed the "Escape" key.

VARIABLES:
- Array(*) Array of tunnel conditions, laser parameters, graph scales, etc.
- NameS(*) Names for the variables in Array(*).
- ImageS(*) Image formats for the variables in Array(*).
- UnitsS(*) Units for the variables in Array(*).
- TypeS Indicates which type of data is to be entered.
- TypeS="VALUES" has the user enter a new value for the selected variable.
- TypeS="NAMES" has the user enter a new name for the selected variable.
- TypeS="IMAGES" has the user enter a new image format for the selected variable.
- TypeS="UNITS" has the user enter new units for the selected variable.
- X Used as in index to the above arrays and string arrays.
- Y Used as in index to the above arrays and string arrays.

DISABLE K$=KB$S
IF KS="" THEN 15990
SELECT NUM(K$[1,1])
CASE 27 ! ESC
Done=1
CASE 255
CALL Select(TypeS,X,Y,Y1,Y2,0,Array(*),NameS(*),ImageS(*),UnitsS(*))
CASE 27,80 ! Break,Stop
PAUSE CASE 124 ! Menu
Done=1
CASE 38 ! Select
CALL Select(TypeS,X,Y,Y1,Y2,1,Array(*),NameS(*),ImageS(*),UnitsS(*))
SELECT TypeS
CASE "VALUES"
IF NameS(Y,X)="" THEN CALL Enter_string("Name for "&NameS(Y,X),"X")
IF ImageS(Y,X)="" THEN CALL Enter_string("Image for "&NameS(Y,X),"X")
CALL Enter_name(NameS(Y,X),Array(Y,X),ImageS(Y,X))
CASE "NAMES"
CALL Enter_string("Name for "&NameS(Y,X),"X")
CASE "UNITS"
CALL Enter_string("Units for "&NameS(Y,X),"X")
CASE "IMAGES"
CALL Enter_string("Image for "&NameS(Y,X),"X")
END SELECT
CASE Select(TypeS,X,Y,Y1,Y2,0,Array(*),NameS(*),ImageS(*),UnitsS(*))
IF X=SIZE(Array,2) THEN Y=Y+1
X=X+1 ! Left
X=X+1 ! Right
X=X+1 ! Up
Y=Y-1 ! Down
Y=Y+1 ! First
X=1
Y=1
END SELECT
X=(X-1) MOD SIZE(Array,2)+1
Y=(Y-Y1+1) MOD (Y2-Y1+1)+Y1
IF X<1 THEN X=SIZE(Array,2)
IF Y<1 THEN Y=Y2
CALL Select(TypeS,X,Y,Y1,Y2,1,Array(*),NameS(*),ImageS(*),UnitsS(*))
END SELECT
ENABLE
SUBEXIT
SUBEND
This subroutine converts a single real precision variable into two 16 bit words. The initial real precision variable is converted to a 32 bit integer and then separated into high and low 16 bit integers. The most significant 16 bits will be in the "High" variable while the least significant 16 bits will be placed in the "Low" variable. The main purpose of this subroutine is to provide a means to send a 32 bit integer to the LVDAS over the 16 bit high speed interface.

Variables:
- Real Initial real precision value for the variable.
- Hex+ Hex value of "Real". String length will be 8 bytes for 32 bits.
- High Most significant 16 bits of integerized "Real".
- Low Least significant 16 bits of integerized "Real".

Example:
- Hex+$=DVALS(Real,16)
- High=IVAL(Hex$,1,16)
- Low=IVAL(Hex$,5,16)

This subprogram is used to create a lookup table array. The lookup table array facilitates the rapid conversion of raw encoded Macrodyne data into a usable frequency. Once the table has been filled, the raw Macrodyne data can be used as an index to the table array.

Variables:
- Table(*) Lookup table of frequencies.
- Mantissa(*) The 10 bit mantissa part of the raw Macrodyne data (0..1023).
- Exponent The 4 bit Exponent part of the raw Macrodyne data.
- Time(*) An array of measurement times for a given number of Fringes and Exponent.
- Freq(*) An array of measured frequencies for a given number of Fringes and Exponent.

Example:
- REAL Mantissa(0:1023), Time(0:1023), Freq(0:1023)
- If last entry in the table is not zero then the table has already been created.

IF Table(32766) THEN SUBEXIT
- FOR Bin=0 TO 1023
  - Fill Mantissa array.
- NEXT Bin
- NEXT Bin
- Min=0
- FOR Fringes=0 TO 1
  - 0 indicates 16 fringes while 1 indicates 8 fringes.

Example:
- MAX=MIN^1023
- IF MAX=32767 THEN Maximum size of an array is 32766.
- MAX=32766
- REDIM Mantissa(0:1022), Time(0:1022), Freq(0:1022)
- END IF
- DISP Fringes, Exponent
- IF Max Time= Mantissa*(2*(Exponent-1))/500000000 THEN Use this line with new macrodynes.
- MAT Time= Mantissa*(2*(Exponent-3))/500000000 THEN Use this line with old macrodynes.
The raw data from array $I_i$ are converted into interarrival times by multiplying their values by $2^{At_{exp}}$ over 10 to get us. The raw data from array $C_i$ are converted into voltages by multiplying their initial values by $2^Ct_{exp}$.

The 1/0 path has a select code of 12 and is initialized to perform unformatted word transfers without any end of line designations. The DIP switches on the HP98622-66501 Rev B transfers are initially set to one by the LVDAS.

The LVDAS data reduce subroutine samples the two analog, three digital, and two external trigger channels from the LVDAS. The HP sends a "CS" to sample the LVDAS data with coincidence. Following the "CS" the HP sends the LVDAS an additional eight words to specify the acquisition and coincidence times, the interarrival and coincidence time exponents, the coincidence mask, and the number of desired samples. After the desired number of samples is acquired or the desired acquisition time expires then the LVDAS sends to the HP an updated number of samples ($N_{sam}$). The updated $N_{sam}$ may be less than the original $N_{sam}$ if the desired acquisition time expires before the desired $N_{sam}$ samples are realized.

### Variables:
- $At_t$ The maximum desired acquisition time (seconds).
- $Ct_t$ The maximum desired coincidence time (seconds).
- $At_1$ The upper word of integer of 10000000*Atime.
- $At_2$ The lower word of integer of 10000000*Atime.
- $Ct_1$ The upper word of integer of 10000000*Ct ime.
- $Ct_2$ The lower word of integer of 10000000*Ct ime.
- $At_{exp}$ Exponent for interarrival times.
- $Ct_{exp}$ Exponent for coincidence times.
- $N_{sam}$ Number of desired samples.
- $Cmask$ Coincidence Mask for $U, V, W$ selection.
- $Raw(*)$ Array of raw data acquired LVDAS data.
- $Data$ Array of raw data acquired LVDAS data.

### Description:
This subroutine is used to initialize the HP98622-66501 Rev B 16-bit General Purpose Input Output (GPIB) interface. The subroutine also opens the LVDAS path on the HP computer for command and data transfers. The 1/0 path is given the name "$\text{@Lvdas}$". Data transferred from the HP to the LVDAS will use the "OUTPUT $\text{@Lvdas}$" statement. Data transferred to the HP from LVDAS will use the "ENTER $\text{@Lvdas}$" statement.

If $N_{sam}$=0 THEN SUBEXIT

Call Convert2words($Ctime*10000000^{At_{exp}}$,$Ct2$)

### Output:
- @Lvdas USING "#,#,AAEOL"
- OUT $\text{@Lvdas}$ USING "$W$";Nonam
- IF Nonam=0 THEN SUBEXIT
- REDIM Raw(1:Nonam,1:10)
- ENTER @Lvdas USING "$W$";Raw(*)

### Description:
This subroutine does the ten by Nsam $Valld(*)$ data arrays and columns 7,8 of the Raw data array. If I'th sample acquired contains valid data, then Valid(i) will be equal to one, and zero otherwise. All values for the Valid array are initially set to one by the LVDAS.

The raw data from arrays $U$, $V$, $W$ are converted into frequencies by using their initial values as indexes to the frequency look up table array $Table(*)$. The raw data from arrays $A$, $B$, $I$ are converted into voltages by multiplying their initial values by 5 volts over 2^15. The raw data from array $I_1$ are converted into interarrival times by multiplying their initial values by $2^{Ct_{exp}}$ over 10 to get us. The raw data from array $C_1$ are converted into coincidence times by multiplying their initial values by $2^{Ct_{exp}}$ over 10 to get us.

### Variables:
- $Table(*)$ Lookup table of frequencies.
- $Raw(*)$ Array of raw data acquired LVDAS data.
- $UI(*)$ Array of extracted raw $U$ frequency data.
- $VI(*)$ Array of extracted raw $V$ frequency data.
DATA CLIP

DESCRIPTION:
This subprogram compares each of the instantaneous U, V, and W frequencies with user selectable minimum and maximum frequencies. If the instantaneous value is less than the desired minimum, then the validation word is set to zero. Also, if the instantaneous value is greater than the desired maximum, then the validation word is set to zero. The setting of the validation words to zero will have the net effect of discarding the data samples from the data set. In other words, the data is weighted as zero for the average, stdv, normal and shear stress calculations.

VARIABLES:
Nsam Number of samples acquired.
Ui(*) Array of instantaneous U frequencies (MHz).
Vi(*) Array of instantaneous V frequencies (MHz).
Wi(*) Array of instantaneous W frequencies (MHz).
Valid(*) Array of sample validation words.

OPTION BASE 1

SUBEND
DESCRIPTION:

This subprogram takes the frequency values from the arrays U1, V1, W1 and replaces them with velocities after doing the frequency to velocity conversion.

Variables:

- Array(*) An array containing relevant LDV laser and tunnel condition parameters
- Frng_spc(*) Fringe Spacings extracted from Array(*).
- Brг_frq(*) Bragg Frequencies extracted from Array(*)
- Mix_frq(*) Mixing Frequencies extracted from Array(*)
- Mea_sgn(*) Measured Frequencies. Signs extracted from Array(*).
- Mix_sgn(*) Mixing Signs. Signs extracted from Array(*).
- Mix_frq(*) Measured Frequencies. Signs extracted from Array(*).
- Mix_sgn(*) Mixing Frequencies. Signs extracted from Array(*).
- U(*) Array of instantaneous U data.
- V(*) Array of instantaneous V data.
- W(*) Array of instantaneous W data.

Equations:

The following equations are used to convert the frequencies to velocities:

\[
\text{Velocity} = \text{Ftotal} = \text{MeaSgn} \times \text{Fmix} + \text{MeaSgn} \times \text{MixFrq}
\]

OPTION BASE 1

DATA/ REAL (*), INTEGER (*), TABLE (*), DATA (*), TEXT (*), EXEC (*), SUB EXEC (*), NAME (*), LENGTH (*)

DIM Frng_spc(3), Brг_frq(3), Mix_frq(3), Mea_sgn(3), Mix_sgn(3)

DISP "Converting Data"

MAT Frng_spc= Array(35,1:3)

MAT Brг_frq= Array(36,1:3)

MAT Mix_frq= Array(37,1:3)

MAT Mea_sgn= Array(38,1:3)

MAT Mix_sgn= Array(39,1:3)

MAT Mix_sgn= Array(40,1:3)

MAT U1= U1*(Mea_sgn(1))

MAT V1= V1*(Mea_sgn(2))

MAT W1= W1*(Mea_sgn(3))

MAT U1= U1*(Mix_sgn(1))*Mix_sgn(2)*Mix_sgn(3)

MAT V1= V1*(Mix_sgn(2))*Mix_sgn(3)

MAT W1= W1*(Mix_sgn(3))

MAT U1= U1*(Frng_spc(1))

MAT V1= V1*(Frng_spc(2))

MAT W1= W1*(Frng_spc(3))

DESCRIPTION:

This subprogram performs the summations on the instantaneous LDV and analog data. Data will be weighted as zero in the summations if the value of the validation word is set to zero. Intermediate arrays will be made so that summations of the products of the LDV and analog data can be determined.

Variables:

- Nsam(*) Number of samples acquired.
- Valid(*) Array of sample validation words.
- U(*) Array of instantaneous U frequency or velocity samples.
- V(*) Array of instantaneous V frequency or velocity samples.
- W(*) Array of instantaneous W frequency or velocity samples.
- A(*) Array of instantaneous A point velocity samples.
- B(*) Array of instantaneous B point velocity samples.
- C(*) Array of coincidence times.

Data summation subprogram takes the instantaneous velocity samples U, V, W, A and B and provides the sum of the instantaneous velocity products as well as their average.

SUBEND

782
OPTION BASE 1

COM /Data/ INTEGER Raw(*),Valid(*),REAL Table(*),Ui(*),Vi(*),
Wi(*),Ai(*),Bi(*),li(*),Ci(*)

REAL Uu(1000),Vv(1000),Ww(1000),Aa(1000),Bb(1000),I2(1000),C2(1000)

REDIM Uu(Nsam),Vv(Nsam),Ww(Nsam),Aa(Nsam),Bb(Nsam),I2(Nsam),C2(Nsam)

REAL Uv(1000),Vw(1000),Wu(1000),Ab(1000),Va(1000),Wa(1000)

REDIM Uv(Nsam),Vw(Nsam),Wu(Nsam),Ab(Nsam),Va(Nsam),Wa(Nsam)

DISP "Summing Data"

! Description:
! This subprogram uses the summations on the instantaneous LDV and analog data as well as the
Variables:

Nsam  The number of valid samples.
Sum(1,1) Summation of the array UI.
Sum(2,1) Summation of the array Vi.
Sum(3,1) Summation of the array Wi.
Sum(4,1) Summation of the array Ai.
Sum(5,1) Summation of the array Bi.
Sum(6,1) Summation of the array Ii.
Sum(7,1) Summation of the array Ci.
Sum(1,2) Summation of the array Uu.
Sum(2,2) Summation of the array Vv.
Sum(3,2) Summation of the array Ww.
Sum(4,2) Summation of the array Aa.
Sum(5,2) Summation of the array Bb.
Sum(6,2) Summation of the array Ii.
Sum(7,2) Summation of the array Cc.
Sum(1,3) Summation of the array Vv.
Sum(2,3) Summation of the array Ww.
Sum(3,3) Summation of the array Uu.
Sum(4,3) Summation of the array Ab.
Sum(5,3) Summation of the array Bb.
Sum(6,3) Summation of the array Vv.
Sum(7,3) Summation of the array Cc.

U  Average U frequency or velocity.
V  Average V frequency or velocity.
W  Average W frequency or velocity.
A  Average A voltage.
I  Average interarrival time.
C  Average coincidence time.
U1  Standard deviation for U frequency or velocity.
V1  Standard deviation for V frequency or velocity.
W1  Standard deviation for W frequency or velocity.
A1  Standard deviation for A voltage.
B1  Standard deviation for B voltage.
I1  Standard deviation for interarrival time.
Cl  Standard deviation for coincidence time.
Ulal  Velocity:Velocity Shear Stress.
Vlal  Velocity:Velocity Shear Stress.
Albl  Voltage:Voltage Shear Stress.
Wlul  Velocity:Velocity Shear Stress.
Vlwl  Velocity:Velocity Shear Stress.
Ulvl  Velocity:Velocity Shear Stress.
Wlal  Velocity:Velocity Shear Stress.
Vlal  Velocity:Velocity Shear Stress.
Ulal  Velocity:Velocity Shear Stress.

DISP "Calculating Results"
Nsam=N(1,1)
IF Nsam>0 THEN
U=Sum(1,1)/Nsam
V=Sum(2,1)/Nsam
W=Sum(3,1)/Nsam
A=Sum(4,1)/Nsam
B=Sum(5,1)/Nsam
I=Sum(6,1)/Nsam
C=Sum(7,1)/Nsam
Ul=SQRT(SUM(SUM((SUM(1,1)/Nsam-U)*U))
Vl=SQRT(SUM(SUM((SUM(2,1)/Nsam-V)*V))
Wl=SQRT(SUM(SUM((SUM(3,1)/Nsam-W)*W))
A1=SQRT(SUM(SUM((SUM(4,2)/Nsam-A)*A))
Bl=SQRT(SUM(SUM((SUM(5,2)/Nsam-B)*B))
I1=SQRT(SUM(SUM((SUM(6,2)/Nsam-I)*I))
Cl=SQRT(SUM(SUM((SUM(7,2)/Nsam-C)*C))
Ulal=Sum(1,1)/Nsam-U*V
Vlal=Sum(2,1)/Nsam-V*W
Albl=Sum(3,1)/Nsam-W*U
Wlul=Sum(4,2)/Nsam-U*A
Vlwl=Sum(5,2)/Nsam-A*B
Ulvl=Sum(6,2)/Nsam-A*B
Wlal=Sum(7,3)/Nsam-A*B

ELSE
U=0
V=0
W=0
A=0
B=0
I=0
C=0
Ul=0
Vl=0

84
20800    WI=0
20810    A1=0
20820    B1=0
20830    T1=0
20840    C1=0
20850    Ulv1=0
20860    Vlv1=0
20870    Wlv1=0
20880    A1bl=0
20890    Ulal=0
20900    Vlal=0
20910    Wlal=0
20920    END IF
20930 SUBEND
20940 Data_trnsfrm: SUB Data_trnsfrm(REAL R(*),U,V,W, Ul,Vl,Wl,Ulv1,Vlv1,Wlv1, Ulal,Vlal,Wlal)
20950 ! Description:
20960 ! This subprogram performs a coordinate system transformation on the averages, standard
20970 ! deviations, and shear stresses. The coordinate system transformation to be applied is passed
20980 ! through the "K3X3" array. If a LASER to TUNNEL coordinate system transformation is to be
20990 ! performed, then the array "Ldv2tun" array will be passed to the "K3X3" array. If a TUNNEL to
21000 ! MODEL coordinate system transformation is to be performed, then the array "Tun2mod" array will
21010 ! be passed to the "K3X3" array.
21020 ! Variables:
21030 ! U  Average U velocity.
21040 ! V  Average V velocity.
21050 ! W  Average W velocity.
21060 ! Ul Standard deviation for U velocity.
21070 ! Vl Standard deviation for V velocity.
21080 ! Wl Standard deviation for W velocity.
21090 ! Ulul Velocity:Velocity Normal Stress.
21100 ! Vlvl Velocity:Velocity Shear Stress.
21110 ! Wlwl Velocity:Velocity Shear Stress.
21120 ! Ulul Velocity:Velocity Normal Stress.
21130 ! Vlul Velocity:Velocity Shear Stress.
21140 ! Wlul Velocity:Velocity Shear Stress.
21150 ! Ulvl Velocity:Velocity Shear Stress.
21160 ! Vlvl Velocity:Velocity Shear Stress.
21170 ! Wlvl Velocity:Velocity Shear Stress.
21180 ! Ulwl Velocity:Velocity Shear Stress.
21190 ! Vlwl Velocity:Velocity Shear Stress.
21200 ! Wlwl Velocity:Velocity Shear Stress.
21210 ! R(*) Original U,V,W.
21220 ! F(*) Original Ulal,Vlal,Wlal.
21230 ! P(*) Original stress terms Ulul,Ulvl,....Wlwl.
21240 ! K3X3 Coordinate system transformation matrix for average and velocity:velocity shear stress
21250 ! conversions.
21260 ! K9X9 Coordinate system transformation matrix for velocity:velocity normal and shear stress
21270 ! conversions.
21280 ! S(*) Transformed U,V,W.
21290 ! H(*) Transformed Ulal,Vlal,Wlal.
21300 ! Q(*) Transformed stress terms Ulul,Ulvl,....Wlwl.
21310 OPTION BASE 1
21320, REAL R(3),S(3),F(3),H(3),P(9),Q(9),K3x3(3,3),K9x9(9,9)
21330 DISP "Transforming Results"
21340 ! Calculate Ulul,Vlul,Wlul using Ul,Vl,Wl.
21350 Ulul=U1*U1
21360 Vlul=V1*V1
21370 Wlul=W1*W1
21380 ! Set Ul1,Vl1,Wl1 equal to Ulul,Vlul,Wlul.
21390 Ul1=Ulul
21400 Vl1=Vlul
21410 Wl1=Wlul
21420 ! Fill the matrix R with U,V,W.
21430 R(1)=U
21440 R(2)=V
21450 R(3)=W
21460 ! Fill the matrix F with Ulal,Vlal,Wlal.
21470 F(1)=Ulal
21480 F(2)=Vlal
21490 F(3)=Wlal
21500 ! Fill the matrix P with Ulul,Ulvl,Wlul,Ulv1,Vlv1,Wlv1.
21510 P(1)=Ulul
21520 P(2)=Ulv1
21530 P(3)=Ul1
21540 P(4)=Vlul
21550 P(5)=Vlv1
21560 P(6)=Wl1
21570 P(7)=Ulal
21580 P(8)=Vlal
21590 P(9)=Wlal

85
! Define the matrix K9x9 using products of the elements from then matrix K3x3.
21600 FOR Y=1 TO 9
21610 FOR X=1 TO 9
21620 Y1=((Y-I) DIV 3)+1
21630 X1=((X-I) DIV 3)+1
21640 Y2=((Y-I) MOD 3)+1
21650 X2=((X-I) MOD 3)+1
21660 K9x9(Y,X)=K3x3(Y1,X1)*K3x3(Y2,X2)
21670 NEXT Y
21680 NEXT X
21690 X2=((X-I) MOD 3)+1
21700 XI=((X-I) DIV 3)+1
21710 Y1=((Y-I) DIV 3)+1
21720 ! Extract the transformed U,V,W from the matrix S.
21730 U=S(1)
21740 V=S(2)
21750 W=S(3)
21760 ! Extract the transformed Ulal,Vlal,Wlal from the matrix H.
21770 U1al=H(1)
21780 Vlal=H(2)
21790 Wlal=H(3)
21800 ! Extract the transformed Ulul,Uvl,v,Wlul,Wvl,v,Wlul,tlal1 from the matrix Q.
21810 Ulul=q(1)
21820 Uvl,v=q(2)
21830 Vlal=q(3)
21840 Vlul=q(4)
21850 Vlul=q(5)
21860 Vlal=q(6)
21870 Wlul=q(7)
21880 Wlul=q(8)
21890 Wlul=q(9)
21900 ! Calculate U1,V1,W1 using Ulul,Vlul,Wlul.
21910 U1=SQR(ABS(Ulul))
21920 V1=SQR(ABS(Vlul))
21930 W1=SQR(ABS(Wlul))
21940 ! Return transformed U,V,W,U1,V1,W1,U1al,Vlal,Wlal to main program.
21950 SUBEND
22000 Print:
22010 ! This subprogram prints the averages, standard deviations, and shear & normal stress in tabular form. This subprogram may be called several times. The first call might print the reduced velocity data when their units are in frequency (MHz). Subsequent calls will print the reduced data when their units are in velocity (m/s). These subsequent calls will print the data in one of three coordinate systems: LASER, TUNNEL, and MODEL.
22020 ! Description:
22030 ! This subprogram prints the averages, standard deviations, and shear & normal stress in tabular form. This subprogram may be called several times. The first call might print the reduced velocity data when their units are in frequency (MHz). Subsequent calls will print the reduced data when their units are in velocity (m/s). These subsequent calls will print the data in one of three coordinate systems: LASER, TUNNEL, and MODEL.
22040 ! Variables:
22050 ! U Average U velocity.
22060 ! V Average V velocity.
22070 ! W Average W velocity.
22080 ! A Average A velocity.
22090 ! B Average B voltage.
22100 ! C Average coincidence time.
22110 ! Ul Standard deviation for U velocity.
22120 ! Vl Standard deviation for V velocity.
22130 ! Wl Standard deviation for W velocity.
22140 ! Al Standard deviation for A velocity.
22150 ! Bl Standard deviation for B voltage.
22160 ! Il Standard deviation for interarrival time.
22170 ! Cl Standard deviation for coincidence time.
22180 ! Ulul Velocity:Velocity Shear Stress.
22190 ! Vlul Velocity:Velocity Shear Stress.
22200 ! Wlul Velocity:Velocity Shear Stress.
22210 ! Alul Voltage:Voltage Shear Stress.
22220 ! Ulal Voltage:Voltage Shear Stress.
22230 ! Vlal Voltage:Voltage Shear Stress.
22240 ! Wlal Voltage:Voltage Shear Stress.
22250 ! Axia$ Indicates one of the three axes X,Y,Z being traversed.
22260 ! Pos Current Traverse Position.
22270 ! Nsam Number of samples acquired.
22280 ! CS Indicates units and/or coordinate system of data printed.
22290 ! DISP "Printing Results"
22300 ON ERROR CALL Error
22310 ! PRINTER IS FPT;WIDTH 144
22320 ! Call Error Subroutines.
22330 PRINT USING 22490:15,Pos,U1,UL1
86
![Raw Text](image-url)
for command and data transfer. The I/O path is given the name "$Tcs8". Data transferred
from the HP to the TCS8 will use the "OUTPUT $Tcs8 statement. Data transferred to the HP
from TCS8 will use the "ENTER $Tcs8 statement.
The I/O path has a select code of 9 and is initialized to perform unformatted byte
transfers without any end of line designations.

ASSIGN @Tcs8 TO 9;BYTE,FORMAT OFF,EOL ""

CONTROL 9,0;3i Reset interface.
CONTROL 9,3;9600 i Select a baud rate of 9600.
CONTROL 9,4;31 i Select even parity, enable parity, 2 stop bits, 8 bits per character.
CONTROL 9,12;IVAL("EF",16) i Enable Carrier Detect. Disable Data Set Ready. Disable Clear To Send.
CONTROL 9,13;9600 i Default baud rate of 9600.
CONTROL 9,14;31 i Default character format: Even parity enabled, 2 stop, 8 bits/char.

SELECT Commands
SUBEND

DIM View(8,1),Set(8,2),Name$(8,1),Image$(8,1),Units$(8,1)

OUTPUT @Tcs8 USING "K,USING "K, View(*) ! Tell the TCS8 we want to View data.
ENTER @Tcs8 USING "K, View(*) ! Enter the parameter specified by Command$.

! Initialize the Name$, Image$, Units$ and Set arrays.
READ Name$(*)
MAT Image$="6D.3D"

DATA X1,X2,Y1,Y2,21,22,A1,A2
FOR Channel=1 TO 8
Set(Channel,1)=Channel
SELECT Command$
CASE "P" ! Command$="P" indicates we want to view the encoder Positions in inches.
Name$(Channel,1)=Name$(Channel,1) & " (pos)"
Units$(Channel,1)="in"
CASE "U" ! Command$="U" indicates we want to view the Units in counts per inch.
Name$(Channel,1)=Name$(Channel,1) & " (cnt)"
Units$(Channel,1)="cnt"
CASE "V" ! Command$="V" indicates we want to view the Velocity in revolution per second.
Name$(Channel,1)=Name$(Channel,1) & " (vel)"
Units$(Channel,1)="rev/s"
CASE "A" ! Command$="A" indicates we want to view the Acceleration in revolution per second^2.
Name$(Channel,1)=Name$(Channel,1) & " (acc)"
Units$(Channel,1)="rev/s^2"
CASE "C" ! Command$="C" indicates we want to view the current - direction limit switches.
Name$(Channel,1)=Name$(Channel,1) & " (+LS)"
Units$(Channel,1)="
CASE "-" ! Command$="-" indicates we want to view the current - direction limit switches.
Name$(Channel,1)=Name$(Channel,1) & " (-LS)"
Units$(Channel,1)="
CASE "S" ! Command$="S" indicates we want to view the current motor Stall indication status.
Name$(Channel,1)=Name$(Channel,1) & " (STALL)"
Units$(Channel,1)="
CASE "H" ! Command$="H" indicates we want to view the current Home limit switches.
Name$(Channel,1)=Name$(Channel,1) & " (HS)"
Units$(Channel,1)="
END SELECT
NEXT Channel
CALL Change("VALUES",View(*),Name$(*) ,Image$(*),Units$(*) )
The "Set" parameters command is now sent to the TCS8
SELECT Command$
CASE "P","U","R","V","A"
MAT Set(*)=View(*)
OUTPUT @Tcs8 USING 23940; "S"&Command$, Set(*)
IMAGE K,6,0,"",M6D.4D,"," /
END SELECT

SUBEND

Tcs8read: SUB Tcs8read(@Tcs8,Mod(*),Tun(*),Tcs1(*),Tcs2(*),Tcs2tun1(*),Tcs2tun2(*),Tun2mod(*))

Description:
This subprogram reads the current TCS8 positions. The 8 positions are read in TCS coordinates with the units being in inches. Four of the eight positions (X1,Y1,Z1,A1) which are the transmitting side traverse positions are entered into the Tcs1 array. The other four positions (X2,Y2,Z2,A2) which are the receiving side traverse positions are entered into the Tcs2 array. The Tcs1 & Tcs2 arrays are converted from TCS to TUNNEL to MODEL coordinates. The current updated positions in the three coordinate systems are printed on the top of the CRT. They are also returned to the main program.

Variables:

Tcs1(*) TCS8 transmitting side traverse positions (X1,Y1,Z1,A1) in TCS coordinates.
Tcs2(*) TCS8 receiving side traverse positions (X2,Y2,Z2,A2) in TCS coordinates.
Tun(*) Traverse positions (X,Y,Z) in TUNNEL coordinates.
Mod(*) Traverse positions (X,Y,Z) in MODEL coordinates.
Tcs2tun1(*) Coordinate system transformation matrix for converting Tcs1(*) to Tun(*).
Tcs2tun2(*) Coordinate system transformation matrix for converting Tcs2(*) to Tun(*).
Tun2mod(*) Coordinate system transformation matrix for converting Tun(*) to Mod(*).

Output:

Get Tcs8 using "#(K),Tcs1(1),Tcs2(1),Tcs2(2),Tcs1(3),Tcs2(3),Tcs1(4),Tcs2(4)

MAT Tun=Tcs2tun1*Tcs1
REDIM Tun(1:13),Mod(1:13)
MAT Mod=Tun2mod*Tun
REDIM Tun(1:14),Mod(1:14)
Mod(4)=0
Tun(4)=0
CALL Tcs8print (Mod(*),Tun(*),Tcs1(*),Tcs2(*))

Description:
This subprogram allows for the movement of the probe volume and collecting optics in one of the three coordinate systems. The three coordinate systems implemented are the TSC, TUNNEL and MODEL coordinate systems. Two movements modes are available. The first movement mode makes movements relative to the current position. The second movement mode makes movements to an absolute fixed position. Both the transmitting side and receiving side traverse are moved in tandem or separately.

Variables:

Tcs1(*) TCS8 transmitting side traverse positions (X1,Y1,Z1,A1) in TCS coordinates.
Tcs2(*) TCS8 receiving side traverse positions (X2,Y2,Z2,A2) in TCS coordinates.
Tun(*) Traverse positions (X,Y,Z) in TUNNEL coordinates.
Mod(*) Traverse positions (X,Y,Z) in MODEL coordinates.
Tcs2tun1(*) Coordinate system transformation matrix for converting Tcs1(*) to Tun(*).
Tcs2tun2(*) Coordinate system transformation matrix for converting Tcs2(*) to Tun(*).
Tun2mod(*) Coordinate system transformation matrix for converting Tun(*) to Mod(*).

Description:
This subprogram allows for the movement of the probe volume and collecting optics in one of the three coordinate systems. The three coordinate systems implemented are the TCS, TUNNEL and MODEL coordinate systems. Two movements modes are available. The first movement mode makes movements relative to the current position. The second movement mode makes movements to an absolute fixed position. Both the transmitting side and receiving side traverse are moved in tandem or separately.

Variables:

Tcs1(*) TCS8 transmitting side traverse positions (X1,Y1,Z1,A1) in TCS coordinates.
Tcs2(*) TCS8 receiving side traverse positions (X2,Y2,Z2,A2) in TCS coordinates.
Tun(*) Traverse positions (X,Y,Z) in TUNNEL coordinates.
Mod(*) Traverse positions (X,Y,Z) in MODEL coordinates.
Tcs2tun1(*) Coordinate system transformation matrix for converting Tcs1(*) to Tun(*).
Tcs2tun2(*) Coordinate system transformation matrix for converting Tcs2(*) to Tun(*).
Tun2mod(*) Coordinate system transformation matrix for converting Tun(*) to Mod(*).

SideS Indicates which sides are to be moved:

Tx : Transmitting side only.
Rx : Receiving side only.
Tx & Rx : Both sides together.

Coors Indicates which coordinate system the movement is to be made:

MODEL : MODEL coordinates.
TUNNEL : TUNNEL coordinates.
TCS : TCS coordinates.

ModeS Indicates which movement mode is to be completed:

RELATIVE: Movements are relative to current positions.
ABSOLUTE: Movements are absolute positions.

K Indicates which axis of the four axes is to be moved.

Movement Indicates the desired movement for the selected axis.

C(*) Array of viewed TCS8 "Currents On" parameters.

C(*) Array of viewed TCS8 "Currents On" parameters.
OPTION BASE 1
DIM LS[100]
REAL Move(R(2),I(8),C(8))
IF SUM(I)=8 THEN OUTPUT @Tcs8 USING "K/","V10"
ENTER @Tcs8 USING "X(K)","(*)"
IF SUM(I)<8 THEN OUTPUT @Tcs8 USING "K/","S10"
EXIT
OPTION BASE 1
Thetal(*) Laser beam angles outside the water tunnel.

Move(6)=Move(4)
Move(3)=Move(2,2)=Tcs2(1)
Move(Channel,l)=Channel
FOR Channel=l TO 8
! File the move array.
END SELECT
FOR Channel=1 TO 8
Move(Channel,1)=Channel
NEXT Channel
Move(1,2)=Tcs2(1)
Move(2,2)=Tcs2(1)
Move(3,2)=Tcs2(1)
Move(4,2)=Tcs2(2)
Move(5,2)=Tcs2(3)
Move(6,2)=Tcs2(3)
Move(7,2)=Tcs2(4)
Move(8,2)=Tcs2(4)
! Initiate the start of the move.
IF Mode$="ABSOLUTE" THEN OUTPUT @Tcs8 USING 25280;"MA",Move(*)
IF Mode$="RELATIVE" THEN OUTPUT @Tcs8 USING 25280;"MR",Move(*)
IMAGE K,B(D) (0)
END IF
END IF
END IF
! If the movement mode is to be RELATIVE, then clear all of the previously read positions.
! Index(2) : Index of refraction for Glass.
! Index(3) : Index of refraction for Water.

Ctm Idv: SUB Ctm Idv(Theta(*),Tun2ldv(*),Ldv2tun(*))
END IF
Ctm Idv: SUB Ctm Idv(Theta(*),Tun2ldv(*),Ldv2tun(*))
25580 ! matrix "Tun21dv(*)". However, the desired coordinate system transformation matrix "Ldv2tun(*)" is 25590 ! required. It is the matrix inverse of "Tun21dv(*)".
25600 ! Variables:
25610 ! Theta(*) Laser beam angles inside the water tunnel.
25620 ! Tun21dv(*) Coordinate system transformation matrix for converting from TUNNEL to LASER.
25630 ! Ldv2tun(*) Coordinate system transformation matrix for converting from LASER to TUNNEL.
25640 OPTION BASE 1
25650 ! Tun21dv converts TUNNEL coordinates to LASER coordinates.
25660 Tun21dv(1,1)=COS(Theta(1,1))
25670 Tun21dv(1,2)=COS(Theta(1,2))
25680 Tun21dv(1,3)=COS(Theta(1,3))
25690 Tun21dv(2,1)=COS(Theta(2,1))
25700 Tun21dv(2,2)=COS(Theta(2,2))
25710 Tun21dv(2,3)=COS(Theta(2,3))
25720 Tun21dv(3,1)=COS(Theta(3,1))
25730 Tun21dv(3,2)=COS(Theta(3,2))
25740 Tun21dv(3,3)=COS(Theta(3,3))
25750 ! Ldv2tun converts LASER coordinates to TUNNEL coordinates.
25760 MAT Ldv2tun= INV(Tun21dv)
25770 SUBEND
25780 Ctm_mod: SUB Ctm_mod(Angle(*),Mod2tun(*),Tun2mod(*))
25790 ! Description:
25800 ! This subprogram computes directly the MODEL to TUNNEL coordinate system transformation matrix "Mod2tun(*)". However, the desired coordinate system transformation matrix "Tun2mod(*)" is required. It is the matrix inverse of "Mod2tun(*)".
25820 ! Variables:
25830 ! Angle(*) Coordinate system transformation matrix for converting from MODEL to TUNNEL.
25840 ! T1(*) Partial coordinate system transformation matrix for converting from MODEL to TUNNEL coordinates. Takes into account a model at an angle of yaw.
25850 ! T2(*) Partial coordinate system transformation matrix for converting from MODEL to TUNNEL coordinates. Takes into account a model at an angle of roll.
25860 ! T3(*) Partial coordinate system transformation matrix for converting from MODEL to TUNNEL coordinates. Takes into account a model at an angle of attack.
25870 ! Mod2tun(*) Coordinate system transformation matrix for converting from MODEL to TUNNEL.
25880 ! Tun2mod(*) Coordinate system transformation matrix for converting from TUNNEL to MODEL.
25890 OPTION BASE 1
25900 REAL T1(3,3),T2(3,3),T3(3,3),Temp(3,3)
25910 ! Define 1st coordinate transformation matrix for Mod2tun.
25920 ! Rotation in the x-y plane about the r-axis.
25930 ! Used when model is at an angle of attack.
25940 T1(1,1)=COS(Angle(1))
25950 T1(1,2)=SIN(Angle(1))
25960 T1(1,3)=0
25970 T1(2,1)=SIN(Angle(1))
25980 T1(2,2)=COS(Angle(1))
25990 T1(2,3)=0
26000 T1(3,1)=0
26010 T1(3,2)=0
26020 T1(3,3)=1
26030 ! Define 2nd coordinate transformation matrix for Mod2tun.
26040 ! Rotation in the x-z plane about the y-axis.
26050 ! Used when model is at an angle of yaw.
26060 T2(1,1)=COS(-Angle(2))
26070 T2(1,2)=0
26080 T2(1,3)=SIN(-Angle(2))
26090 T2(2,1)=0
26100 T2(2,2)=1
26110 T2(2,3)=0
26120 T2(3,1)=0
26130 T2(3,2)=0
26140 T2(3,3)=1
26150 ! Define 3rd coordinate transformation matrix for Mod2tun.
26160 ! Rotation in the y-z plane about the x-axis.
26170 ! Used when model is at an angle of roll.
26180 T3(1,1)=1
26190 T3(1,2)=0
26200 T3(1,3)=0
26210 T3(2,1)=0
26220 T3(2,2)=COS(-Angle(3))
26230 T3(2,3)=SIN(-Angle(3))
26240 T3(3,1)=0
26250 T3(3,2)=SIN(-Angle(3))
26260 T3(3,3)=COS(-Angle(3))
26270 ! Mod2tun converts MODEL coordinates to TUNNEL coordinates.
26280 MAT Temp= T2*Temp
26290 ! T2*Temp= T2*T1
26300 ! MAT Mod2tun= T3*Temp
26310 ! MAT Mod2tun= INV(Mod2tun)
26320 ! T2*Temp= T2*T1
26330 ! T3*Temp= T3*T1
26340 SUBEND
26350 Ctm_tcs: SUB Ctm_tcs(Tcs2tunnel(*),Tcs2tun(*),Tun2tans(*),Tun2tcs(*),Fz,Fx,Br,Index(*),Ts,Ta)
26380 ! Description:
26390 ! This subprogram computes the TUNNEL to TCS coordinate system transformation matrices
26400 "Tun2tcs1(*)" and "Tun2tcs2(*)". The coordinate system transformation matrices "Tcs2tun1" and
26410 "Tcs2tun1" are the matrix inverses of "Tun2tcs1(*)" and "Tun2tcs2(*)" respectively.
26420 ! Variables:
26430 Tcs2tun1(*) Sending side coordinate transformation matrix converting Tcs(*) to Tun(*).
26440 Tun2tcs1(*) Sending side coordinate transformation matrix converting Tun(*) to Tcs(*).
26450 Tcs2tun2(*) Receiving side coordinate transformation matrix converting Tcs(*) to Tun(*).
26460 Tun2tcs2(*) Receiving side coordinate transformation matrix converting Tun(*) to Tcs(*).
26470 Fs Focal length for sending side onaxis and offaxis lenses.
26480 Fr Focal length for receiving side offaxis.
26490 Bs Beam spacings for sending side onaxis and offaxis beam pairs.
26500 Br Beam spacing for receiving side offaxis.
26510 Index(*) Array of indexes of refraction for air, glass, and water.
26520 Ta Angle of offaxis sending side beam pair.
26530 Tr Angle of offaxis receiving side beam pair.
26540 Ys_offr Starting offaxis auxiliary traverse angle. Returned to main program.
26550 Xs_on,Ys_on Starting coordinates of onaxis sending side lens.
26560 Xs_offs,Ys_offs Starting coordinates of offaxis sending side lens.
26570 Xs_offr,Ys_offr Starting coordinates of offaxis receiving side lens.
26580 Xc,Yc The common point in air of two beam path equations.
26590 Ba,Bb The Y intercepts of two beam path equations.
26600 Theta(*) Array of angles in which each beam contacts the window.
26610 Xs_on起码 Array of X coordinates for the points in which each beam contacts the window.
26620 Yposition The Y coordinate of the point where all beams cross in the water.
26630 Y1,X2,Y2,X3,Y3 Temporary variables to hold the results of the first call to Findstart.
26640 Thickness The thickness of the window.
26650 Beam Subscript used while determining the X(*) array.
26660 OPTIONS BASE 1
26660 REAL Xs_on,Ys_on,Xs_offs,Ys_offs,Xs_offr,Ys_offr,Xc,Yc,Ba,Bb,Theta(6),X(6)
26680 REAL Yposition,Y1,X2,Y2,X3,Y3,Thickness
26690 INTEGER Beam
26700 Thickness=1.25
26710 Yposition=0
26720 GOSUB Findstart
26730 Y1=Ys_on
26740 X2=Xs_offs
26750 Y2=Ys_offs
26760 X3=Xs_offr
26770 Y3=Ys_offr
26780 Yposition=1
26790 GOSUB Findstart
26800 MAT Tun2tcs1= IDN
26810 MAT Tun2tcs2= IDN
26820 Tun2tcs1(2,2)=Ys_on+Y1
26830 Tun2tcs1(4,4)=SQR1((Xs_offs-X2)^2+(Ys_offs-(Ys_on+Y1)+Y2)^2)
26840 Tun2tcs1(4,4)=0
26850 Ta=ATN((Xs_offs-X2)/(Ys_offs-(Ys_on+Y1)+Y2))
26860 Tun2tcs2(1,2)=Xs_offs-X3
26870 Tun2tcs2(2,2)=Ys_offs+Y3
26880 Tun2tcs2(4,4)=0
26890 MAT Tcs2tun= INV(Tun2tcs1)
26900 MAT Tcs2tun2= INV(Tun2tcs2)
26910 Tcs2tun1(4,4)=0
26920 Tcs2tun2(4,4)=0
26930 SUBEXIT
26940 ! Findstart: ! This subroutine finds the starting coordinates for the onaxis (Xs_on,Ys_on) and offaxis (Xs_offs,
26950 ! Xs_offr) sending side lenses and the offaxis (Xs_offr,Ys_offr) receiving side lens given the point
26960 ! at which all beams cross in the tunnel. The crossing point is given to be (0,Yposition). The
26970 ! method in which the starting coordinates are found involves solving simultaneously the equations for
26980 ! the path of each beam pair in air yielding the common point (Xc,Yc). Given the focal length of the
26990 ! lens, the starting coordinate can be calculated. The equation for each beam path in air is obtained
27000 ! by determining the angle and the point a beam contacts the window.
27010 !
27020 ! These six equations find the six angles.
27030 Theta(1)=ATN(Ba/(2*Fs))
27040 Theta(2)=ATN(Bb/(2*Fs))
27050 Theta(3)=Ts+ATN(Ba/(2*Fs))
27060 Theta(4)=Tr+ATN(Bb/(2*Fs))
27070 Theta(5)=Tr+ATN(Ba/(2*Fs))
27080 Theta(6)=Tr+ATN(Bb/(2*Fs))
27090 ! This equation finds the X coordinate of the six points. The Y coordinate is equal to -Thickness.
27100 FOR Beam=1 TO 6
27110 X(Beam)=Yposition*TAN(ASN(Index(1)/Index(3))*SIN(Theta(Beam))))-
27120 Thickness*TAN(ASN(Index(1)/Index(2))*SIN(Theta(Beam))))
27130 NEXT Beam
27140 ! Determine the Y intercepts for the onaxis beam paths.
27150 Ba=-Thickness*X(1)/TAN(Theta(1))
27160 Bb=-Thickness*X(2)/TAN(Theta(2))
27170 ! Solve for the common point.
! Determine the Y intercepts for the offaxis receiving beam paths.
! of the offaxis sending side lens.

Xs_offs=Yc-Fs*SIN(0)
Ys_offs=Yc-Fs*COS(0)

! of the offaxis sending side lens.

Xc=(Bb-Ba)

! Solve for the common point.

Xc=(Bb-Ba)/(1/TAN(Theta(3))-1/TAN(Theta(4)))

Yc=Xc/TAN(Theta(4))
Bb=Thickness-X(4)/TAN(Theta(4))

! Determine the Y intercepts for the offaxis receiving side beam paths.

Bb=Thickness-X(6)/TAN(Theta(6))

! Solve for the common point.

Xc=(Bb-Ba)/(1/TAN(Theta(3))-1/TAN(Theta(6)))

Yc=Xc/TAN(Theta(6))
Bb=Thickness-X(6)/TAN(Theta(6))

! With the focal length and the offaxis angle calculate the starting coordinate

Xc=(Bb-Ba)/TAN(Theta(3))-1/TAN(Theta(4)))

Yc=Xc/TAN(Theta(4))+Bb

! With the focal length and the offaxis angle calculate the starting coordinate

Xc=(Bb-Ba)/(1/TAN(Theta(3))-1/TAN(Theta(6)))

Yc=Xc/TAN(Theta(6))
Bb=Thickness-X(6)/TAN(Theta(6))

! With the focal length and the offaxis angle calculate the starting coordinate

Xc=(Bb-Ba)/(1/TAN(Theta(3))-1/TAN(Theta(6)))

Yc=Xc/TAN(Theta(6))
Bb=Thickness-X(6)/TAN(Theta(6))

! With the focal length and the offaxis angle calculate the starting coordinate

Xc=(Bb-Ba)/(1/TAN(Theta(3))-1/TAN(Theta(6)))

Yc=Xc/TAN(Theta(6))
Bb=Thickness-X(6)/TAN(Theta(6))

! With the focal length and the offaxis angle calculate the starting coordinate

Xc=(Bb-Ba)/(1/TAN(Theta(3))-1/TAN(Theta(6)))

Yc=Xc/TAN(Theta(6))
Bb=Thickness-X(6)/TAN(Theta(6))

! With the focal length and the offaxis angle calculate the starting coordinate

Xc=(Bb-Ba)/(1/TAN(Theta(3))-1/TAN(Theta(6)))

Yc=Xc/TAN(Theta(6))
Bb=Thickness-X(6)/TAN(Theta(6))

! With the focal length and the offaxis angle calculate the starting coordinate

Xc=(Bb-Ba)/(1/TAN(Theta(3))-1/TAN(Theta(6)))

Yc=Xc/TAN(Theta(6))
Bb=Thickness-X(6)/TAN(Theta(6))

! With the focal length and the offaxis angle calculate the starting coordinate

Xc=(Bb-Ba)/(1/TAN(Theta(3))-1/TAN(Theta(6)))

Yc=Xc/TAN(Theta(6))
Bb=Thickness-X(6)/TAN(Theta(6))

! With the focal length and the offaxis angle calculate the starting coordinate

Xc=(Bb-Ba)/(1/TAN(Theta(3))-1/TAN(Theta(6)))

Yc=Xc/TAN(Theta(6))
Bb=Thickness-X(6)/TAN(Theta(6))

! With the focal length and the offaxis angle calculate the starting coordinate

Xc=(Bb-Ba)/(1/TAN(Theta(3))-1/TAN(Theta(6)))

Yc=Xc/TAN(Theta(6))
Bb=Thickness-X(6)/TAN(Theta(6))

! With the focal length and the offaxis angle calculate the starting coordinate

Xc=(Bb-Ba)/(1/TAN(Theta(3))-1/TAN(Theta(6)))

Yc=Xc/TAN(Theta(6))
Bb=Thickness-X(6)/TAN(Theta(6))

! With the focal length and the offaxis angle calculate the starting coordinate

Xc=(Bb-Ba)/(1/TAN(Theta(3))-1/TAN(Theta(6)))

Yc=Xc/TAN(Theta(6))
Bb=Thickness-X(6)/TAN(Theta(6))

! With the focal length and the offaxis angle calculate the starting coordinate

Xc=(Bb-Ba)/(1/TAN(Theta(3))-1/TAN(Theta(6)))

Yc=Xc/TAN(Theta(6))
Bb=Thickness-X(6)/TAN(Theta(6))

! With the focal length and the offaxis angle calculate the starting coordinate

Xc=(Bb-Ba)/(1/TAN(Theta(3))-1/TAN(Theta(6)))

Yc=Xc/TAN(Theta(6))
Bb=Thickness-X(6)/TAN(Theta(6))

! With the focal length and the offaxis angle calculate the starting coordinate

Xc=(Bb-Ba)/(1/TAN(Theta(3))-1/TAN(Theta(6)))

Yc=Xc/TAN(Theta(6))
Bb=Thickness-X(6)/TAN(Theta(6))
Variables:

Array(*) Array containing the plot positions and scales.
ImageS(*) String array containing image formats for the axes labeling.
Wdws(*) Array containing the plot's scales.
Vwpri(*) Array containing the plot's CRT position.
Xdiv(*) Array containing the number of X divisions for the plot's X axis.
Ydiv(*) Array containing the number of Y divisions for the plot's Y axis.
Xlabels(*) String array containing labels for the X axis.
Ylabels(*) String array containing labels for the Y axis.
TitleS(*) String array containing labels for the plots.
Images(*) String array containing image formats for the X axis labeling.
Images(*) String array containing image formats for the Y axis labeling.
LegendS(*) Array containing labels for each symbol in a profile plot.
Symbols(*) Array of Symbol arrays. Each symbol array contains a distinct geometric symbol.

G Used as an index to the above arrays. Specifies one of nine plots.

OPTION BASE 1

COM /Graph\ Wdws(*),Vwpri(*),Xdiv(*),Ydiv(*),Xlabels(*),Ylabels(*),TitleS(*),Images(*),Images(*),
LegendS(*)

MAT Wdws= Array(61:69,*)
MAT Vwpri= Array(71:79,*)
MAT Xdivi(15) = Array(81:85,1)
MAT Xdiv(61) = Array(81:84,3)
MAT Ydivi(15) = Array(81:85,2)
MAT Ydiv(61) = Array(81:84,4)
MAT Images= Images(61:69,1)
MAT Images= Images(61:69,3)
FOR G=1 TO 9
READ G,Xlabels(G)
FOR I=1 TO SIZE(LegendS,2)
READ LegendS(G,I)
NEXT I
SELECT G
CASE 1 TO 5
CASE 6 TO 9
Ylabels(G)=CHR$(NUM("X")+Paxis-1)
END SELECT
CALL Set_up(G,Symbols(*))
NEXT G
SUBEXIT

DATA 1, "", "", "", "", ""
DATA 2, "", "", "", "", ""
DATA 3, "", "", "", "", ""
DATA 4, "", "", "", "", ""
DATA 5, "", "", "", "", ""
DATA 6, "Velocities / Uedge"
DATA 7, "Velocities / Uedge"
DATA 8, "Shear Stress / Uedge2"
DATA 9, "Fluorescence: C,C1 (volts)"

SUBEND

28500 Set_up: SUB Set_up(G,Symbols(*))
28510 \ Description:
28520 \ This subprogram clears and then redraws one of nine empty plots on the CRT screen.
28530 \ Variables:
28540 \ Wdws(*) Array containing the plot's scales.
28550 \ Vwpri(*) Array containing the plot's CRT position.
28560 \ Xdiv(*) Array containing the number of X divisions for the plot's X axis.
28570 \ Ydiv(*) Array containing the number of Y divisions for the plot's Y axis.
28580 \ Xlabels(*) String array containing labels for the X axis.
28590 \ Ylabels(*) String array containing labels for the Y axis.
28600 \ TitleS(*) String array containing labels for the plots.
28610 \ Images(*) String array containing image formats for the X axis labeling.
28620 \ Images(*) String array containing image formats for the Y axis labeling.
28630 \ LegendS(*) String array containing labels for each symbol in a profile plot.
28640 \ Symbols(*) Array of Symbol arrays. Each symbol array contains a distinct geometric symbol.
28650 \ G Used as an index to the above arrays. Specifies one of nine plots.
28660 \ OPTION BASE 1
28670 COM /Graph\ Wdws(*),Vwpri(*),Xdiv(*),Ydiv(*),Xlabels(*),Ylabels(*),TitleS(*),Images(*),Images(*),
LegendS(*)
28680 DIM LS(80)
28690 ON ERROR CALL Error
28700 PLOTTER IS CRT,"INTERNAL"
28710 \ Define the pen numbers for the colors black and white.
28720 Black=1
28730 White=1
CSIZE 100*15/1023  ! Select a character labeling size of 15 pixels high.

Define the values for the left, right, bottom, top ends of the horizontal and vertical scales.

Xmin=Wndw(G,1)
Xmax=Wndw(G,2)
Ymin=Wndw(G,3)
Ymax=Wndw(G,4)

Define the step size between grid lines, axis tick marks, and axis labels.

Xstep=(Xmax-Xmin)
Ystep=(Ymax-Ymin)

Define the amount of scale X and Y which equals the size of one pixel (picture element).

Xpixel=(Xmax-Xmin)/(Xpix2-Xpix1)
Ypixel=(Ymax-Ymin)/(Ypix2-Ypix1)

Clear the plots back ground & plot area and also draw the plot's borders, grids, and axes.

AREA PEN Black
PEN White
PEN Black
GOSUB Back_ground
GOSUB Axes
GOSUB Grid
GOSUB Plot_area
! Draw the X and Y axis labels.
CLIP OFF
GOSUB Xlabel
GOSUB Ylabel
!
CREATE a legend to define which symbol is used with which data.
CALL Legend(G,Symbols(*))
OFF ERROR
SUBEXIT

Back_ground:
! This subroutine clears the plot's background.
VIEWPORT (Xpix1-75)/10.23,(Xpix2+25)/10.23,(Ypix1-33)/10.23,(Ypix2+6)/10.23
WINDOW -1.8+9.1,0.9+9,-1.8+9,1.6+9
MOVE 0,0
WINDOW 0,1,0,1
MOVE 0,0
RECTANGLE 1.1,1.1,FILL
RETURN

Axes:
! This subroutine draws the plot's X and Y axes.
VIEWPORT (Xpix1-1)/10.23,(Xpix2+1)/10.23,(Ypix1-6)/10.23,(Ypix2+6)/10.23
WINDOW Xmin,Xmax,0,1
AXES Xstep,2,Xmin,0,1,1,1
VIEWPORT (Xpix1-1)/10.23,(Xpix2+1)/10.23,(Ypix1-6)/10.23,(Ypix2+6)/10.23
WINDOW Xmin,Xmax,0,1
AXES Xstep,2,Xmin,0,1,1,1
VIEWPORT (Xpix1-6)/10.23,(Xpix1-1)/10.23,(Ypix1-1)/10.23,(Ypix2+1)/10.23
WINDOW 0,1,0,Ymin,Ymax
AXES 2,Ystep,0,Ymin,0,1,1,1
VIEWPORT (Xpix2+1)/10.23,(Xpix2+6)/10.23,(Ypix1-1)/10.23,(Ypix2+1)/10.23
WINDOW 0,1,0,Ymin,Ymax
AXES 2,Ystep,0,Ymin,0,1,1,1
RETURN

Grid:
! This subroutine draws the plot's X and Y grid lines.
VIEWPORT (Xpix1-1)/10.23,(Xpix2+1)/10.23,(Ypix1-1)/10.23,(Ypix2+1)/10.23
WINDOW Xmin,Xmax,Ymin,Ymax
LINE TYPE 4
GRID Xstep,Ystep,Xmin,Ymin
LINE TYPE 1
RETURN

Plot_area:
! This subroutine selects part of the CRT plot area and give it scales for the X and Y axes.
VIEWPORT Xpix1/10.23,Xpix2/10.23,Ypix1/10.23,Ypix2/10.23
WINDOW Xmin,Xmax,Ymin,Ymax
RETURN

Xlabel:
! This subroutine labels the X axis and also names the X axis.
LOG 5
FOR X=Xmin TO Xmax+Xstep/100 STEP Xstep
MOVE X,Ymin-12*Ypixel
OUTPUT LS USING Ximage4(G);X
LABEL TRIM(LS)
NEXT X
MOVE (Xmin+Xmax)/2,Ymin-25*Ypixel
LABEL Xlabel(G)
RETURN

Ylabel:
! This subroutine labels the Y axis and also names the Y axis.
LOG 8
Len=0
FOR Y=Ymin TO Ymax+Ystep/100 STEP Ystep
MOVE Xmin-5*Ypixel,Y

95
!This subprogram plots real time histograms within five of the nine plots on the CRT screen.

Variables Defined in Main Program:

Histo(*), Repeat

Variables Defined in this Program:

N(*) Number of samples in the previous histogram.
Nnew Number of samples in the most up to date histogram.
N(*) Number of samples for each histogram of the five separate channels.

Local Variables:

Histo(*) Array of bin numbers, old histogram bin heights, and new histogram bin heights.

Description: This subprogram plots real time histograms within five of the nine plots on the CRT screen.

The histogram data is acquired from the LVDAS over a specified acquisition time.

The number of coordinates in which when connected produce a distinct geometric symbol.

The number of coordinates which the previous histogram.

The number of coordinates for each symbol in a profile plot.

The number of coordinates to index the Legends array.

The number of coordinates which each symbol arrays contains a distinct geometric symbol.

Used as an index to the above arrays. Specifies one of nine plots.

For each symbol put up a sample symbol and its label.

The maximum length of all of the symbol labels.

For each symbol in a profile plot.

The maximum length of individual histogram vertical bars.

Minimum value for histogram. Left side of histogram scale.

Maximum value for histogram. Right side of histogram scale.

Upper 16 bits of integerized Min.

Lower 16 bits of integerized Min.

Upper 16 bits of integerized histogram acquisition time.

Lower 16 bits of integerized histogram acquisition time.

Number of samples in the most up to date histogram.

Number of samples for each histogram of the five separate channels.
Call Convert2words(Min,F1,F2).  
CASE 4.5  
Kw=32768/5  
Min=Kw*Wndw(G,1)  
Max=Kw*Wndw(G,2)  
Bin=INT(LGT((Max-Min)/100)/LGT(2)+1)  
Ww=2*Bin  
CALL Convert2words(Min,F1,F2)  
CASE ELSE  
END SELECT  
HEND:  
! Tell the LVDAS to Take a Histogram.
OUTPUT @Lvdas USING "AA,6(W);"TH,F1,F2,Bin,A1,A2,Channel
ENTER @Lvdas USING ",W,W;Histo(*)
! Redimension the Histo(*) and the enter the histogram data.
IF Nbins>0 THEN
  REDIM Histo(Nbins,3)
  ENTER @Lvdas USING ",W,W;Histo(*)
END IF
! Enter the number of samples for the previous and current histogram.
ENTER @Lvdas USING ",W,W;New,Nold
! Scale part of the CRT for the histogram plotting.
VIEWPORT Wvprt(G,1)/10.23, Wvprt(G,2)/10.23, Wvprt(G,3)/10.23, Wvprt(G,4)/10.23
WINDOW Kw*Wndw(G,1), Kw*Wndw(G,2), Wndw(G,3), Wndw(G,4)
Xpixel=Kw*(Wndw(Channel,2)-Wndw(Channel,1))/ (Wvprt(Channel,2)-Wvprt(Channel,1))
N1=N(Channel)
N2=N(Channel)-Nold+Nnew
H(Channel)=N(Channel)-Nold+Nnew
FOR I=1 TO Nbins  
Old=MIN(Histo(I,3),Wndw(Channel,4))  
New=MIN(Histo(I,2),Wndw(Channel,4))  
AREA PEN SGN(Old-New)  
X1=Histo(I,1)+Ww+Min  
X2=Ww  
Y1=Old  
Y2=New-Old  
IF X1<Xw*Wndw(G,1) THEN X1=Xw*Wndw(G,1)  
IF X1<Xmin then set X1=Xmin  
IF (X1>Xw*Wndw(G,2)-X2) THEN X1=Xw*Wndw(G,2)-X2  
IF X1>Xmax then set X1=Xmax  
MOVE X1,Y1  
RECTANGLE X2-Xpixel,Y2,FILL  
NEXT I
NEXT Channel
UNTIL KBDS<>"" OR NOT Repeat  
! Quit if any key on the keyboard has been pressed.
SUBEXIT
31130 Hdone: Done=1
31140 RETURN
31150 SUBEND
31160 Histo: !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
31170 Pt_histo: SUB Pt_histo(Symbols(*),Run,File,Pos,INTEGER Naam)
31180 ! Description:
31190 ! This subprogram plots post time histograms within five of the nine plots on the CRT screen.
31200 ! The histogram data is acquired from the LVDAS over a specified acquisition time.
31210 ! Variables Defined in Main Program:
31220 ! Wndw(*),Vwprt(*),Xdiv(*),Ydiv(*),Xlabel(*),Ylabel(*),Title(*),Ximage(*),Yimage(*),Legend(*)
31230 ! Local Variables:
31240 ! Histo(*) Array of histogram bin heights indexed by bin number.
31250 ! Data(*) Array of instantaneous U,V,W velocity or A,B voltage data.
31260 ! Naam Number of samples acquired.
31270 ! Xmin Minimum value for histogram, left side of histogram scale.
31280 ! Xmax Maximum value for histogram, right side of histogram scale.
31290 ! Xwin Window width of each vertical histogram bar.
31300 ! K Used as an index to the above arrays. Specifies one of 100 bins.
31310 ! L Used as an index to the Histo(*). Specifies one of 100 bins.
31320 ! Channel Selects one of the 5 channels of Ui(*),Vi(*),Wi(*),Ai(*),Bi(*) data.
31330 ! G Used as an index to the graphics arrays. Specifies one of nine plots.
31340 ! Xpixel Horizontal length of one picture on the CRT in scale units.
31350 ! Data(*) Array of instantaneous U,V,W velocity or A,B voltage data.
31360 ! Naam Number of samples acquired.
31370 ! Xmin Minimum value for histogram, left side of histogram scale.
31380 ! Xmax Maximum value for histogram, right side of histogram scale.
31390 ! Xwin Window width of each vertical histogram bar.
31400 ! K Used as an index to the above arrays. Specifies one of 100 bins.
31410 ! L Used as an index to the Histo(*). Specifies one of 100 bins.

OPTION BASE 1

COM /Data/ INTEGER Raw(*),Valid(*),REAL Table(*),Ui(*),Vi(*),Wi(*),Ai(*),Bi(*)

COM /Graph/ Wndw(*),Vwprt(*),Xdiv(*),Ydiv(*),Xlabel(*),Ylabel(*),Title(*),Ximage(*),Yimage(*),Legend(*)

INTEGER Histo(0:100)
31400 REAL Data(1000)
31410 REDIM Data(Naam)
31420 FOR Channel=1 TO 5
31430 ! Fill the data array with Ui(*), Vi(*), Wi(*), Ai(*), or Bi(*) depending on Channel.
31440 G=Channel
31450 IF Channel=1 THEN MAT Data= Ui
31460 IF Channel=2 THEN MAT Data= Vi
31470 IF Channel=3 THEN MAT Data= Wi
31480 IF Channel=4 THEN MAT Data= Ai
31490 IF Channel=5 THEN MAT Data= Bi
31500 ! Draw a new empty histogram plot.
31510 CALL Set_up(Channel,Symbols(*))
31520 Hsort:
31530 Xmin=Wndw(Channel,1)
31540 Xmax=Wndw(Channel,2)
31550 Xmin=(Xmax-Xmin)/100
31560 ! Sort the data into a histogram.
31570 MAT Data= Data-(Xmin)
31580 MAT Data= Data/((Xmax-Xmin)/100)
31590 MAT Histo= (0)
31600 FOR K=1 TO Naam
31610 Histo(K)=MAX(MIN(Data(K),100),0)
31620 NEXT K
31630 Hplot:
31640 ! Scale part of the CRT for histogram plotting.
31650 VIEWPORT Vwprt(G,1)/10.23,Vwprt(G,2)/10.23,Vwprt(G,3)/10.23,Vwprt(G,4)/10.23
31660 WINDOW 0,100,Wndw(G,3),Wndw(G,4)
31670 Xpixel=(100-0)/(Vwprt(Channel,2)-Vwprt(Channel,1))
31680 ! Draw the histogram.
31690 FOR K=0 TO 100
31700 IF Histo(K) THEN
31710 MOVE K-.5,0
31720 AREA PEN SGN(1) ! Positive pens will plot while negative histograms erase.
31730 END IF
31740 NEXT K
31750 NEXT Channel
31760 SUBEXIT
31770 SUBEND
DONE
A three-dimensional laser fluorescence anemometer has been designed, built, and demonstrated for use in the Langley 16-by 24-Inch Water Tunnel. Innovative optical design flexibility combined with compact and portable data acquisition and control systems have been incorporated into the instrument. This will allow its use by NASA in other test facilities. A versatile fiber optic system facilitates normal and off-axis laser beam alignment, removes mirror losses and improves laser safety. This added optical flexibility will also enable simple adaptation for use in the adjacent jet facility. New proprietary concepts in transmitting color separation, light collection, and novel prism separation of the scattered light have also been designed and built into the system. Off-axis beam traverse and alignment complexity led to the requirement for a specialized, programmable traverse controller and the inclusion of an additional traverse for the off-axis arm. To meet this challenge, an “in-house” prototype unit was designed and built and traverse control software developed specifically for the water tunnel traverse applications. A specialized data acquisition interface was also required. This was designed and built for the Laser Fluorescence Anemometer system.

Subject Category 02