Final Report
Iowa Highway Research Board
Project HR-269

DATA ACQUISITION AND COMPUTER PLOTTING OF DELAMTECT DATA

D & D Digital Systems, Inc.

September 1985

In cooperation with the
Highway Division

Iowa Department
of Transportation

D & D Digital Systems

"Computer Engineering"

P.O. Box 1802
111 Lynn
Ames, Iowa 50010
(515) 292-0490

FINAL REPORT

PROJECT : HR - 269

DATA ACQUISITION AND COMPUTER PLOTTING OF DELAMTECT DATA

SUBMITTED TO: IOWA DEPARTMENT OF TRANSPORTATION

SUBMITTED BY: D & D DIGITAL SYSTEMS INC
111 LYNN
AMES, IOWA 50010

DATE: JULY 1, 1985

1.0 SYSTEM OVERVIEW

The overall system is designed to permit automatic collection of delamination field data for bridge decks. In addition to measuring and recording the data in the field, the system provides for transferring the recorded data to a personal computer for processing and plotting. This permits rapid turnaround from data collection to a finished plot of the results in a fraction of the time previously required for manual analysis of the analog data captured on a strip chart recorder.

1.1 DELAMTECT

In normal operation the Delamtect provides an analog voltage for each of two channels which is proportional to the extent of any delamination. These voltages are recorded on a strip chart for later visual analysis. An event marker voltage, produced by a momentary push button on the handle, is also provided by the Delamtect and recorded on a third channel of the analog recorder.

1.2 D & D DATA ACQUISTION SYSTEM (DAS)

A distance measuring wheel was added to provide digital pulses indicating distance traveled.

A microprocessor based digital computer was designed to sample both analog signals from the Delamtect, convert them to digital numbers, and transmit these numbers to a cassette recorder. The Delamtect event marker switch and distance measuring wheel are monitored by this computer to control when to begin and stop the process of sampling the analog signals and how often samples should be taken.

1.3 TECHTRAN RECORDER

A digital cassette recorder was added to the system to record the digital numbers which are equivalent to the analog voltages from the two Delamtect channels. Digital recorders have the ability to directly receive digital data and record at high densities. The recorder has been mounted such that it can be removed from the Delamtect and transported to a computer site. There the digitally recorded data can be read and transferred from the recorder to the computer for further processing and plotting.

1.4 SPERRY/ZENITH/IBM PC PLOTTING PROGRAMS

Two programs have been developed to be run on the SPERRY/ZENITH/IBM PC.

The first program is designed to read data files recorded on the Techtran. Each pass across the bridge constitutes a file of data. As each data file is read, it is checked for correct sequence and length characteristics before being written to a disk file on the PC.

The second program is designed to process the data files and produce a plot of the results on a dot matrix printer operating in graphics mode which visually shows where delaminations exist in the area surveyed. Selected choices for type of plot and printer are provided. The voltage level to be used as a threshold for determining the presence of a delamination is also provided as a choice.

Communication between the Techtran and the PC is through the asynchronous COM1 port. The printer is attached through the Centronics compatible PRN port.

1.5 TEXAS INSTRUMENTS 855 PRINTER

The TI 855 dot matrix printer was specified to be the primary print device. Choices for the kind of plot desired include one which fills the width of the paper and one which presents uniform horizontal and vertical scales.

Header information describing the bridge ID, date, total area, delaminated area, etc. is also printed on each plot.

2.0 THEORY OF OPERATION

The following sections describe some of the technical aspects of the design by functional component. Technical design details can be found from the circuit schematics and program listings in the appendices of this report.

2.1 DELAMTECT

Each channel of the Delamtect provides an analog signal with a voltage ranging from 0 to approximately 5 volts. In areas of no delamination, the voltage is approximately 0.1 to 0.3 volt. Major delamination areas have been judged to occur where changes in voltage exceed 400 millivolts which correspond to 4 mm deflection on the chart paper. Where extreme delamination occurs, voltage changes or fluctuations in excess of 2 volts occur.

The sensitivity of the strip chart recorder is typically set for 1 volt/cm or 100 millivolt/mm.

The event marker switch creates a signal of 0 or approximately 2.0 volts when it is open or closed.

Power supply voltages of +12 and -12 volts DC are supplied by the Delamtect.

No modifications have been made to the original Delamtect circuits.

Only wiring taps to the power supply, channel voltages and event marker switch have been added to route these signals to the D & D DAS for processing. These taps are wired to a new connector added to the Delamtect but do not alter the original signals. An external wiring harness extends these signals to the DAS mounted in the lid of the Techtran recorder.

2.2 D & D DATA ACQUISITION SYSTEM (DAS)

A distance measuring wheel was added to provide 10 pulses per inch of travel. These become 5 volt pulse inputs to the DAS. This was chosen to avoid problems with integrating a DC voltage from the tachometer on the Techtran to compute distance traveled.

Two eight-bit analog-to-digital converters (one for each channel) are used with a 5 volt maximum input voltage which provides for a 20 millivolt per bit sensitivity. This sensitivity corresponds to a 0.2 mm deflection on the strip chart recorder.

In 3 inches of travel, 30 distance pulses are received. The analog voltages from the Delamtect are sampled and digitized after 6 pulses (0.6 inch) are received This is repeated five times during every 3 inches of travel. The average of the first 4 samples is computed and transmitted to the Techtran recorder after the fifth sample period. The fifth sample point is ignored. Thus, each 3 inches of travel results in a value being sent to the recorder for each channel (two bytes; left channel followed by right channel). These values represent the computed average of 4 sample values during 3 inch interval.

An offset of 32 (base 10) is added to each digitized value. The maximum value after offset adjustment is limited to 127 (base 10). This insures no spurious CTRL characters (0 to 31) are sent to the recorder which might alter its operational status. Additionally, characters are transmitted over a serial RS-232C interface in 7 bit even parity codes which limits the maximum value to 127. This corresponds to 1.9 volts which is well in excess of any reasonable delamination threshold level.

To insure the tape is properly positioned with respect to tape leader in the Techtran recorder, a sequence of four one-second read commands followed by a rewind command is initiated on powerup. The green light on the top of the recorder case momentarily turns on and then off at the beginning of the powerup sequence.

The green light on the top of the recorder case should be on when a pass begins and off at the end of the pass. This is accomplished by the operator confidently pushing the marker event switch once at the beginning and end of each pass to toggle the light from off to on or from on to off.

Data files for each pass are transmitted to the Techtran recorder which begin with a 'Pass number sequence character' (A,B,C,D,E, etc.), followed by pairs of averaged data bytes (Left, Right), and ending with a file termination character CTRL-S.

A custom program stored in an EPROM on the DAS controls all this activity among the Delamtect, distance wheel, and recorder.

2.3 TECHTRAN RECORDER

The 9600PRL recorder has several switch settings which should remain unchanged throughout the operations.

LINE MODE SWITCH = OFF (DOWN POSITION)
BINARY MODE SWITCH = ONLINE (CENTER POSITION)

DIP SWITCHES BENEATH THE LIFT OFF COVER PLATE ARE SET TO

1:	+	9600 BAUD
2:	+	9600 BAUD
3:	-	DISABLE BS
4:	-	HALF DUPLEX
5:	-	DISABLE DELAY
6:	-	LF LINEMODE
7:	-	BIN CTRL OFF
8:	-	DELAY ON LF
9:	-	EVEN PARITY
10:	+	ENABLE PARITY

Cassette insertion and removal is done by pressing the POWER pushbutton (it will illuminate), then manually lifting the door latch allowing the door to swing open, and inserting or removing the cassette. The magnetic tape side should face down into the recorder, the cassette label should be visible through the window, and the large tape spool should be on the left if it is rewound.

New tapes should be rewound forward and backward to remove any static binding which may initially exist.

Used tapes should be demagnetized with the bulk eraser before being removed. Approximately 5 - 10 seconds is long enough to remove all previously recorded data.

The internal battery pack should be charged during the night preceding any field tests. Techtran verbally estimates a minimum of 5-6 hour operation time between charges.

Each digital cassette tape is rated to hold up to 220,000 characters or bytes which is well within the maximum limit of 64,000 characters or bytes permitted by the applications program for the SPERRY/ZENITH/IBM PC.

During data collection on a bridge, the TERMINAL PORT is used to connect the Techtran to the DAS. The MODEM/CPU PORT should be disconnected during this operation.

After data has been collected, the MODEM/CPU PORT is used to connect the Techtran to the SPERRY/ZENITH/IBM PC COM1 port.

The TERMINAL PORT should be disconnected during this operation.

2.4 SPERRY/ZENITH/IBM PC PROGRAMS

Two programs have been written to process the data collected on the Techtran recorder called TAPEREAD and BRIDGE.

TAPEREAD reads the data from cassette tape, does some verification of the data, and writes it to a disk file for subsequent processing.

BRIDGE reads the data from the disk file and plots it on a dot matrix printer according to user selected options.

The maximum length on the bridge for a single pass is software constrained to 6000 feet.

A data space of 64K bytes (8 bits = 1 byte) is reserved in the PC for a bit map of the bridge deck where each bit represents a 3 inch distance traveled in a 9 inch wide path. Two channels provide for combined 18 inch wide passes.

Every foot of travel provides 4 data values (one for every 3 inches of travel) for each of 2 channels. Thus, a total of 8 data values exist for every foot of travel. Each of these values represents the condition of a 3 inch by 9 inch surface area which is either delaminated or not delaminated. Once yes or no decision about delamination has been determined, only a single bit is required to represent that result (1 = delaminated, 0 = not delaminated). Thus, 1 byte (8 bits) can be used to represent the status of 8 areas each 3x9 inches. This is equivalent to a total area of 12 x 18 inches or 1 foot of travel for the 18 inch wide Delamtect path.

Thus, the maximum surface area which may be represented in a 64K (64,000) bit map can be found as follows:

PASSES	LENGTH	TOTAL
10	6000 ft	60,000 bytes
21	3000	63,000
42	1500	63,000
64	1000	64,000
128	500	64,000

etc.

There is also a limitation of the maximum surface area which may be surveyed at one time based upon the disk capacity associated with the SPERRY/ZENITH/IBM PC. Floppy disk drives have a maximum capacity of 320K or 360K bytes depending upon the version of DOS which is being used.

Two files are created by the TAPEREAD program. The first is a temporary file which contains a copy of the data read from the Techtran recorder. This temporary file is always written to the default system disk drive. The second file is the disk file which contains the data without the sequence numbers and end of file characters. In addition, the second file contains a header record with information about the bridge. A user prompt for the file name can also include a disk drive designation if one other than the present default drive is desired.

Since each pass contains 2 channels of data and pairs of data are recorded for every 3 inches of travel, 8 data bytes are recorded for every foot of travel on a pass. Thus, the maximum pass-foot distance which can be written to a 320K disk file is 40K and the maximum for a 360K disk file is 45K.

This would lower the size of a deck surface which can be handled for a 320K diskette to

PASSES	LENGTH	TOTAL
10	4000 ft	40,000 bytes
20	2000	40,000

etc.

A PC with a winchester (hard) disk drive does not have this size constraint since files in excess of 100,000 bytes present no space problem provided the disk is not full with other information.

A 64 byte header record is used in the PC data file for bridge identification and layout parameters.

Documented in the source code listing, this 64 character string variable contains

Information	Bytes In File			Comment
Bridge Id	0		32	
# Passes		33		A,B,C, ...
Length in samples	34	-	35	Binary, high byte 1st
Normal or Slewed		36		N,n,S,s
Distance in inches (R)	37	-	38	Binary, high byte 1st
Distance in inches (L)	39	-	40	Binary, high byte 1st
Start Right/Left		37		R,r,L,l
Date Info	42	-	54	
Extra space	55	-	63	
Bridge data	64	-	??	

Bridge data in the PC files do not alternate between left and right channels every byte. Instead, all left channel data for one pass is written to the disk, then all right channel data for the same pass is written. This pattern then repeats for the number of valid passes associated with the bridge deck.

Disk filenames follow standard MS DOS conventions and can have up to 8 characters with an optional 3 character extension, such as 'BR125437.DAT'. Disk drive information in addition to this limit is allowed, i.e. B:MAXWELL.DAT .

Bridge names can be up to 32 characters long.
Dates can be up to 10 characters long.

It is important that the length of the first pass be accurate since it is used as the reference for all following passes. Passes which are shorter in length have additional low voltage data added equally to both ends of the file to fill it out to a standard length. Passes which are longer in length have data truncated equally from both ends until the standard length is reached.

If the first pass is short (less than 10 feet) the program will display a prompt on the PC screen to ask if it is to really be used. If the answer is n (no), it will read the second pass and treat it as a new first pass.

If a subsequent pass is short (less than 90% of the first pass), a displayed prompt will request if it is to be used or ignored. This could occur in the middle of a field test if the event marker switch was inadvertently pushed at the wrong time.

If a record is out of sequence (i.e. record 5 follows record 3), an option is displayed to permit a choice of using it, skipping it, or inserting a blank (no delaminations) record of data before this record. This should not occur, but gremlins occasionally show up in any system, and this provides a degree of partial recovery of the data.

The goal of the TAPEREAD program features described above is to create a disk file of sequential records from cassette files. This disk file will have identical data lengths for each pass with appropriate master header information about the name, date of test, size, and type of bridge being tested.

The BRIDGE program processes data from the PC data file previously created by TAPEREAD.

While reading in a disk data file but prior to plotting, an average baseline value for each side of each pass is computed by finding the most frequent value. Delaminations are then determined by deviations above this baseline value by some selected amount. A default of 400 millivolts (20 decimal above the baseline value) may be altered by a user prompt. This information is then placed into a memory bit map which is referred to by subsequent plotting programs.

Two delamination plots can be selected.

1: 8 Inches Wide
2: 4 Dots Per 3 Inches

An 8 inches wide plot utilizes the full width of the paper. Horizontal scaling varies with the number of passes.

A 4 dots per 3 inches of deck surface plot achieves uniform horizontal and vertical scaling within the physical limits of the printer.

One percentage plot can also be selected.

P: Percentages

A percentage plot separates the deck into 4 ft by 3 ft grids and prints the percentage of delaminated area in each grid box in addition to an overall percentage of delaminated area for the entire deck.

Bridge decks wider than 30 feet (20 passes) will be plotted in a compressed mode when using the percentage plot (P). The maximum width in this mode is 60 feet.

Bridge decks wider than 33 (22 passes) feet will be plotted in a compressed mode for graphical delamination plots (1, 2). The maximum width in this mode is 66 feet.

Bridge decks can be rectangular or skewed. If skewed, the distance in inches from the left and from the right rectangular starting line to the actual deck is requested by the TAPEREAD program.

Each plot prints information about the bridge name, date of field test, delamination threshold, length and width of the bridge deck, total area, and the percentage of the total area where delaminations exist. If the bridge is skewed, the left and right skew distances are also printed. Calculations of total area and percentage of delaminated area do not include the skewed area outside of the bridge deck.

Passes for a bridge can start on the left or on the right. Once started, subsequent passes are always in the opposite direction to the previous pass in an up and back manner. The bottom of the printed plot will correspond to the starting end of the bridge survey.

2.5 TEXAS INSTRUMENTS 855 PRINTER

Printers generally have more problems with compatibility than computers. While the printers may indeed have comparable capabilities, programming them into the same operating mode is often done in ways unique to each printer.

There is a big difference between functional compatibility and hardware compatibility. Therefore, each new printer should be approched with doubt as to having it function in anything but a standard manner without hardware or software modifications.

A choice of 4 printer types is given in the BRIDGE program.

T : TI 855 Printer
S : Star Radix Printer
E : Epson Printer
A : Alphanumeric Printer

The Alphanumeric Printer assumes no graphics dot matrix capability is present, so it will only print header summary information typically found at the beginning of each delamination plot. In addition, it will also plot the Percentages (P) plot since it prints only standard alphanumeric characters.

The TI 855 printer is placed into DP mode under program control and left there when done. If this is not the normal mode, it can easily be restored by turning the power switch of the printer off and then back on.

TI and the Star RADIX printers have a 72 dots/inch plotting density. Epson MX/FX printers have a 60 dots/inch plotting density. Even more of a problem is that each type of printer typically has a different code sequence to put it into a so-called 'compatible' mode.

The uniform vertical and horizontal scale plot at 4 dots per 3 inches is achieved by occasionally skipping a dot in the vertical direction. Perfect scaling is not feasible. However, when reduced to a 10 scale size (1 in = 10 ft), the perturbation should not be observable.

3.0 OPERATING INSTRUCTIONS

3.1 DATA ACQUISTION PROCEDURES

1. TURN ON THE DELAMTECT FOR WARMUP AND RUN NORMAL CALIBRATION PROCEDURES.

2. PRESS POWER SWITCH OF RECORDER ON (SWITCH LIGHT WILL TURN ON). SET RECORDER LINE MODE SWITCH OFF (DOWN) AND BINARY SWITCH ONLINE (CENTER). DO NOT CLOSE THE LID YET.

3. INSERT TAPE IN RECORDER (TAPE EDGE IN FIRST & LABEL SHOWING IN WINDOW). PLUG IN RIBBON CABLE FROM RECORDER LID TO TERMINAL CONNECTOR ON RECORDER.

4. LOWER THE DISTANCE WHEEL AND INSERT HOLDING PIN IN OTHER HOLE.

5. TURN POWER SWITCH OF DELAMTECT OFF & ON.

   (**DO NOT TURN POWER SWITCH OFF AGAIN UNTIL SURVEY IS COMPLETE**).

   THE TRANSMITTER AND OPERATE/CALIBRATE SWITCH CAN BE CYCLED ON/OFF AT WILL.

   (** THE GREEN LIGHT ON THE RECORDER LID WILL CYCLE ON AND OFF ONCE. THE TAPE RECORDER READ SWITCH SHOULD THEN AUTOMATICALLY CYCLE ON & OFF 4 TIMES FOLLOWED BY AN AUTOMATIC REWIND CYCLE TO CORRECTLY POSITION THE TAPE LEADER. THE REWIND, READ, AND WRITE SWITCH LIGHTS SHOULD BE OFF AND THE POWER SWITCH LIGHT SHOULD BE ON AT THIS TIME. **)

   REPEAT THIS STEP UNTIL THE CORRECT RECORDER STATUS IS OBSERVED.

6. CLOSE THE RECORDER LID AND MOVE INTO POSITION ON BRIDGE IF NOT ALREADY THERE. IF IT IS A SKEWED BRIDGE, NOTE THE DISTANCE FROM BASE LINE TO BRIDGE DECK IN INCHES OF THE RIGHT AND LEFT SIDES.

7. TURN ON DELAMTECT TRANSMITTER (THIS MAY BE TURNED ON AND OFF AS NEEDED DURING THE SURVEY).

8. PUSH EVENT SWITCH ONLY ONCE TO START A PASS (GREEN LIGHT ON RECORDER SHOULD MUST BE ON TO RECORD!!) IF INADVERTENTLY PUSHED TWICE ON FIRST PASS, GO BACK TO STEP 5 AND START OVER. IF ADAVERTENTLY PUSHED TWICE ON SUBSEQUENT PASSES, PUSH AGAIN UNTIL LIGHT COMES ON AND

PROCEED BUT NOTE THIS OCCURENCE SO THAT THE SHORT RECORD OF DATA CAN BE TOSSED OUT BY THE COMPUTER OPERATOR WHEN LATER READ BY THE COMPUTER FOR PROCESSING.

9. AT END OF EACH PASS, PUSH EVENT SWITCH ONCE TO TURN GREEN RECORDER LIGHT OFF AND STOP DATA COLLECTION FOR THAT PASS.

10. TURN THE DELAMTECT AROUND (PRESS DOWN ON THE HANDLE TO ALLOW THE DISTANCE MEASURING WHEEL TO RAISE FROM THE SURFACE AND AVOID DAMAGE BY TWISTING).

11. PUSH EVENT BUTTON TO START THE NEXT PASS (THE GREEN RECORDER LIGHT SHOULD TURN ON AGAIN, ETC).

12. WHEN DONE WITH LAST PASS, TURN OFF THE DELAMTECT TRANSMITTER, OPEN THE RECORDER LID, PUSH THE REWIND PUSHBUTTON SWITCH OF THE RECORDER TO REWIND TAPE. REMOVE THE TAPE, TURN THE TAPE RECORDER POWER OFF, AND THEN TURN DELAMTECT POWER OFF. RAISE DISTANCE WHEEL AND LOCK UP WITH THE HOLDING PIN.

3.2 COMPUTER PLOTTING PROCEDURES

1. A SYSTEM DISK SHOULD BE FORMATTED ON THE TARGET COMPUTER AND A COPY OF TAPEREAD.EXE AND BRIDGE.EXE COPIED TO IT. THIS BECOMES A MASTER PROGRAM DISK. TO CALL UP EITHER PROGRAM, SIMPLY TYPE 'TAPEREAD' OR 'BRIDGE'.

   AT LEAST 2 FORMATTED DATA DISKS SHOULD BE AVAILABLE TO USE WITH THE TAPEREAD PROGRAM. FOR SMALL BRIDGES, BOTH THE TEMPORARY FILE AND FINAL DATA FILE CAN PROBABLY FIT ON ONE DATA DISKETTE IN DRIVE B:. FOR LARGER BRIDGES, A DATA DISK MIGHT BE NECESSARY IN BOTH FLOPPY DRIVES (A: & B:). AFTER THE TAPEREAD PROGRAM IS LOADED, THE EXTRA DATA DISK CAN BE PLACED IN THE SYSTEM DEFAULT DRIVE A:.

2. AN RS-232 CABLE IS REQUIRED TO CONNECT THE TECHTRAN RECORDER TO THE COM1 PORT ON THE SPERRY/ZENITH/IBM PC. THE SPERRY COM1 PORT IS FOUND ON THE BACK NEAR THE CENTER BOTTOM IN A HORIZONTAL POSITION.

3. AFTER CONNECTING THE CABLE BETWEEN THE RECORDER AND THE COMPUTER, AND BOOTING THE PROGRAM DISK, TURN THE TAPE RECORDER ON AND INSERT THE CASSETTE TAPE WITH THE BRIDGE SURVEY DATA.

4. PROGRAM 'TAPEREAD' IS USED FIRST TO TRANSFER THE DATA FROM THE RECORDER TO THE COMPUTER. PROMPTS ARE DISPLAYED AND RESPONSED NEED TO BE TYPED FOR

   BRIDGE ID:
   DATE:
   START ON RIGHT OR LEFT SIDE
   NORMAL OR SKEWED:
   IF SKEWED, RIGHT AND LEFT SKEW DISTANCE:

   INFORMATION ABOUT THE PASS NUMBER AND FILE LENGTHS ARE DISPLAYED HERE. IF SHORT PASSES OR SEQUENCE PROBLEMS EXIST, PROMPTS FOR DESIRED OPERATOR ACTIONS ARE REQUESTED HERE.

   DISK FILE NAME:

   THE DISK FILE NAME CAN ALSO BE PRECEEDED WITH THE DISK DRIVE ID IF OTHER THAN THE DEFAULT, IE. B:BR123456.DAT

   IF NO DRIVE IS SPECIFIED, FILES WILL BE WRITTEN TO THE DEFAULT DRIVE.

5. THE 'BRIDGE' PROGRAM CAN NOW BE USED TO CALCULATE A BIT MAP AND PLOT THE RESULTS ON THE ATTACHED TI 855 PRINTER.

A PROMPT FOR THE DISK FILE NAME WITH THE SURVEY DATA IS ISSUED. THE BRIDGE NAME IS DISPLAYED ON THE SCREEN, AND A PROMPT IS ISSUED FOR THE DELAMINATION VOLTAGE WITH 400 MV BEING THE DEFAULT VALUE WHICH IS USED IF THE RETURN KEY IS SIMPLY PUSHED RATHER THAN A NUMBER.

AT THIS TIME THE DATA IS READ IN, AVERAGES COMPUTED, AND DELAMINATION DECISIONS ARE WRITTEN TO A MEMORY BIT MAP. THE LONGER THE DATA FILE, THE LONGER THIS TAKES. THE DISK DRIVE MAY BE OBSERVED TO DETERMINE IF RECORDS ARE STILL BEING READ IN AND PROCESSED PRIOR TO COMMENCEMENT OF PLOTTING.

PLOTTING CHOICES OF PRINTER TYPE ARE DISPLAYED WITH THE CURRENT CHOICE HIGHLIGHTED ON THE SCREEN.

THREE TYPES OF PLOTS MAY NOW BE SELECTED BY TYPING

1: 8 INCH WIDE GRAPHICS PLOT
2: 4 DOTS PER 3 INCH GRAPHICS PLOT WITH EQUAL SCALES
OR P: PERCENTAGES IN EACH 4' X 3' GRID BOX

THE SAME DATA CAN BE PLOTTED AGAIN WITH DIFFERENT CHOICES FOR DELAMINATION THRESHOLD VOLTAGE.

A.0 APPENDICES : TECHNICAL REFERENCE MATERIALS

A.1 DELAMTECT SCHEMATICS AND MODIFICATIONS

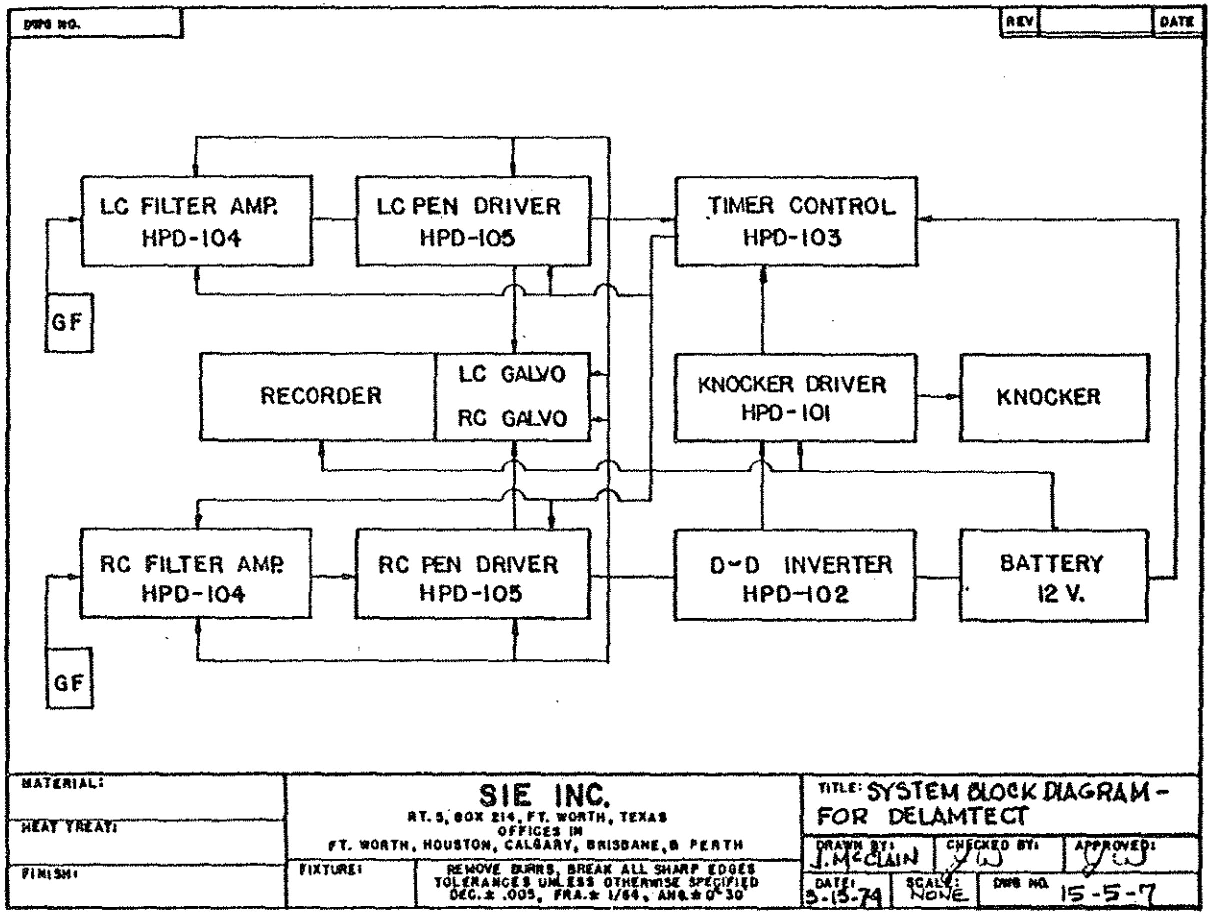
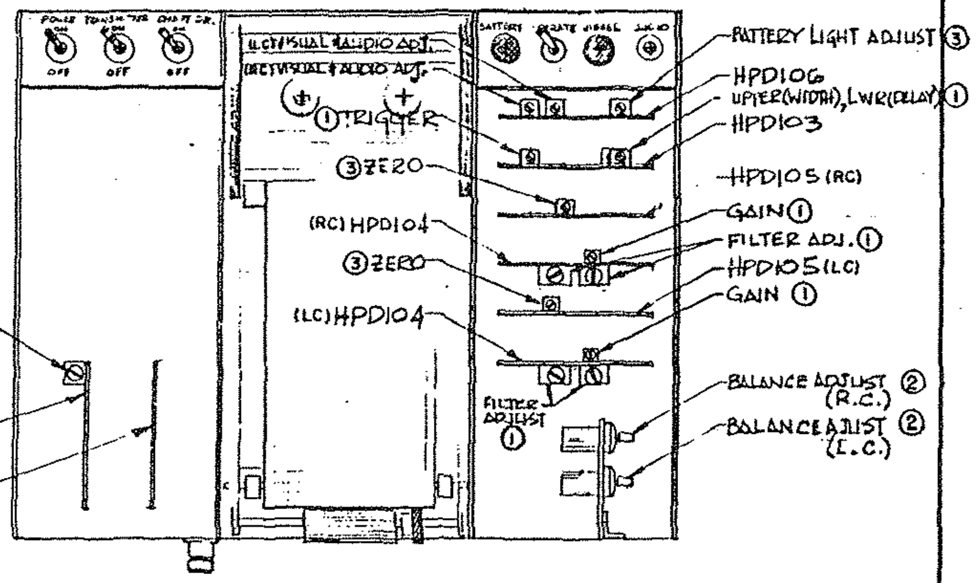

DWG NO.
REV
DATE
LC FILTER AMP.
HPD-104
LC PEN DRIVER
HPD-105
TIMER CONTROL
HPD-103
GF
RECORDER
LC GALVO
RC GALVO
KNOCKER DRIVER
HPD-101
KNOCKER
RC FILTER AMP.
HPD-104
RC PEN DRIVER
HPD-105
D-D INVERTER
HPD-102
BATTERY
12 V.
GF
MATERIAL:
HEAT TREAT:
FINISH:
SIE INC.
RT. 5, BOX 214, FT. WORTH, TEXAS
OFFICES IN
FT. WORTH, HOUSTON, CALGARY, BRISBANE, & PERTH
FIXTURE:
REMOVE BURRS, BREAK ALL SHARP EDGES
TOLERANCES UNLESS OTHERWISE SPECIFIED
DEC. ± .005, FRA. ± 1/64, ANG. ± 0° 30'
TITLE: SYSTEM BLOCK DIAGRAM - FOR DELAMTECT
DRAWN BY: J. McCLAIN
CHECKED BY: YW
APPROVED: YW
DATE: 5-15-74
SCALE: NONE
DWG NO. 15-5-7

Figure 15

DWG NO. 15-5-6

REV		DATE



① Preset at factory. Consult SIE prior to adjustment

② See Section 5 for adjustment instructions

③ See Section 3 for adjustment instructions

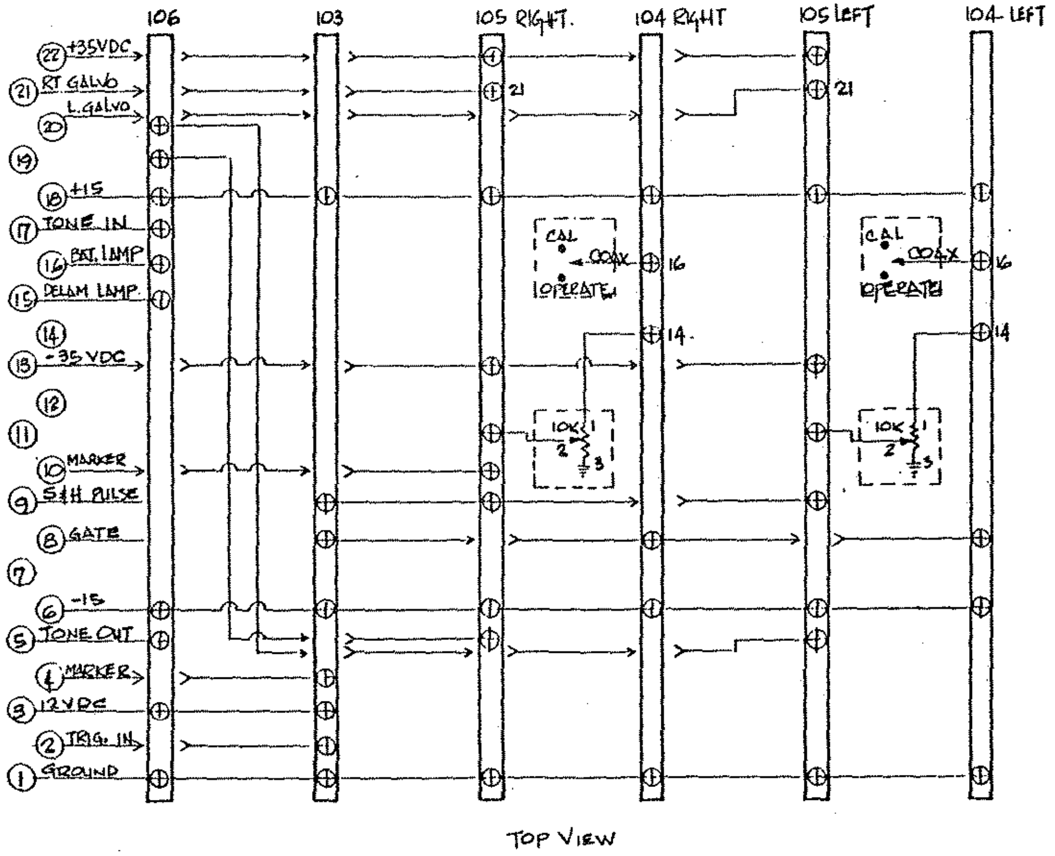
LC - Left Channel

RC - Right Channel

PLAN VIEW OF ELECTRONICS UNIT
LOCATION OF CIRCUIT BOARDS

MATERIAL:	SIE INC. RT. 5, BOX 214, FT. WORTH, TEXAS OFFICES IN FT. WORTH, HOUSTON, CALGARY, BRISBANE, & PERTH		TITLE: "DELAMTECT"		
HEAT TREAT:			DRAWN BY: J. McCLAIN	CHECKED BY:	APPROVED:
FINISH:	FIXTURE:	REMOVE BURRS, BREAK ALL SHARP EDGES TOLERANCES UNLESS OTHERWISE SPECIFIED DEC. ± .005, FRA. ± 1/64, ANG ± 0° 30'	DATE: 5-21-74	SCALE: NONE	DWG NO. 15-5-6

Figure 14



JAN 14 1983

NEXT ASSY	REQD	MATERIAL	TOLERANCES FRACTIONS ± .015 .XX ± .01 .XXX ± .00	RADIATION ENG. & MFG. CO. FORT WORTH, TEXAS
		FIXTURE:		TITLE: DIAGRAM FOR P.C. BOARDS — DELAMTEC
		DRAWN BY: J. McCLAIN	DATE: 1-14-76	SCALE: NONE DWG NO. 15-5-17

Fig. 18

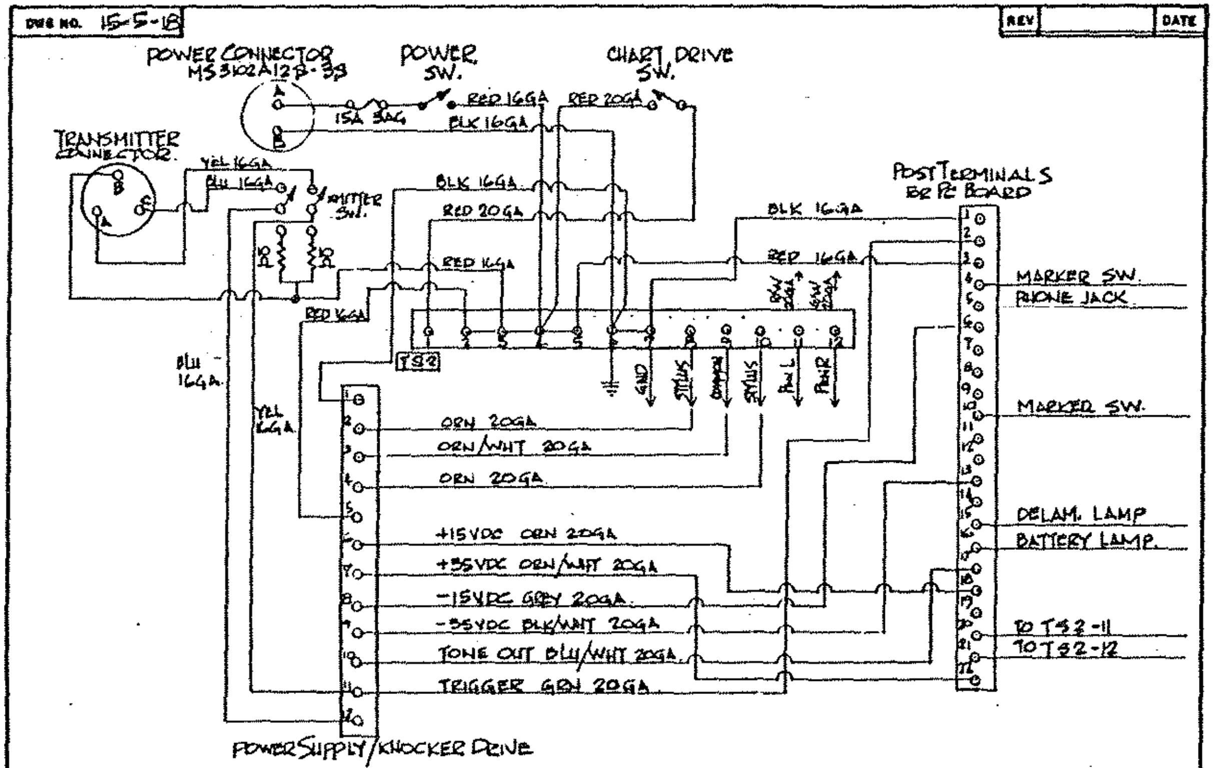


Fig. 19

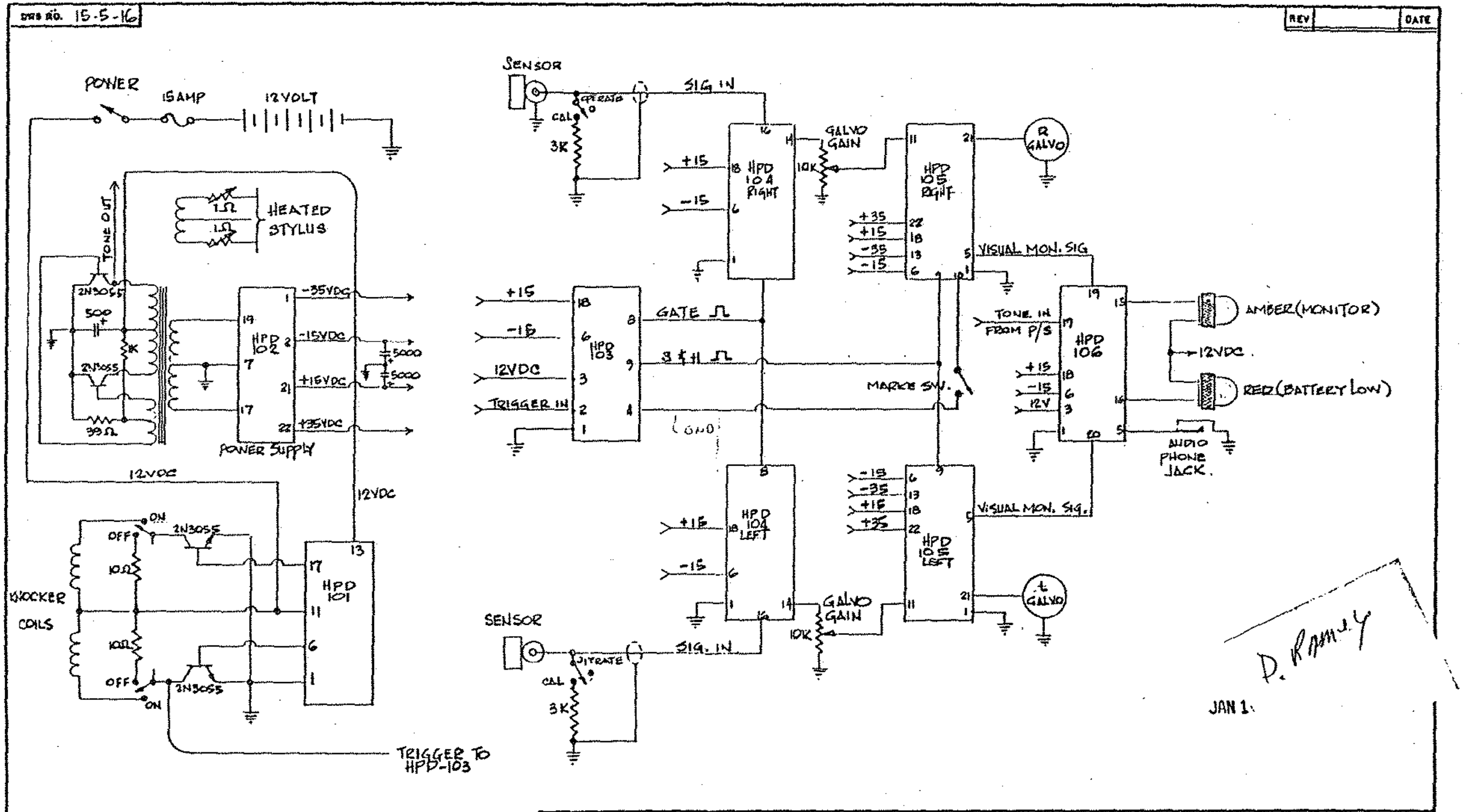
MATERIAL:	**SIE INC.** RT. 5, BOX 214, FT. WORTH, TEXAS OFFICES IN FT. WORTH, HOUSTON, CALGARY, BRISBANE, & PERTH	TITLE: ELECTRONICS CABINET -DELAMTEC
HEAT TREAT:		DRAWN BY: J. MCCLAIN | CHECKED BY: | APPROVED:
FINISH:	FIXTURE: | REMOVE BURRS, BREAK ALL SHARP EDGES TOLERANCES UNLESS OTHERWISE SPECIFIED DEC. ± .005, FRA. ± 1/64, ANG. ± 0° 30'	DATE: 12-8-76 | SCALE: NONE | DWG NO 15-5-18

5"

8.5"

1"

11"

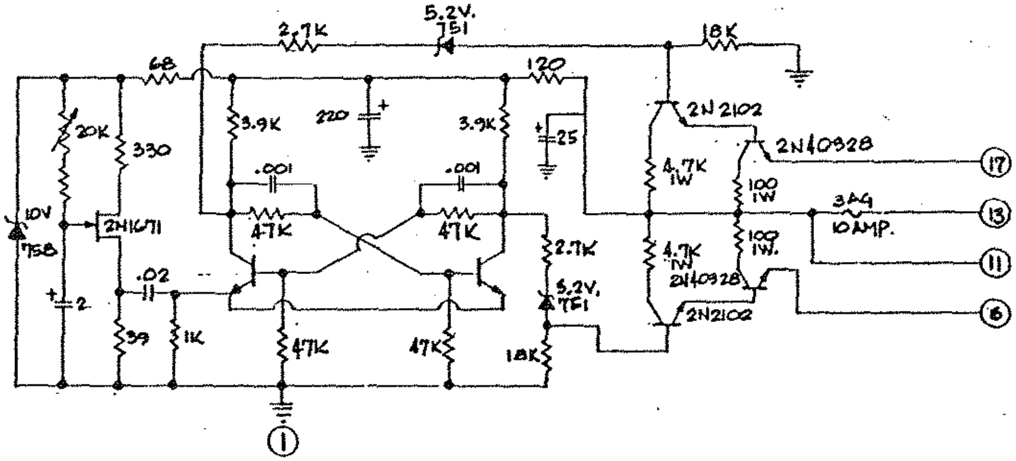


MATERIAL:	SIE INC. RT. 5, BOX 214, FT. WORTH, TEXAS OFFICES IN FT. WORTH, HOUSTON, CALGARY, BRISBANE, & PERTH		TITLE: SYSTEMS SCHEMATIC FOR DELAMTEC.		
HEAT TREAT:			DRAWN BY: J. McCLAIN	CHECKED BY:	APPROVED:
FINISH:	FIXTURE:	REMOVE BURRS, BREAK ALL SHARP EDGES TOLERANCES UNLESS OTHERWISE SPECIFIED DEC. ± .005, FRA. ± 1/64, ANG. ± 0°30'	DATE: 1-14-76	SCALE: NONE	DWG NO. 15-5-16

Fig. 17

DWG NO. 15-5-19

REV		DATE



PIN USAGE

1	GROUND
6	TO 2N3055B
11	FUSED 12V. OUT
13	12V. IN
17	TO 2N3055B

MATERIAL:	**SIE INC.** RT. 5, BOX 214, FT. WORTH, TEXAS OFFICES IN FT. WORTH, HOUSTON, CALGARY, BRISBANE, & PERTH	TITLE: KNOCKER DRIVER HPD-101, DELAMTEC
HEAT TREAT:		DRAWN BY: J. McClain — CHECKED BY: — APPROVED:
FINISH:	FIXTURE: — REMOVE BURRS, BREAK ALL SHARP EDGES TOLERANCES UNLESS OTHERWISE SPECIFIED DEC. ± .005, FRA. ± 1/64, ANG. ± 0°30'	DATE: 10-18-79 — SCALE: NONE — DWG NO. 15-5-19

FIG. 20

5"

8.5"

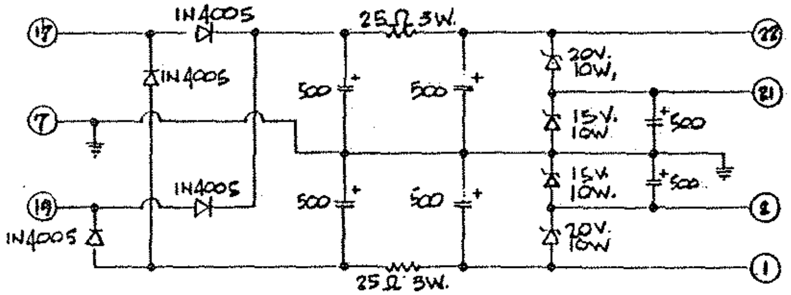
1"

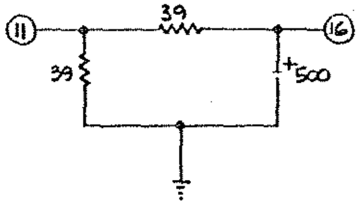
11"

JAN 14 1983

DWG NO. 15-5-20

REV		DATE





PIN USAGE

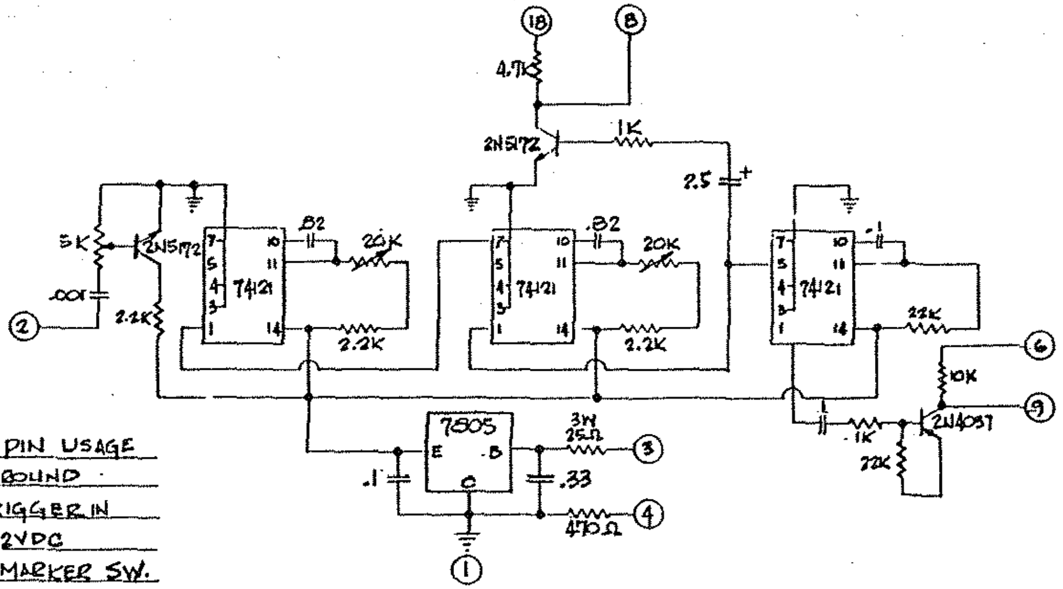
1	-35V.D.C.
2	-15V.DC.
7	GND
11	TO RENY 4
16	12V.D.C.
17	AC IN
19	AC IN
21	+15V.D.C.
22	+35VDC

Fig. 21

MATERIAL:	**SIE INC.** RT. 5, BOX 214, FT. WORTH, TEXAS OFFICES IN FT. WORTH, HOUSTON, CALGARY, BRISBANE, & PERTH	TITLE: POWER SUPPLY — HPD-102, DELAMTEC
HEAT TREAT:		DRAWN BY: J. McCLAIN · CHECKED BY: · APPROVED:
FINISH:	FIXTURE: · REMOVE BURRS, BREAK ALL SHARP EDGES TOLERANCES UNLESS OTHERWISE SPECIFIED DEC. ± .005, FRA. ± 1/64, ANG. ± 0° 30'	DATE: 10-20-75 · SCALE: NONE · DWG NO. 15-5-20

DWG NO. 15-5-21

REV | DATE



PIN USAGE

1	GROUND
2	TRIGGER IN
3	12VDC
4	TO MARKER SW.
6	-15VDC
8	GATE PULSE ⎍
9	8, 4 H. PULSE ⎍
18	+15VDC

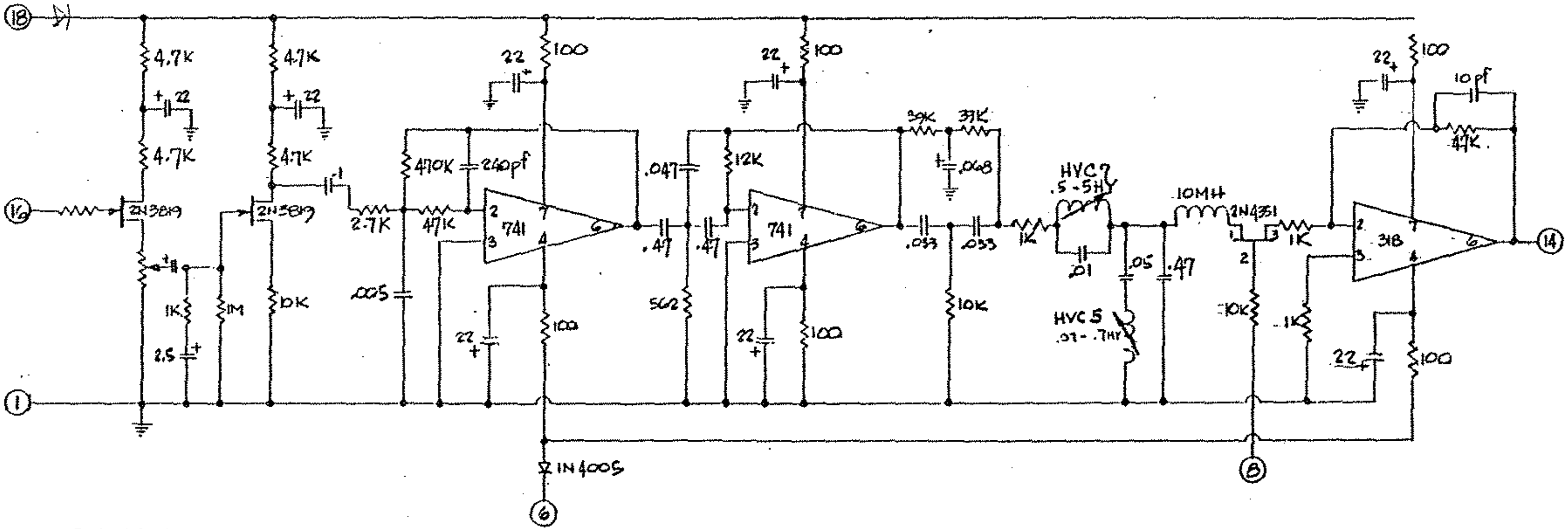
Fig. 22

MATERIAL:	**SIE INC.** RT. 5, BOX 214, FT. WORTH, TEXAS OFFICES IN FT. WORTH, HOUSTON, CALGARY, BRISBANE, & PERTH	TITLE: PULSE GENERATOR - HPD -103, DELAMTEC
HEAT TREAT:		DRAWN BY: J. McCLAIN · CHECKED BY: · APPROVED:
FINISH:	FIXTURE: · REMOVE BURRS, BREAK ALL SHARP EDGES TOLERANCES UNLESS OTHERWISE SPECIFIED DEC. ± .005, FRA. ± 1/64, ANG. ± 0° 30'	DATE: · SCALE: NONE · DWG NO. 15-5-21

JAN 14 1983

DWG NO. 15-5-22

REV		DATE



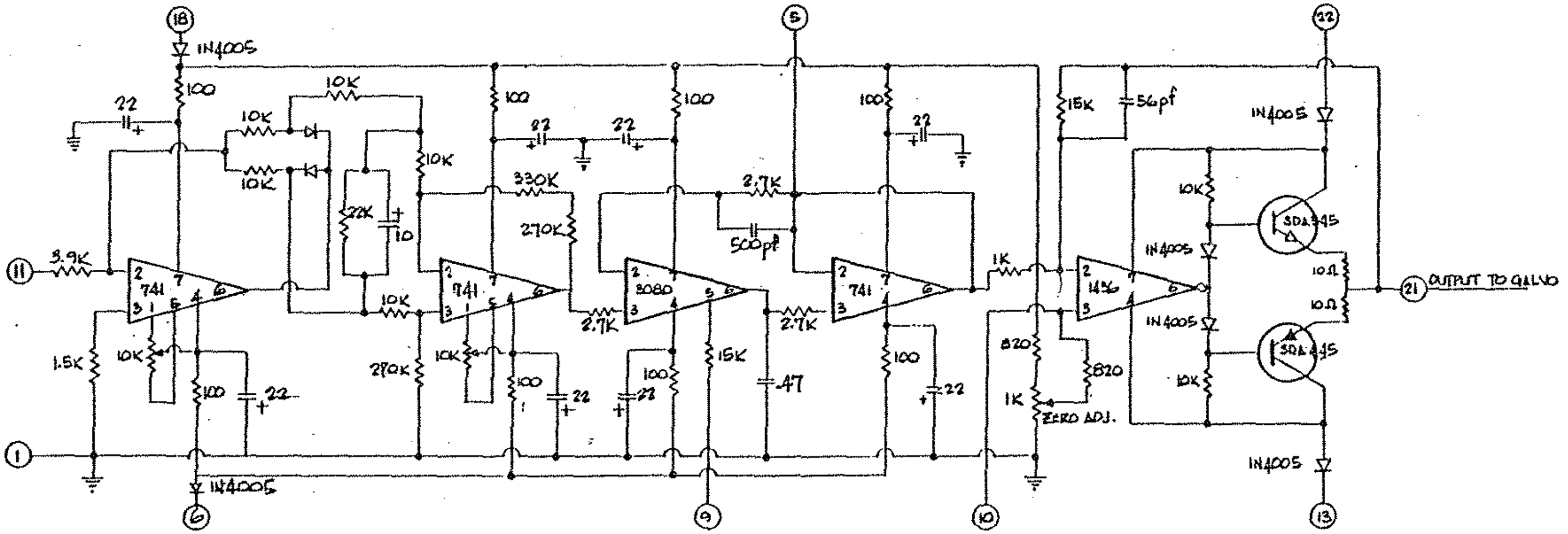
PIN USAGE

1	GND
6	-15VDC
8	GATE ⎍
14	OUTPUT
16	SIGNAL IN
18	+15VDC

JAN 14 1983

MATERIAL:	**SIE INC.** RT. 5, BOX 214, FT. WORTH, TEXAS OFFICES IN FT. WORTH, HOUSTON, CALGARY, BRISBANE, & PERTH	TITLE: AMPLIFIER/FILTER BOARD HPD-104, DELAMTEC		
HEAT TREAT:		DRAWN BY: J.McCAIN	CHECKED BY:	APPROVED:
FINISH:	FIXTURE: REMOVE BURRS, BREAK ALL SHARP EDGES TOLERANCES UNLESS OTHERWISE SPECIFIED DEC. ± .005, FRA. ± 1/64, ANG. ± 0° 30'	DATE: 1-14-76	SCALE: NONE	DWG NO. 15-5-22

Fig. 23



Pin Usage

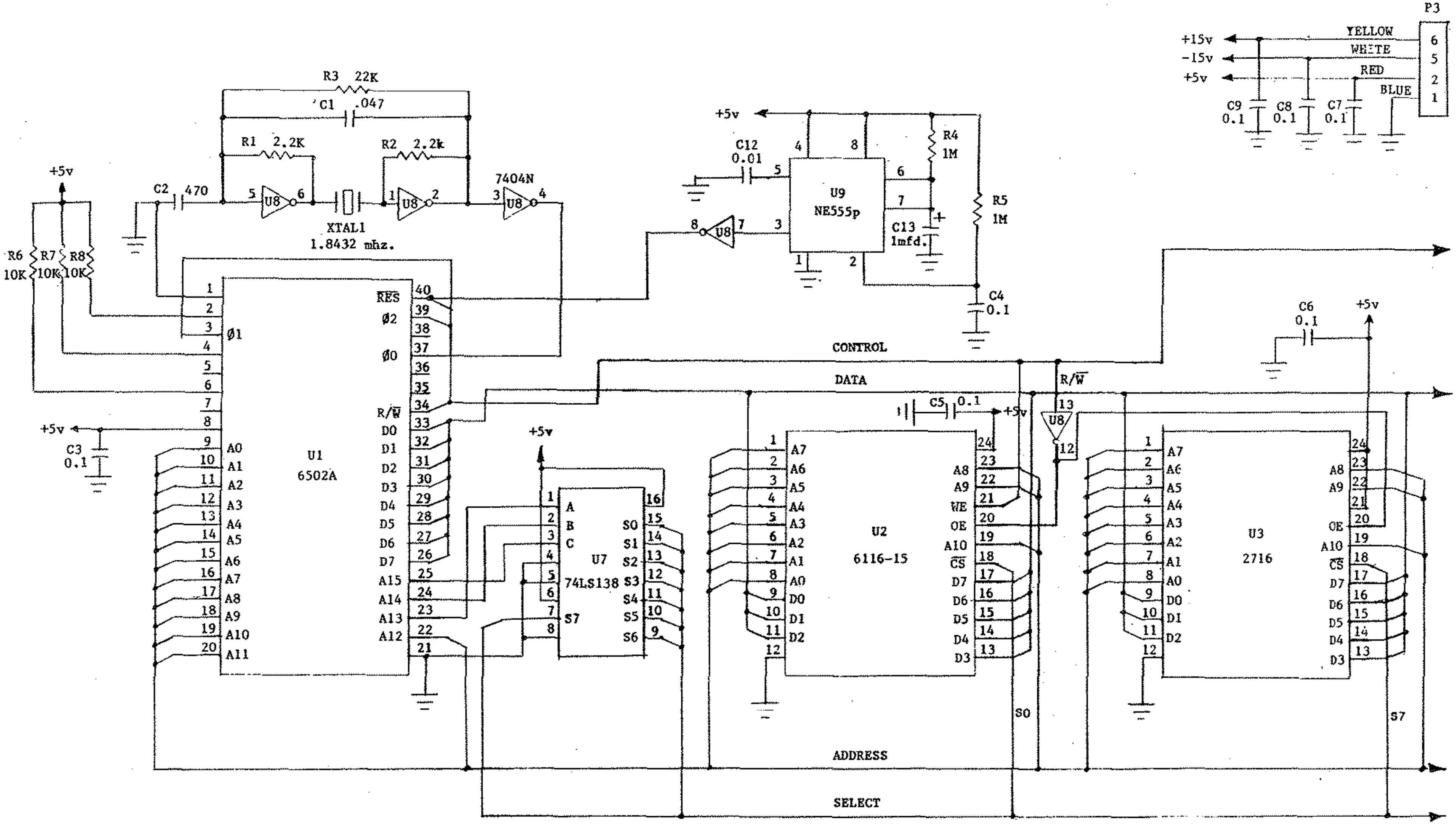
1	GROUND
5	MONITOR SIG.
6	-15VDC
9	S+H PULSE ⎍
10	MARKER SW.
11	INPUT
13	-35VDC.
18	+15VDC.
22	+35VDC.

JAN 14 1983

NEXT ASSY	REQD	MATERIAL	TOLERANCES	
			FRACTIONS ± .015 .XX ± .01 .XXX ± .00	RADIATION ENG. & MFG. CO. FORT WORTH, TEXAS
		FIXTURE:		TITLE: GALVO DRIVE BOARD – HPD-105, DELAMTEC
		DRAWN BY: J. McCLAIN	DATE: 12-9-75	SCALE: NONE DWG NO. 15-5-23

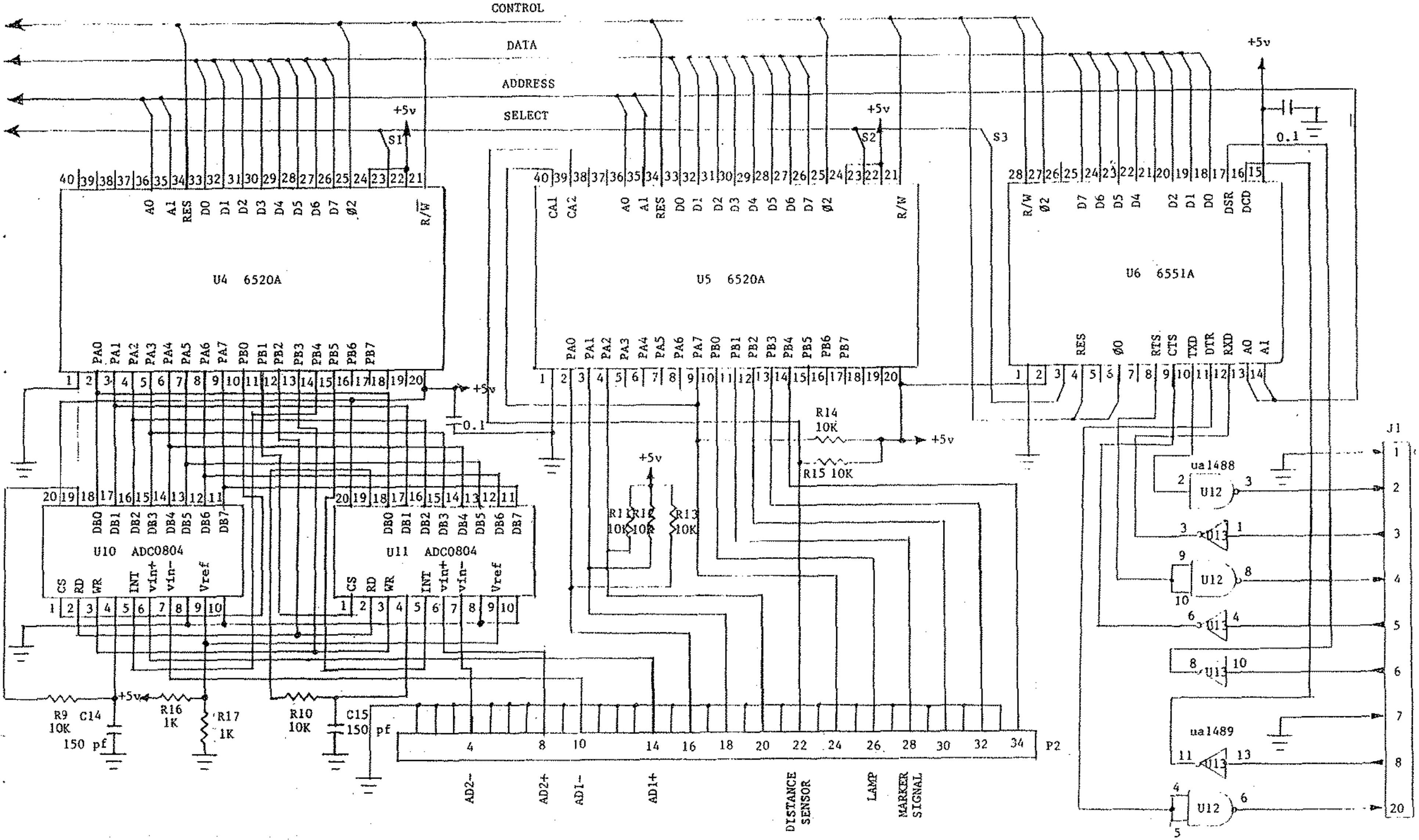
Fig. 24

A.2 D & D DATA ACQUISITION SYSTEM



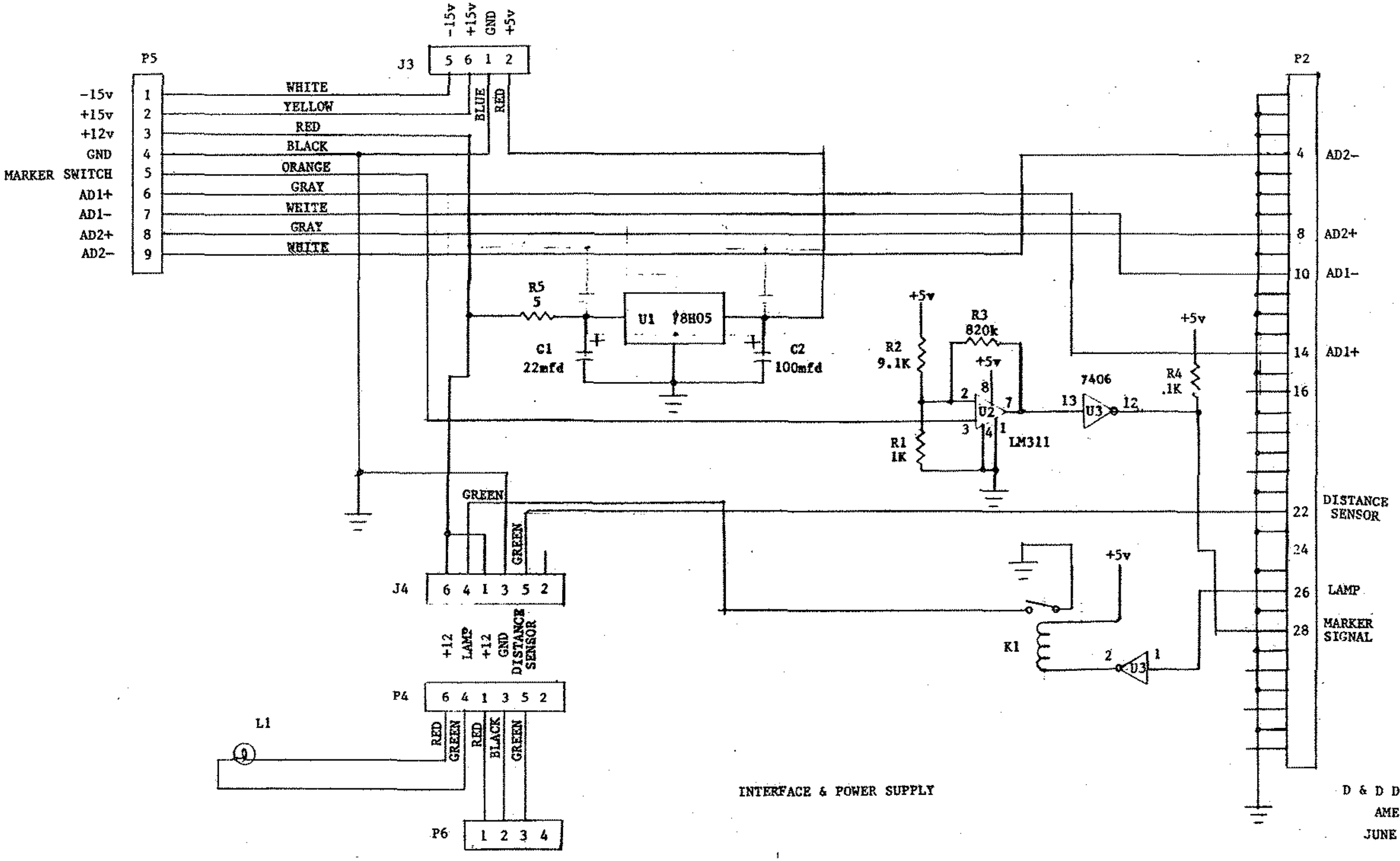
6502 A/D BOARD
SHT 1

D & D DIGITAL SYSTEMS
AMES, IOWA
JUNE 30, 1985



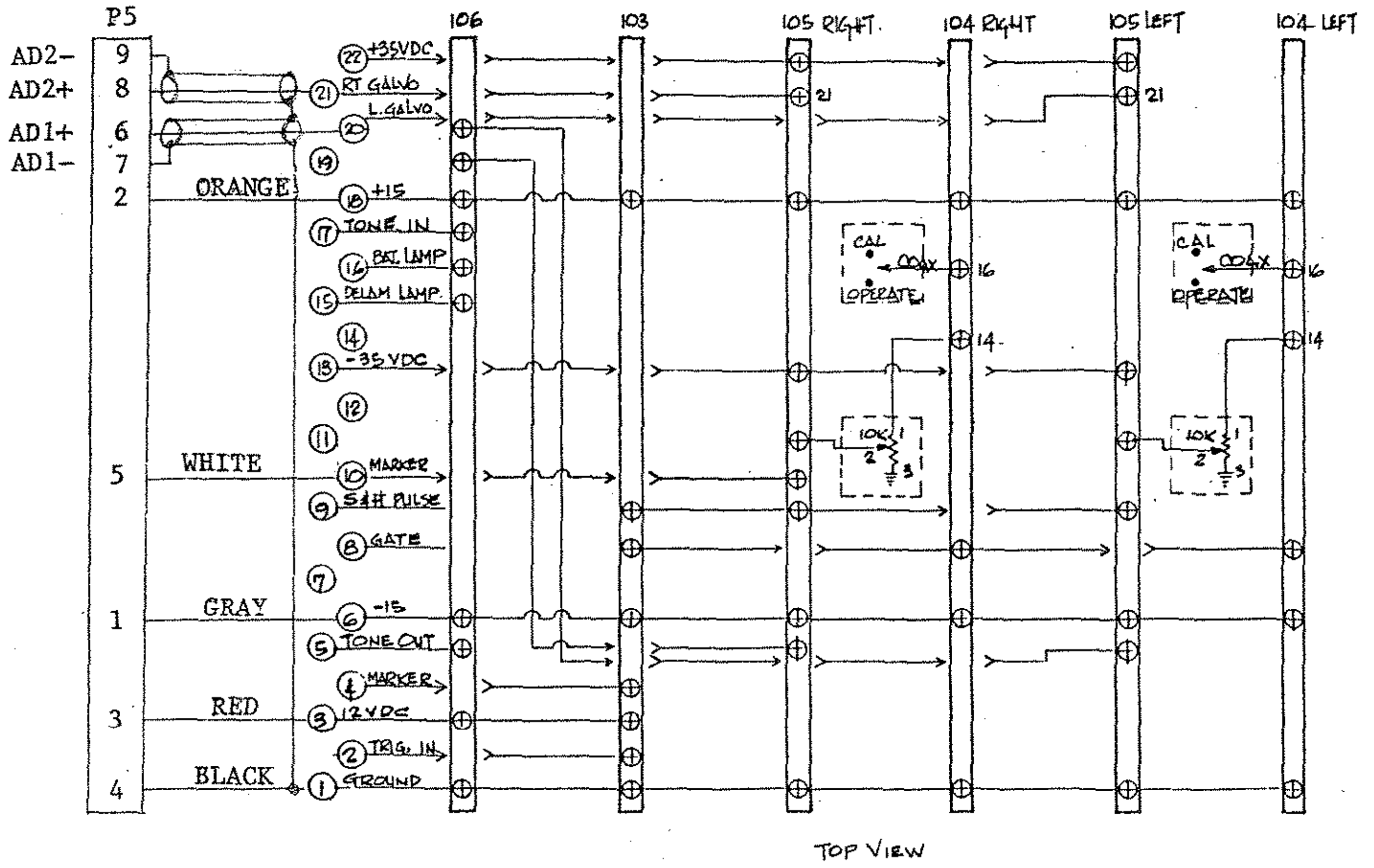
6502 A/D BOARD SHT 2

D & D DIGITAL SYSTEMS
AMES, IOWA
JUNE 30, 1985



INTERFACE & POWER SUPPLY

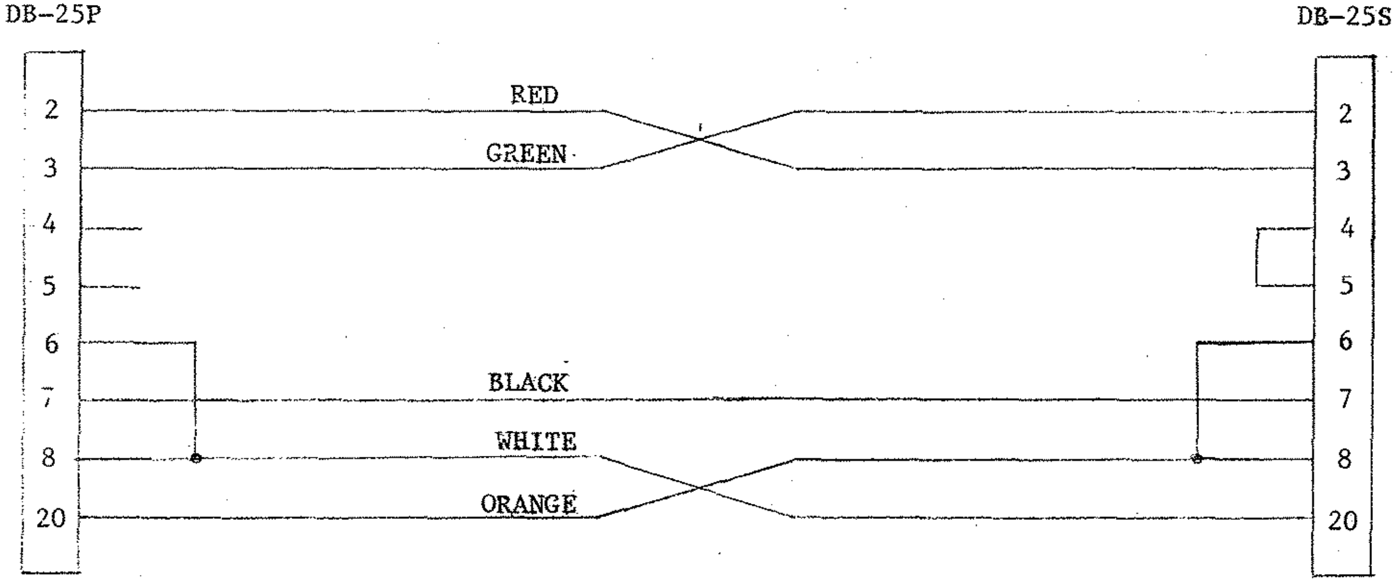
D & D DIGITAL SYSTEMS
AMES, IOWA
JUNE 30, 1985



JAN 14 1983

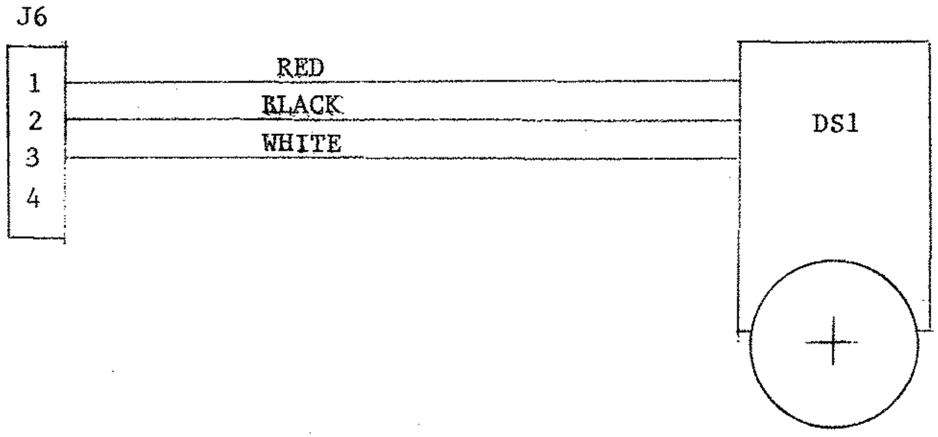
NEXT ASSY	REQD	MATERIAL	TOLERANCES FRACTIONS ± .015 .XX ± .01 .XXX ± .00	RADIATION ENG. & MFG. CO. FORT WORTH, TEXAS
		FIXTURE:		TITLE: DIAGRAM FOR P.C. BOARDS — DELAMTEC
		DRAWN BY: J. McCLAIN	DATE: 1-14-76	SCALE: NONE DWG NO. 15-5-17

Fig. 18



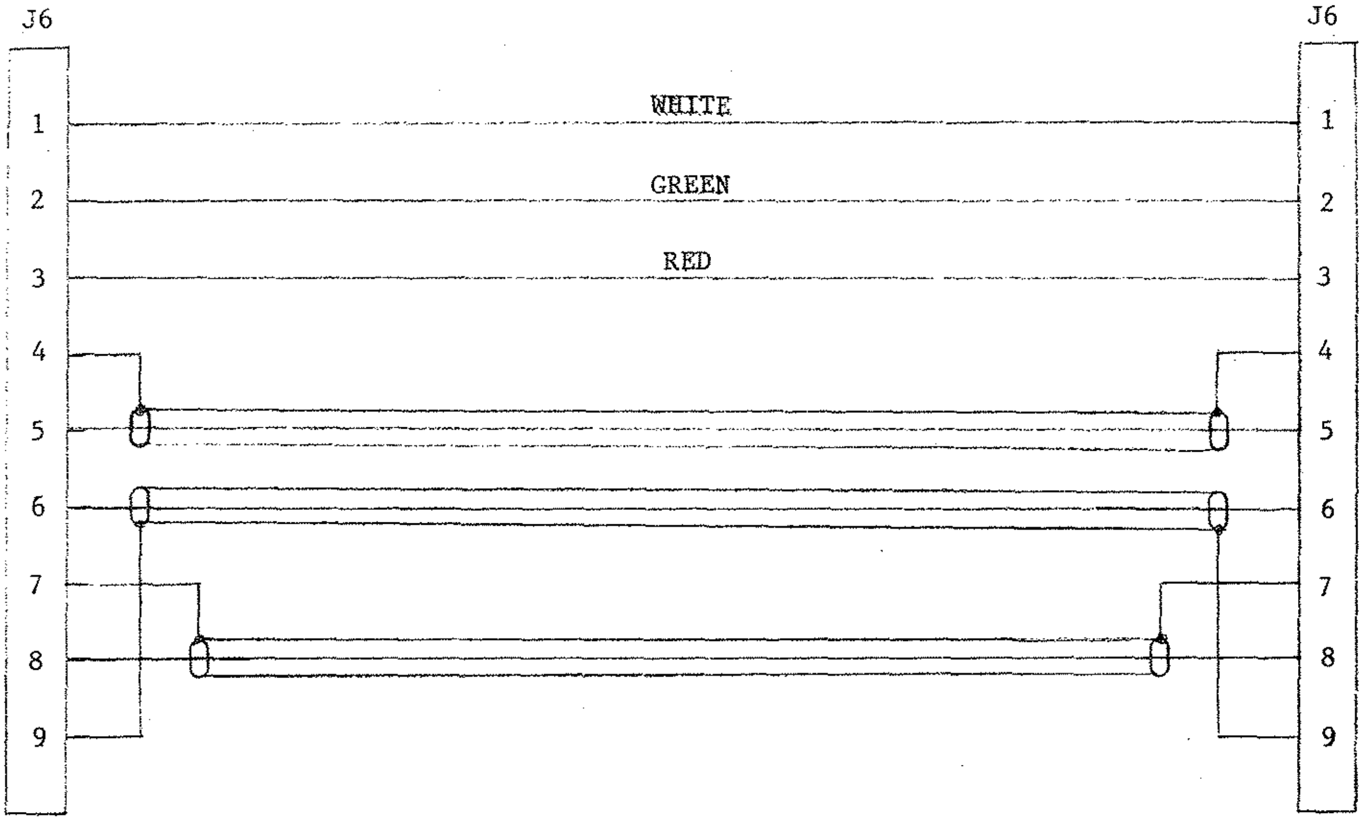
COMPUTER CABLE

D & D DIGITAL SYSTEMS
AMES, IOWA
JUNE 30, 1985



DISTANCE SENSOR

D & D DIGITAL SYSTEMS
AMES, IOWA
JUNE 30, 1985



SIGNAL & POWER CABLE

D & D DIGITAL SYSTEMS

AMES, IOWA

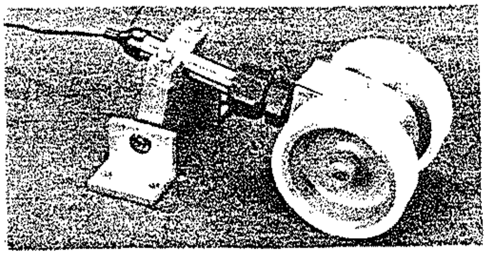
JUNE 30, 1985

RED LION CONTROLS
Willow Springs Circle, RD 5, York, Pa. 17402 • (717) 767-6511 • TWX: 510 657 4214 RLC YRK

BULLETIN NO. LSC-001
REVISED 12/12/83

MODEL LSC MEASURING WHEEL LENGTH SENSOR

AN ECONOMICAL ANSWER TO HIGH SPEED, UNI-DIRECTIONAL LENGTH MEASUREMENT FOR PAPER, TEXTILES, FILM, FOIL, FOAM & METAL STRIP PRODUCTS RUNNING THROUGH REWINDERS, PRINTING PRESSES, SLITTERS, SPOOLERS, SHEETERS & OTHER WEB, STRIP OR RIBBON HANDLING APPLICATIONS.



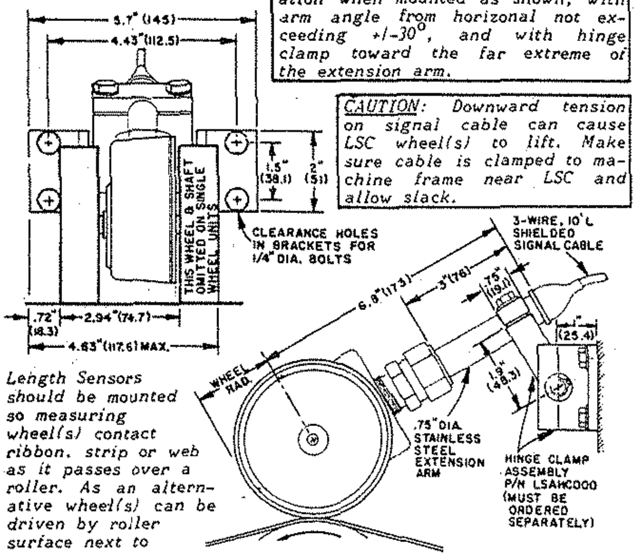
- CURRENT SINK OUTPUT
- AVAILABLE WITH:
  - ONE OR TWO WHEELS
  - ROUND, FLAT & KNURLED TREAD
  - ENGLISH & METRIC UNITS
- FOR MEASURING
  - INCHES, FEET, YARDS
  - CENTIMETERS, METERS
  - FT/MIN, YDS/MIN, MTRS/MIN

Model LSC Length Sensors are designed for accurate reliable operation at high speeds. A slotted, shaft-mounted disc in the cast aluminum housing is scanned by an L.E.D. and photo sensor. The resulting signal is amplified, shaped, and appears as a current-sinking output pulse, compatible with most RLC Counters, Motion Monitors and controls. The unit will operate in either direction of rotation provided the direction does not change. In applications where reversing occurs, the up/down counter direction must be controlled by an external switch contact to correspond to the direction of motion. (As an alternative an RPGB with quadrature output can be outfitted as a length sensor, see following page)

DIMENSIONS & MOUNTING

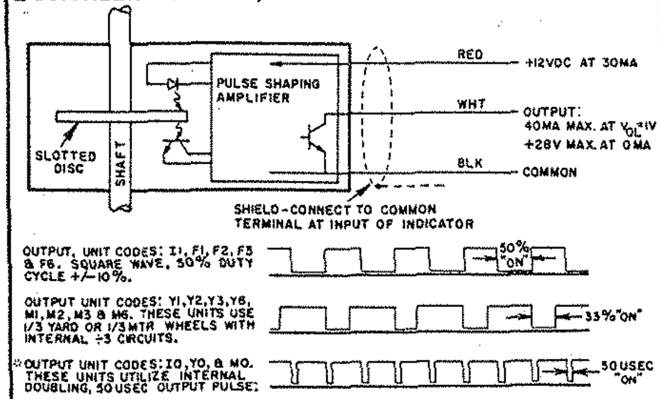
NOTE: Weight of LSC provides sufficient traction for accurate operation when mounted as shown, with arm angle from horizonal not exceeding +/-30°, and with hinge clamp toward the far extreme of the extension arm.

CAUTION: Downward tension on signal cable can cause LSC wheel(s) to lift. Make sure cable is clamped to machine frame near LSC and allow slack.


INCHES(MM)
5.7" (145)
4.43"(112.5)
1.5" (38.1)
2" (51)
THIS WHEEL & SHAFT OMITTED ON SINGLE WHEEL UNITS
CLEARANCE HOLES IN BRACKETS FOR 1/4" DIA. BOLTS
.72" (18.3)
2.94"(74.7)
4.63"(117.6) MAX.
3-WIRE, 10'L SHIELDED SIGNAL CABLE
6.8"(173)
3"(76)
.75" (19.1)
1" (25.4)
1.9" (48.3)
WHEEL RAD.
.75" DIA. STAINLESS STEEL EXTENSION ARM
HINGE CLAMP ASSEMBLY P/N LSAHC000 (MUST BE ORDERED SEPARATELY)

Length Sensors should be mounted so measuring wheel(s) contact ribbon, strip or web as it passes over a roller. As an alternative wheel(s) can be driven by roller surface next to material being measured.

EQUIVALENT CIRCUIT, CONNECTIONS & WAVEFORMS


SLOTTED DISC
SHAFT
PULSE SHAPING AMPLIFIER
RED
+12VDC AT 30MA
WHT
OUTPUT:
40MA MAX. AT V_OL=1V
+28V MAX. AT 0 MA
BLK
COMMON
SHIELD-CONNECT TO COMMON TERMINAL AT INPUT OF INDICATOR
OUTPUT, UNIT CODES: I1, F1, F2, F3 & F6. SQUARE WAVE, 50% DUTY CYCLE +/-10%.
50% "ON"
OUTPUT UNIT CODES: Y1,Y2,Y3,Y6, M1,M2,M3 & M6. THESE UNITS USE 1/3 YARD OR 1/3 MTR WHEELS WITH INTERNAL ÷3 CIRCUITS.
33% "ON"
*OUTPUT UNIT CODES: I0, Y0, & M0. THESE UNITS UTILIZE INTERNAL DOUBLING, 50USEC OUTPUT PULSE:
50USEC "ON"

* *Check counter or indicator using this output for compatibility. Cannot be frequency doubled externally.*

Unlike mechanical measuring wheels the LSC has no internal cam switches or dividing gears to wear or create drag. This means lighter wheel pressures since the only resistance the wheel encounters is due to very low bearing friction. Low wheel pressures in turn, means less wear on the wheel tread and less danger of marking the product.

Other Specifications include:

OUTPUT: NPN Open-Collector, current-limited at 40ma 28 VDC maximum.

MAX SHAFT SPEED: 3600 RPM (See wheel information, following page for wheel speed restrictions)

CONSTRUCTION: Cast Aluminum housing, Stainless Steel Tube extension. Oil impregnated sintered bronze bearings, lifetime lubricated. Shielded, 3-wire, signal cable, 10'long. Operating Temperature range 0°to 60°C, weight 1.8 lbs.

TO ORDER: ASSEMBLE COMPLETE PART NO. AS FOLLOWS:

PART NO.-**LSC** _ - _ _ - _ _ /A

NO. OF WHEELS
S for Single
D for Dual

UNITS CODE

WHEEL CODES
(See description of wheels & codes next page)
OO - Less Wheels
OR - Round Neoprene
OF - Flat Tread Polyurethane
OK - Flat Tread Knurled

DESIRED UNITS OF MEASURE	OUTPUT PULSE RATE (PULSES/UNIT)	INSERT 2-DIGIT CODE
INCHES	1/INCH	I1
INCHES IN 1/10THS	10/INCH	I0 **
FEET	1/FOOT	F1
FEET IN 1/10THS	10/FOOT	F2
FEET IN 1/100THS	100/FOOT	F3
YARDS	1/YARD	Y1
YARDS IN 1/10THS	10/YARD	Y2
YARDS IN 1/100THS	100/YARD	Y3
METERS	1/METER	M1
METERS IN 1/10THS	10/METER	M2
METERS IN 1/100THS (CM)	100/METER	M3
*FEET/MIN	60/FOOT	F6
*YARDS/MIN	60/YARD	Y6
*YARDS/MIN IN 1/10THS	600/YARD	Y0 **
*METERS/MIN	60/METER	M6
*METERS/MIN IN 1/10THS	600/METER	M0 **

* *Primarily used for Tachometers & Motion Monitors*

** *Output from I0, Y0 & M0 units code are 50usec pulses (See waveforms at left). Check compatibility of instrument, control, or counter for use with this waveform. Cannot be frequency doubled externally.*

ABOUT ACCURACY IN LENGTH SENSOR APPLICATIONS

USING THE MODEL RMX RATE MULTIPLIER FOR CORRECTION, UNITS CONVERSION & WHEEL WEAR COMPENSATION

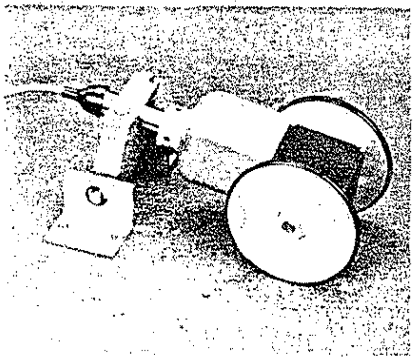
Length Sensor wheels have a nominal accuracy of 0.1% which means that under ideal conditions the measurement should be accurate to within 1 part in 1000. Ideal conditions are realized when measuring hard, thin and strong materials such as metal strip, foil or hard paper. However, materials that are thick, soft, spongy or elastic can present problems in obtaining true readings directly, since the surface geometry may not be predictable.

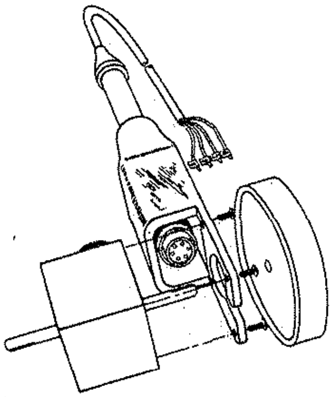
The great majority of these situations can be accommodated by feeding the output from the Length Sensor to a Model RMX Rate Multiplier prior to counting or speed measurement. The Rate Multiplier applies a presettable correction multiplier to the pulse train by dropping or adding pulses as required to obtain a corrected measurement. The presettable multiplier is entered in via thumbwheel switches to an accuracy of four decimal places.

In addition to correcting for elastic and compliance errors, the Rate Multiplier can also be used in applications where English/Metric Conversions must be made and for compensating for wheel wear. (For more information see data sheet on the Model RMX Rate Multiplier in Accessories section of the Catalog)

LENGTH SENSOR CONVERSION BRACKET (P/N LSCB-1000)

ADAPTS RPGB ROTARY PULSE GENERATOR TO LENGTH MEASUREMENT





- FOR BI-DIRECTIONAL MOTION APPLICATIONS REQUIRING QUADRATURE

- FOR FINE RESOLUTION, HIGH-PULSE-RATE APPLICATIONS

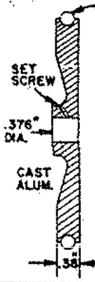
This conversion bracket allows the customer to assemble a custom length sensor by purchasing the following items separately.

1. Length Sensor Conversion Bracket (P/N LSCB-1000)
2. RPGB with appropriate PPR and Single Channel or Quadrature Output (See RPGB data sheet, Section D of the Catalog)
3. One or two measuring wheels (Listed below)
4. Hinge Clamp Assembly (Listed below)

The tubular arm length of this bracket, related to the wheel axis center-line of the RPGB is 6.8" similar to the LSC (see previous page). The 10' long, 4-wire, shielded cable (included with conversion bracket) has the same color coding as described for the RPGB cable P/N CCA-RPG-01 in the RPGB data sheet. Screws for mounting the conversion bracket to the RPGB are included. To order see table below.

SEPARATE LENGTH MEASURING WHEELS

WHEEL CODE **OR**

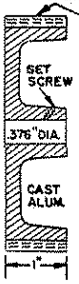


Round Section Replaceable Tire .210" Section Dia. Black Neoprene

FOR USE ON: Metal, paper, foil, film and hard plastics. Line contact on material being measured, convenient when available measuring track is narrow or for measuring on end of roller beside passing material.

Max. Speed: 3000 RPM

WHEEL CODE **OF**

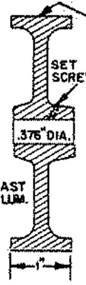


White, Smooth Polyurethane Tread

FOR USE ON: Soft, smooth materials such as soft paper, matting, cardboard, fine weave textiles. Broad wheel tread minimizes contact pressure and white polyurethane tread minimizes marking.

Max. Speed:
Standard wheels - 600 RPM
Balance to 3000 RPM
Available on special order.

WHEEL CODE **OK**



Diamond Knurled Aluminum Tread

FOR USE ON: Rubber, coarse weave fabrics, rough wood surfaces, foam, insulation.

MAX. SPEED:
Standard wheel - 600 RPM
Balance to 3000 RPM
Available on special order.

SELECTING APPROPRIATE WHEEL SIZE & PPR (Pulses Per Rev.) OF ROTARY PULSE GENERATOR

When the desired output of an RPGB and wheel combination is either in feet or inch units, selection of the proper combination is relatively straight forward. For example, with a 1-foot wheel circumference a 1 PPR Rotary Pulse Generator will deliver 1 pulse/ft, 12 PPR would deliver 12 pulses/ft (or 1 pulse/inch); 100 PPR would yield 100 pulses/ft; and 120 PPR would permit measuring to 1/10th of an inch (1/120th of a foot).

Measuring in yards or meters however is a bit more involved since a 1 yard or 1 meter circumference wheel would be prohibitively large. Instead, 4/10 yard and 4/10 meter wheels can be used in conjunction with RPGB's in either of two ways. First, RPGB's with PPR's of 1, 10 and 100 can be used with a Model BDMD (Bi-Directional Motion Decoder, See Accessories Section of catalog). The BDMD can quadruple the quadrature signal input and deliver 4, 40 or 400 PPR respectively when used with these RPGB's, allowing measurement of yards or meters in increments of 1/10th, 1/100th or 1/1000th. The second approach would be to use an RPGB with 4, 40 or 400 PPR (available on special order) to generate the desired measuring increments directly. With either of these approaches, the largest measuring increment available is 1/10th of a meter or yard, however this is rarely a handicap when 6-digit counting capacity is available. (Model LSC's described on the previous page, are available in pulse rates of 1 pulse/yard and 1 pulse/meter. This is done by using a 1/3 yard or 1/3 meter wheel with a PPR of 1 internally divided by 3. This capability is not available with Model RPGB Rotary Pulse Generators)

ORDERING INFORMATION WHEELS & REPLACEMENT TIRES FOR CODE - OR WHEELS

WHEEL CODE	CIRCUMFERENCE	PART NO.
OR	1 foot (1/3 yd)	**WF-1000-OR** /A
	1/3 meter	**WM-0333-OR** /A
	4/10ths yard	**WY-0400-OR** /A
	4/10ths meter	**WM-0400-OR** /A
OF	1 foot (1/3 yd)	**WF-1000-OF** /A
	1/3 meter	**WM-0333-OF** /A
	4/10ths yard	**WY-0400-OF** /A
	4/10ths meter	**WM-0400-OF** /A

WHEEL CODE	CIRCUMFERENCE	PART NO.
OK	1 foot (1/3 yd)	**WF-1000-OK**/A
	1/3 meter	**WM-0333-OK**/A
	4/10ths yard	**WY-0400-OK**/A
	4/10ths meter	**WM-0400-OK**/A
REPLACEMENT TIRES FOR OR WHEELS	1 foot (1/3 yd)	**TOR-F-1000**/A
	1/3 meter	**TOR-M-0333**/A
	4/10ths yard	**TOR-Y-0400** /A
	4/10ths meter	**TOR-M-0400**/A

ACCESSORIES

DESCRIPTION	PART NO.
LENGTH SENSOR CONVERSION BRACKET FOR RPGB	**LSCB-1000** /A
HINGE CLAMP ASSEMBLY FOR MODEL LSC & CONVERSION BRACKET (above)	**LSA-HCO-00** /A ✓

PARTS LIST

6502 A/D BOARD

COMPONENT NUMBER OR QUANTITY	DESCRIPTION
C1	0.047 ufd. Capacitor
C2	470 pf. Mica Capacitor
C3,C4,C5,C6,C7,C8,C9,C10,C11	0.1 ufd. Ceramic Capacitor
C12	0.01 ufd. Ceramic Capacitor
C13	1 ufd. 50 v. Electrolytic Capacitor
C14,C15	150 pf. Mica Capacitor
J1	DB-25S Rt. Angle P.C. Mount Connector
P2	34 Pin P.C. Mount Header
P3	6 Pin Amp Mate-N-Lock w/pins
R1,R2	2.2K Ohm 1/4 Watt 5% Resistor
R3	22K Ohm 1/4 Watt 5% Resistor
R4,R5	1M Ohm 1/4 Watt 5% Resistor
R6,R7,R8,R9,R10,R11,R12,R13,R14,R15	10K Ohm 1/4 Watt 5% Resistor
R16,R17	1K Ohm 1/4 Watt 1% Resistor
U1	65C02A Microprocessor I.C.
U2	6116-15 Ram I.C.
U3	2716 Eprom I.C.
U4,U5	68C21A Peripheral Interface Adaptor
U6	6551A Communications Interface I.C.
U7	SN74LS138N TTL I.C.
U8	SN7404 TTL I.C.
U9	NE555P Timer I.C.
U10,U11	ADC0804LCN A/D Converter
U12	ua1488 RS-232c Driver I.C.
U13	ua1489 RS-232c Receiver I.C.
XTAL1	1.8432 Mhz. Crystal

Interface Cable:

2	34 Contact Ribbon Socket Connectors
3 1/2"	28 AWG 34 Conductor Ribbon Cable

Tape Recorder Cable:

2	DB-25P Ribbon Connectors
2'	28 AWG 25 Conductor Ribbon Cable

Hardware:

4	4-40 X 3/8" Hex Posts
8	4-40 X 3/8" Machine Screws
1	24 Pin I.C. Socket
1/2'	22 AWG Red Hookup Wire
1/2'	22 AWG Yellow Hookup Wire
1/2'	22 AWG Blue Hookup Wire
1/2'	22 AWG White Hookup Wire

INTERFACE & POWER SUPPLY BOARD

COMPONENT NUMBER OR QUANTITY	DESCRIPTION
C1	22 ufd. 25 v. Tantalum Capacitor
C2	100 ufd. 25 v. Electrolytic Capacitor
J3,J4	6 Pin Amp Mate-N-Lock Connector w/sockets
K1	Reed Relay-Radio Shack 275-232
L1	12v. Lamp Holder-Radio Shack 272-324
P2	34 Pin P.C. Mount Header
P4	6 Pin Amp Mate-N-Lock Connector w/pins
P5	9 Pin Receptacle-Amp CPC w/pins
P6	4 Pin Receptacle-Amp CPC w/pins
R1,R4	1K Ohm 1/4 Watt 5% Resistor
R2	9.1K Ohm 1/4 Watt 5% Resistor
R3	820K Ohm 1/4 Watt 5% Resistor
R5	5 Ohm 5 Watt 1% Resistor
U1	ua78H05 Voltage Regulator
U2	LM311N Comparator I.C.
U3	SN7406N TTL I.C.
	Hardware:
1	#1487 Lamp
2	4-40 Standoff Terminals
4	4-40 X 3/8" Hex Posts
1	TO-3 Spacesaver Heat Sink HS102
20	4-40 X 3/8" Machine Screws
10	4-40 Hex Nuts
10	4-40 Lockwashers
2'	22 AWG Black Hookup Wire
3'	22 AWG Red Hookup Wire
1/2'	22 AWG Orange Hookup Wire
1 1/2'	22 AWG Yellow Hookup Wire
2'	22 AWG Green Hookup Wire
1'	22 AWG Blue Hookup Wire
1'	22 AWG Gray Hookup Wire
2'	22 AWG White Hookup Wire

DELAMTECT MODIFICATION

COMPONENT NUMBER OR QUANTITY	DESCRIPTION
P5	9 Pin Receptacle-Amp CPC w/pins
	Hardware:
4	4-40 X 3/8" Machine Screws
4	4-40 Lockwashers
4	4-40 Hex Nuts
2'	Single Conductor Shielded Wire
1'	22 AWG Orange Hookup Wire
1'	22 AWG White Hookup Wire
1'	22 AWG Gray Hookup Wire
1'	22 AWG Red Hookup Wire
1'	22 AWG Black Hookup Wire

DISTANCE SENSOR

COMPONENT NUMBER OR QUANTITY	DESCRIPTION
DS1	Distance Measuring Transducer Red Lion LSCD-IO-OF Wheel Sensor Red Lion LSA-HCO-OO Bracket
J6	6 Pin Plug-Amp CPC w/sockets

COMPUTER CABLE

COMPONENT NUMBER OR QUANTITY	DESCRIPTION
1	DB-25P D Connector-Solder Type
1	DB-25S D Connector-Solder Type
2	DB-25 D Connector Hoods
12'	22 AWG 7 Conductor Cable

SIGNAL & POWER CABLE

COMPONENT NUMBER OR QUANTITY	DESCRIPTION
2	6 Pin Plugs-Amp CPC w/sockets
2 1/2'	22 AWG 4 Conductor Cable
7 1/2'	Single Conductor Shielded Wire
2 1/2'	Spiral Cable Wrap

A.3 TECHTRAN RECORDER

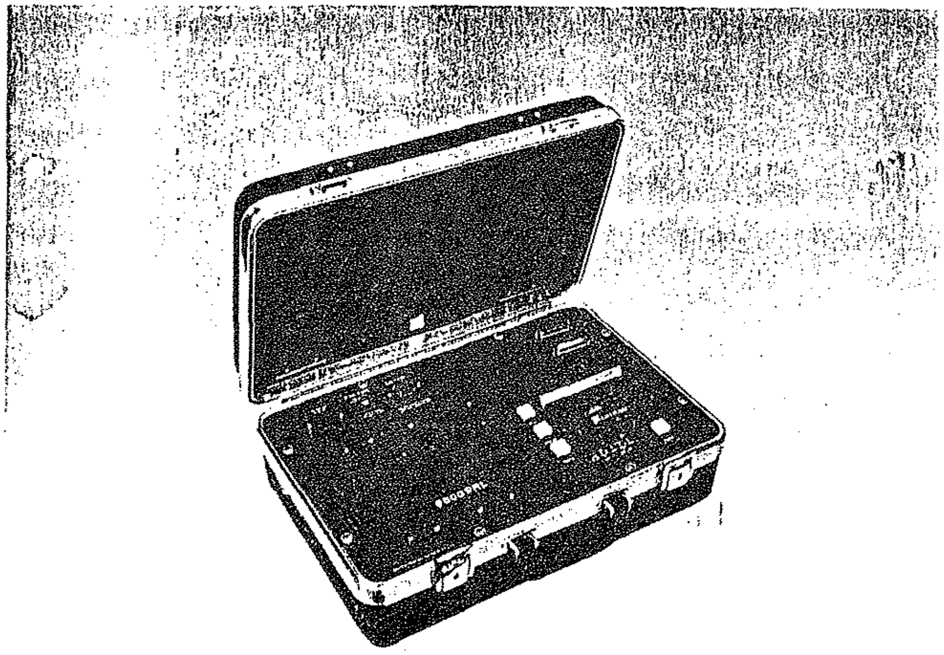
COMMAND REFERENCE

FUNCTION	BINARY SWITCH POSITION	MANUAL COMMAND	REMOTE COMMAND (Dip Switch 7 ON)
Start READ	any	READ Pushbutton	(CTRL Q)
Stop READ	any	READ Pushbutton	(CTRL S)
Start WRITE	BINARY	WRITE Pushbutton	N/A
	ONLINE/OFFLINE	WRITE Pushbutton	(CTRL R)
Stop WRITE	BINARY	WRITE Pushbutton	N/A
	ONLINE/OFFLINE	WRITE Pushbutton	(CTRL T)
END OF FILE Marker	BINARY	N/A	(CTRL S)
	ONLINE/OFFLINE	N/A	(CTRL S)
Start FILE SKIP	ONLINE/OFFLINE	N/A	(CTRL O)
Stop FILE SKIP	ONLINE/OFFLINE	READ Pushbutton	(CTRL S)
REWIND	BINARY	REWIND Pushbutton	N/A
	ONLINE/OFFLINE	REWIND Pushbutton	(CTRL Z)
CHARACTER DELETE	ONLINE/OFFLINE	N/A	(CTRL X)

NOTE: The LINE MODE switch affects the READ function as follows:

LINE MODE switch ON - READ line by line
LINE MODE switch OFF - READ file by file

9600PRL INSTALLATION AND OPERATING INSTRUCTIONS



TECHTRAN
INDUSTRIES, INC

200 Commerce Drive • Rochester, N.Y. 14623
Telephone (716) 334-9640 • TWX 510-253-3246

Contents

1 SYSTEM OVERVIEW 1

Introduction. 9600PRL Operation. Control Panel. Cassette Description. Cassette Insertion and Removal. Control Codes. Preventive Maintenance. Manual Organization.

2 INSTALLATION AND SPECIFICATIONS 5

Introduction. Cassette Specifications. Terminal Port Interface. Pin Assignments. Signal Characteristics. General Information. Environmental Specifications. ONLINE Data Flow. Dip Switch Settings. Baud Rate. Transmit/Disable (BS) Character. Full/Half Duplex. Enable/Disable READ Delay. Line Terminator. Remote READ Control. READ Delay Character. Odd/Even Parity. Enable/Disable Parity. Cable Connections.

3 OPERATING INSTRUCTIONS 15

Introduction. Write. Read. Rewind. Character Delete. File Skip.

4 OPTIONAL FEATURES 19

Ready/Busy Output. Start/Stop Input. Online Speed Control.

Illustrations

Figure 1: Local 9600PRL System 2
Figure 2: Remote 9600PRL System.. 2
Figure 3: Control Panel of the 9600PRL.. 3
Figure 4: ONLINE Data Flow.. 7
Figure 5: Dip Switch Panel 8
Figure 6: OFFLINE Mode, Full and Half Duplex 10
Figure 7: ONLINE Mode, Full and Half Duplex 11

1

System Overview

INTRODUCTION

This manual is designed for the user of the 9600PRL. All of the required operations associated with the unit are explained including:

- installation procedures
- cassette usage and care
- preventive maintenance
- operating instructions

This section should be thoroughly read and understood before proceeding to any of the above operations. It contains valuable information about using the 9600PRL.

9600PRL OPERATION

The 9600PRL is a high-speed portable data recorder/program loader designed for storing, transmitting and receiving ASCII-coded data. Among the standard operating features are:

- *WRITE* - recording data at selected speeds
- *READ* - displaying the contents of a cassette
- *FILE SKIP* - providing rapid forward tape advance
- *CHARACTER DELETE* - eliminating undesired characters during data recording

The 9600PRL has the versatility for numerous applications including:

- loading programs into PBX systems and process control devices
- loading diagnostic routines into programmable systems and devices
- digital recording for datalogging systems
- remote testing and troubleshooting

Figures 1 and 2 illustrate typical system configurations of the 9600PRL.

CONTROL PANEL

The control panel of the 9600PRL contains several switches and pushbuttons used to operate the unit (refer to Figure 3). The READ, WRITE, and REWIND pushbuttons are used for manual control of the unit; remote control (if activated) is implemented by control codes received at the MODEM/CPU or TERMINAL ports.

During installation, the dip switches (located beneath the lift-off cover) are set to meet desired interface requirements. AC power requirements can also be selected from the control panel.

The Line Mode and Binary switches provide special operating features. Each operation provided in this manual specifies the correct switch settings and explains various options available.

In addition, two ports are supplied for equipment interface. The MODEM/CPU port can be attached to a modem, acoustic coupler, CPU or intelligent device. The TERMINAL port can be attached to a hardcopy terminal, printer, CRT or electronic device.

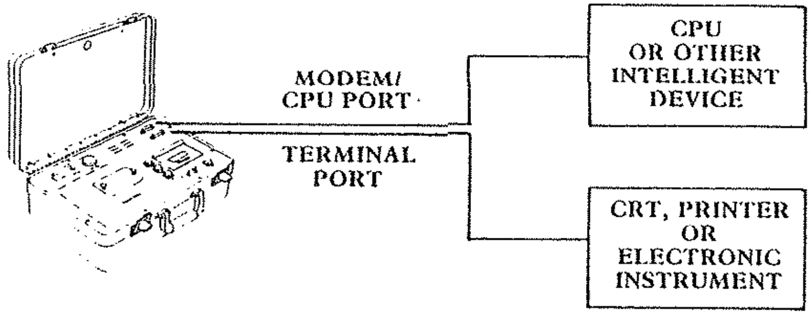


Figure 1: Local 9600PRL System

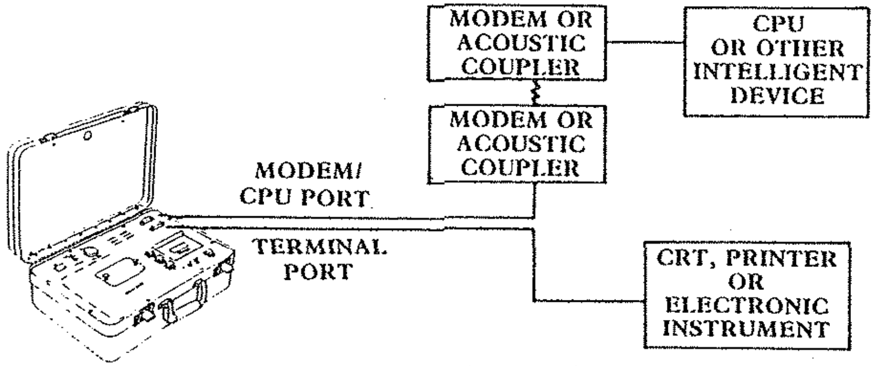


Figure 2: Remote 9600PRL System

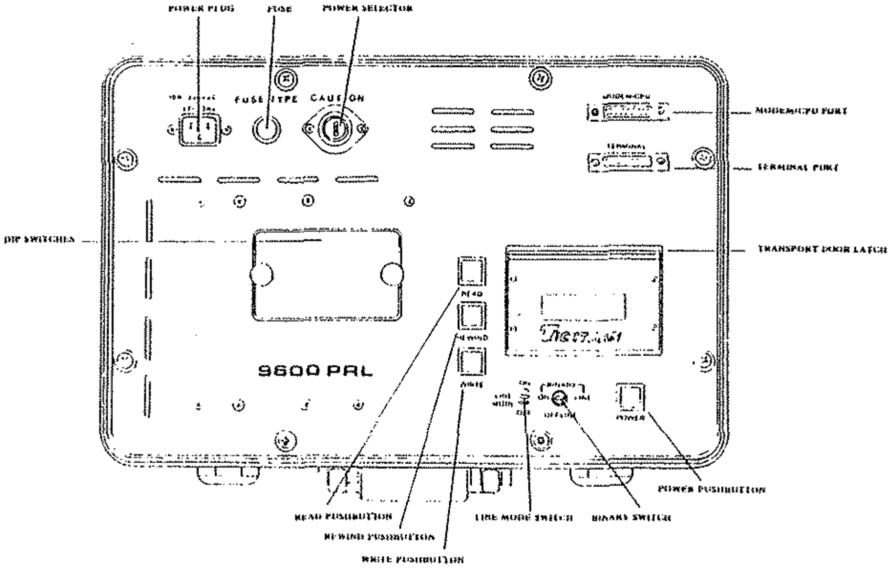


Figure 3: Control Panel of the 9600PRL

CASSETTE DESCRIPTION

Cassettes are a magnetic medium used to store recorded data. *Techtran cassettes (P/N 4300001) or an approved equal should be used with the 9600PRL;* otherwise, damage may occur and the machine warranty may be voided (refer to Section 2).

Data is only recorded on one side of the cassette. Cassettes recorded on the 9600PRL can only be interchanged with cassettes from another 9600PRL or from a Techtran Series 800 Datacassette with the 9600 baud option. Do not expose cassettes to strong magnetic fields or temperatures in excess of 104°F (40°C).

CASSETTE INSERTION AND REMOVAL

1. Turn the unit on by pressing the POWER pushbutton (the pushbutton will illuminate). Lift the transport door latch, allowing the door to swing open.
2. Fully insert the cassette into the tape guides with the label facing forward and the large tape spool on the left. Close the transport door.

To remove a cassette, make sure that no functions are in progress. Lift the transport door latch and remove the cassette.

CONTROL CODES

Several 9600PRL operations are facilitated by using control codes. All control codes in this manual are identified as follows:

(CTRL X)

CTRL represents the CONTROL key on the terminal and X represents a designated character. To use a control code, press the CTRL key and simultaneously press the designated character. All control codes are listed on the back cover.

PREVENTIVE MAINTENANCE

The READ/WRITE tape head of the recorder must be cleaned *daily* using a soft cotton swab dampened in isopropyl alcohol. Neglecting this care may result in abnormal tape wear, transport damage or operational errors.

MANUAL ORGANIZATION

This manual contains three additional sections. Section 2 explains how to install the 9600PRL and provides unit specifications. Section 3 provides operating instructions for the 9600PRL, and Section 4 lists available options. Proceed to Section 2 for installation procedures.

2

Installation and Specifications

INTRODUCTION

The following section contains information about installing the 9600PRL. Once the unit is unpacked, check that the following standard equipment was included:

- 9600PRL
- one cassette
- male to female EIA RS-232C cable
- power cord
- *9600PRL Installation and Operating Instructions*

Installation consists of setting the dip switches, connecting cables and powering on. Before beginning installation, verify that the peripheral equipment in use meets the specifications listed subsequently.

CASSETTE SPECIFICATIONS

- Philips-type
- 300 feet (length)
- 1600 bits per inch (density)

Only Techtran cassettes (P/N 4300001) or an approved equal should be used with 9600PRL. Use of other cassette types may cause equipment damage and could void the machine warranty.

TERMINAL PORT INTERFACE

A device connected to the TERMINAL port must have the following characteristics:

- Full or Half Duplex, asynchronous
- 8-level USASCII coded
- EIA RS-232C/CCITT V.24

If using a Model 33* or similar terminal having a current interface, the terminal must be prepared to operate in the Full Duplex 20 milliampere Neutral mode.

*Model 33 is a trademark of Teletype Corporation.

PIN ASSIGNMENTS

TERMINAL Port		MODEM/CPU Port	
Pin	**Function**	**Pin**	**Function**
1 AA	Protective Ground	1 AA	Protective Ground
2 BA	Transmitted Data (Out)	2 BA	Transmitted Data (In)
3 BB	Received Data (In)	3 BB	Received Data (Out)
4 CA	Request to Send	5 CB	Clear to Send
5 CB	Clear to Send	6 CC	Data Set Ready
7 AB	Signal Ground	7 AB	Signal Ground
8 CF	Data Carrier Detect	8 CF	Data Carrier Detect
16	Ready/Busy Output ①	16	Ready/Busy Output ①
20 CD	Data Terminal Ready	20 CD	Data Terminal Ready
25	Start/Stop Input ②	25	Start/Stop Input ②

TERMINAL Port – 20mA Current Loop

Pin	Function
2	Transmitted Data (In)
3	Received Data (Out)
10	Receive Common
13	Transmit Common

① Optional (Ready = ∅V, Busy = + 5V)

② Optional (Start = ∅V, Stop = + 5V)

NOTES

- 9600PRL supplies +5V on pins 5, 6 and 8 of the TERMINAL port and pins 4 and 20 of the MODEM/CPU port.
- 9600PRL requires +3V to +25V on pin 4 or 20 of the TERMINAL port to enable READ Delay.
- 9600PRL requires +3V to +25V on pin 5 of the MODEM/CPU port to enable interface.
- MODEM/CPU port is a DB-25P connector; TERMINAL port is a DB-25S connector.
- MODEM/CPU interface is only active when the attached device sends a signal to pin 5 (CB, Clear to Send) and the Binary switch is ONLINE or BINARY. Future reference to this condition: pin 5 at the MODEM/CPU port *enabled.*

GENERAL INFORMATION

- Power: Selected from the control panel (100, 115, 200, 220, 230, and 240VAC, 47-63Hz, 12W)
- Baud Rates: 110, 300, 1200, or 9600
- Characters per cassette: 220,000 (nominal)
- Recording Format: Techtran NRZ Dual-Track

SIGNAL CHARACTERISTICS

- EIA RS-232C/CCITT V.24
- Receive - Mark: -3 to -25 volts
  Space: +3 to +25 volts
- Transmit - Mark: -8 volts with 3K load
  Space: +8 volts with 3K load
- Maximum short circuit current: 500 mA
- Terminating Impedance: 3K to 7K

ENVIRONMENTAL SPECIFICATIONS

- Operating Temperature: 50-104°F (10-40°C)
- Operating Humidity: 20-90% relative humidity, non-condensing

ONLINE DATA FLOW

The 9600PRL is ONLINE when the Binary switch is in BINARY or ONLINE and pin 5 at the MODEM/CPU port is enabled. Figure 4 demonstrates how data travels in the ONLINE mode.

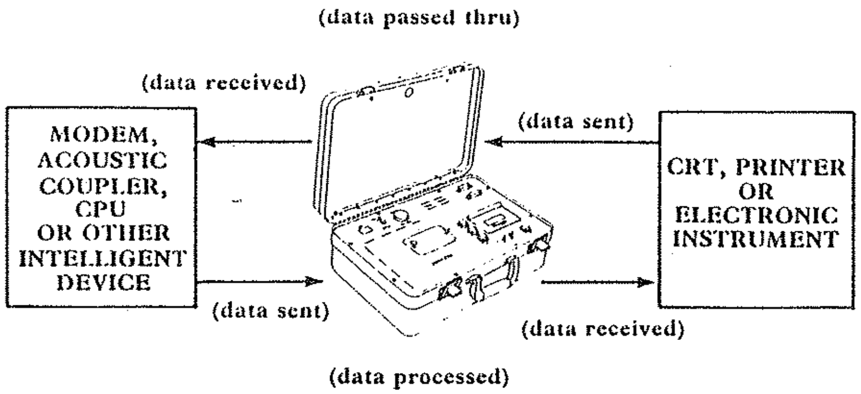

(data passed thru)
(data received)
(data sent)
MODEM, ACOUSTIC COUPLER, CPU OR OTHER INTELLIGENT DEVICE
CRT, PRINTER OR ELECTRONIC INSTRUMENT
(data sent)
(data received)
(data processed)

Figure 4: ONLINE Data Flow

DIP SWITCH SETTINGS

The dip switches are housed beneath a lift-off cover as indicated in Figure 3. Figure 5 illustrates the dip switch panel; follow the instructions provided to set these switches. Use a pencil tip to manipulate the switches.

NOTE: Each switch sets the interface requirements for *all* devices attached to the unit. Data to be recorded must come to the MODEM/CPU port under the following two conditions:

- Pin 5 at the MODEM/CPU port is enabled
- Binary switch set to ONLINE or BINARY

Input must come from the TERMINAL port if pin 5 at the MODEM/CPU port is *disabled* or the Binary switch is OFFLINE.

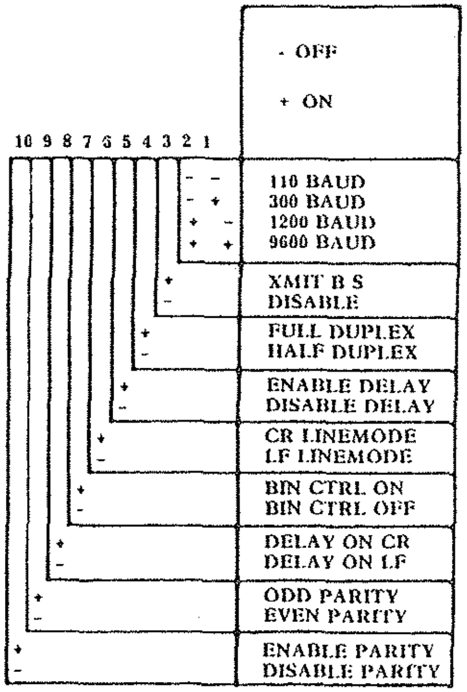


Figure 5: Dip Switch Panel

Baud Rate

Determine the appropriate baud (transmission) rate of the attached devices and set switches 1 and 2 accordingly. Refer to the upper right of the dip switch panel to determine the correct switch positions (ON or OFF). Characters with eleven bits (two stop bits) will be transmitted at the 110 speed; ten bit characters (one stop bit) will be transmitted at all other speeds. If using the ONLINE SPEED CONTROL option, refer to Section 4 before setting these switches.

Transmit/Disable (BS) Character

Switch 3 affects the CHARACTER DELETE function (refer to Section 3). When removing unwanted characters from data during this function, you may set this switch to cause the following results at the TERMINAL and MODEM/CPU ports:

- If the switch is set to ON(+), then once a character is removed from data, a (BS) code is sent and the cursor of the attached terminal device will move back to the position of the removed character. The next character received will be placed in this position.

- If the switch is set to OFF(-), then once a character is removed from data, no (BS) code is sent and the cursor prints the next character received in the following position.

Full/Half Duplex

The effect of switch 4 varies according to the position of the Binary switch on the control panel (refer to Figures 6 and 7).

- When the Binary switch is OFFLINE or pin 5 at the MODEM/CPU port is *disabled* with the Binary switch ONLINE, data received at the TERMINAL port is affected as follows:

  Full Duplex: All data received is echoed back to the source. In other words, if the device sends a character, it will be processed by the 9600PRL and also sent back to the originating device as verification.

  Half Duplex: No data is echoed back to the source.

When the Binary switch is ONLINE and pin 5 at the MODEM/CPU port is *enabled,* no data is echoed back from the 9600PRL. Any full duplex device connected to the TERMINAL port is expected to receive data as a result of an echoback from a device attached to the MODEM/CPU port.

- When the Binary switch is in ONLINE or BINARY and pin 5 at the MODEM/CPU port is *enabled,* the READ function is affected as follows (refer to Section 3):

  Full Duplex: Data is sent to the MODEM/CPU port only. Any device connected to the TERMINAL port is expected to receive data as a result of an echoback received at the MODEM/CPU port from the attached device.

  Half Duplex: Data is sent to both ports; therefore, a device connected to the TERMINAL port will receive the same data as a device connected to the MODEM/CPU port.

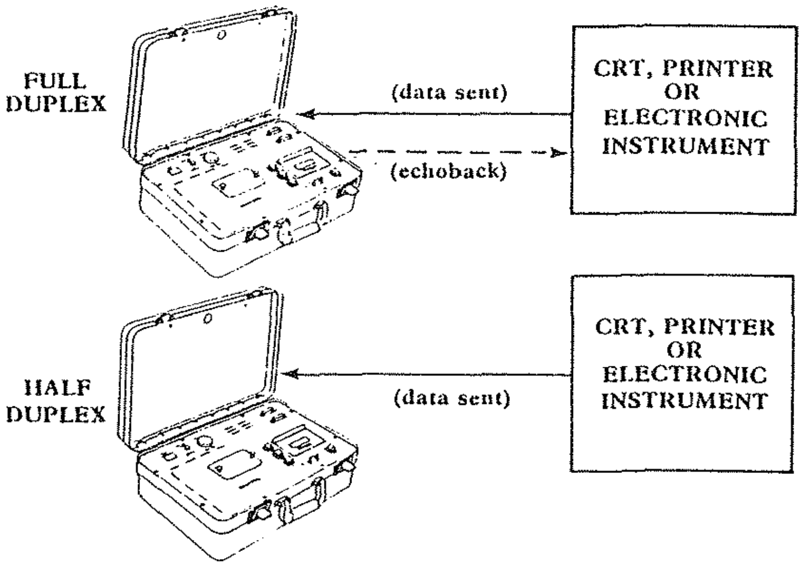

FULL DUPLEX
(data sent)
(echoback)
CRT, PRINTER OR ELECTRONIC INSTRUMENT
HALF DUPLEX
(data sent)
CRT, PRINTER OR ELECTRONIC INSTRUMENT

Figure 6: OFFLINE Mode, Full and Half Duplex

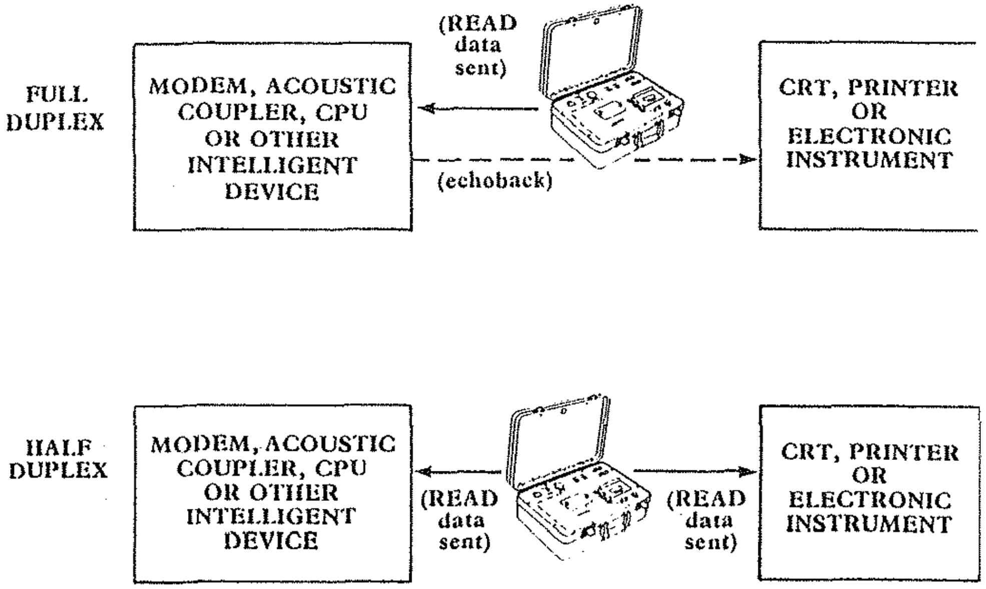

FULL DUPLEX
MODEM, ACOUSTIC COUPLER, CPU OR OTHER INTELLIGENT DEVICE
(READ data sent)
(echoback)
CRT, PRINTER OR ELECTRONIC INSTRUMENT
HALF DUPLEX
MODEM, ACOUSTIC COUPLER, CPU OR OTHER INTELLIGENT DEVICE
(READ data sent)
(READ data sent)
CRT, PRINTER OR ELECTRONIC INSTRUMENT

Figure 7: ONLINE Mode, Full and Half Duplex

Enable/Disable READ Delay

Switch 5 determines if an automatic 300 millisecond delay will occur during the READ function after each line of data sent from the tape. Set the switch to the ON(+) position to enable the READ delay. The READ delay is functional only when an active device is attached to the TERMINAL port and +3V to +25V is present at pin 4 or 20 of the TERMINAL port, but it will affect the READ function at both ports. If the switch is set to the OFF(-) position, no READ delay will occur.

Line Terminator

When the Line Mode switch on the control panel is ON, either a Carriage Return (CR) or Line Feed (LF) character (as set by this switch) is recognized as the line terminator. Determine which character ends lines of data for your application and set switch 6 accordingly.

Remote READ Control

Switch 7 determines if remote control with the READ function is permitted when the Binary switch is in BINARY. Using remote control, the READ function can be controlled by sending control codes Q and S to the MODEM/CPU and TERMINAL ports. If remote control is desired, set this switch to the ON(+) position. If remote control is not required, set this switch to the OFF(-) position and only manual control panel commands will be recognized (READ, WRITE and REWIND pushbuttons).

READ Delay Character

Set switch 8 only if dip switch 5 is set to the ON(+) position. The READ delay selected by dip switch 5 (Enable/Disable READ Delay) is activated by either a (CR) or (LF) character (as set by this dip switch). Specify the character to cause the delay by setting this switch.

Odd/Even Parity

Set switch 9 only if odd or even parity is required for the attached devices. This switch identifies the odd or even parity requirements of the attached devices. Specify if the devices in use have odd or even parity by setting this switch accordingly.

Enable/Disable Parity

Switch 10 further specifies the parity requirements of the attached devices. If this switch is set to the ON(+) position, odd or even parity will be sent as specified by dip switch 9. If this switch is set to the OFF(-) position, data will be recorded as it was received (8-bit bytes) and sent as it was recorded.

CABLE CONNECTIONS

The 9600PRL can be connected to peripheral devices by either direct or remote means as follows:

1. Connect the male to female cable supplied from the MODEM/CPU port on the unit to one of the following devices:

   - CPU or Intelligent Device (direct connection)
   - Modem or Acoustic Coupler (remote connection)

   Refer to Figures 1 and 2. *Pin 5 must be enabled* by the attached device. Local cable distances should not exceed 50 feet, according to EIA RS-232C specification.

2. Connect a male to male EIA RS-232C cable (not supplied) from the TERMINAL port on the unit to the terminal, printer, or electronic device in use. Be sure to activate the Remote/Online mode on a terminal or printer.

3. Attach the power cord supplied from the power plug on the unit to a local AC power source. Select the appropriate local AC power requirements on the control panel *before* turning the unit on. Press the POWER pushbutton on the control panel to turn the unit on (pushbutton will illuminate). Turn on peripheral devices.

3

Operating Instructions

INTRODUCTION

The 9600PRL can be operated manually from the control panel using the READ, WRITE, and REWIND pushbuttons. If remote READ control is selected (dip switch 7 ON), the TERMINAL and MODEM/CPU ports will recognize control codes Q and S when the Binary switch is in BINARY.

Before beginning the READ or WRITE operations, refer to the guidelines listed below.

IF SENDING DATA TO THE TERMINAL PORT, ***ONE*** OF THE FOLLOWING CONDITIONS MUST BE MET (OFFLINE MODE):

1. Binary switch set to OFFLINE (pin 5 at the MODEM/CPU port enabled or disabled)
2. Pin 5 at the MODEM/CPU port disabled (Binary switch set to ONLINE or OFFLINE)
3. No device connected to the MODEM/CPU port

IF SENDING DATA TO THE MODEM/CPU PORT, ***BOTH*** OF THE FOLLOWING CONDITIONS MUST BE MET (ONLINE MODE):

1. Binary switch set to ONLINE or BINARY
2. Pin 5 at the MODEM/CPU port enabled

Use the back cover for future command reference.

WRITE

The WRITE function permits data recording onto a cassette. The 9600PRL records data in 8-bit bytes and treats control codes as it does any other character when the Binary switch is in BINARY (code transparent). Once all of the information has been entered, it must be stored in a manner that is easy to retrieve later. All recorded information is stored in *files* that you terminate when data recording is ended.

Before beginning to WRITE:

- adhere to the guidelines listed previously.
- be sure that the cassette in use is not write-protected. The plastic tab on the top left of the cassette must be in place.

To record data:

1. Insert the cassette and close the transport door.

2. Enter (CTRL R) or press the WRITE pushbutton. The WRITE pushbutton will illuminate. Note: (CTRL R) cannot be used if the Binary switch is in BINARY.

3. Enter the information to be recorded.

4. Enter (CTRL S) to identify the end of a file.

To terminate the WRITE function, enter (CTRL T) or press the WRITE pushbutton. The WRITE function *must* be terminated to insure that all information is recorded on the cassette.

READ

The READ function permits viewing of the cassette contents. There are three ways to READ cassettes (notice the required switch settings for each method):

- READ cassette information line by line (Line Mode switch set to ON, Binary switch set to ONLINE or OFFLINE)

- READ cassette information file by file (Line Mode switch set to off, Binary switch set to ONLINE or OFFLINE

- READ entire cassette without stops (Binary switch set to BINARY)

Before you begin to READ, adhere to the guidelines listed previously.

To READ a cassette:

1. Insert the cassette and close the transport door.

2. Enter (CTRL Q) or press the READ pushbutton. The READ pushbutton will illuminate. The information will be displayed as previously indicated by the switch settings.

To end the READ function, enter (CTRL S) or press the READ pushbutton. The READ function will automatically terminate if:

- the end of all data is reached
- a blank cassette is inserted
- the end of the cassette tape is reached
- the unit encounters a file terminator, (CTRL S), recorded on the cassette (unless the Binary switch is in BINARY)
- the unit encounters a line terminator, (CR) or (LF), recorded on the cassette (if the Line Mode switch is ON)

REWIND

You can rewind the cassette tape provided no other function is in progress. The tape rewinds completely and cannot be stopped once begun. The transport door should never be opened during REWIND.

To REWIND the tape, enter (CTRL Z) or press the REWIND pushbutton. The REWIND pushbutton will illuminate and the tape will fully rewind. Always rewind the tape before removal. If the tape does not respond to this command, follow these steps:

1. Enter (CTRL Q) or press the READ pushbutton. Allow the tape to advance for a few seconds.

2. Enter (CTRL S) or press the READ pushbutton.

3. Enter (CTRL Z) or press the REWIND pushbutton.

CHARACTER DELETE

This function removes unwanted characters during data recording. When the unit receives the CHARACTER DELETE command (CTRL X), the last character sent to the unit is erased. The (CTRL X) command may be used consecutively to remove up to the last 64 characters. This function is inoperative when the Binary switch is in BINARY.

FILE SKIP

The FILE SKIP function permits rapid forward tape advance on the cassette without data display. Cassette data is advanced file by file. This function is inoperative when the Binary switch is in BINARY. Follow these steps to implement the FILE SKIP:

1. Insert the cassette and close the transport door.

2. Enter (CTRL O). The READ pushbutton will illuminate and the tape will advance to the next stop. Continue the FILE SKIP as required.

The FILE SKIP function does not need to be terminated once a stop point is reached if you wish to begin another operation. This function automatically stops when:

- the end of the cassette tape is reached
- a blank cassette is inserted
- the unit encounters a file terminator, (CTRL S), recorded on the cassette

To stop a FILE SKIP in progress, enter (CTRL S) or press the READ pushbutton.

4

Optional Features

READY/BUSY OUTPUT

This option indicates when the cassette tape is in motion by registering the following voltages on pin 16 of either port:

- Tape Stopped: ϕV
- Tape in Motion: +5V +/- 10% (maximum 10mA)

START/STOP INPUT

This option can be used once the READ function has been initiated. When READ is in progress, the following voltages sent to pin 25 of the active port will interrupt and resume the READ function:

- Start Read: ϕV
- Stop Read: +5V +/- 10% (minimum 20mA)

This option cannot be used to begin or end the READ function.

ONLINE SPEED CONTROL

This option automatically selects a preset alternate baud rate for both ports when the Binary switch is ONLINE and pin 5 at the MODEM/CPU port is *enabled*. This eliminates the need for altering dip switch settings 1 and 2 if the baud rate differs during ONLINE and OFFLINE functions.

When using this option, set dip switches 1 and 2 to reflect the baud rate of the TERMINAL port (OFFLINE operations); the baud rate of the MODEM/CPU port will be factory set and is selected automatically with this option whenever ONLINE operations are activated.

LIMITED WARRANTY

Techtran Industries, Inc. warrants that this product is free from defects in material and workmanship for a period of ninety (90) days from the date of shipment. Techtran's obligation under this limited warranty shall be to replace or repair, at its option, at its designated site, any part or parts thereof (except expendable parts thereof, defective cassettes and damage caused in shipment) that within the warranty period are returned to Techtran in the original shipping package under a Return Authorization (RA) number issued by Techtran and that are found by Techtran to be defective in proper usage. Techtran reserves the right to refuse to accept delivery of any shipment containing any shipping cartons which do not have the RA number displayed on the outside. Buyer shall ship freight prepaid to Techtran's designated site. If Techtran determines that the product is not defective within the terms of this limited warranty, Buyer shall pay Techtran the cost of repairs at the then prevailing Techtran repair rates.

> The foregoing limited warranty is in lieu of all warranties, either expressed or implied, including without limitation any implied warranty of merchantability or fitness for a particular purpose, and of any other obligation on the part of Techtran. Techtran shall not be liable for any injury, loss or damage, direct or consequential to persons or property caused either directly or indirectly by the use or inability to use the equipment and/or this or related documents. Such limitations in liability shall remain in full force and effect even when Techtran may have been advised of the possibility of such injuries, losses or damages.

Any provision herein to the contrary notwithstanding, in no event shall Techtran be liable for indirect, incidental or consequential damages and in no event shall the liability to Techtran arising in connection with any product (whether such liability arises from a claim based on contract, warranty, tort or otherwise) exceed the actual amount paid to Techtran for the defective product.

Before purchasing or using the product, Buyer shall determine the suitability of Techtran products and related documents for his or her intended use and assumes all risk and liability whatsoever in connection therewith.

A.4 MONITOR PROGRAM

```
2500 A.D. 6502 CROSS ASSEMBLER   -   VERSION 3.00e
--------------------------------------------------

          INPUT  FILENAME :  DOTREH3.ASM
          OUTPUT FILENAME :  DOTREH3.OBJ
```

```
                ;
                ;  DOTREH3.ASM        6-11-85
                ;
                ;       DOT DATA COLLECTION PROGRAM
                ;
                ;
                ;
                        .PW     80      ;PAGE WIDTH 80 COLS
                ;
                ;
                ;***********************************************
                 ************
                ;
                ;               PAGE ZERO VARIABLES
                ;
                ;***********************************************
                 ************
                ;
00 00           RAM     EQU     $0000   ;RAM AREA
01 00           PLSCNT  EQU     $01     ;PULSE COUNT STORAGE
02 00           DATCNT  EQU     $02     ;DATA BYTE COUNT STORAG
                                        E
03 00           AD1     EQU     $03     ;AD1 DATA STORAGE
04 00           AD2     EQU     $04     ;AD2 DATA STORAGE
05 00           OUT1H   EQU     $05     ;DATA1  UPPER BYTE
06 00           OUT1L   EQU     $06     ;DATA1 LOWER BYTE
07 00           OUT2H   EQU     $07     ;DATA2 UPPER BYTE
08 00           OUT2L   EQU     $08     ;DATA2 LOWER BYTE
09 00           PDATA   EQU     $09     ;PRINT DATA FLAG-80H=DA
                                        TA1,40H=OUT2
0A 00           COUNT1  EQU     $0A     ;DELAY COUNTER1
0B 00           COUNT2  EQU     $0B     ;DELAY COUNTER2
0C 00           PASS    EQU     $0C     ;PASS NUMBER
0D 00           SWFLAG  EQU     $0D     ;MARKER SWITCH FLAG
0E 00           SWCNT   EQU     $0E     ;MARKER SW COUNT
0F 00           CHAN1   EQU     $0F     ;CHANNEL1 DATA
10 00           CHAN2   EQU     $10     ;CHANNEL2 DATA
                ;
                ;***********************************************
                 ************
                ;
                ;               CONSTANTS
                ;
                ;***********************************************
                 ************
                ;
                ;
11 00           RDFILE  EQU     $11     ;CNTL-Q, READ TAPE FILE
```

```
          12 00               OPFILE  EQU     $12     ;CNTL-R, OPEN TAPE FILE
          14 00               CLFILE  EQU     $14     ;CNTL-T, CLOSE TAPE FIL
                                                      E
          1A 00               REWND   EQU     $1A     ;CNTL-Z, REWIND TAPE FI
                                                      LE
          13 00               ENFILE  EQU     $13     ;CNTL-S, ENDREAD TAPE F
                                                      ILE
          07 00               ACK_RW  EQU     $07     ;CNTL-G, REWIND ACK
          06 00               ACK_WR  EQU     $06     ;CNTL-F, WRITE ACK
                              ;
                              ;
                              *****************************************
                               ************
                              ;
                              ;               PERIPHERAL CHIPS
                              ;
                              ;****************************************
                               ************
                              ;
          00 20               ADDRA   EQU     $2000   ;A TO D 6821-DIR REG A
          00 20               ADPRA   EQU     $2000   ;PER REG A
          01 20               ADCRA   EQU     $2001   ;CNT REG A
          02 20               ADDRB   EQU     $2002   ;DIR REG B
          02 20               ADPRB   EQU     $2002   ;PER REG B
          03 20               ADCRB   EQU     $2003   ;CNT REG B
                              ;
          00 40               GPDRA   EQU     $4000   ;GEN P 6821-DIR REG A
          00 40               GPPRA   EQU     $4000   ;PER REG A
          01 40               GPCRA   EQU     $4001   ;CNT REG A
          02 40               GPDRB   EQU     $4002   ;DIR REG B
          02 40               GPPRB   EQU     $4002   ;PER REG B
          03 40               GPCRB   EQU     $4003   ;CNT REG B
                              ;
          00 60               ACIADAT EQU     $6000   ;SERIAL PORT-DATA REG
          01 60               ACIASTA EQU     $6001   ;STATUS REG
          02 60               ACIACMD         EQU     $6002   ;CMD REG
          03 60               ACIACTL         EQU     $6003   ;CTL REG
          6B 00               ICMD            EQU     $6B     ;DTR=L,IRQ=H,RT
                                                              S=L,NO ECHO,EVE
                                                              N PAR
          3E 00               ICTL            EQU     $3E     ;9600 BAUD,7 BI
                                                              T,1 STOP
                              ;
                              ;
F800                                  .ORG    F800H   ;BEGIN PROGRAM ASSEMBLY
                              ;
                              ;
                              ;****************************************
                               ************
                              ;
                              ;               INITIALIZATION
                              ;
                              ;****************************************
                               ************
                              ;
F800                          INITCPU
F800      A2 FF                       LDX     #$FF
```

```
F802   9A                   TXS             ;TOP OF STACK
F803   78                   SEI             ;DISABLE CPU INTERRUPTS
F804   D8                   CLD             ;BINARY MODE
                  ;
F805              INITSER
F805   A9 6B                LDA   #ICMD     ;INIT COM REG
F807   8D 02 60             STA   ACIACMD
F80A   A9 3E                LDA   #ICTL     ;INIT CTL REG
F80C   8D 03 60             STA   ACIACTL
F80F              INITAD
F80F   A9 04                LDA   #$04      ;SET TO PER REG
F811   8D 03 20             STA   ADCRB
F814   A9 0F                LDA   #$0F      ;SET A/D CNTL LINES HIG
                                            H
F816   8D 02 20             STA   ADPRB
F819   A9 00                LDA   #$00      ;SET TO DIR REGS
F81B   8D 01 20             STA   ADCRA
F81E   8D 03 20             STA   ADCRB
F821   8D 00 20             STA   ADDRA     ;SET PA0-PA7 INPUTS
F824   A9 0F                LDA   #$0F      ;PB0-PB3 OUTPUTS,PB4-PB
                                            7 INPUTS
F826   8D 02 20             STA   ADDRB
F829   A9 04                LDA   #$04      ;SET TO PER REG
F82B   8D 01 20             STA   ADCRA
F82E   8D 03 20             STA   ADCRB
F831   A9 04                LDA   #$04      ;CS1,CS2,WR=L
F833   8D 02 20             STA   ADPRB
F836   A9 0F                LDA   #$0F      ;SET AD LINES HIGH
F838   8D 02 20             STA   ADPRB
F83B   A9 08                LDA   #$08      ;CS1,CS2,RD=L
F83D   8D 02 20             STA   ADPRB
F840   A9 0F                LDA   #$0F      ;SET AD LINES HIGH
F842   8D 02 20             STA   ADPRB
F845              INITGP
F845   A9 00                LDA   #$00      ;SET TO DIR REG
F847   8D 01 40             STA   GPCRA
F84A   8D 03 40             STA   GPCRB
F84D   8D 00 40             STA   GPDRA     ;PA0-PA7 INPUTS
F850   A9 01                LDA   #$01      ;PB0 OUT, PB1-7 IN
F852   8D 02 40             STA   GPDRB
F855   A9 0D                LDA   #$0D
F857   8D 01 40             STA   GPCRA     ;CA1,2 NEG TRANS, PER R
                                            EG
F85A   8D 03 40             STA   GPCRB     ;CB1,2 NEG TRANS, PER R
                                            EG
F85D              INITREAD
F85D   A0 04                LDY   #04       ;READ 4 TIMES
F85F   A9 11      TAPERD    LDA   #RDFILE   ;CNTL-Q  START READ
F861   20 C7 F9             JSR   PRINT
F864   A2 05                LDX   #05       ;ONE SEC LOOP INIT
F866   20 B6 F9   ONESEC    JSR   DELAY     ;200 MS DELAY
F869   CA                   DEX
F86A   D0 FA                BNE   ONESEC
F86C   A9 13                LDA   #ENFILE   ;CNTL-S  STOP READ
F86E   20 C7 F9             JSR   PRINT
F871   88                   DEY
F872   D0 EB                BNE   TAPERD    ;NEXT READ
```

```
F874                INITRWD
F874  A9 1A                  LDA   #REWND   ; REWIND TAPE
F876  20 C7 F9               JSR   PRINT
F879                INITLP1
F879  20 33 FA               JSR   READTP   ; GET CHAR
F87C  C9 FF                  CMP   #$0FF
F87E  F0 F4                  BEQ   INITRWD
F880  C9 07                  CMP   #ACK_RW  ; ACK REWIND CNTL-G
F882  D0 F5                  BNE   INITLP1  ;TRY NEXT CHAR IN BUFFE
                                            R
F884                INITPASS
F884  A9 41                  LDA   #$41     ;SET FIRST PASS TO "A"
F886  85 0C                  STA   PASS
F888                INITVAR
F888  A9 64                  LDA   #100     ;SWITCH COUNT
F88A  85 0E                  STA   SWCNT
F88C  A9 00                  LDA   #00      ;SWITCH FLAG
F88E  85 0D                  STA   SWFLAG
                    ;
                    ;
                    ;**********************************************
                     ************
                    ;
                    ;                MAIN PROGRAM
                    ;
                    ;**********************************************
                     ************
                    ;
                    ;
                    START                           ;WAIT FOR START BUTTON
F890  AD 02 40               LDA   GPPRB    ;READ SWITCH PORT
F893  29 02                  AND   #$02     ;MASK FOR SWITCH
F895  D0 F9                  BNE   START    ;SWITCH OPEN
F897  20 94 F9               JSR   DEBOUN   ;DEBOUNCE SWITCH
F89A  A5 0D                  LDA   SWFLAG
F89C  F0 F2                  BEQ   START    ;FALSE SWITCH CLOSE
                    FILEO                           ;OPEN TAPE FILE
F89E  A9 12                  LDA   #OPFILE  ;CNTL-R  START WRITE
F8A0  20 C7 F9               JSR   PRINT
F8A3                FILLP1
F8A3  20 33 FA               JSR   READTP   ; GET ACK
F8A6  C9 FF                  CMP   #$0FF
F8A8  F0 F4                  BEQ   FILEO
F8AA  C9 06                  CMP   #ACK_WR  ;ACK WRITE CNTL-F
F8AC  D0 F5                  BNE   FILLP1   ;TRY NEXT CHAR IN BUFFE
                                            R
F8AE  A9 01                  LDA   #01      ;TURN ON COLLECT LIGHT
F8B0  8D 02 40               STA   GPPRB
F8B3  A5 0C                  LDA   PASS     ;SEND PASS NUMBER TO TA
                                            PE
F8B5  20 C7 F9               JSR   PRINT
                    CLEAR                           ;CLEAR VARIABLES
F8B8  A9 00                  LDA   #$00
F8BA  85 03                  STA   AD1
F8BC  85 04                  STA   AD2
F8BE  85 09                  STA   PDATA
F8C0  85 0F                  STA   CHAN1
```

```
F8C2    85 10                         STA     CHAN2
F8C4    AD 00 40                      LDA     GPPRA     ;RESET CA1/CA2
                            LOOP                        ;CLEAR 4 SAMPLE SUM
F8C7    A9 00                         LDA     #$00
F8C9    85 05                         STA     OUT1H
F8CB    85 06                         STA     OUT1L
F8CD    85 07                         STA     OUT2H
F8CF    85 08                         STA     OUT2L
F8D1                        RESTART
F8D1    A9 06                         LDA     #$06      ;INIT PULSE COUNTER
F8D3    85 01                         STA     PLSCNT
F8D5    A9 05                         LDA     #$05      ;INIT DATA COUNTER
F8D7    85 02                         STA     DATCNT
                            ;
                            PCNTR                       ;PULSE COUNTING ROUTINE
F8D9    AD 02 40                      LDA     GPPRB     ;MARKER SWITCH PORT
F8DC    29 02                         AND     #$02      ;SWITCH MASK
F8DE    D0 04                         BNE     PULSE     ;SWITCH STILL OPEN
F8E0    C6 0E                         DEC     SWCNT     ;SW CLOSURE COUNT
F8E2    F0 30                         BEQ     CLOSE     ;END OF PASS
F8E4    2C 01 40            PULSE     BIT     GPCRA     ;DISTANCE PULSE?
F8E7    50 F0                         BVC     PCNTR     ;NO PULSE RECEIVED
F8E9    AD 00 40                      LDA     GPPRA     ;RESET CA1/CA2
F8EC    C6 01                         DEC     PLSCNT
F8EE    D0 11                         BNE     PCHAR     ;NOT YET
F8F0    A9 06                         LDA     #$06      ;RESET PULSE COUNTER
F8F2    85 01                         STA     PLSCNT
F8F4    20 D3 F9                      JSR     READAD    ;GET DATA
F8F7    C6 02                         DEC     DATCNT
F8F9    F0 0C                         BEQ     LASTDAT   ;LAST DATA BYTE
F8FB    20 32 F9                      JSR     TOTAL1    ;ADD DATA
F8FE    4C D9 F8                      JMP     PCNTR     ;GET MORE DATA
                            PCHAR                       ;PRINT DATA TO TAPE
F901    20 6A F9                      JSR     CHAR      ;CHAR SEND ROUTINE
F904    4C D9 F8                      JMP     PCNTR     ;GET MORE DATA
                            LASTDAT                     ;DONE TAKING DATA
F907    20 1E FA                      JSR     DIV1      ;DIVIDE DATA BY 4
F90A    20 4D F9                      JSR     ADJUST    ;ADJUST DATA
F90D    A9 C0                         LDA     #$C0      ;SET PRINT DATA FLAG
F90F    85 09                         STA     PDATA
F911    4C C7 F8                      JMP     LOOP      ;GET MORE DATA
F914                        CLOSE
F914    20 94 F9                      JSR     DEBOUN    ;DEBOUNCE SWITCH
F917    A9 00                         LDA     #00       ;TURN OFF COLLECT LIGHT

F919    8D 02 40                      STA     GPPRB
F91C    A9 13                         LDA     #ENFILE   ;SEND END OF FILE TO TA
                                                        PE
F91E    20 C7 F9                      JSR     PRINT
F921    A9 14                         LDA     #CLFILE   ;CLOSE TAPE FILE
F923    20 C7 F9                      JSR     PRINT
F926    E6 0C                         INC     PASS      ;SET NEXT PASS NUMBER
F928    A9 64                         LDA     #100      ;INIT MARKER SW COUNTER
F92A    85 0E                         STA     SWCNT
F92C    AD 00 40                      LDA     GPPRA     ;RESET CA1/CA2
F92F    4C 90 F8                      JMP     START
                            ;
```

```
              ;************************************************
               ************
              ;
              ;                ADD DATA BYTE SUBROUTINE
              ;
              ;************************************************
               ************
              ;
              TOTAL1                      ;ADD DATA
F932  A5 03            LDA     AD1        ;GET DATA1
F934  18               CLC
F935  65 06            ADC     OUT1L      ;LOW BYTE
F937  85 06            STA     OUT1L
F939  A9 00            LDA     #$00       ;ADD CARRY TO HI BYTE
F93B  65 05            ADC     OUT1H      ;HI BYTE
F93D  85 05            STA     OUT1H
              TOTAL2                      ;ADD DATA BYTE 2
F93F  A5 04            LDA     AD2        ;GET DATA2
F941  18               CLC
F942  65 08            ADC     OUT2L      ;LOW BYTE
F944  85 08            STA     OUT2L
F946  A9 00            LDA     #$00       ;ADD CARRY TO HI BYTE
F948  65 07            ADC     OUT2H      ;HI BYTE
F94A  85 07            STA     OUT2H
F94C  60               RTS                ;RETURN
              ;
              ;************************************************
               ************
              ;
              ;                ADJUST DATA
              ;
              ;************************************************
               ************
              ;
F94D          ADJUST
F94D  A9 20            LDA     #32        ;ADD OFFSET
F94F  18               CLC
F950  65 06            ADC     OUT1L      ;ADJUST BYTE 1
F952  85 0F            STA     CHAN1
F954  A9 20            LDA     #32        ;ADD OFFSET
F956  18               CLC
F957  65 08            ADC     OUT2L      ;ADJUST BYTE 2
F959  85 10            STA     CHAN2
              OVER                        ;DATA OUT OF BOUNDS TES
                                           T
F95B  A9 7F            LDA     #127
F95D  C5 0F            CMP     CHAN1      ;TEST BYTE 1
F95F  B0 02            BCS     NEXTB      ;OKAY
F961  85 0F            STA     CHAN1      ;LIMIT TO $FF
F963          NEXTB
F963  C5 10            CMP     CHAN2      ;TEST BYTE 2
F965  B0 02            BCS     FIXD       ;OKAY
F967  85 10            STA     CHAN2      ;LIMIT TO $FF
F969  60      FIXD     RTS                ;RETURN
              ;
              ;************************************************
               ************
```

```
                              ;
                              ;                CHARACTER PRINT
                              ;
                              ;********************************************
                               ************
                              ;
F96A                          CHAR
F96A      24 09                         BIT     PDATA    ;DATA
F96C      10 11                         BPL     CHAR2    ;NO
F96E      AD 01 60                      LDA     ACIASTA  ;ACIA READY?
F971      29 10                         AND     #$10
F973      F0 1E                         BEQ     CHARDN   ;NOT READY
F975      A5 0F                         LDA     CHAN1
F977      8D 00 60                      STA     ACIADAT  ;SEND CHARACTER
F97A      A9 40                         LDA     #$40     ;CLEAR 7,SET 6
F97C      85 09                         STA     PDATA
F97E      60                            RTS              ;RETURN
F97F                          CHAR2
F97F      24 09                         BIT     PDATA    ;MORE DATA
F981      50 10                         BVC     CHARDN   ;NO
F983      AD 01 60                      LDA     ACIASTA  ;ACIA READY?
F986      29 10                         AND     #$10
F988      F0 09                         BEQ     CHARDN   ;NOT READY
F98A      A5 10                         LDA     CHAN2
F98C      8D 00 60                      STA     ACIADAT  ;SEND CHARACTER
F98F      A9 00                         LDA     #$00     ;RESET PRINT FLAG
F991      85 09                         STA     PDATA
F993                          CHARDN
F993      60                            RTS              ;RETURN
                              ;
                              ;********************************************
                               ************
                              ;
                              ;                DEBOUNCE SUBROUTINE
                              ;
                              ;********************************************
                               ************
                              ;
F994                          DEBOUN
F994      20 B6 F9                      JSR     DELAY    ;WAIT 200 MSEC.
F997      AD 02 40                      LDA     GPPRB    ;SWITCH PORT
F99A      29 02                         AND     #$02     ;SWITCH MASK
F99C      D0 13                         BNE     FALSE    ;EARLY SW OPEN
F99E                          SHUT
F99E      20 B6 F9                      JSR     DELAY    ;WAIT 200 MSEC
F9A1      AD 02 40                      LDA     GPPRB    ;SWITCH PORT
F9A4      29 02                         AND     #$02     ;SWITCH MASK
F9A6      F0 F6                         BEQ     SHUT     ;SWITCH IS STILL CLOSED
F9A8      20 B6 F9                      JSR     DELAY    ;OPEN, BUT WAIT 200 MSE
                                                         C
F9AB      A9 01                         LDA     #01
F9AD      85 0D                         STA     SWFLAG   ;VALID SWITCH CLOSURE
F9AF      D0 04                         BNE     RETRN
F9B1      A9 00               FALSE     LDA     #00
F9B3      85 0D                         STA     SWFLAG   ;FALSE SWITCH CLOSURE
F9B5      60                  RETRN     RTS
                              ;
```

```
                         ;**********************************************
                          ************
                         ;
                         ;               DELAY SUBROUTINE
                         ;
                         ;
                         ;**********************************************
                          ************
                         ;
F9B6                     DELAY
F9B6   A9 C8                     LDA     #200     ;200 MS DELAY
F9B8   85 0B                     STA     COUNT2
F9BA                     ONEMS
F9BA   A9 FA                     LDA     #$FA     ;1 MILLISEC COUNT
F9BC   85 0A                     STA     COUNT1
F9BE                     DOWN1
F9BE   C6 0A                     DEC     COUNT1
F9C0   D0 FC                     BNE     DOWN1
F9C2   C6 0B                     DEC     COUNT2
F9C4   D0 F4                     BNE     ONEMS
F9C6   60                        RTS
                         ;
                         ;
                         ;**********************************************
                          ************
                         ;
                         ;               PRINT SUBROUTINE
                         ;
                         ;**********************************************
                          ************
                         ;
F9C7                     PRINT
F9C7   AA                        TAX              ;MOVE CHARACTER
F9C8                     TSTXMT
F9C8   AD 01 60                  LDA     ACIASTA  ;ACIA READY
F9CB   29 10                     AND     #$10
F9CD   F0 F9                     BEQ     TSTXMT   ;NOT READY
F9CF   8E 00 60                  STX     ACIADAT  ;SEND CHARACTER
F9D2   60                        RTS              ;RETURN
                         ;
                         ;**********************************************
                          ************
                         ;
                         ;               READ A/D CONVERTERS
                         ;
                         ;**********************************************
                          ************
                         ;
F9D3                     READAD
                         CONST                    ;START CONVERSION
F9D3   A9 0C                     LDA     #$0C     ;CS1,CS2=L
F9D5   8D 02 20                  STA     ADPRB
F9D8   A9 04                     LDA     #$04     ;WR=L
F9DA   8D 02 20                  STA     ADPRB
F9DD   A9 0F                     LDA     #$0F     ;CNT LINES=H
F9DF   8D 02 20                  STA     ADPRB
                         DONCV                    ;DONE CONVERSION
```

```
F9E2   AD 02 20                 LDA     ADPRB
F9E5   29 30                    AND     #$30      ;MASK INTR BITS
F9E7   D0 F9                    BNE     DONCV     ;NOT READY
F9E9                    READ1
F9E9   A9 0E                    LDA     #$0E      ;CS1=L
F9EB   8D 02 20                 STA     ADPRB
F9EE   A9 0A                    LDA     #$0A      ;RD=L
F9F0   8D 02 20                 STA     ADPRB
F9F3   AD 00 20                 LDA     ADPRA     ;READ DATA
F9F6   C9 7F                    CMP     #127
F9F8   90 02                    BCC     R_OK1
F9FA   A9 7F                    LDA     #127
F9FC                    R_OK1
F9FC   85 03                    STA     AD1
F9FE   A9 0F                    LDA     #$0F      ;CONTROL LINES=H
FA00   8D 02 20                 STA     ADPRB
FA03                    READ2
FA03   A9 0D                    LDA     #$0D      ;CS2=L
FA05   8D 02 20                 STA     ADPRB
FA08   A9 09                    LDA     #$09      ;RD=L
FA0A   8D 02 20                 STA     ADPRB
FA0D   AD 00 20                 LDA     ADPRA     ;READ DATA
FA10   C9 7F                    CMP     #127
FA12   90 02                    BCC     R_OK2
FA14   A9 7F                    LDA     #127
FA16                    R_OK2
FA16   85 04                    STA     AD2
FA18   A9 0F                    LDA     #$0F      ;CONTROL LINES=H
FA1A   8D 02 20                 STA     ADPRB
FA1D   60                       RTS               ;RETURN
                        ;
                        ;********************************************
                         ************
                        ;
                        ;               SHIFT SUBROUTINE
                        ;
                        ;********************************************
                         ************
                        ;
FA1E                    DIV1
FA1E   18                       CLC
FA1F   66 05                    ROR     OUT1H     ;HIGH BYTE
FA21   66 06                    ROR     OUT1L     ;LOW BYTE
FA23   18                       CLC
FA24   66 05                    ROR     OUT1H
FA26   66 06                    ROR     OUT1L
FA28                    DIV2
FA28   18                       CLC
FA29   66 07                    ROR     OUT2H     ;HIGH BYTE
FA2B   66 08                    ROR     OUT2L     ;LOW BYTE
FA2D   18                       CLC
FA2E   66 07                    ROR     OUT2H
FA30   66 08                    ROR     OUT2L
FA32   60                       RTS               ;RETURN
                        ;********************************************
                         ************
                        ;
```

```
                          ;                READ UART
                          ;
                          ;        WAIT 250 MS FOR RESPONSE
                          ;        RETURN FF IF NO RESPONSE
                          ;        RETURN CHAR IF RESPONSE
                          ;
                          ;**********************************************
                           ************

FA33                      READTP
FA33   A2 FF                       LDX      #$OFF    ;256 TRIES
FA35                      READLP1
FA35   A9 01                       LDA      #01
FA37   85 0B                       STA      COUNT2   ;WAIT ONE MS
FA39   20 BA F9                    JSR      ONEMS
FA3C   CA                          DEX
FA3D   FO 0B                       BEQ      NO_READ  ;GIVE UP
FA3F   AD 01 60                    LDA      ACIASTA  ;GET STATUS
FA42   29 08                       AND      #$08     ;RDRF FLAG
FA44   FO EF                       BEQ      READLP1  ;TRY AGAIN
FA46   AD 00 60                    LDA      ACIADAT  ;GET DATA
FA49   60                          RTS
FA4A                      NO_READ
FA4A   A9 FF                       LDA      #$OFF
FA4C   60                          RTS
                          ;
                          ;**********************************************
                           ************
                          ;
                          ;                VECTORS
                          ;
                          ;**********************************************
                           ************
                          ;
FFFC                               .ORG     FFFCH    ;RESET VECTOR
FFFC   00                          .BYTE    00H      ;LOW BYTE
FFFD   F8                          .BYTE    F8H      ;HIGH BYTE
                          ;
                          ;**********************************************
                           ************
                          ;
FFFE                               .END
```

```
************   C R O S S    R E F E R E N C E    T A B L E    ************
ACIACMD    = 6002  :      96
ACIACTL    = 6003  :      98
ACIADAT    = 6000  :     313     324     387     478
ACIASTA    = 6001  :     309     320     384     475
ACK_RW     = 0007  :     150
ACK_WR     = 0006  :     183
AD1        = 0003  :     191     257     418
AD2        = 0004  :     192     265     431
ADCRA      = 2001  :     105     111
ADCRB      = 2003  :     101     106     112
ADDRA      = 2000  :     107
ADDRB      = 2002  :     109
ADJUST       F94D  :     232
ADPRA      = 2000  :     413     426
ADPRB      = 2002  :     103     114     116     118     120     399     401
                         403     405     410     412     420     423     425
                         433
CHAN1      = 000F  :     194     284     291     293     312
CHAN2      = 0010  :     195     288     295     297     323
CHAR         F96A  :     228
CHAR2        F97F  :     308
CHARDN       F993  :     311     319     322
CLEAR        F8B8  :
CLFILE     = 0014  :     242
CLOSE        F914  :     214
CONST        F9D3  :
COUNT1     = 000A  :     366     368
COUNT2     = 000B  :     363     370     471
DATCNT     = 0002  :     207     223
DEBOUN       F994  :     173     237
DELAY        F9B6  :     136     337     342     346
DIV1         FA1E  :     231
DIV2         FA28  :
DONCV        F9E2  :     407
DOWN1        F9BE  :     369
ENFILE     = 0013  :     139     240
FALSE        F9B1  :     340
FILEO        F89E  :     182
FILLP1       F8A3  :     184
FIXD         F969  :     296
GPCRA      = 4001  :     123     129     215
GPCRB      = 4003  :     124     130
GPDRA      = 4000  :     125
GPDRB      = 4002  :     127
GPPRA      = 4000  :     196     217     247
GPPRB      = 4002  :     170     186     210     239     338     343
ICMD       = 006B  :      95
ICTL       = 003E  :      97
INITAD       F80F  :
INITCPU      F800  :
INITGP       F845  :
INITLP1      F879  :     151
INITPASS     F884  :
INITREAD     F85D  :
```

```
INITRWD       F874  :    149
INITSER       F805  :
INITVAR       F888  :
LASTDAT       F907  :    224
LOOP          F8C7  :    235
NEXTB         F963  :    292
NO_READ       FA4A  :    474
ONEMS         F9BA  :    371      472
ONESEC        F866  :    138
OPFILE     =  0012  :    177
OUT1H      =  0005  :    199      262      263      444      447
OUT1L      =  0006  :    200      259      260      283      445      448
OUT2H      =  0007  :    201      270      271      451      454
OUT2L      =  0008  :    202      267      268      287      452      455
OVER          F95B  :
PASS       =  000C  :    154      187      244
PCHAR         F901  :    219
PCNTR         F8D9  :    216      226      229
PDATA      =  0009  :    193      234      307      315      318      326
PLSCNT     =  0001  :    205      218      221
PRINT         F9C7  :    134      140      145      178      188      241      243
PULSE         F8E4  :    212
RAM        =  0000  :
RDFILE     =  0011  :    133
READ1         F9E9  :
READ2         FA03  :
READAD        F9D3  :    222
READLP1       FA35  :    477
READTP        FA33  :    147      180
RESTART       F8D1  :
RETRN         F9B5  :    349
REWND      =  001A  :    144
R_OK1         F9FC  :    415
R_OK2         FA16  :    428
SHUT          F99E  :    345
START         F890  :    172      175      248
SWCNT      =  000E  :    157      213      246
SWFLAG     =  000D  :    159      174      348      351
TAPERD        F85F  :    142
TOTAL1        F932  :    225
TOTAL2        F93F  :
TSTXMT        F9C8  :    386


          LINES ASSEMBLED :   496              ASSEMBLY ERRORS :     0
```

A.5 TAPEREAD PROGRAM

```
{ Link Taperead+Util,,Nul,Pascal+Ibm3 }

{ $INCLUDE:'SystemIO.Int' }

{ $INCLUDE:'Screen.INT'    }

Program Tape_Read (Input,Output);

USES SystemIO;
```

```
USES Screen;

{ Program To Read Information Off The Tape Unit And Store It In A
  Format Usable By Bridge                                          }
{ Tape Unit 9600 Baud 7 Bits Even Parity }

Const
  CtrlZ         =       Chr(26);
  CtrlS         =       Chr(19);
  CtrlQ         =       Chr(17);
  CtrlO         =       Chr(15);
  Pad           =       Chr(31);

Type
  Buff_Type     =       Array [1..24000] Of Char;       { 6000 Feet }

Var
  TimeOut       :       Boolean;
  Dev_Setup     :       Dev;
  Dev_Setup2    :       Dev;
  Length        :       Integer;
  Bridge_Id     :       LString(32);
  Date          :       LString(12);
  Square        :       Boolean;
  Right         :       Boolean;
  DistR         :       Integer;
  DistL         :       Integer;
  No_Pass       :       Integer;
  Ch            :       Char;
  Left_Buff     :       Buff_Type;
  Left_L        :       Integer;
  Right_Buff    :       Buff_Type;
  Right_L       :       Integer;
  Temp          :       File Of Char;
  Data          :       File Of Char;
  OK            :       Boolean;
  Pass          :       Char;
  Expected      :       Char;
  First         :       Char;

Function Dosxqq ( Command, Parameter : Word ) : Byte; Extern;

Function Get_Int ( Limit : Integer ) : Integer; Extern;

Procedure Fit ( Var Data : Buff_Type; Var Len : Integer );

Const
  Debug         =       False;

Var
  i,j           :       Integer;
```

```
BEGIN
  If (Debug) Then
    Writeln(' Procedure Fit');
  If (Len < Length) Then
    BEGIN
      i := (Length - Len) Div 2;
      For j := (Len + i) DownTo i + 1 Do
        Data[j] := Data[j - i];
      For j := 1 To i Do
        Data[j] := Pad;
      For j := (Len + i + 1) To Length Do
        Data[j] := Pad;
    END
  Else BEGIN
         i := (Len - Length) Div 2;
         For j := 1 To Length Do
           Data[j] := Data[j + i];
       END;
END;

Function Char_From_Tape : Char;

Const
  Debug           =       False;

Var
  Timer           :       Integer;
  Buff1           :       Stat;
  Buff2           :       Stat;
  i               :       Integer;
  Ch              :       Char;

BEGIN
  If (Debug) Then
    Writeln(' Function Char_From_Tape');
  REPEAT
    Queue_Status(Buff1,Buff2);
    Timer := i;
  UNTIL (Ord(Buff1[6]) = 0);    { Check To See If Everything Has Been }
  i := 2000;                    { Sent To The Tape                    }
  While ((i > 0) And (Ord(Buff1[2]) = 0)) Do
    BEGIN                       { Loop To Wait For Something To Be   }
      i := i - 1;               { Returned From The Tape Or Until A }
      Timer := i;
      Queue_Status(Buff1,Buff2);{ Number (2000) Of Checks Have Been }
    END;                               { Made And A TimeOut Occurs         }
  If (Ord(Buff1[2]) <> 0) Then
    Ch := GetChar(1)
  Else BEGIN
         TimeOut := True;
         Ch := Pad;
       END;
  Char_From_Tape := Ch;
END;

Procedure Write_File;
{ Procedure That Writes The Buffers To The Tempoary File To Await }
{ Further Processing                                              }

Var
  i               :       Integer;

BEGIN
  For i := 1 To Length Do
    Write(Temp,Left_Buff[i]);
  For i := 1 To Length Do
    Write(Temp,Right_Buff[i]);
    Write(Temp,Right_Buff[i]);
END;

Procedure Insert_Blank;

Var
  i             :       Integer;

BEGIN
  For i := 1 To Length Do               { Left Pass All Pad Chars }
    Write(Temp,Pad);
  For i := 1 To Length Do               { Right Pass All Pad Chars }
    Write(Temp,Pad);
END;

Procedure Get_File;

Const
  Debug         =       False;

Var
  Buff1         :       Stat;
  Buff2         :       Stat;
  InChar        :       Char;

BEGIN
  If Debug Then
    Writeln('Procedure Get_File');
  OK := True;
  Left_L := 0;
  Right_L := 0;
  Queue_Status(Buff1,Buff2);
  If (Ord(Buff1[2]) = 0) Then   { If Nothing Is In The Tape Buffer }
    PutChar(1,CtrlQ);           { Then Send The Read Command CtrlQ }
  Pass := Char_From_Tape;
  If (Not TimeOut) And (InChar <> CtrlS) Then
    BEGIN
      REPEAT
        InChar := Char_From_Tape;
        If (InChar <> CtrlS) And (Not TimeOut) Then
          BEGIN
            Left_L := Left_L + 1;
            Left_Buff[Left_L] := InChar;
            InChar := Char_From_Tape;
            If (InChar <> CtrlS) And (Not TimeOut) Then
              BEGIN
                Right_L := Right_L + 1;
                Right_Buff[Right_L] := InChar;
              END;
          END;
        UNTIL (TimeOut) Or (InChar = CtrlS);
    END;
END;

Procedure Fit_Buffers;

BEGIN
  If (Left_L <> Length) Then
    Fit(Left_Buff,Left_L);
  If (Right_L <> Length) Then
    Fit(Right_Buff,Right_L);
  No_Pass := No_Pass + 1;
  Expected := Succ(Expected);
END;

Procedure Decide ( Var Ok : Boolean );
```

```
Var
  Ch               :       Char;

BEGIN
  Ok := True;
  If (Not TimeOut) Then
    BEGIN
      Writeln(' Pass #',(Ord(Pass) - Ord(First) + 1):3,Left_L:6,' Samples Taken'
);
      If (Pass = Expected) Then
        If (No_Pass = 0) Then
          If (Left_L > 30) Then         { First Pass Long Enough }
            BEGIN
              Length := Left_L;
              Fit_Buffers;
            END
          Else BEGIN
                 Writeln(' The First Pass Is Rather Short');
                 Write(' Do You Want To Use It ?');
                 REPEAT
                   Ch := Chr(Dosxqq(6,255));
                 UNTIL (Ch In ['n','N','y','Y']);
                 Writeln;
                 If (Ch In ['y','Y']) Then
                   BEGIN
                     Length := Left_L;
                     Fit_Buffers;
                   END
                 Else Ok := False;
               END
        Else If (Left_L > 0.9 * Length) Then
               Fit_Buffers
             Else BEGIN
                    Writeln(' This Pass Is Only ',(Left_L / Length *  100):5:2,'
% As Long As The First');
                    Write(' Do You Want To Use It ?');
                    REPEAT
                      Ch := Chr(Dosxqq(6,255));
                    UNTIL (Ch In ['n','N','y','Y']);
                    Writeln;
                    If (Ch In ['y','Y']) Then
                      Fit_Buffers
                    Else Ok := False;
                  END
      Else BEGIN
             Writeln(' Pass #',(Ord(Expected) - Ord(First) + 1):3,' Was Expected
');
             Writeln(' Do You Want To Use It For Pass #',(Ord(Expected) - Ord(Fi
rst) + 1):3,', ');
             Write('    Not Use It Or Insert A Blank Pass (Y,N,I) ?');
             REPEAT
               Ch := Chr(Dosxqq(6,255));
             UNTIL (Ch In ['n','N','y','Y','i','I']);
             Writeln;
             If (Ch In ['y','Y']) Then
               BEGIN
                 Expected := Pass;
                 If (No_Pass = 0) Then
                   First := Pass;
                 Decide(Ok);
               END
             Else If (Ch In ['i','I']) Then
                    BEGIN
                      If (No_Pass = 0) Then
                        Length := Left_L;
                      Insert_Blank;
                      No_Pass := No_Pass + 1;
                      Expected := Succ(Expected);
                      Decide(Ok);
                    END
             Else Ok := False;
           END;
    END
  Else Ok := False;
END;

Procedure Trim_Edges ( Var Buff : Buff_Type; p, c : Integer );

Const
  Debug           =       False;

Var
  i               :       Integer;
  Off_Top         :       Integer;
  Off_Bottom      :       Integer;

BEGIN
  If (Not Right) Then
    p := c - p - 1;
  p := p * 9 + 5;
  c := c * 9 + 4;
  Off_Top := (DistR + ((c - p) * (DistL - DistR)) Div c) Div 3;
  Off_Bottom := DistR Div 3 + DistL Div 3 - Off_Top;
  If (Debug) Then
    Writeln(' Trim  Top = ',Off_Top:3,' Bottom = ',Off_Bottom:3);
  For i := 1 To Off_Top Do
    Buff[i] := Pad;
  For i := Length DownTo (Length - Off_Bottom) Do
    Buff[i] := Pad;
END;

Procedure Write_Data_File;

Const
  Debug           =       False;

Var
  Name            :       LString(32);
  i,j             :       Integer;

BEGIN
  Write(' Data File Name : ');
  Readln(Name);
  Assign(Data,Name);
  Reset(Temp);
  Rewrite(Data);
  For i := 0 To 32 Do
    Write(Data,Bridge_Id[i]);
  Write(Data,Chr(No_Pass));
  Write(Data,Chr(Length Div 256),Chr(Length Mod 256));
  If (Square) Then
    Write(Data,'N')
  Else Write(Data,'S');
  Write(Data,Chr(DistR Div 256),Chr(DistR Mod 256));
  Write(Data,Chr(DistL Div 256),Chr(DistL Mod 256));
  If (Right) Then
    Write(Data,'R')
  Else Write(Data,'L');
  For i := 42 To 54 Do
    Write(Data,Date[i - 42]);
  For i := 55 To 63 Do
```

```
    Write(Data,    );
  For i := 0 To (No_Pass - 1) Do
    BEGIN
      If (Odd(i)) Then
        For j := 1 To Length Do
          Read(Temp,Left_Buff[j])
      Else For j := Length DownTo 1 Do
             Read(Temp,Left_Buff[j]);
      If (Odd(i)) Then
        For j := 1 To Length Do
          Read(Temp,Right_Buff[j])
      Else For j := Length DownTo 1 Do
             Read(Temp,Right_Buff[j]);
      If (Not Square) Then
        Trim_Edges(Left_Buff,2 * i,2 * No_Pass - 1);
      For j:= 1 To Length Do
        Write(Data,Left_Buff[j]);
      If (Not Square) Then
        Trim_Edges(Right_Buff,2 * i + 1,2 * No_Pass - 1);
      For j := 1 To Length Do
        Write(Data,Right_Buff[j]);
    END;
  Close(Data);
END;

BEGIN
  Clear_Screen;
  Home;
  Writeln('                        IOWA D.O.T.');
  Writeln;
  Setup1;
  Dev_Status(Dev_Setup,Dev_Setup2);
  Dev_Setup[0] := Chr(13);
  Dev_Setup[1] := Chr(2);
  Dev_Setup[2] := Chr(120);
  Dev_Setup[3] := Chr(0);
  Dev_Setup[4] := Chr(0);
  Dev_Setup[5] := Chr(0);
  Dev_Setup[6] := CtrlS;
  Dev_Setup[7] := CtrlQ;
  Dev_Control1(Dev_Setup);
  Dtr1_True;
  PutChar(1,CtrlZ);             { Rewind The Tape }
  Write(' Bridge ID : ');
  First := 'A';
  Readln(Bridge_Id);
  Write(' Date : ');
  Readln(Date);
  TimeOut := False;
  REPEAT
    Write(' Start On The Right Or Left : ');
    Readln(Ch);
  UNTIL (Ch In ['l','L','r','R']);
  Right := (Ch In ['r','R']);
  REPEAT
    Write(' Normal Or Slewed : ');
    Readln(Ch);
  UNTIL (Ch In ['n','N','s','S']);
  Square := (Ch In ['n','N']);
  DistR := 0;
  DistL := 0;
  If (Not Square) Then
    BEGIN
      Write(' Distance In Inches On The Right : ');
      DistR := Get_Int(4);
      Write(' Distance In Inches On The Left : ');
      DistL := Get_Int(4);
    END;
  Length := 0;
  Expected := 'A';
  Assign(Temp,'Scratch.me');
  Rewrite(Temp);
  Ch := Char_From_Tape;
  While (Not TimeOut) Do
    BEGIN
      Get_File;
      Decide(Ok);
      If (Ok) Then
        Write_File;
    END;
  PutChar(1,Ctrl1Z);     { Rewind The Tape }
  Disable1:
  If (No_Pass <> 0) Then
    Write_Data_File
  ELse Writeln(' No Passes Found Or Used ');
  Discard(Temp);
END.
```

A.6 BRIDGE PLOT PROGRAM

```
{ Link Bridge+Util,,Nul,Pascal+Ibm3 }

{ $INCLUDE:'Screen.INT'    }
```

```
Program Bridge (Input,Output);

USES Screen(Home, Clear_Screen, Clear_Line, Up, Down, Left, Right, Pos,
              Reverse, Norm, Setup_Screen, Cursor_Off, Cursor_On);

{ Program to analyse digitied soundings of bridge sections }

CONST
  Max_Length    =       24000;          { 6000 Feet }
  Min_Value     =        31;
  Max_Value     =       127;
  Def_Delam     =       400;            { In Millivolts }
  Increment     =        21;            { In Millivolts }
  First         =      'A';
  Space         =         8;
  Esc           =       Chr(27);
  Def_Printer   =       1;              { TI 855 Compatible }

Type
  Tape_Type     =       Array [1..Max_Length] Of Char;
  Ord_Type      =       Array [1..Max_Length] Of Integer;
  Dist_Type     =       Array [0..Max_Value] Of Integer;
  Bit_Map_Type  =       Super Array [1..*,1..*] Of Byte;
  Bit_Map_Ptr   =       ^Bit_Map_Type;
  ID_Type       =       LString(32);
  Date_Type     =       LString(12);
  Count_Type    =       Array [0..255] Of Integer;
  Dens_Type     =       LString(2);

VAR
  Bridge_ID     :       ID_Type;
  Date          :       Date_Type;
  Num_Passes    :       Integer;
  Normal        :       Boolean;
  Dr, Dl        :       Integer;
  Start_Right   :       Boolean;
  Delam         :       Integer;
  Left_Data     :       Tape_Type;
  Right_Data    :       Tape_Type;
  Length        :       Integer;
  Width         :       Integer;
  Pass          :       Integer;
  OK            :       Boolean;
  Data_Valid    :       Boolean;
  Init          :       Boolean;
  Init_Printer  :       Boolean;
  Bit_Map       :       Bit_Map_Ptr;
  Offset        :       Integer;
  Map_Length    :       Integer;
  Prn           :       Text;
  Info          :       File Of Char;
  Power         :       Array [0..7] Of Integer;
  Total         :       Integer;
  Bad           :       Integer;
  Percent       :       Real;
  Printer_Type  :       Integer;
  Star          :       Dens_Type;
```

```
Function Dosxqq ( Command, Parameter : Word ) : Byte; Extern;

Function Get_Int ( Limit : Integer ) : Integer; Extern;


Procedure Get_Information;

{ Procedure to get from the user the name of a file containing
  the bridge data with the following format

        Information                     Bytes In File

        Bridge_Id                0           -         32
        # Passes                                      33
        Length in samples        34          -         35
        Normal or Slewed                     36
        Distance in inches (R)   37          -         38
        Distance in inches (L)   39          -         40
        Start_Right                          41
        Date Info                42          -         54
        Extra space              55          -         63
        Bridge data              64          -         ?        }

CONST
  Debug          =        False;

VAR
  Ch             :        Char;
  File_Name      :        ID_Type;
  Found          :        Boolean;
  i              :        Integer;

BEGIN
  Data_Valid := True;
  Num_Passes := 0;
  Length := 0;
  Normal := True;
  Dr := 0;
  Dl := 0;
  Start_Right := True;
  REPEAT
    Clear_Screen;
    Home;
    Writeln('                  IOWA D.O.T.');
    Writeln;
    Write(' Bridge Data File : ');
    Readln(File_Name);
    Assign(Info,File_Name);
    Info.Trap := True;           { Enable Error Trapping }
    Reset(Info);
    If (Info.Errs <> 0) Then     { Error With File }
      BEGIN
        Info.Errs := 0;          { Reset Error }
        Writeln(' ',File_Name:Ord(File_Name[0]),' Not Found');
        Found := False;
      END
    Else Found := True;
  Until (Found);                 { No Errors With File }
  Writeln;
  For i := 0 To 32 Do                     { Read Bridge ID }
    If (Not Eof(Info)) Then
      Read(Info,Bridge_Id[i])
    Else Data_Valid := False;
  If (Data_Valid) Then
    Writeln(Bridge_Id:48)
```

```
Else Writeln(' ERROR IN DATA FILE');
Writeln;
If (Not Eof(Info)) And (Data_Valid) Then       { Get Number Of  }
  BEGIN                                        { Passes In File }
    Read(Info,Ch);
    Num_Passes := Ord(Ch);
  END
Else Data_Valid := False;
If (Not Eof(Info)) And (Data_Valid) Then
  BEGIN
    Read(Info,Ch);
    Length := Ord(Ch);
  END
Else Data_Valid := False;
If (Not Eof(Info)) And (Data_Valid) Then
  BEGIN
    Read(Info,Ch);
    Length := Length * 256 + Ord(Ch);
  END
Else BEGIN
       Data_Valid := False;
       Length := 0;
     END;
If (Not Eof(Info)) And (Data_Valid) Then
  BEGIN
    Read(Info,Ch);
    Normal := (Ch In ['n','N']);
  END
Else Data_Valid := False;
If (Not Eof(Info)) And (Data_Valid) Then
  BEGIN
    Read(Info,Ch);
    Dr := Ord(Ch);
  END
Else Data_Valid := False;
If (Not Eof(Info)) And (Data_Valid) Then
  BEGIN
    Read(Info,Ch);
   Dr := Dr * 256 + Ord(ch);
  END
Else BEGIN
       Data_Valid := False;
       Dr := 0;
     END;
If (Not Eof(Info)) And (Data_Valid) Then
  BEGIN
    Read(Info,Ch);
    Dl := Ord(Ch);
  END
Else Data_Valid := False;
If (Not Eof(Info)) And (Data_Valid) Then
  BEGIN
    Read(Info,Ch);
   Dl := Dl * 256 + Ord(ch);
  END
Else BEGIN
       Data_Valid := False;
       Dl := 0;
     END;
If (Not Eof(Info)) And (Data_Valid) Then
  BEGIN
    Read(Info,Ch);
    Start_Right := (Ch In ['r','R']);
  END
Else Data_Valid := False;
For i := 42 To 54 Do
```

```
    If (Not Eof(Info)) Then
      Read(Info,Date[i - 42]);
  For i := 55 To 63 Do
    If (Not Eof(Info)) Then
      Read(Info,Ch)
    Else Data_Valid := False;
  REPEAT
    Write(' Delamination In mV (',Def_Delam:3,') : ');
    Delam := Get_Int(4);
    If (Delam = 0) Then Delam := Def_Delam;
  UNTIL (Delam > 0);
END;


Procedure Read_Data_From_Tape;

CONST
  Debug           =       False;

VAR
  L_Length        :       Integer;
  R_Length        :       Integer;
  Pass_Found      :       Integer;
  Answer          :       Char;
  i               :       Integer;

BEGIN
  If Debug Then
      Writeln(' Read Data From Tape');
  For i := 1 To Length Do
    If (Not Eof(Info)) Then
      Read(Info,Left_Data[i])
    Else Data_Valid := False;
  For i := 1 To Length Do
    If (Not Eof(Info)) Then
      Read(Info,Right_Data[i])
    Else Data_Valid := False;
  Ok := Data_Valid;
END;

Function BaseLine (Var Data : Tape_Type) : Integer;

{ Determines The Baseline Of The Data. Assumes That The Most
  Frequently Occuring Value Is The Baseline                 }

CONST
  Debug           =       False;

VAR
  Dist            :       Dist_Type;
  Base            :       Integer;
  i               :       Integer;

BEGIN
  If Debug Then Writeln(' BaseLine');
  For i := Min_Value To Max_Value Do             { Zeroes The Distribution }
    Dist[i] := 0;                                { Array                   }
  For i := 1 To Length Do                        { Adds Up The Occurances  }
    Dist[Ord(Data[i])] := Dist[Ord(Data[i])] + 1;{ Of The Values           }
  i := Min_Value + 1;
  Base := i;
  While (i <= 100) Do                    { Searches For The Most   }
    BEGIN                                { Frequent Value In The   }
      If (Dist[Base] < Dist[i]) Then     { Array                   }
        Base := i;
      i := i + 1;
    END;
  BaseLine := Base;
END;

Procedure Insert_Into_BitMap;

{ Inserts The Tape Data Into The Bit Map }

CONST
  Debug          =      False;

VAR
  i, j           :       Integer;
  Row            :      Integer;
  Col           :       Integer;
  Base_L         :       Integer;
  Base_R         :       Integer;
  Delam_L        :       Integer;
  Delam_R        :       Integer;
  Bit            :       Byte;

BEGIN
  If Debug Then Writeln(' Insert_Into_BitMap');
  If (Init) Then                                   { Dynamically Allocates }
    BEGIN                                          { The Memory Needed     }
      Map_Length := ((Length - 1) Div 8 + 1);
      Offset := Map_Length * 8 - Length;
      New(Bit_Map,Map_Length,Width);
      For i := 1 To Map_Length Do
        For j := 1 To Width Do
          Bit_Map^[i,j] := 0;
      Init := False;
    END;
  If (Start_Right) Then
    Col := Width - Pass * 2 + 1
  Else Col := Pass * 2 - 1;
  Base_L := BaseLine(Left_Data);
  Base_R := BaseLine(Right_Data);
  Delam_L := Base_L + Delam Div Increment;
  Delam_R := Base_R + Delam Div Increment;
  For i := (Length - 1) DownTo 0 Do
    BEGIN
      Bit := Power[(i + Offset) Mod 8];
      Row := (i + Offset) Div 8 + 1;
      If (Ord(Left_Data[i + 1]) >= Delam_L) Then
        Bit_Map^[Row,Col] := Bit_Map^[Row,Col] + Bit;
      If (Ord(Right_Data[i + 1]) >= Delam_R) Then
        Bit_Map^[Row,Col + 1] := Bit_Map^[Row,Col + 1] + Bit;
    END;
END;

Procedure Percentage;

VAR
  i, j, k        :       Integer;

BEGIN
  Total := 0;
  Bad := 0;
  Percent := 0;
  For i := 1 To Map_Length Do
    For j := 1 To Width Do
      For k := 0 To 7 Do
        If (Odd(Bit_Map^[i,j] Div Power[k])) Then
          Bad := Bad + 1;
  Total := Length * Width;
  If (Not Normal) Then
    Total := Total - (Dr + Dl) * Width;
  Percent := Bad / Total;
END;

Procedure Process_Section;

CONST
  Debug         =       False;

BEGIN
  If Debug Then Writeln(' Process_Section');
  OK := True;
  Init := True;
  Pass := 1;
  Width := (2 * Num_Passes);
  While (Pass <= Num_Passes) Do
    BEGIN
      Read_Data_From_Tape;
      If (Ok) Then
        Insert_Into_BitMap
      Else Pass := Num_Passes;
      Pass := Pass + 1;
    END;
  Percentage;
END;

Procedure Header;

BEGIN
  If (Init_Printer) And (Printer_Type = 1) Then
    BEGIN
      Writeln(Prn,Esc,'@');
      Write(Prn,Esc,'q');
    END
  Else If (Init_Printer) And (Printer_Type = 2) Then
         BEGIN
           Writeln(Prn,Esc,'@');
           Write(Prn,Esc,'G');
         END
       Else If (Init_Printer) And (Printer_Type = 3) Then
              BEGIN
                Writeln(Prn,Esc,'@');
                Write(Prn,Esc,'4');
              END;
  Init_Printer := False;
  Writeln(Prn,'IOWA D.O.T.':45);
  Writeln(Prn,'DELAMINATION MAP':48);
  Writeln(Prn);
  Writeln(Prn,Bridge_Id:46);
  Writeln(Prn);
  Writeln(Prn,Date:(Ord(Date[0]) + 18),' ':(21 - Ord(Date[0])),'Delamination at
',Delam:5,' mV');
  Writeln(Prn);
  Write(Prn,'Length = ':20,((Length * 3) Div 12):2,' Feet ',((Length * 3) Mod 12
):2,' Inches');
  Writeln(Prn,'Width = ':11,((Width * 9) Div 12):2,' Feet ',((Width * 9) Mod 12)
:2,' Inches');
  Writeln(Prn);
  If (Not Normal) Then
    BEGIN
      Write(Prn,'Skew Distance On Left : ':26,Dl:4,'Inches':7);
      Writeln(Prn,'Skew Distance On Right : ':27,Dr:4,'Inches':7);
      Writeln(Prn);
    END;
  Write(Prn,'Total Area = ':24,(Total / 144 * 27):7:2,' Square Feet');
  Writeln(Prn,'Percentage Bad = ':20,(Percent * 100):4:2,'%');
  Writeln(Prn);
END;

Procedure Dumb_Printer;

BEGIN
  Header;
END;

Procedure Print_Top ( Density : Dens_Type );

VAR
  i               :       Integer;

BEGIN
  Write(Prn,Esc,Density,Chr((12 * Width + 5) Mod 256),Chr((12 * Width + 2) Div 2
56));
  For i := 1 To (12 * Width + 5) Do
    Write(Prn,Chr(1));
END;

Procedure Print_Bar ( Density : Dens_Type );

BEGIN
  Write(Prn,Esc,Density,Chr(1),Chr(0),Chr(255));
END;

Procedure Print_Tic ( Density : Dens_Type; Which : Integer );

BEGIN
  Write(Prn,Esc,Density,Chr(3),Chr(0));
  If (Odd(Which)) Then
    Write(Prn,Chr(128),Chr(128))
  Else Write(Prn,Chr(0),Chr(0));
  Write(Prn,Chr(128));
END;

Procedure Print_Bottom ( Density : Dens_Type );

VAR
  i,j             :       Integer;
  Spacing         :       Integer;

BEGIN
  Write(Prn,Esc,Density,Chr((12 * Width + 5) Mod 256),Chr((12 * Width + 2) Div 2
56));
  Write(Prn,Chr(128),Chr(128),Chr(128));
  For i := 0 To (Width * 12 - 1) Do
    If (i Mod 16 = 0) Then
      Write(Prn,Chr(248))
    Else If (i Mod 8 = 0) Then
           Write(Prn,Chr(224))
         Else Write(Prn,Chr(128));
  If ((Width * 12) Mod 16 = 0) Then
    Write(Prn,Chr(248),Chr(248))
  Else If ((Width * 12) Mod 8 = 0) Then
         Write(Prn,Chr(224),Chr(224))
       Else Write(Prn,Chr(128),Chr(192));
  Writeln(Prn);
  Write(Prn,'0':6);
  If (Printer_Type = 1) Or (Printer_Type = 3) Then
    Spacing := 18
  Else If (Printer_Type = 2) Then
         Spacing := 20
       Else Spacing := 1;
  For i := 1 To ((Width * 3) Div 8) Do
    BEGIN
      Write(Prn,Esc,Density,Chr(Spacing),Chr(0));
      For j := 1 To Spacing Do
        Write(Prn,Chr(0));
      Write(Prn,(i + i):2);
    END;
END;

Procedure Scale_4_To_3 ( Density : Dens_Type );

CONST
  Debug         =       False;

VAR
  N1            :       Char;
  N2            :       Char;
  i, j, k, m, n :       Integer;
  Ch            :       Integer;
  Bit           :       Byte;

BEGIN
  Header;
  n := (Length - 1) Div 8;
  If Debug Then
    Writeln(' Scale 4 To 3 - Density = ',Density);
  N1 := Chr((Width * 12) Mod 256);
  N2 := Chr((Width * 12) Div 256);
  Writeln(Prn,Esc,'3',Chr(24));          { Sets Line Spacing }
  Writeln(Prn);
  Writeln(Prn,'FEET':5);
  Write(Prn,' ':5);
  Print_Top(Density);
  Writeln(Prn);
  For i := 1 To Map_Length Do
    BEGIN
      Write(Prn,(n - i + 2) * 2:5);
      Print_Tic(Density,1);
      Print_Bar(Density);
      Write(Prn,Esc,Density,N1,N2);
      For k := 1 To Width Do
        BEGIN
          Bit := 1;
          If (Odd(Bit_Map^[i,k] Div Bit)) Then
            Ch := 240
          Else Ch := 0;
          Bit := Bit + Bit;
          If (Odd(Bit_Map^[i,k] Div Bit)) Then
            Ch := Ch + 15;
          For m := 1 To 12 Do
            Write(Prn,Chr(Ch));
        END;
      Print_Bar(Density);
      Writeln(Prn);
      Write(Prn,' ':5);
      Print_Tic(Density,2);
      Print_Bar(Density);
      Write(Prn,Esc,Density,N1,N2);
      For k := 1 To Width Do
        BEGIN
          Bit := 4;
          If (Odd(Bit_Map^[i,k] Div Bit)) Then
            Ch := 240
          Else Ch := 0;
          Bit := Bit + Bit;
          If (Odd(Bit_Map^[i,k] Div Bit)) Then
```

```
            Ch := Ch + 15;
          For m := 1 To 12 Do
            Write(Prn,Chr(Ch));
        END;
      Print_Bar(Density);
      Writeln(Prn);
      Write(Prn,' ':5);
      Print_Tic(Density,3);
      Print_Bar(Density);
      Write(Prn,Esc,Density,N1,N2);
      For k := 1 To Width Do
        BEGIN
          Bit := 16;
          If (Odd(Bit_Map^[i,k] Div Bit)) Then
            Ch := 240
          Else Ch := 0;
          Bit := Bit + Bit;
          If (Odd(Bit_Map^[i,k] Div Bit)) Then
            Ch := Ch + 15;
          For m := 1 To 12 Do
            Write(Prn,Chr(Ch));
        END;
      Print_Bar(Density);
      Writeln(Prn);
      Write(Prn,' ':5);
      Print_Tic(Density,4);
      Print_Bar(Density);
      Write(Prn,Esc,Density,N1,N2);
      For k := 1 To Width Do
        BEGIN
          Bit := 64;
          If (Odd(Bit_Map^[i,k] Div Bit)) Then
            Ch := 240
          Else Ch := 0;
          Bit := Bit + Bit;
          If (Odd(Bit_Map^[i,k] Div Bit)) Then
            Ch := Ch + 15;
          For m := 1 To 12 Do
            Write(Prn,Chr(Ch));
        END;
      Print_Bar(Density);
      Writeln(Prn);
    END;
  Write(Prn, 0:5);
  Print_Bottom(Density);
  Writeln(Prn);
  If Start_Right Then
    If (Printer_Type = 2) Then
      Writeln(Prn,'START':(8 + Width * 2))
    Else Writeln(Prn,'START':(8 + Width + (8 * Width) Div 10))
  Else Writeln(Prn,'START':8);
  Writeln(Prn,Esc,'2',Chr(12));
END;

Procedure Graph ( Density : Dens_Type );

CONST
  Debug         =       False;

VAR
  n, m          :       Integer;
  n1, n2        :       Integer;
  mL, mO        :       Integer;
  i, j          :      Integer;
  Bit           :       Integer;
  Set_Bit       :       Integer;
  Test_Bit              :       Integer;
  Row                   :       Integer;
  Col                   :       Integer;
  Dist                  :       Integer;
  Edge                  :       Integer;
  Line                  :       Array [-4..600] Of Integer;

BEGIN
  If (Debug) Then
    Writeln(' Procedure Graph');
  If (Printer_Type = 1) Or (Printer_Type = 3) Then
    n := (512 Div Width)
  Else  n := (420 Div Width);   { Number of dots per bit horizontally }
  m := (n Div 3);               { Number of dots vertically }
  If (m = 0) Then
    m := 1;
  mL := m * Length;
  mO := m * Offset;
  Dist := 0;
  n := 3 * m;                   { Set for true 3 : 1 ratio }
  n1 := (Width * n + 6) Mod 256;
  n2 := (Width * n + 6) Div 256;
  If (Debug) Then
    Writeln('   Vert = ',m:4,'  Horz = ',n:4,' n1, n2 ',n1:3,n2:4);
  Line[0] := 255;
  Writeln(Prn,Esc,'3',Chr(24));
  Write(Prn,' ':6);
  Write(Prn,Esc,Density,Chr(n1),Chr(n2),Chr(0),Chr(0),Chr(0),Chr(0));
  For i := 1 To (Width * n + 2) Do
    Write(Prn,Chr(1));
  Writeln(Prn);
  Set_Bit := 128;
  For i := -4 To 480 Do
    Line[i] := 0;
  If ((Length Mod 4) = 0) Then
    BEGIN
      Line[-1] := 128;
      Line[-2] := 128;
    END;
  If ((Length Mod 8) = 0) Then
    BEGIN
      Line[-3] := 128;
      Dist := (Length Div 8) * 2;
      Line[-4] := 128;
    END;
  For Bit := 0 To (mL - 1) Do
    BEGIN
      Row := ((Bit + mO) Div m) Div 8 + 1;
      i := ((Bit + mO) Div m) Mod 8;
      Test_Bit := Power[i];
      If (((mL - Bit) Mod (4 * m)) = 0) Then
        BEGIN
          Line[-1] := Set_Bit;
          Line[-2] := Set_Bit;
        END;
      If (((mL - Bit) Mod (8 * m)) = 0) Then
        BEGIN
          Line[-3] := Set_Bit;
          Line[-4] := Set_Bit;
          Dist := (mL - Bit) Div (8 * m) * 2;
        END;
      For Col := 1 To Width Do
        If (Odd(Bit_Map^[Row,Col] Div Test_Bit)) Then
          Line[Col] := Line[Col] + Set_Bit;
      Set_Bit := Set_Bit Div 2;
      If (Set_Bit = 0) Then
```

```
        BEGIN
          If (Dist = 0) Then
            Write(Prn,' ':6)
          Else Write(Prn,Dist:6);
          Dist := 0;
          Write(Prn,Esc,Density,Chr(n1),Chr(n2));
          For i := -4 To 0 Do
            BEGIN
              Write(Prn,Chr(Line[i]));
              Line[i] := 0;
            END;
          For i := 1 To Width Do
           BEGIN
              For j := 1 To n Do
                Write(Prn,Chr(Line[i]));
              Line[i] := 0;
           END;
          Writeln(Prn,Chr(255));
          Line[0] := 255;
          Set_Bit := 128;
        END;
    END;
  For i := -4 To -1 Do
    Line[i] := 0;
  For i := -4 To Width Do
    Line[i] := Line [i] + Set_Bit;
  Edge := 255-(Set_Bit Div 2)-(Set_Bit Div 4)-(Set_Bit Div 8)-(Set_Bit Div 16)-(
Set_Bit Div 32)-(Set_Bit Div 64)-(Set_Bit Div 128);
  Line[0] := Edge;
  Write(Prn,'0':6);
  m := 2;
  Set_Bit := Set_Bit Div 2;
  If (Set_Bit = 0) Then
    BEGIN
      Dist := 0;
      Write(Prn,Esc,Density,Chr(n1),Chr(n2));
      For i := -4 To 0 Do
        BEGIN
          Write(Prn,Chr(Line[i]));
          Line[i] := 0;
        END;
      For i := 1 To Width Do
       BEGIN
          For j := 1 To n Do
            Write(Prn,Chr(Line[i]));
          Line[i] := 0;
       END;
      Writeln(Prn,Chr(255));
      Line[0] := 255;
      Set_Bit := 128;
      Write(Prn,' ':6);
    END;
  For Bit := 1 To 4 Do
    BEGIN
      For Col := 0 To (Width Div m) Do
        Line[Col * m] := Line[Col * m] + Set_Bit;
      If (Bit = 2) Then
        m := m + m;
      Set_Bit := Set_Bit Div 2;
      If (Set_Bit = 0) Then
        BEGIN
          Write(Prn,Esc,Density,Chr(n1),Chr(n2));
          For i := -4 To 0 Do
            BEGIN
              Write(Prn,Chr(Line[i]));
              Line[i] := 0;
      END;
     For i := 1 To Width Do
      BEGIN
       For j := 2 To n Do
        Write(Prn,Chr(Line[1]));
       Write(Prn,Chr(Line[i]));
       Line[i] := 0;
      END;
     Writeln(Prn,Chr(255));
     Line[0] := 255;
     Set_Bit := 128;
     Write(Prn,' ':6);
    END;
  END;
 If (Set_Bit <> 128) Then
  BEGIN
   Write(Prn,Esc,Density,Chr(n1),Chr(n2));
   For i := -4 To 0 Do
    Write(Prn,Chr(Line[i]));
   For i := 1 To Width Do
    BEGIN
     For j := 2 To n Do
      Write(Prn,Chr(Line[1]));
     Write(Prn,Chr(Line[i]));
    END;
   Writeln(Prn,Chr(0));
  END;
 Write(Prn,'0':7);
 For i := 1 To (Width Div 4) Do
  BEGIN
   Write(Prn,Esc,Density,Chr((4 * n - 14) Mod 256),Chr((4 * n - 14) Div 256))
;
   For j := 1 To (4 * n - 14) Do
    Write(Prn,Chr(0));
   Write(Prn,(i * 3):2);
  END;
 Writeln(Prn);
 If (Start_Right) Then
  Writeln(Prn,'START':76)
 Else Writeln(Prn,'START':9);
 Writeln(Prn,Esc,'2');
END;

Procedure Map;

BEGIN
 Header;
 If (Printer_Type = 1) Then
  Graph('N')
 Else If (Printer_Type = 2) Then
     Graph('K')
    Else If (Printer_Type = 3) Then
       Graph(Star)
      Else Dumb_Printer;
 Writeln(Prn,Chr(12));
END;

Procedure Init_Count ( Var Count : Count_Type );

CONST
 Debug       =      False;

VAR
 i,j         :      Integer;

BEGIN
```

```
 If (Debug) Then
  Writeln(' Procedure Init_Count');
 For i := 0 To 255 Do
  BEGIN
   If (Odd(i Div Power[0])) Then
    Count[i] := 1
   Else Count[i] := 0;
   If (Odd(i Div Power[1])) Then
    Count[i] := 1 + Count[i];
   If (Odd(i Div Power[2])) Then
    Count[i] := 1 + Count[i];
   If (Odd(i Div Power[3])) Then
    Count[i] := 1 + Count[i];
   If (Odd(i Div Power[4])) Then
    Count[i] := 1 + Count[i];
   If (Odd(i Div Power[5])) Then
    Count[i] := 1 + Count[i];
   If (Odd(i Div Power[6])) Then
    Count[i] := 1 + Count[i];
   If (Odd(i Div Power[7])) Then
    Count[i] := 1 + Count[i];
  END;
 If (Debug) Then
  BEGIN
   Write(Prn,'     ');
   For i := 0 To 15 Do
   Write(Prn,i:4);
   Writeln(Prn);
   Write(Prn,'    __');
   For i := 0 To 15 Do
   Write(Prn,'____');
   Writeln(Prn);
   For i := 0 To 15 Do
    BEGIN
     Write(Prn,i:4,' |');
     For j := 0 To 15 Do
      Write(Prn,Count[i * 16 + j]:4);
     Writeln(Prn,Chr(12));
    END;
  END;
END;

Procedure Map_Percent;

CONST
 Debug      =   False;

VAR
 Count      :   Count_Type;
 Totals     :   Array [0..64] Of Integer;
 Col_Off    :   Integer;
 Row_Off    :   Integer;
 No_Percent :   Integer;
 i,j        :   Integer;
 Index      :   Integer;

BEGIN
 If (Debug) Then
  Writeln('Procedure Map_Percent');
 If (Width > 40) AND (Printer_Type In [1,2]) Then       { Compressed }
  Write(Prn,Chr(15));                                   { Print Mode }
 Header;
 If  (Odd(Map_Length)) Then
  Row_Off := 1
 Else Row_Off := 0;
 If (Start_Right) Then
```

```
    Col_Off := ((Width - 1) Div 4 + 1) * 4 - Width
  Else Col_Off := 0;
  No_Percent := ((Width - 1) Div 4);
  Init_Count(Count);
  For i := 0 To 64 Do
    Totals[i] := 0;
  Writeln(Prn,'   FEET');
  Write(Prn,(((Map_Length - 1) Div 2 + 1) * 4):6,'--');
  For i := 0 To No_Percent Do
    Write(Prn,'-------');
  Writeln(Prn);
  For i := 1 To Map_Length Do
    BEGIN
      For j := 0 To (Width - 1) Do
        BEGIN
          Index := Bit_Map^[i,j + 1];
          If (Index < 0) Then
            Index := Index + 256;
          Totals[(j + Col_Off) Div 4] := Totals[(j + Col_Off) Div 4] + Count[Ind
ex];
        END;
    If (Not Odd(i + Row_Off)) Then
      BEGIN
        Write(Prn'¦':8);
        For j := 0 To No_Percent Do
         Write(Prn,'¦':7);
        Writeln(Prn);
        Write(Prn,'¦':8);
        For j := 0 To No_Percent Do
          Write(Prn,(Totals[j] / 0.64):5:1,' ¦');
        Writeln(Prn);
        Write(Prn'¦':8);
        For j := 0 To No_Percent Do
         Write(Prn,'¦':7);
        Writeln(Prn);
        Write(Prn,(((Map_Length - i) Div 2) * 4):6,'--');
        For j := 0 To No_Percent Do
          Write(Prn,'-------');
        Writeln(Prn);
        If (Debug) Then
          BEGIN
            For j := 0 To No_Percent Do
              Write(Totals[j]:5);
            Writeln;
          END;
        For j := 0 To No_Percent Do
          Totals[j] := 0;
      END;
    END;
  Write(Prn,'¦':8);
  For i := 0 To No_Percent Do
    Write(Prn,'¦':7);
  Writeln(Prn);
  Write(Prn,'0':8);
  For i := 1 To No_Percent + 1 Do
    Write(Prn,(i * 3):7);
  Writeln(Prn);
  If (Start_Right) Then
    Writeln(Prn,'START':(18 + 7 * No_Percent))
  Else Writeln(Prn,'START':11);
  If (Width > 40) AND (Printer_Type In [1,2]) Then      { Compressed }
    Write(Prn,Chr(18));                                 { Print Mode }
  Writeln(Prn,Chr(12));         { Form Feed }           {     Off    }
END;

Procedure Display_Options;
```

```
CONST
  Debug            =       False;
  Valid            =       ['1','2','p','P','t','T','s','S','e','E','a','A'];

VAR
  In_Ch            :              Integer;
  Ch               :       Char;
  Quit             :              Boolean;
  String           :       LString(80);

BEGIN
  If Debug Then Writeln(' Display_Options');
  Quit := False;
  REPEAT
    Clear_Screen;
    Home;
    Writeln;
    Writeln('                IOWA D.O.T.');
    Writeln;
    Writeln('            DELAMINATION MAPS');
    Writeln('                1 : 8 Inches Wide');
    Writeln('                2 : 4 Dots Per 3 Inches');
    Writeln;
    Writeln('            FUNCTIONS');
    Writeln('                P : Percentages');
    Write('                  ');
    String := 'T : TI 855 Printer';
    If (Printer_Type = 1) Then
      Reverse(String)
    Else Write(String);
    Writeln;
    Write('                  ');
    String := 'S : Star Printer';
    If (Printer_Type = 3) Then
      Reverse(String)
    Else Write(String);
    Writeln;
    Write('                  ');
    String := 'E : Epson Printer';
    If (Printer_Type = 2) Then
      Reverse(String)
    Else Write(String);
    Writeln;
    Write('                  ');
    String := 'A : Alphanumeric Printer';
    If Not (Printer_Type In [1,2,3]) Then
      Reverse(String)
    Else Write(String);
    Writeln;
    Writeln('                Q : Quit');
    REPEAT
      In_Ch := Dosxqq(6,255);
    UNTIL (In_Ch <> 0);
    Ch := Chr(In_Ch);
    Writeln;
    Writeln;
    If (Ch In Valid) Then
      Write('    Working...');
    Cursor_Off;
    Case Ch Of
      '1'     : Map;
      '2'     : If (Printer_Type = 1) Then
                  If (Width < 45) Then
                    Scale_4_To_3('N')
                  Else Scale_4_To_3('O')
```

```
                Else If (Printer_Type = 2) Then
                       If (Width < 35) Then
                         Scale_4_To_3('K')
                       Else Scale_4_To_3('L')
                     Else If (Printer_Type = 3) Then
                            Scale_4_To_3(Star)
                          Else Dumb_Printer;
      'p','P' : Map_Percent;
      't','T' : BEGIN
                  Init_Printer := True;
                  Printer_Type := 1;
                END;
      'e','E' : BEGIN
                  Init_Printer := True;
                  Printer_Type := 2;
                END;
      's','S' : BEGIN
                  Init_Printer := True;
                  Printer_Type := 3;
                END;
      'a','A' : BEGIN
                  Init_Printer := False;
                  Printer_Type := -1;
                END;
      'Q','q' : Quit := True;
     Otherwise { };
     END;
  UNTIL (Quit);
  Cursor_On;
END;

BEGIN
  Star[1] := '*';
  Star[2] := Chr(5);
  Star[0] := Chr(2);
  Power[0] := 1;
  Power[1] := 2;
  Power[2] := 4;
  Power[3] := 8;
  Power[4] := 16;
  Power[5] := 32;
  Power[6] := 64;
  Power[7] := 128;
  Assign(Prn,'PRN');
  Rewrite(Prn);
  Printer_Type := Def_Printer;
  Init_Printer := True;
  REPEAT
    Get_Information;
    Process_Section;
    If Ok Then Display_Options;
    Writeln;
    Write(' Continue With Another Section (<cr> = No) ? ');
    Dispose(Bit_Map);
    Close(Info);
    Readln(Bridge_ID);
  UNTIL (Ord(Bridge_ID[0]) = 0) Or (Bridge_ID[1] In ['n','N']);
  If (Printer_Type = 1) Or (Printer_Type = 2) Then
    Write(Prn,Esc,'@');
  Close(Prn);
END.
```

```
Module Utilities;

Function Get_Int ( Limit : Integer ) : Integer;

CONST
  Digits = ['1','2','3','4','5','6','7','8','9','0'];

VAR
  i      : Integer4;
  j      : Integer;
  Minus  : Boolean;
  Ch     : Char;

BEGIN
  i := 0;
  j := 0;
  Minus := False;
  If Not Eoln Then Read(ch) Else Ch := '?';
  While (Not (Ch In Digits)) And (Ch <> '-') And (Not Eoln) Do
    Read(ch);
  If (Not Eoln) And (Ch = '-') Then
    BEGIN
      Minus := True;
      Read(Ch);
    END;
  If (Not (Ch In Digits)) And (Not Eoln) Then
    BEGIN
      Minus := False;
      i := Get_Int(Limit);
    END
  Else While (Ch In Digits) And (j < Limit) Do
         BEGIN
           i := i * 10 + Ord(Ch) - Ord('0');
           j := j + 1;
           If Eoln Then j := Limit
           Else Read(ch);
         END;
  If Minus Then i := -i;
  If (i <= MaxInt) And (i >= -MaxInt) Then
    Get_Int := Retype(Integer,i)
  Else BEGIN
         Writeln(' Integer Quantity Overflow; Value Set To ZERO');
         Writeln('  Value Must Be Between',MaxInt:7,' And',-MaxInt:8);
         Get_Int := 0;
       END;
  Readln;
END;

END.
```

A.7 SAMPLE PLOTS

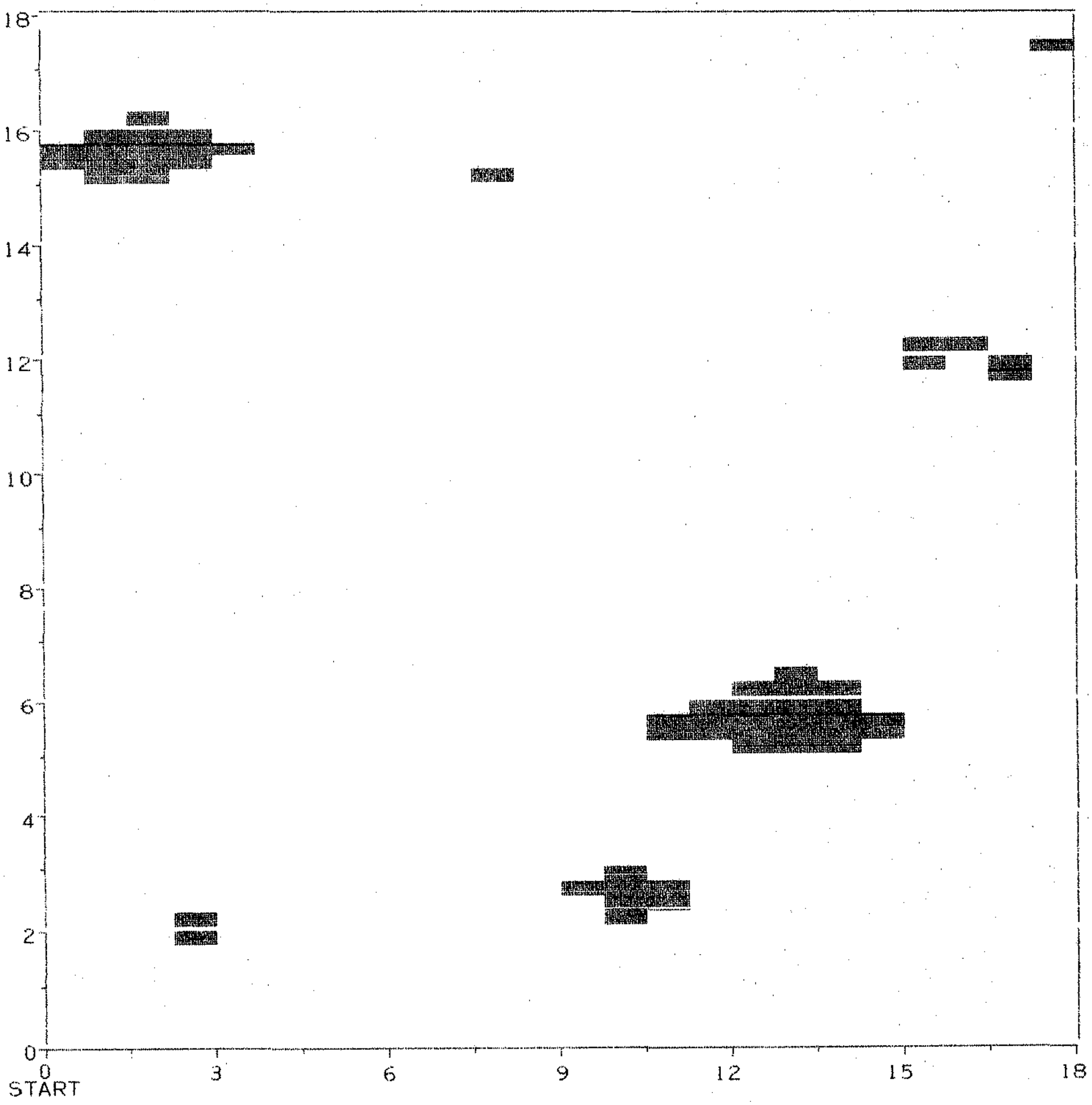
IOWA D.O.T.
DELAMINATION MAP

D.O.T. Example Plot

July 1 1984 Delamination at 400 mV

Length = 18 Feet 0 Inches Width = 18 Feet 0 Inches

Total Area = 324.00 Square Feet Percentage Bad = 3.13%



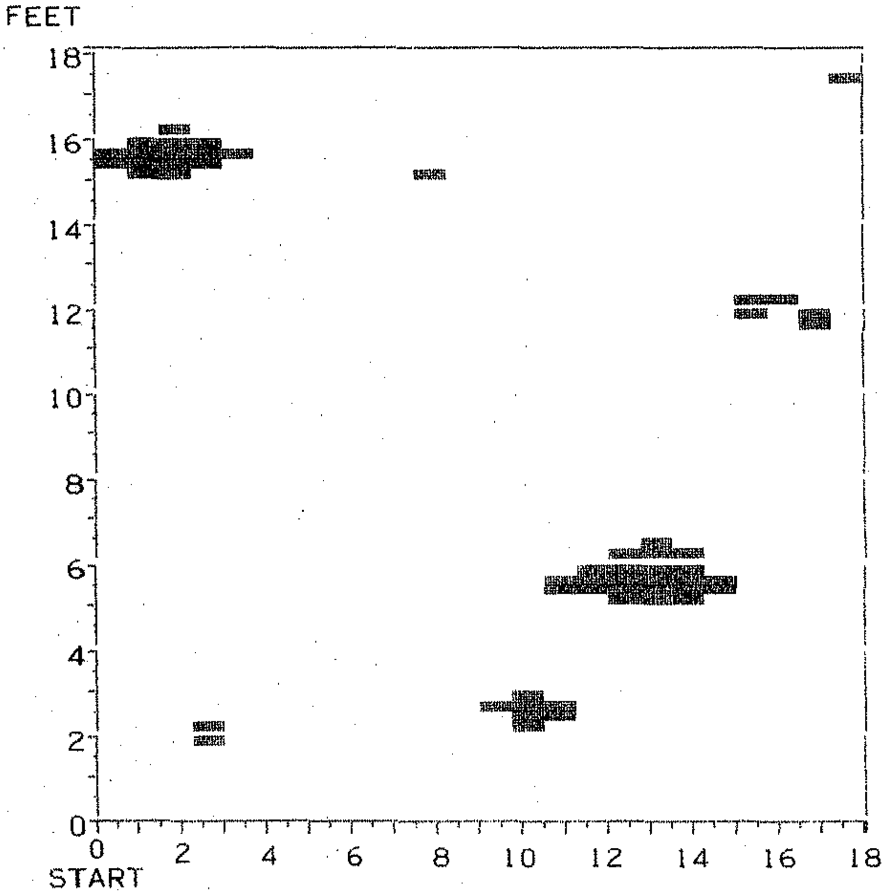
IOWA D.O.T.
DELAMINATION MAP

D.O.T. Example Plot

July 1 1984 Delamination at 400 mV

Length = 18 Feet 0 Inches Width = 18 Feet 0 Inches

Total Area = 324.00 Square Feet Percentage Bad = 3.13%



IOWA D.O.T.
DELAMINATION MAP

D.O.T. Example Plot

July 1 1984   Delamination at   400 mV

Length = 18 Feet 0 Inches   Width = 18 Feet 0 Inches

Total Area = 324.00 Square Feet   Percentage Bad = 3.13%

```
FEET
20-----------------------------------------------------
   |       |       |       |       |       |       |
   |  1.6  |  0.0  |  0.0  |  0.0  |  0.0  |  1.6  |
   |       |       |       |       |       |       |
16-----------------------------------------------------
   |       |       |       |       |       |       |
   | 20.3  |  1.6  |  1.6  |  0.0  |  0.0  |  3.1  |
   |       |       |       |       |       |       |
12-----------------------------------------------------
   |       |       |       |       |       |       |
   |  0.0  |  0.0  |  0.0  |  0.0  |  0.0  |  4.7  |
   |       |       |       |       |       |       |
 8-----------------------------------------------------
   |       |       |       |       |       |       |
   |  0.0  |  0.0  |  0.0  |  7.8  | 28.1  |  0.0  |
   |       |       |       |       |       |       |
 4-----------------------------------------------------
   |       |       |       |       |       |       |
   |  3.1  |  0.0  |  0.0  | 10.9  |  0.0  |  0.0  |
   |       |       |       |       |       |       |
 0-----------------------------------------------------
   |       |       |       |       |       |       |
   0       3       6       9      12      15      18
  START
```

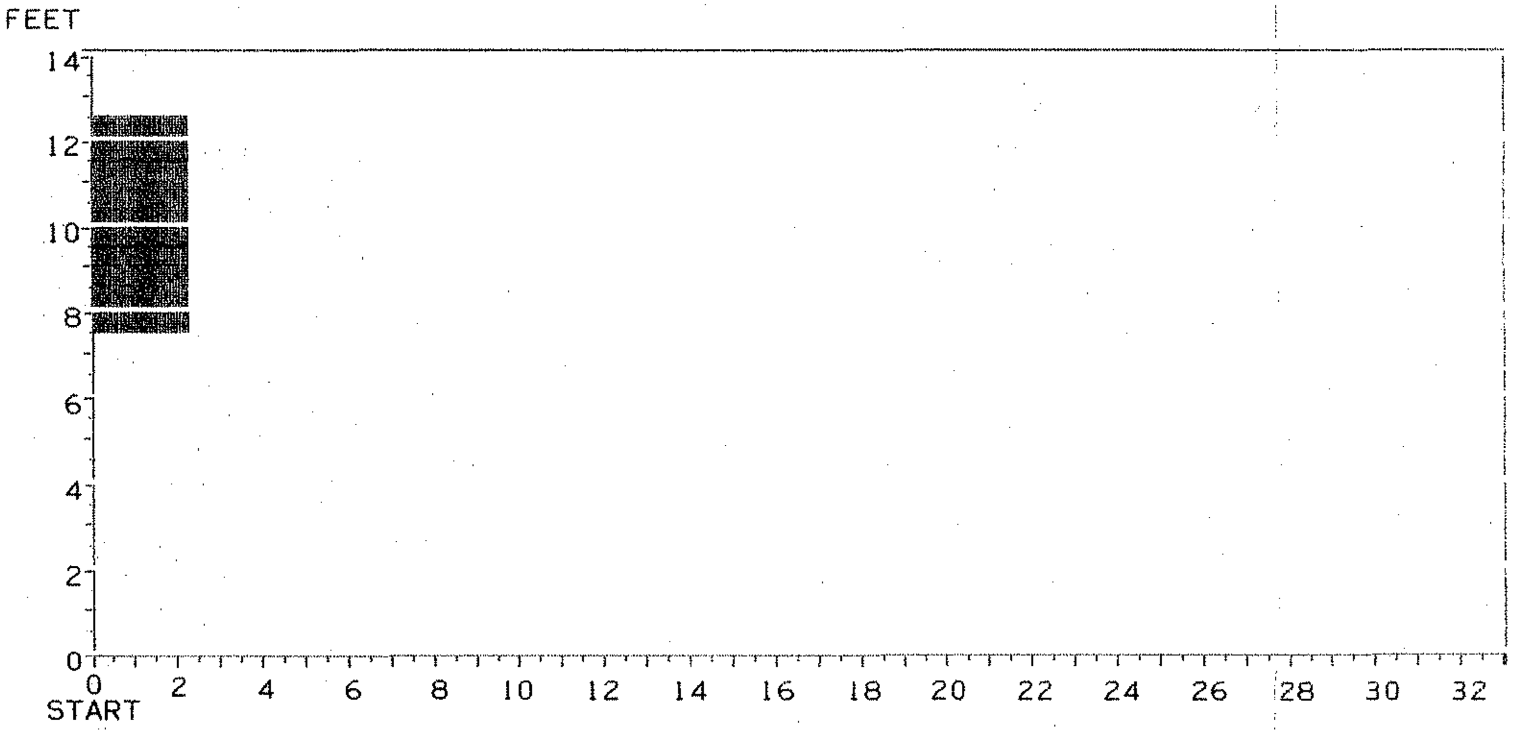
IOWA D.O.T.
DELAMINATION MAP

Very Wide Bridge

6 - 20 -85 Delamination at 400 mV

Length = 12 Feet 6 Inches Width = 33 Feet 0 Inches

Total Area = 412.50 Square Feet Percentage Bad = 2.73%



```
                         IOWA D.O.T.
                       DELAMINATION MAP

                     Very Wide Bridge

    6 - 20 -85          Delamination at   400 mV

  Length = 12 Feet  6 Inches   Width = 33 Feet  0 Inches

  Total Area =  412.50 Square Feet   Percentage Bad = 2.73%

FEET
16-----------------------------------------------------------------------
   |      |      |      |      |      |      |      |      |      |      |      |
   | 9.4  | 0.0  | 0.0  | 0.0  | 0.0  | 0.0  | 0.0  | 0.0  | 0.0  | 0.0  | 0.0  |
   |      |      |      |      |      |      |      |      |      |      |      |
12-----------------------------------------------------------------------
   |      |      |      |      |      |      |      |      |      |      |      |
   | 75.0 | 0.0  | 0.0  | 0.0  | 0.0  | 0.0  | 0.0  | 0.0  | 0.0  | 0.0  | 0.0  |
   |      |      |      |      |      |      |      |      |      |      |      |
 8-----------------------------------------------------------------------
   |      |      |      |      |      |      |      |      |      |      |      |
   | 9.4  | 0.0  | 0.0  | 0.0  | 0.0  | 0.0  | 0.0  | 0.0  | 0.0  | 0.0  | 0.0  |
   |      |      |      |      |      |      |      |      |      |      |      |
 4-----------------------------------------------------------------------
   |      |      |      |      |      |      |      |      |      |      |      |
   | 0.0  | 0.0  | 0.0  | 0.0  | 0.0  | 0.0  | 0.0  | 0.0  | 0.0  | 0.0  | 0.0  |
   |      |      |      |      |      |      |      |      |      |      |      |
 0-----------------------------------------------------------------------
   |      |      |      |      |      |      |      |      |      |      |      |
   0      3      6      9      12     15     18     21     24     27     30     33
   START
```