9950-976

DEPARTMENT OF MECHANICAL ENGINEERING AND MECHANICS
SCHOOL OF ENGINEERING
OLD DOMINION UNIVERSITY
NORFOLK, VIRGINIA

THE DEVELOPMENT OF A MICROPROCESSOR-CONTROLLED LINEARLY-ACTUATED VALVE ASSEMBLY

By

Raymond H. Wall

Robert L. Ash, Principal Investigator

Supplemental Technical Report

Prepared for the
Jet Propulsion Laboratory
California Institute of Technology
Pasadena, California 91103

Under
Contract No. 955731
Warren L. Dowler, Technical Manager

May 1984

OLD DOMINION UNIVERSITY RESEARCH FOUNDATION

DEPARTMENT OF MECHANICAL ENGINEERING AND MECHANICS
SCHOOL OF ENGINEERING
OLD DOMINION UNIVERSITY
NORFOLK, VIRGINIA

THE DEVELOPMENT OF A MICROPROCESSOR-CONTROLLED LINEARLY ACTUATED VALVE ASSEMBLY

By

Raymond H. Wall

Robert L. Ash, Principal Investigator

Supplemental Technical Report

Prepared for the
Jet Propulsion Laboratory
California Institute of Technology
Pasadena, California 91103

Under
Contract No. 955731
Warren L. Dowler, Technical Manager

Submitted by the
Old Dominion University Research Foundation
P.O. Box 6369
Norfolk, Virginia 23508

May 1984

This work was prepared for the Jet Propulsion Laboratory, California Institute of Technology, under a contract with the National Aeronautics and Space Administration.

THE DEVELOPMENT OF A MICROPROCESSOR-CONTROLLED LINEARLY-ACTUATED VALVE ASSEMBLY

By

Raymond H. Wall

Under the direction of Dr. Robert L. Ash, Principal Investigator and
Dr. Ping Tcheng, NASA/Langley Research Center, Hampton, VA

INTRODUCTION

The major part of this report is comprised of a master's thesis (see Appendix), written by Mr. Raymond H. Wall, which investigates the feasibility of a direct, digitally-controlled valve for space system operation. The thesis is relatively self-explanatory. It is important to note, however, that due to severe cost limitations, many of the technology issues which could be solved by modest levels of funding, could not be solved at the level of funding available for this work.

In addition to the thesis, we would like to submit the discussion that follows, concerning technology issues in space. We would also recommend, in addition to the documentation provided with this report, the inclusion of the following three reference documents for our report:

Item 1: Document 74A39119 entitled "Pulse-Modulated Dual-Gas Control Subsystem for Space Cabin Atmosphere," by J.K. Jackson. Summarizes research on a pulse modulated valve for space system application.

Item 2: Report No. 75N22744 entitled "Final Report: Line Fluid Activated Valve Development Program," by R. A. Lynch of the Marquardt Corporation, Van Nuys, California.

Item 3: Document No. 75N24839 entitled "Space Shuttle Main Engine Definition (Phase B) Volume 5: Valves and Interconnects -- for Space Shuttle." This is a Pratt and Whitney Report.

These documents, which are available in the library of the California Institute of Technology, should be included as a package of material addressing technology issues for continuously controllable valve systems.

OBJECTIVES

One of the objectives of this program was to determine the feasibility of avoiding A to D conversion elements in this valve system. Based on the work that followed, it is believed that the control aspects can be accomplished directly, but the pressure measuring aspects of the system remain driven by analog systems with digital conversion.

One problem which will continue to be an issue is absolute position location using digital encoder systems. A recent development called a "gray band encoder" provides an absolute position location rather than a digital output (driven typically by counting black and white bands) and appears to be the preferred mode. Gray band encoders are presently being used by the Army in some of the helicopter development work at Ft. Eustis. While the gray band encoder is not a classified device, apparently the application is classified, and we were not able to determine the precise specifications for gray band position encoders. Digital displacement measurement devices are an essential technology for any system of this type. A digital displacement measuring device can be used both for pressure measurements as well as valve stem position measurements. These devices have not yet been developed at a level which would be acceptable for autonomous operation in space. State of the art linear encoder devices appear to be too large and too sensitive to drift in absolute position due to vibration.

OPINIONS AND CONCLUSIONS

It was clearly the consensus of the three investigators in this project that bellows valves are the preferred valve for space applications. These valves must be custom designed for each application because of the need to match both the bellows spring constant and the valve stem travel to the particular displacement system and flow application. Longer valve stem travel is needed for high flow rate applications than presently is available on commercial valve systems. The actual valve geometry must be a major consideration in the space system design. The geometry affects performance, reliability, repeatability, and survivability. In addition, proper valve design may enable a relatively high level of self calibration capability.

Continuing calibration of the valve/flow control system was identified as an essential element in a long duration valve operation. Presently there are no easy calibration procedures which could be used either continuously, or intermittently in conjunction with the valve and associated differential pressure measuring systems.

The consensus of this group was that the pressure measurements across an orifice plate which was not maintained directly in the flow stream was a viable means of calibrating intermittently the flow control system. The idea would be to have a bypass system whereby the flow could be bypassed through an orifice plate with the corresponding pressure measurements being made for short periods of time. In that way, neither corrosion nor contamination would occur at the same rate as it would occur in the continuously operated stream. By using the calibration loop intermittently, it is possible to maintain an updated calibration for the valve system.

At the conclusion of the thesis, it became apparent that the next step beyond this indirect control approach would be to develop a calibration

relation by appropriate choice of parameters which collapsed the data onto a single curve. That is, the different parametric calibration curves presented in the thesis could be collapsed onto a single curve (to some reasonable level of approximation) and greatly reduce the computation time required in the software. That approach would be feasible if variations of up to 5% in flow are tolerated by the system.

It was determined further that the flow rate measurement, or at least the flow rate calibration, should be separated from direct operation of the valve. Problems with long term degradation of valve geometry and valve performance cannot be identified if both are degrading simultaneously.

SUMMARY

In summary, no technology issues were found which prevent the development of digital valve systems for space applications. The single, most limiting technology, however, appears to be a direct digital displacement measuring system which is not affected seriously by vibration. The valve system developed in this thesis was found to work autonomously and satisfactorily. It is hoped that the information developed in this work, including the micro computer programs, will be useful at the start of a more ambitious valve research program in the future.

APPENDIX

THE DEVELOPMENT OF A MICROPROCESSOR CONTROLLED LINEARLY ACTUATED VALVE ASSEMBLY

by

Raymond H. Wall
B.S. (Mechanical Engineering)
August 1981, Old Dominion University

A Thesis Submitted to the Faculty of
Old Dominion University in Partial Fulfillment of
the Requirements for the Degree of

MASTER OF ENGINEERING

MECHANICAL ENGINEERING

OLD DOMINION UNIVERSITY
May 1984

Approved by:

_____
Ping Tcheng (Director)

_____
Robert L. Ash

_____

ABSTRACT

THE DEVELOPMENT OF A MICROPROCESSOR CONTROLLED LINEARLY ACTUATED VALVE ASSEMBLY

Raymond H. Wall
Old Dominion University, 1984
Director: Dr. Ping Tcheng with
Dr. Robert L. Ash

The development of a proportional fluid control valve/assembly is presented. This electro-mechanical system is needed for space applications to replace the current proportional flow controllers that use bulky hydraulic or pneumatic actuation systems. The flow is controlled by a microprocessor system that monitors the control parameters of upstream pressure and requested volumetric flow rate. The microprocessor achieves the proper valve stem displacement by means of a digital linear actuator. A linear displacement sensor is used to measure the valve stem position. This displacement is monitored by the microprocessor system as a feedback signal to close the control loop. With an upstream pressure between 15-47 psig, the developed system operates between 779 standard cm^3/sec (SCCS) and 1543 SCCS. The delivered volumetric flow rates were within 5% of the requested values. Recommendations made for future work on the design include further modification to the bellows spring and the possibility of a mass flow rate controller.

ACKNOWLEDGEMENT

The author wishes to express his appreciation to his advisors, Dr. Ping Tcheng of NASA Langley Research Center and Dr. R. L. Ash at Old Dominion University, for their advice and assistance in this investigation. Also, thanks to Dr. A. Sidney Roberts, Jr. for his guidance and support through the Aeronautics Program. A special thanks is expressed to Mrs. Diane Pizzeck for her help, suggestions, and the typing of this manuscript. The author offers his gratitude to his family and Ms. Robin Wilkinson for their continuing support during this development.

This work was supported by NASA Langley Research Center through Graduate Engineering Research Participation in Aeronautics, Grant NGR-47-003-052.

TABLE OF CONTENTS

Page

LIST OF TABLES

LIST OF FIGURES

Figure — Page

LIST OF SYMBOLS

E_s	incremental optical linear encoder output
L_s	linear variable differential transformer output
P_u	upstream pressure
P_a	atmospheric pressure
Δp	differential pressure across regulating valve
Q_o	output volumetric flow rate
Q_r	requested volumetric flow rate
x_s	valve stem displacement (steps)

Chapter 1

INTRODUCTION

The control of fluid flow through a valve can be accomplished by either on-off or proportional control techniques. A survey of commercially available equipment for either method reveals limitations in the use of such technology designed for space systems. Examples of on-off type flow controllers would be solenoid valves or explosively actuated valves. Proportional flow control valves would typically be gate valves.

One limitation of an on-off controller is the inability to vary flow rate through the valve. Assuming a constant differential pressure, the fluid must be diverted through a second valve of different orifice size to change the flow rate. This requires an array of solenoid valves for a modest number of flow rate changes. This is prohibitive due to the mass and size restraints on a space system. Proportional flow controllers using gate or globe valves actuated by hydraulic systems, are also unacceptable because of the mass and size restraints on a space system. Additionally, the potential for leaks in hydraulic systems would eliminate their use in space.

A proportional pulse-modulated flow controller was reported by Jackson [1]*. The system achieved proportional control by pulsing solenoid valves at specific rates to maintain constant concentrations of nitrogen and oxygen in a Space Shuttle Orbiter cabin.

* The numbers in brackets indicate references.

While Jackson's results showed it was possible to approximate portional flow control by discrete-time operation of the solenoid, such a device is considered unreliable on long term space missions. That is, the cycling of the valve for extended times would eventually wear down the valve seat and thereby change the flow rate characteristics.

Another, commercially available, device is a digital flow controller developed by the Porter Instrument Company. Their device uses a spring and a diaphragm. Assuming a constant upstream pressure, a change in downstream pressure is compensated by a change in the force the spring exerts on the diaphragm. Thus, changes in the differential pressure across the controller are compensated and a constant flow rate is maintained. However, the device is designed for operation in a controlled laboratory environment and could not function in a space system where temperature and pressure variations can be extreme.

The purpose of this thesis is to describe the development of a prototype electro-mechanical proportional flow control system for use in a space environment. A microprocessor was used in the system to monitor system parameters and digitally control the valve stem displacement. Currently, to the author's knowledge, there exists no similar system or design, even though the necessary technology has existed for several years. The equipment used in this development is described in Chap. 2. A description of the development is presented in Chap. 3. The results of the system final testing are presented and discussed in Chap. 4. Conclusions and recommendations on those areas of the development that need further work are in Chap. 5.

Chapter 2

DESCRIPTION OF EQUIPMENT

The control of volumetric flow rate through a regulating valve can be accomplished by: (1) An accurate and reliable means to move the valve stem; and (2) a method to measure differential pressure across the valve. The equipment listed below was used to accomplish flow control and has been divided into two catagories: (1) Items which were preselected (by others) as the basic components for the preliminary investigation of the problem (designated by an asterisk); and (2) items which were added to the preliminary system during its development, which will be described by this thesis. A summary of the equipment used can be found in Table 1. A more detailed description of each component can be found in the following pages.

1. *Bellows Sealed Regulating Valve, NUPRO Model SS-4BRG. This type of valve (bellows sealed) was chosen because: (1) it eliminates the possibility of fluid leaks which can not be tolerated by space systems; (2) it eliminates the need for wad packing and the accompanying need to tighten the packing nut periodically; and finally, (3) because the bellows design readily adapts to the modifications which convert the valve movement from rotary to translational. (See Fig. 1 for a cross section view of the modified valve).
2. *Digital Linear Actuator, AIRPAX Model L92421-P2. After a survey of the commercially available linear movement

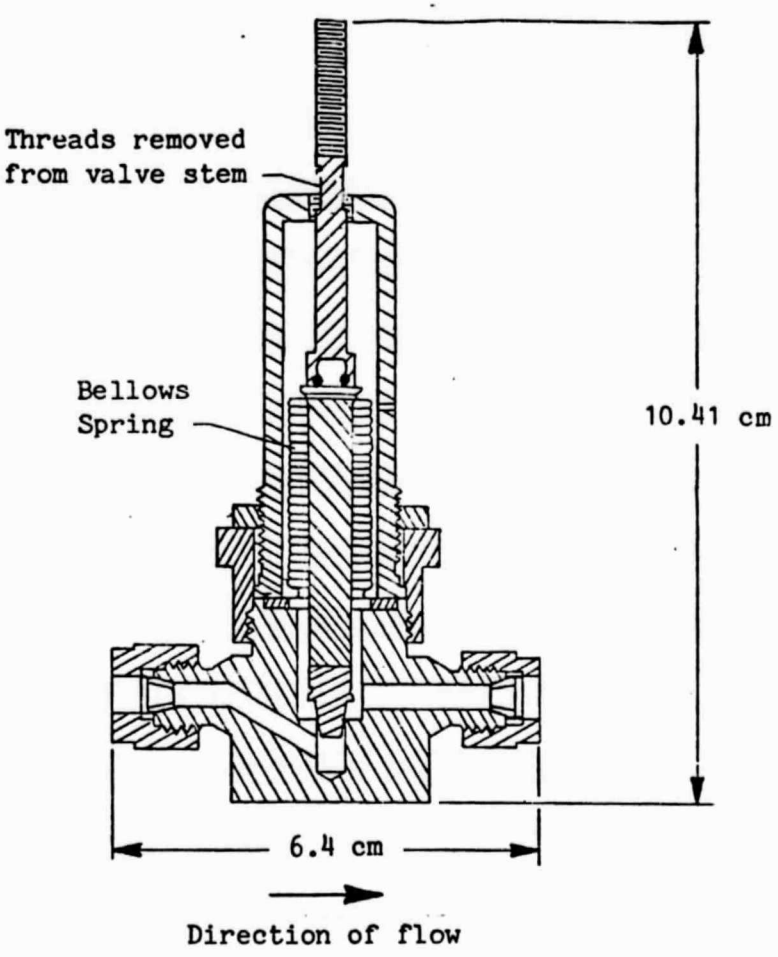

Fig. 1 Cross section of modified bellows sealed regulating valve.

actuators [2], the AIRPAX model 92421-P2 was selected on the basis of size and the magnitude of the linear force it could generate. The actuator is a modified stepping motor that has had its rotary motion converted into a linear force. The linear actuator has a resolution of 0.0051 cm/ step and a maximum linear force of 84.5 Newtons (N) at the slowest stepping rate.

3. *COSMAC Microprocessor Evaluation System, RCA CDP18S025. The RCA 1802 microprocessor was chosen to simulate the controlling computer in a spacecraft for several reasons: (1) The microprocessor evaluation system is built entirely of Complementary Metal-Oxide Silicon (CMOS) logic circuits, which means it uses considerably less power than the Transistor-Transistor Logic (TTL) circuits [3]; (2) the 1802 microprocessor is space qualified; and (3) the 1802 microprocessor is designed to be a process control device; it is not designed to be a support system for a personal computer like many other commercially available microprocessors. The Evaluation System has an eight bit processor with a 4K byte Random Access Memory (RAM) and 16 storage registers [4]. The system used in this development was modified to give it a battery backup RAM, in case of power failure, to prevent the loss of any programs stored in the memory. The RCA CDP18S021 COSMAC Microterminal is a fully assembled, compact, hand-held terminal designed for standard control, communications, and debugging functions

with CDP1802 microprocessor systems.

4. Barocel Electronic Manometer, DATAMETRICS Model 1174. The manometer measures the differential pressure across the valve and outputs a digital display of pressure (in psig) and an output in DC voltage between zero and one volt full scale.
5. Linear Variable Differential Transformer (LVDT), G. L. COLLINS CORP. Model LMT SS-207, S/N5240. By knowing the proportional relationship between flow coefficient and the displacement of the valve stem, the voltage readings from the LVDT can be used as one of the parameters necessary to control the volumetric flow rate through the valve. This was later replaced by the incremental optical linear encoder to eliminate the need for an analog to digital converter (ADC).
6. Turbometer, COX Model GM 16. This was used initially to measure volumetric flow rate through the valve but was later found to be inaccurate and was not used in the development. However, because it did not degrade the performance of the system, it was not removed from the assembly.
7. Rotameter, BROOKS CO. Model R9M-25-1. The rotameter was put in the system downstream as a means to check the accuracy of the turbometer. After the turbometer was found to be inaccurate, the rotameter was used as the only volumetric flow rate measurement device.
8. Globe Valve. Initially it was assumed that the differential pressure variation should be the control parameter in the development and so the valve was added to the downstream

assembly (see Fig. 2 for a photograph of the downstream assembly). While differential pressure is the correct control parameter, upstream pressure was substituted due to the lack of resolution in the Manometer Interface Circuit. Measurements were made with the downstream pressure tap of the manometer open to atmospheric conditions. Consequently, the valve was not necessary to the development but was left in the downstream assembly for the same reason as the turbometer.

9. Incremental Optical Linear Encoder (IOLE), DYNAMICS RESEARCH CORP. Model SST-E-4-1-0. Since the microprocessor is not suited to read the analog signal output by the LVDT, it was necessary to either build an ADC or to replace the LVDT with a digital linear measurement device. The linear encoder has a resolution of 1.016×10^{-3} cm/cycle and has a square wave quadrature output (90° out of phase) that is TTL compatible. An external circuit was necessary to read the output waveforms. The circuit diagram is shown in Fig. A.1, Appendix A, entitled IOLE Interface Circuit.

10. Linear Actuator Driver Board, AIRPAX Model K33505. This modified printed circuit board is the main component in the Interface Box (see Fig. A.2(a), Appendix A) between the microprocessor system and the linear actuator. It takes the clock signal generated from the microprocessor and then processes the signals necessary to drive the linear actuator. A more complete description of the process is

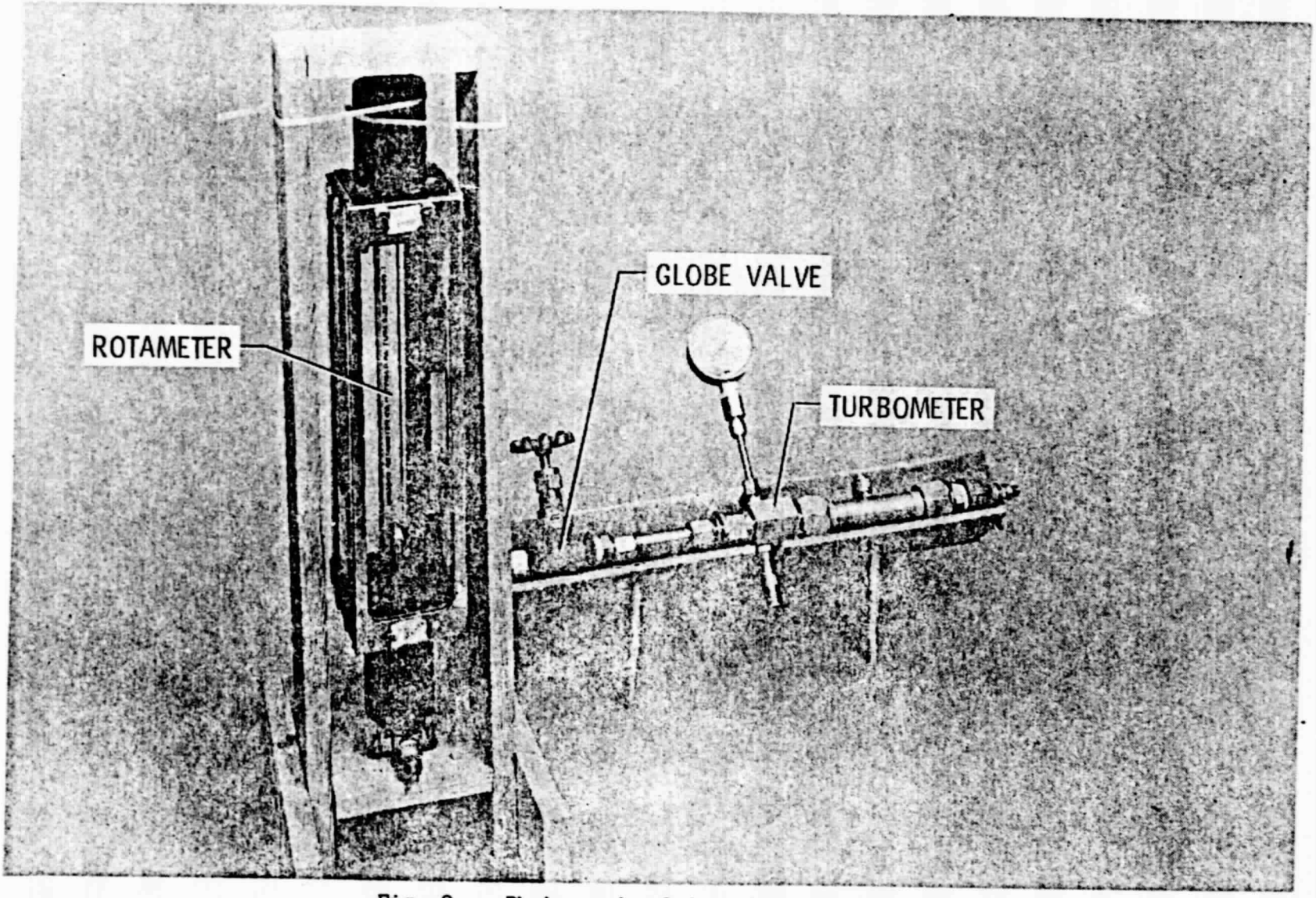

Fig. 2 Photograph of downstream assembly.

given in sec. 3.2.1.

11. Bridgesensor, CALEX Model 166. Contained within the Model 166 Bridgesensor is a voltage to frequency (V/F) converter. By using this isolated component of the bridgesensor, in conjunction with the Manometer Interface Circuit shown in Fig. A.3, Appendix A, it was possible to convert the analog signal output from the manometer into a digital signal usable by the microprocessor.

12. Pressure Regulator, PARKER-HANNIFIN Model R2037/J5. During testing, a range of upstream pressures were desired. This required the installation of the regulator. The regulator has a 0-689.5 kilo Pascals (kPa) gage range.

13. Diverting Valve, CLIPPARD Model MVD-4J. When a signal is sent from the microprocessor system to the attached solenoids (see equipment item No. 14), the spool of the diverting valve changed the path of air flow from the exit port leading to the regulating valve, to the plugged exit port. This allowed the tubing upstream of the regulating valve to depressurize. (See Fig. 3 for a photograph of the upstream assembly).

14. Solenoids (2), CLIPPARD Model AVS-12. These "energize-to-actuate" solenoids were used to move the diverting valve's spool. By using two solenoids, on either side of the diverting valve, a hybrid "latching" solenoid was formed. This reduced power consumption.

15. Data Acquisition System (DAS), CYBORG Isaac Model 91A. The Isaac DAS was an interface device between the external

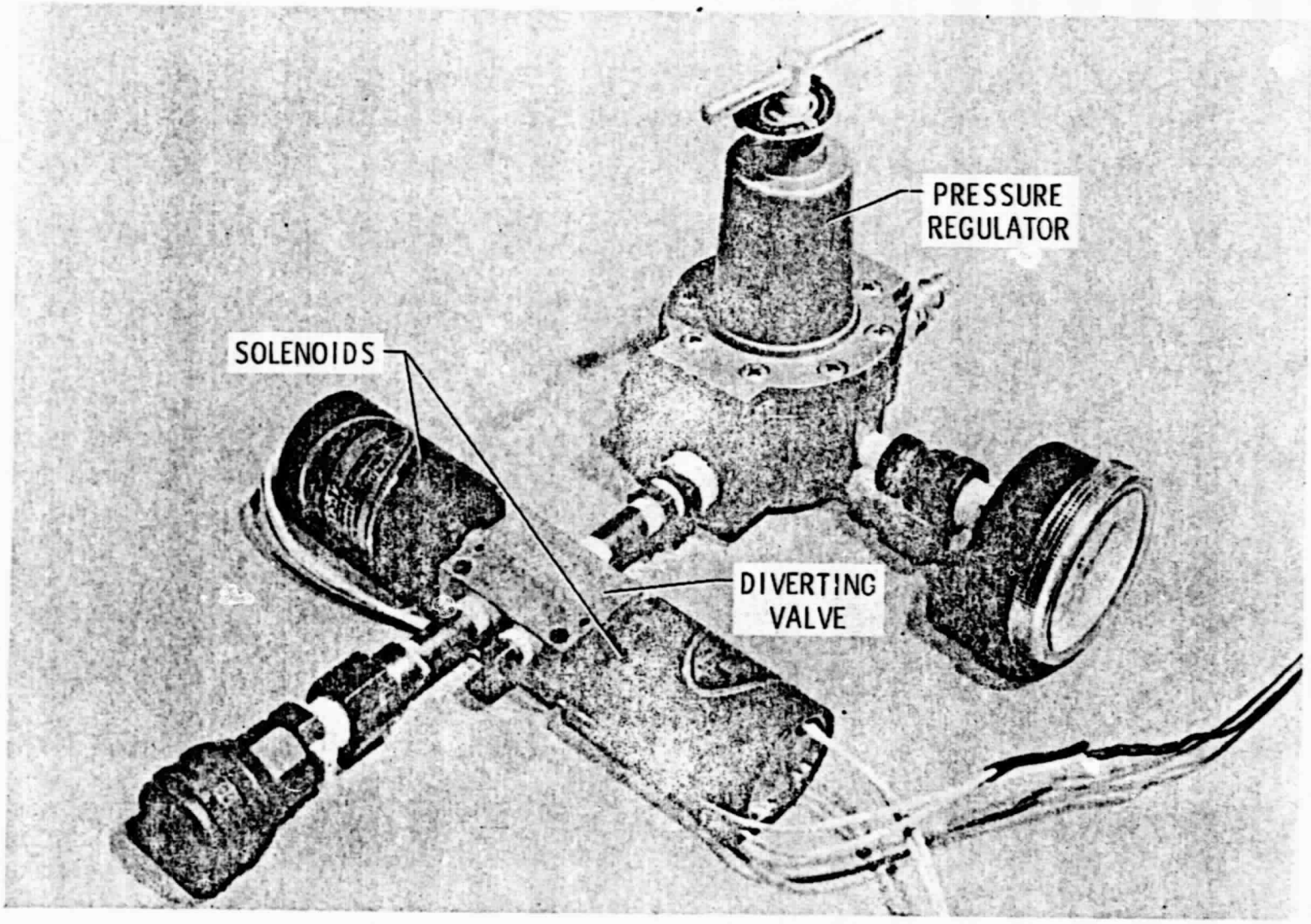

Fig. 3 Photograph of upstream assembly.

sensors and an APPLE IIe microcomputer. During the final testing of the developed assembly, the performance of the system was continuously monitored by the DAS. The values of upstream pressure and valve stem displacement were fed into the microcomputer through the Isaac's 16 bit binary input port. Any changes in either of these variables were recorded on hard copy.

Table 1 Equipment used in development

Item No.	Equipment, Manufacturer and Model No.
1.	Bellows Sealed Regulating Valve, NUPRO Model SS-4BRG
2.	Digital Linear Actuator, AIRPAX Model L92421-P2
3.	COSMAC Microprocessor Evaluation System, RCA CDP18S025
4.	Barocel Electronic Manometer, DATAMETRICS Model 1174
5.	Linear Variable Differential Transformer, G. L. COLLINS CORP. Model LMT SS-207, S/N5240
6.	Turbometer, COX Model GM 16
7.	Rotameter, BROOKS CO. Model R9M-25-1
8.	Globe Valve, Meets Federal Specification WW-V-51A, Type 1
9.	Incremental Optical Linear Encoder, DYNAMICS RESEARCH CORP. Model S[illegible]T-E-4-1-0
10.	Linear Actuator Driver Board, AIRPAX Model K33505
11.	Bridgesensor, CALEX Model 166
12.	Pressure Regulator, PARKER-HANNIFIN Model R2037/J5
13.	Diverting Valve, CLIPPARD Model MJV-4D
14.	Solenoids (2), CLIPPARD Model AVS-12
15.	Data Acquisition System, CYBORG Isaac Model 91A

Chapter 3

DESCRIPTION OF SYSTEM DEVELOPMENT

Treating the three original components listed in Chap. 2 (i.e. the microprocessor system, the regulating valve, and the digital linear actuator) as the basic design, this work was directed toward the modification of the assembly. The modifications and the results of each are described in the following pages.

3.1 Initial Modifications

The turbometer, the globe valve, the rotameter, and an eight switch bit-setter were added to the system at the beginning of the development. The flow meters were added to allow measurements of the volumetric flow rate. The globe valve was added to permit variations of differential pressure, but was unnecessary for the reasons already described under equipment item No. 8, Chap. 2. The eight switch bit-setter was designed and built to allow direct binary input onto the microprocessor system data bus without the need for an ADC. A schematic of the design is shown in Fig. A.4, Appendix A.

3.2 Microprocessor Program Development

After the addition of the equipment listed in sec. 3.1, two microprocessor programs were developed. These programs, the Cycling Program and the Actuator Movement Program, are described in the following pages.

3.2.1 Cycling Program

The cycling program was developed to move the linear actuator/valve assembly in a controlled fashion. The purpose of the program

was to make the actuator/valve assembly cycle from the fully open position to the closed position. The listing of the program is entitled "Cycling Program" in Appendix B. The program operated by generating a clock signal (square wave) that was fed to the Interface Box. The signal was a high voltage (V_{CLK} = +5 volts) on bit zero (DO 0) of the output port, which was maintained while a countdown/delay subroutine was executed, and then V_{CLK} was dropped to 0 volts. The clock signal was formed by repeating this process at a set rate. The Interface Box received this square wave and utilized it to generate the signals used to drive the digital linear actuator When this square wave was accompanied by a high voltage (V_D) on bit one (DO 1) of the output port, the actuator shaft traveled in one direction. If V_D was then brought to zero, the direction of the actuator movement was reversed. By generating continuously the square wave and periodically changing V_D, the valve stem was cycled.

The distance the actuator moved in each direction could be set by placing the desired number of steps of linear travel in a designated holding register. By varying the number of incremental steps, testing showed the maximum linear travel of the valve stem occurred during a test sequence of 80 steps; with a step size of 0.0051 cm/step, this equates to a travel of approximately .408 cm. (An uncertainty may arise because the actuator could skip a step before maximum travel was determined). This value of maximum travel was .17 cm or 71.4% greater than the maximum stem travel documented by the valve manufacturer for the unmodified valve.

3.2.2 Actuator Movement Program

The next step in the development involved developing a second program that moved the actuator/valve assembly to a user specified location. Each location was prescribed by the number of actuator steps the stem was displaced from the "valve closed" position. The eight switch bit-setter was used as the input device to the microprocessor system. The program scanned the switches for the number of displacement steps the actuator/valve assembly was to be moved. The program then generated the clock signal that was input to the actuator driver board. Each square wave corresponded to one step displacement. The driver board output moved the actuator to the user specified position. The AIRPAX Model K33505 board had been modified by removing the on-board 555 timer. This timer controlled the rate at which the actuator was stepped, by means of an externally attached potentiometer. The clock signal required by the board was generated by the microprocessor program in the same manner as described in sec. 3.2.1 and was sent via the output port to the K33505 board. This modification allowed the stepping rate of the actuator to be controlled internally by the microprocessor system. (A listing of this program, entitled "Actuator Movement Program", is given in Appendix B).

3.3 Displacement Sensor

A Linear Variable Differential Transformer (LVDT) was added to the development system at this point to verify the accuracy of the control by the Actuator Movement Program. Accuracy was defined in terms of the program's ability to move the actuator/ valve

assembly to the user specified position on a repeatable basis. Testing of the system showed the actuator shaft wobbled sufficiently to affect the LVDT readings. To eliminate this wobble, two linear bearings were added to the assembly, one at either end of the actuator shaft. Subsequent testing showed that the problem caused by the wobble was eliminated. The result of the mean LVDT calibration test is shown in Fig. 4.

3.4 Flow Rate Calibration

Using the Actuator Movement Program as the stem controller, a series of calibration curves were generated. While maintaining a constant upstream pressure and incrementing the user specified position on the eight switch bit-setter (from the closed position to the fully open setting), data recorded included the volumetric flow rate, the upstream pressure, the differential pressure, and the LVDT voltage at each location. During this testing, it was discovered that the linear actuator could not produce sufficient linear force to close the valve completely at upstream pressures above 267 kPa gage. Analysis revealed that the bellows spring required a linear force of 53.4 N to close the valve after the stem threads had been removed during modification. Since the linear actuator could produce only 84.5 N at the slowest stepping rate, it was clear that at moderate to high upstream pressures, the actuator could not close the valve.

Rather than reduce the range of upstream test pressures and thereby the flow rate resolution, an upstream assembly was added to divert the flow and eliminate the problem of insufficient linear force. A single input/dual output diverting valve was placed between

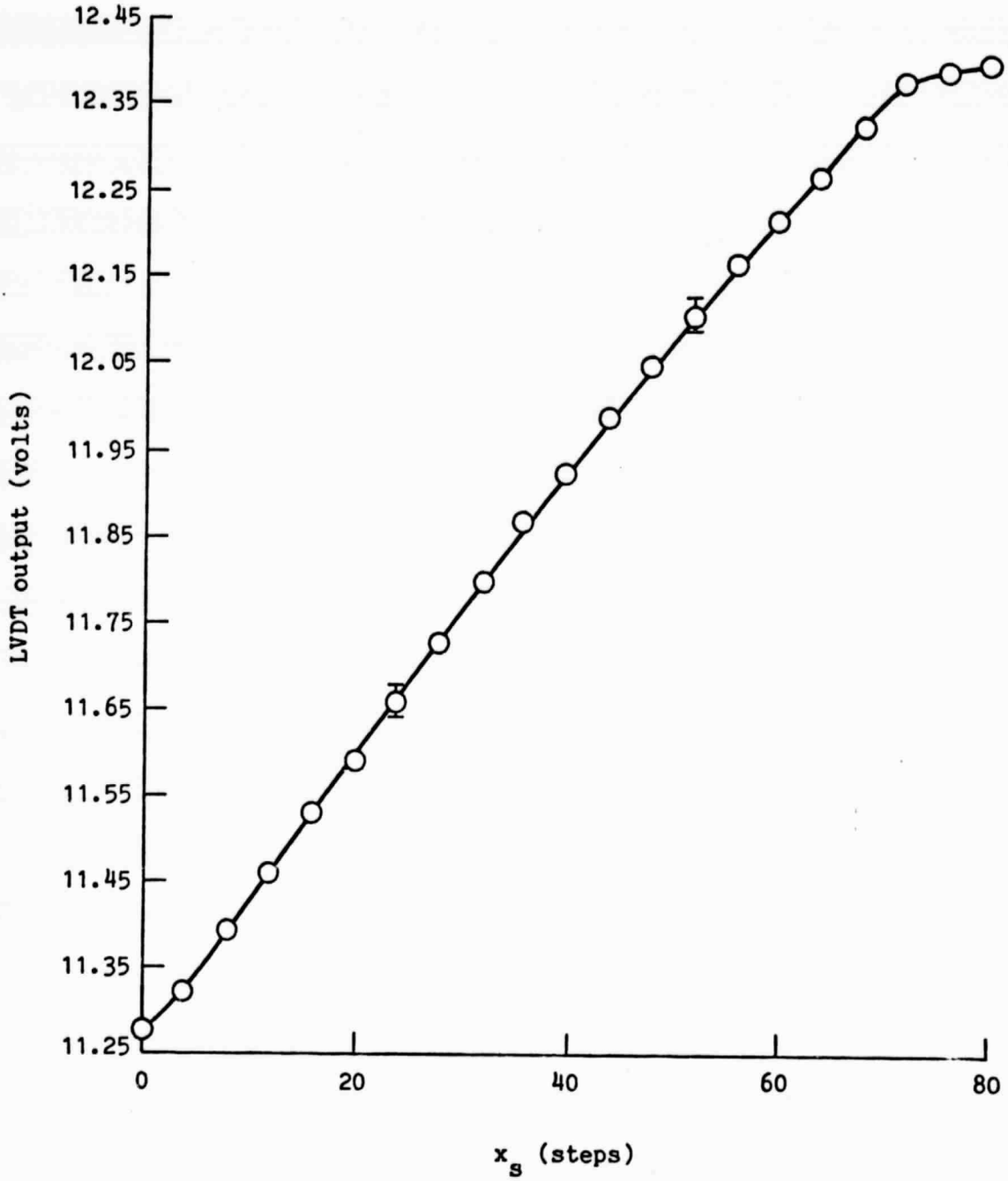

Fig. 4 LVDT calibration curve.

the regulating valve and the upstream pressure regulator, to allow the flow to be stopped during valve stem position changes. This allowed the pressurized volume upstream of the regulating valve to vent to atmosphere; reducing the amount of force required to close the valve. To reduce power usage, the upstream diverting valve was driven by two solenoids -- one on either side of the valve (see Fig. 3). Hence, the upstream flow was cut off by a single pulse of power rather than requiring a continuous current which was required normally to open or close solenoid valves. After the addition of this "latching" solenoid to the upstream assembly, the data presented in Appendix D, Table D.1, were collected without further problems due to the linear actuator. It should be noted that the upstream assembly (i.e., the two solenoids and the diverting valve) was not part of the basic design of the digital valve system, but was added to compensate for the insufficient linear force available from the actuator.

The mean flow rate calibration data are presented in Table 2. Upstream pressure was considered the controlled parameter, and is given in lb_f/in^2 (psi) because the instrumentation operated using those units. While differential pressure is a controlling parameter in the volumetric flow rate of a fluid through a valve (modeling the valve as a venturi [5]), the upstream pressure was used in this development. This decision was based on the lack of resolution in the differential pressure readings across the valve, between successive changes in the valve stem displacement. As shown by the preliminary calibration data shown in Table D.1, Appendix D, the

Table 2 Preliminary volumetric flow rate calibration data

Upstream Pressure [a]	Valve stem displacement (steps from closed position)																				
	00	04	08	12	16	20	24	28	32	36	40	44	48	52	56	60	64	68	72	76	80
	Volumetric flow rate [b]																				
207	0.0	241[c]	481[d]	698	883	1001	1114	1185	1232	1274	1307	1321	1336	1350	1364	1364	1383	1383	1397	1397	1397
30	00	70	140	205	265	305	340	365	380	395	405	410	415	420	425	425	430	430	435	435	435
	0000	0046	008C	00CD	0109	0131	0154	016D	017C	018B	0195	019A	019F	01A4	01A9	01A9	01AE	01AE	01B3	01B3	01B3
276	0.0	293[c]	599[d]	883	1123	1288	1411	1501	1576	1624	1652	1685	1713	1713	1727	1746	1760	1775	1775	1775	1775
40	00	85	175	265	345	400	440	470	495	510	520	530	540	540	545	550	555	560	560	560	560
	0000	0055	00AF	0109	0159	0190	01B8	01D6	01EF	01FE	0208	0212	021C	021C	0221	0226	022B	0230	0230	0230	0230
345	0.0	359[c]	732	1062	1350	1562	1713	1803	1897	1959	2001	2034	2048	2077	2091	2091	2110	2124	2138	2138	2138
50	00	105	215	325	420	490	540	570	600	620	635	645	650	660	665	665	670	675	680	680	680
	0000	0069	00D7	0145	01A4	01EA	021C	023A	0258	026C	027B	0285	028A	0294	0299	0299	029E	02A3	02A8	02A8	02A8
414	0.0	414[d]	897	1260	1576	1822	2001	2138	2228	2289	2336	2364	2412	2440	2440	2440	2473	2487	2501	2501	2501
60	00	120	270	390	495	575	635	680	710	730	745	755	770	780	780	780	790	795	800	800	800
	0000	0078	010E	0186	01EF	023F	027B	02A8	02C6	02DA	02E9	02F3	0302	030C	030C	030C	0316	031B	0320	0320	0320
483	0.0	496[d]	1034	1425	1789	2062	2275	2426	2534	2596	2657	2700	2733	2761	2794	2808	2822	2841	2841	2841	2855
70	00	145	315	445	565	655	725	775	810	830	850	865	875	885	895	900	905	910	910	910	915
	0000	0001	013B	01BD	0235	028F	02D5	0307	032A	033F	0352	0361	036B	0375	037F	0384	0389	038E	038E	038E	0393

a. kilo-Pascals gage
psi gage

b. Volumetric Flow Rate (SCCS)
Rotameter$_{10}$
Rotameter$_{16}$

c. Reading below Rotameter scale, value extrapolated

d. Air spring phenomena. Steady value extrapolated

changes in differential pressure were small. Consequently, because the microprocessor system could not conveniently read fractional changes in the differential readings, upstream pressure was used as the control parameter. This was a valid assumption since the downstream pressure was expected to remain constant in this application. The validity of this was further verified by the data shown in Fig. 5, where a linear relationship between differential pressure and upstream pressure was observed. Therefore, although the use of upstream pressure was not fundamentally correct, it was valid to use as a control parameter under the system conditions.

In Table 2, the SI equivalents to the read-out pressure are listed along the left column of the table along with the psi value. Every fourth user specifiable location (from closed to fully open) was tested in the flow rate calibration. Each of the volumetric flow rate data in Table 2 are listed as three numbers: (1) the volumetric flow rate in standard cm^3/sec (SCCS); (2) the flow rate reading from the rotameter (used in the table to eliminate the need to convert from rotameter scale to SCCS); and (3) the volumetric flow rate reading from the rotameter scale in hexidecimal. A graphical representation of this table is given in Fig. 6. The correction factor between actual flow rate and flow rate at standard temperature and pressure was less than 1%. For this reason, the flow rates are listed at standard conditions. The LVDT calibration data in Fig. 4 shows that the maximum linear travel of the valve stem was not 0.408 cm as expected for a linear travel of 0.0051 cm/step. Rather, the linear distance traveled began to approach asymptotically a maximum

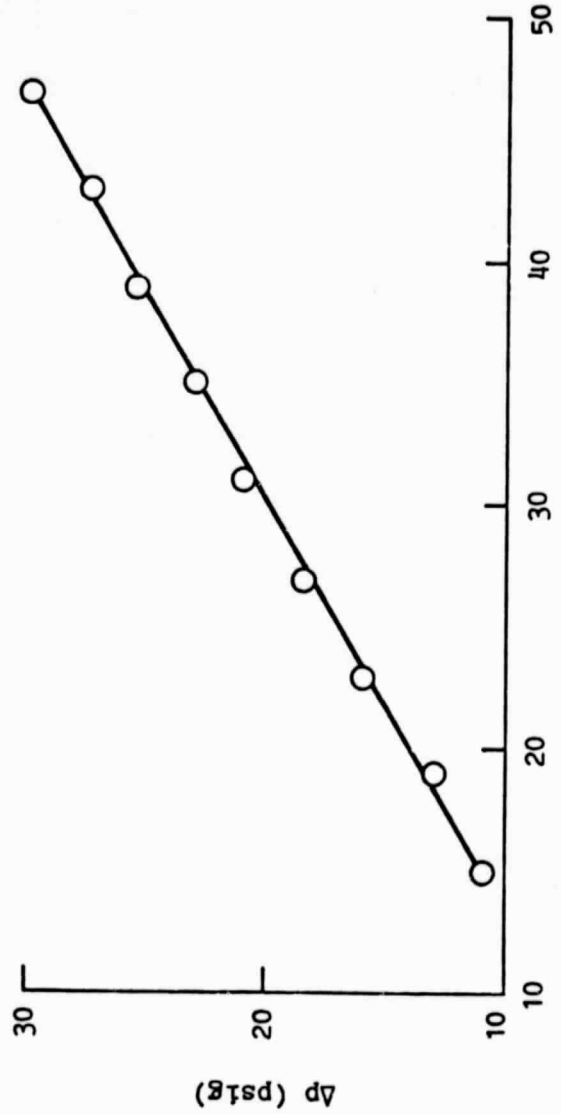

Fig. 5 Differential pressure across the valve vs upstream pressure.

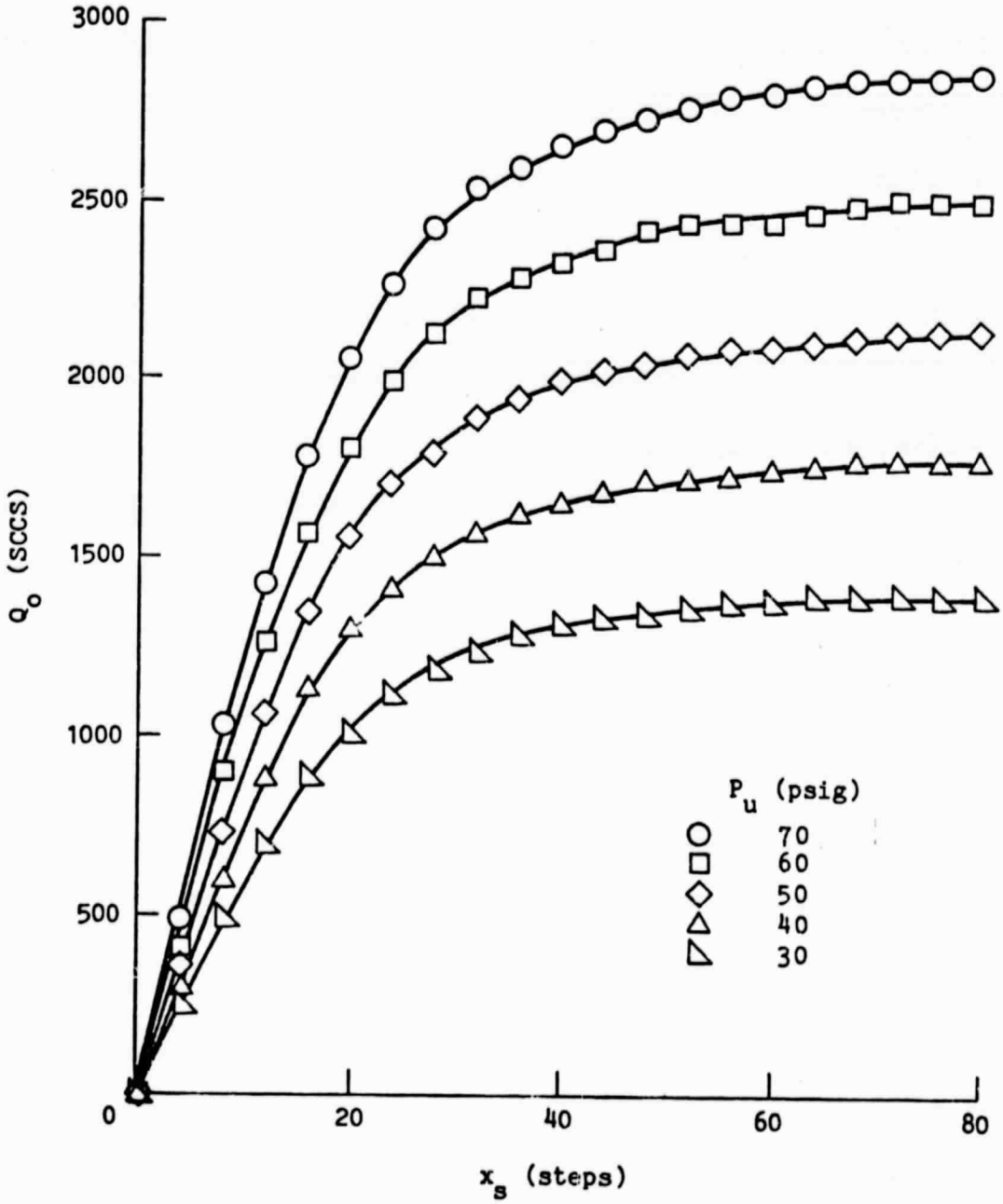

Fig. 6 Preliminary volumetric flow rate calibration curves.

value of 0.3429 cm. The nonlinearity resulted from the condition of the bellows spring, which changed from compression to tension. This tension caused the actuator to slip when the bellows was fully stretched.

3.5 Interpolation Program

The generation of Table 2 was essential in the development of the Interpolation Program for the microprocessor system. The graphical representation, Fig. 6, was used to determine how the interpolation should proceed. The interpolation process was chosen over a table look-up due to the limitation of the RAM size in the microprocessor system. It was not desirable (or even possible) to have a data table stored in the microprocessor system memory with the volumetric flow rate data for each user specifiable location. (Such a table would require 6683 memory bytes rather than the 215 memory bytes required by the data in Table 2). Therefore, because the calibration curves shown in Fig. 6 are non-linear, it became necessary for the microprocessor system to calculate the correct valve stem displacement for a requested flow rate. The calculated displacement was in the number of actuator steps from the closed position, to achieve the specified volumetric flow rate at the monitored upstream pressure. To do this, the microprocessor system was required to interpolate the correct displacement using data similar to that of Table 2. Difficulty arose at this point because the RCA CDP18S025 Evaluation System was incapable of interpolation without the addition of a RCA CDPR582CD Fixed-Point Binary Arithmetic Circuit. Without the addition of this circuit, the microprocessor

system was only capable of multiplication or division by twos. To eliminate the need for this costly and time consuming modification, a special interpolation program was written. A listing of this program can be found under "Interpolation Program" in Appendix B.

The Interpolation Program consisted of two parts: (1) the collection and storage of the necessary data for the interpolation process; and (2) the interpolation of the correct valve stem displacement (in terms of the number of linear actuator steps from the closed position). A flow chart of the data collection element is shown in Fig. 7. The interpolation element of the program is represented by the flow chart in Fig. 8.

The data from Table 2 was stored in the microprocessor system RAM area in such a way as to utilize the memory address locations in the Interpolation Program. First, the data was formed into columns of constant upstream pressure with the pressure reading at the head of each column. Next, these columns were placed into the RAM in a single continuous array, starting with the lowest upstream pressure column, then followed by columns for each successively higher pressure. Because these constant pressure columns were each 43 bytes in length, flow rate data for the same displacement could be found by moving the look-up pointer by 43 counts. For example, at 40 psig the flow rate was 1576 SCCS at 32 steps displacement. By moving the look-up pointer 43 counts ahead, a flow rate of 1897 SCCS was found for the same displacement but at 50 psig.

To illustrate the interpolation process, Fig. 9 has been supplied to show the sequence of the program. The numbers in

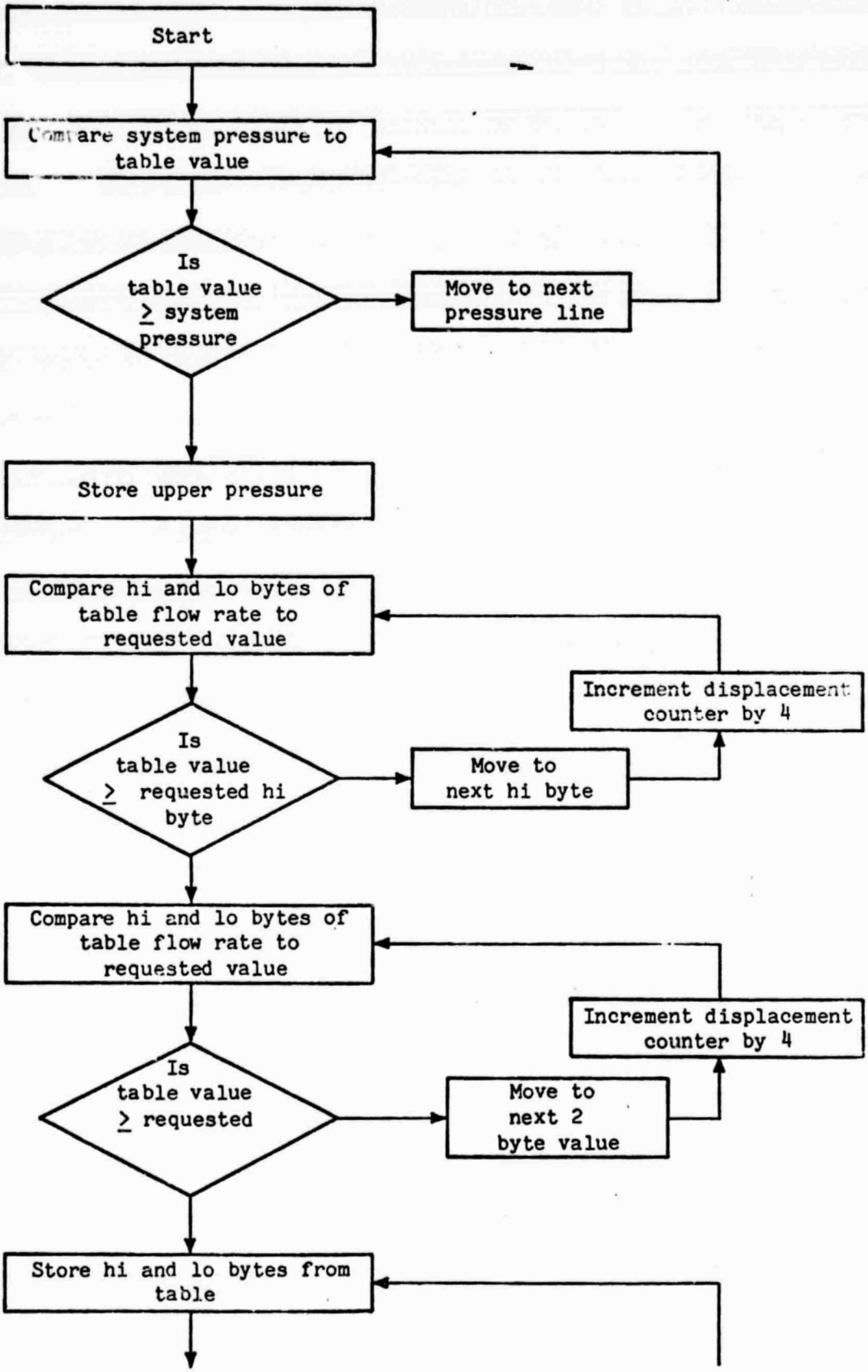

Fig. 7 Flow chart of data collection element of the Interpolation Program.

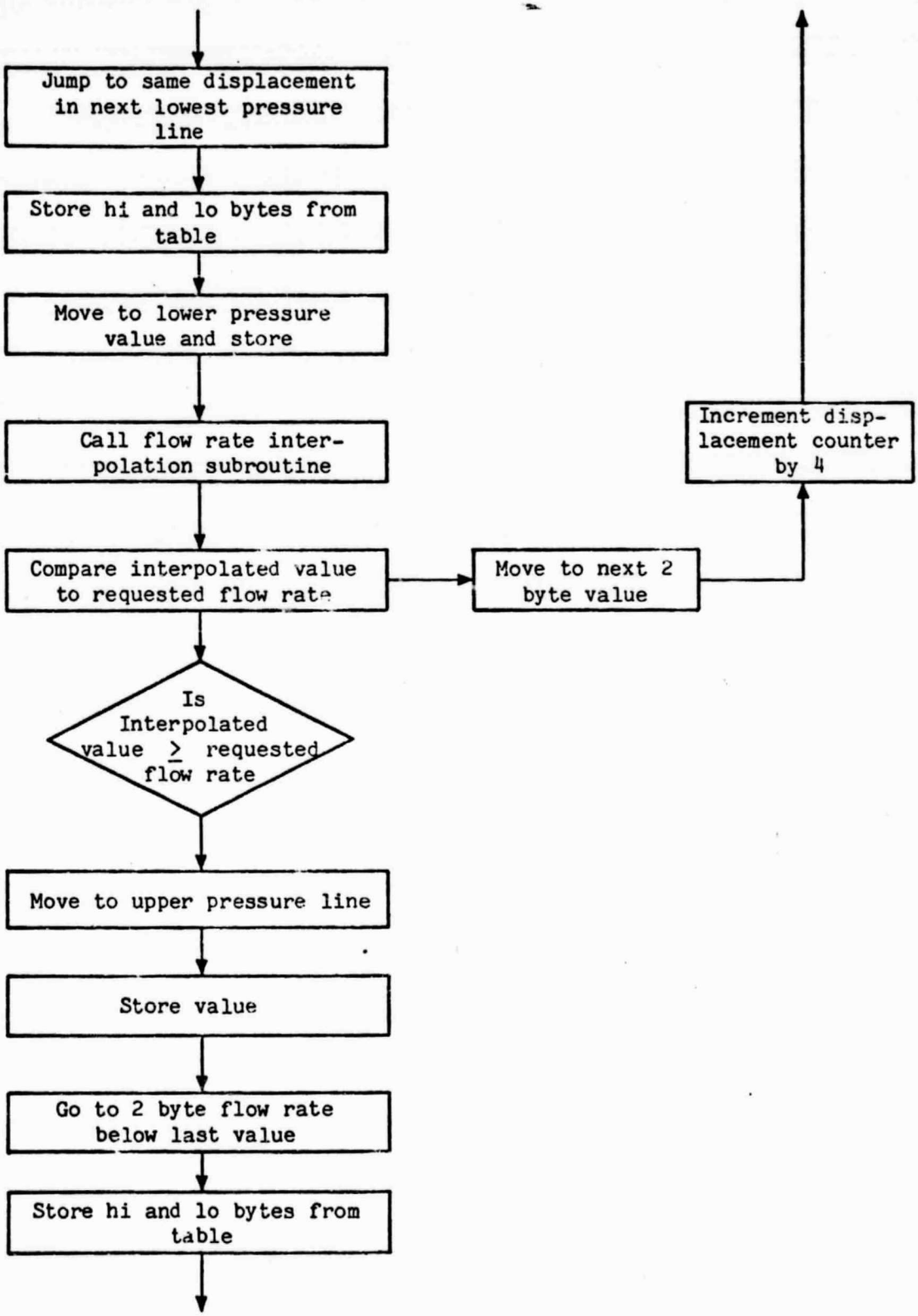

Fig. 7 Continued.

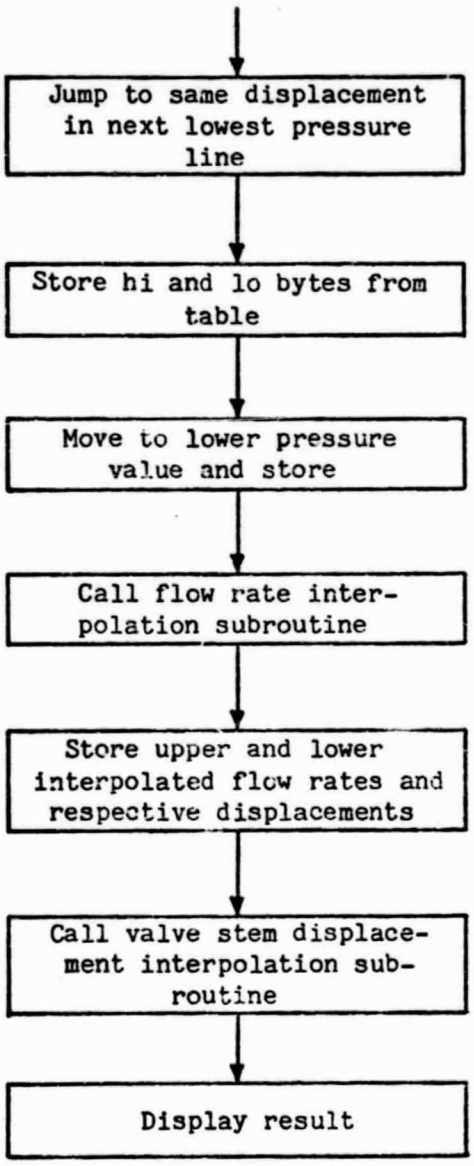

Fig. 7 Concluded.

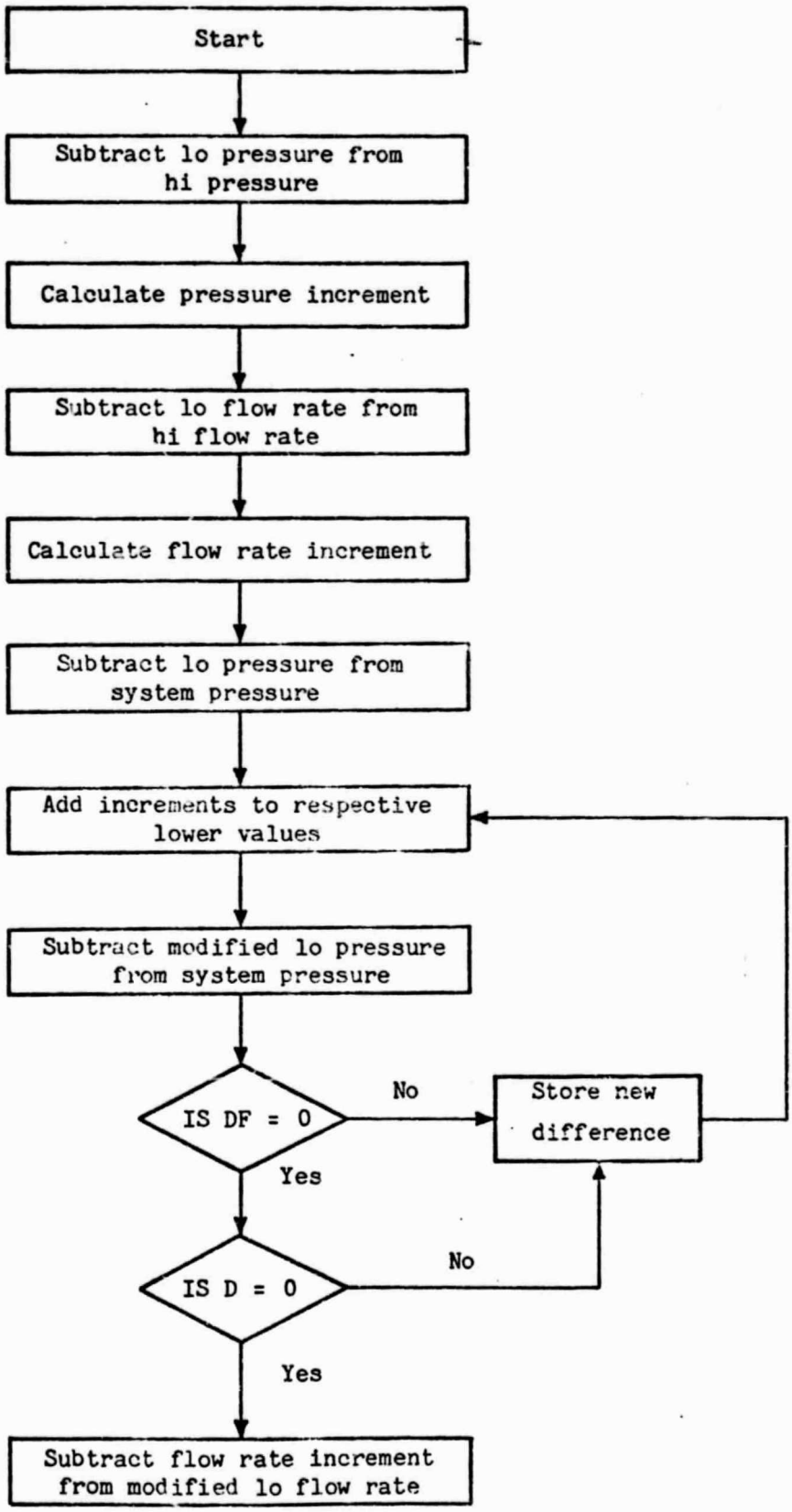

Fig. 8 Flow chart of interpolation element of the Interpolation Program.

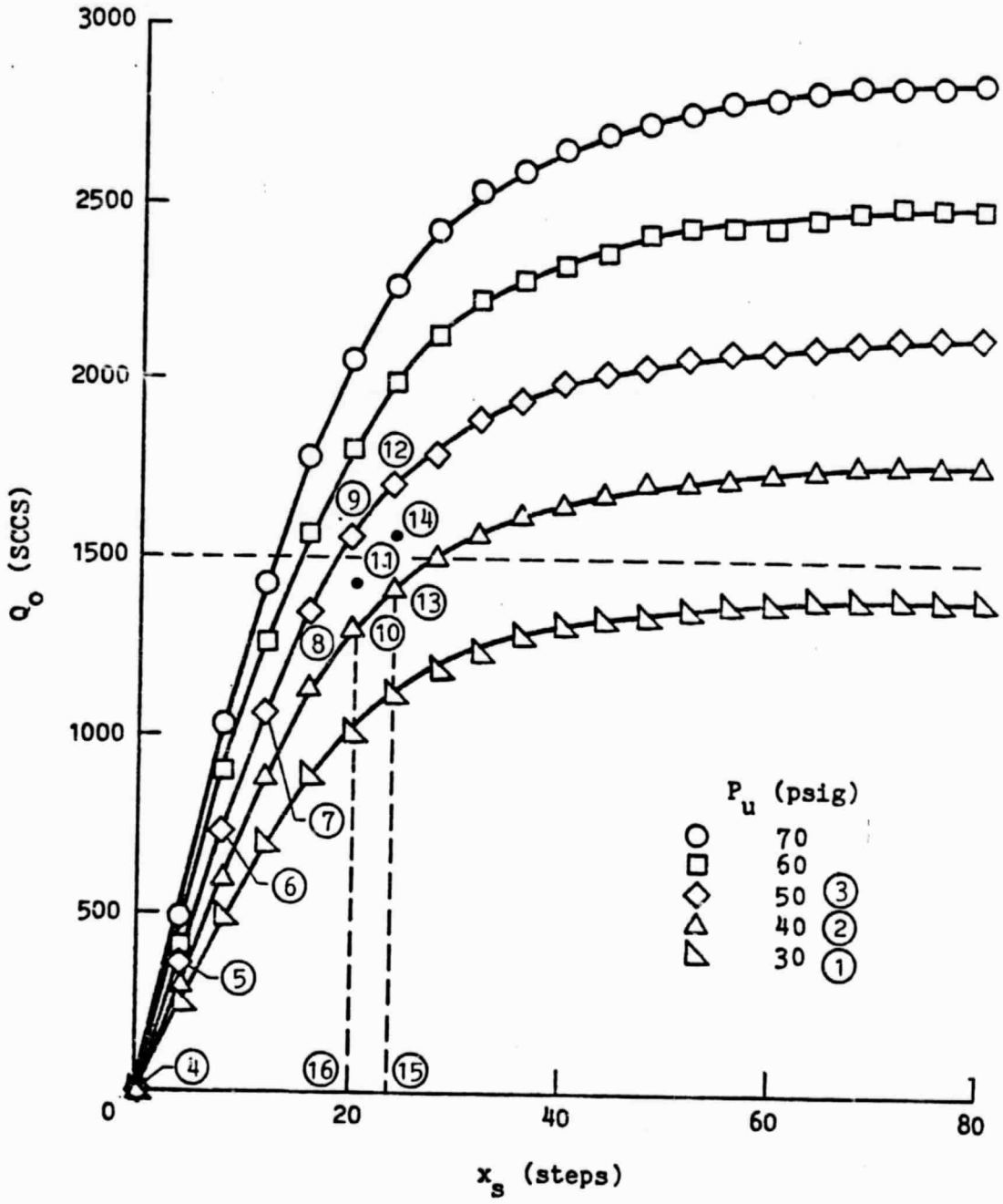

Fig. 9 Illustration of interpolation process.

parentheses () in the following text will correspond to those in Fig. 9. The conditions of the illustration are: P_u = 45 psig; Q_r = 1500 SCCS.

3.5.1 Data Collection Element

The data collection element of the Interpolation Program began by comparing the input value of upstream pressure to the first table value (in this case, 30 psig (1)). If the system pressure was not less than or equal to the memory table valve, the program jumped the subsequent flow rate data to the next highest constant upstream pressure line and repeated this comparison. This process continued until the program reached a value of memory table pressure greater than or equal to the input pressure (2,3). At this point, the upper pressure range had been found for the interpolation process. The lower range would be the last value of memory table pressure not accepted (2). The program stored the upper pressure limit in a specified holding area and then compared the requested volumetric flow rate to the values in the array at that point. The two byte volumetric flow rate requested would be compared to each two byte data value in that upper pressure line until one was found that was greater than or equal to the request value (5-9). Simultaneously, a storage register was increased by increments of four as each successive table flow rate was compared. As a result, the corresponding valve stem displacement (in steps from the closed position) could be determined.

Once a table flow rate was found that was greater than or equal to the requested value, that value was stored in the specified

holding area and the program jumped back in the array to the same valve stem displacement location in the lower pressure range (10). This value of flow rate and then the lower pressure value were also stored in the specified holding area. With these four data values, the interpolation element (to be described later) was used to calculate the volumetric flow rate corresponding to the system pressure, at that valve stem displacement (11). The interpolated flow rate was compared to the requested value to determine if it was greater than or equal to the requested flow rate. If the calculated value was not greater than or equal to the requested value, then the upper boundary of flow rate along the system pressure line had not been determined. The data collection element of the program was repeated to get to the next highest flow rate pair (12,13). These are along the upper (3) and lower (2) table pressure boundaries (having been determined earlier). The process was repeated until the calculated flow rate was greater than or equal to the requested flow rate (14). At this point, the upper (14) and lower (11) boundaries of the flow rate had been determined (along the upstream pressure line of the system). The correct valve stem displacement can then be interpolated between these two values.

3.5.2 Interpolation Element

The initial step for interpolating the volumetric flow rate (14) consisted of subtracting the lower boundary pressure and flow rate values from the upper boundary values (2,13 from 3,12). The differences were then divided into increments. The increment sizes corresponded to resolution of the microprocessor

system. In the case of calculating the volumetric flow rate (along the system pressure line), the divisor is eight. This is for two reasons: (1) Since the microprocessor system can only multiply or divide by twos, the possible divisors were 2, 4, 8, or 16. A divisor of four would cause the increment size to be too large. A divisor of 16 would result in the pressure increment being zero. (2) The microprocessor system had a resolution of one psi for the monitoring of upstream pressure. It was desirable to have the increment size the same as the resolution. This reduced the round-off error.

The interpolation was simulated by adding the pressure and flow rate increments to the respective lower boundaries until the incremented pressure equaled the system pressure. At this point, the incremented lower flow rate (13) was equal to the interpolated value along the system pressure line (14). To achieve this in the programming, a test condition was set-up for the microprocessor system. As a test condition, the lower pressure boundary was subtracted from the system pressure and the result was stored. After the pressure increment was added to the lower pressure boundary, this sum was subtracted from the system pressure. This new difference (DIFF) was compared to the test criteria (TEST) ; if DIFF was less than TEST, the incremented lower pressure value was approaching the system pressure and another increment of pressure and flow rate were added to the respective holding areas. Subsequently, the new DIFF became the test criteria (TEST). This process continued until the new DIFF was greater than or equal to TEST. At that point, the incremented lower number pressure was diverging from the system

pressure so that the old value of incremented volumetric flow rate was the closest approximation possible (with the increment size used).

The process was similar for the valve stem displacement interpolation. The major difference was the requested volumetric flow rate became the measure by which the divergence test was made. The values stored in the specified holding area were the upper and lower interpolated flow rates (14,11) and their respective valve stem displacements (15,16). The incremental values were found by dividing the differences between the upper and lower boundaries by four (the number of steps between adjacent flow rate values on the data table). The incremental value of flow rate was added to the lower interpolated flow rate. The test condition for the microprocessor system was the difference between the lower flow rate (11) and the requested value. After the addition, the modified lower flow rate was subtracted from the requested flow rate. This new difference (DIFF) was compared to TEST; if DIFF was less than TEST, the incremented lower flow rate was approaching the requested value and another increment of flow rate and displacement were added to the respective incremented lower boundaries. As before, the new DIFF became TEST. As in the flow rate interpolation, this process continued until the divergence was detected. At that point, the previous value of displacement would give a flow rate closest the requested value (assuming the system upstream pressure was maintained). Testing of the completed program showed the accuracy of the interpolation process to be within one step in 80 or within 1.25% full scale.

3.6 Development of Overall Controlling Program

With the Interpolation Program completed, work began on a 32 switch bit-setter and the addition of four input ports to the microprocessor system. This new bit-setter was needed for the development of an overall controlling microprocessor program. This program included the Interpolation Program and parts of the Actuator Movement Program. The 32 switch bit-setter was needed to simulate the inputs, which consisted of the upstream pressure, requested volumetric flow rate, and LVDT voltage. The 32 switch bit-setter is equivalent to four eight switch bit-setters combined (see schematic, Fig. A.5, Appendix A). The 32 switch bit-setter and the four input ports were tested and found to function properly.

The execution of the Overall Program is a closed loop (see Fig. 10 for a flow chart of the Overall Program). The microprocessor system read from 16 of the input switches the requested volumetric flow rate, and from another eight the value of upstream pressure from the manometer. Those readings were then used in the Interpolation Program to calculate the correct valve stem displacement. The system then moved the actuator/valve assembly to the calculated position. A subroutine checked the LVDT voltage against a calibration table to determine whether the actuator/valve assembly was correctly positioned. In the event the LVDT voltage showed the actuator/valve assembly to be incorrectly positioned, the Overall Program would calculate the error in the number of actuator steps and correct the position by repositioning the assembly. On test cycles of the Overall Program when the actuator/valve assembly were already

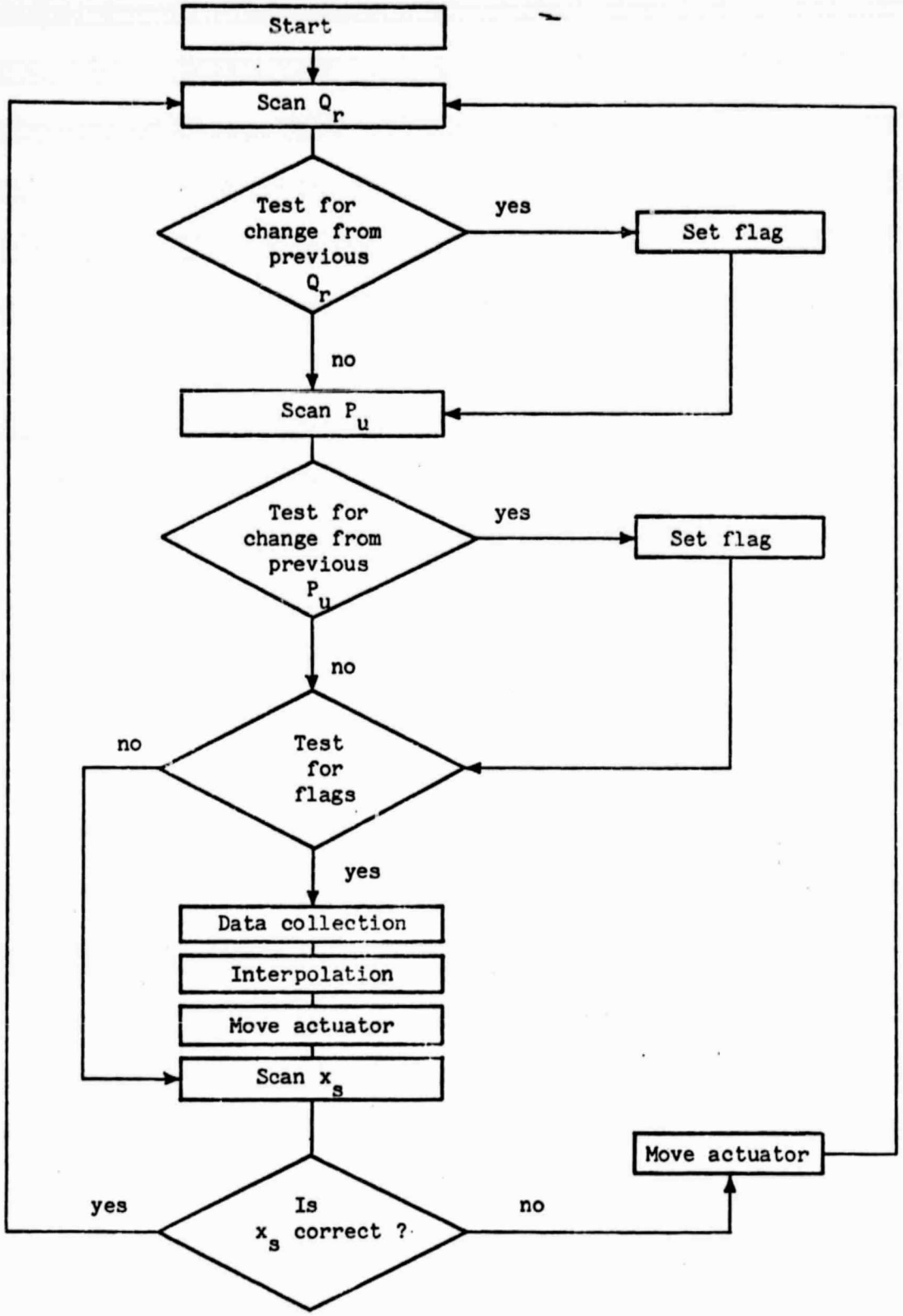

Fig. 10 Flow chart of Overall Program

positioned correctly, the LVDT subroutine was still executed to determine whether the assembly had slipped out of position. A simulation of the program was run using the 32 switch bit-setter to input the values of requested flow rate, upstream pressure, and LVDT voltage. From the results of the testing, the Overall Program was found to function correctly.

3.7 Attachment of Peripheral Sensors

With the Overall Program functional, efforts were focused on improving the peripheral sensing devices (i.e., the manometer and the position sensor) and to attach them to the microprocessor system. The electronic manometer was attached by means of the Manometer Interface Circuit (MIC) shown in Appendix A. This circuit counted the number of pulses received from the CALEX Bridgesensor V/F converter over a one second period and put the total on the data bus. The total was updated once every four seconds.

Because the analog signal from the LVDT was incompatible with the digital microprocessor system, an Incremental Optical Linear Encoder (IOLE) was used to replace the LVDT in the development and thereby eliminate the need for an ADC (see Fig. 11 for main valve assembly). Normally the IOLE has a resolution of 1.016×10^{-3} cm/cycle and a square wave quadrature output (90^{o} out of phase) that is TTL compatible. But, a resolution of 1.016×10^{-3} cm/cycle was too sensitive for the maximum stem travel of 0.3429 cm shown by the LVDT. Such a sensitivity produced a maximum pulse count which exceeded the limiting value of 255 for one byte. Since the input was limited to one byte due to the number of available input ports, and

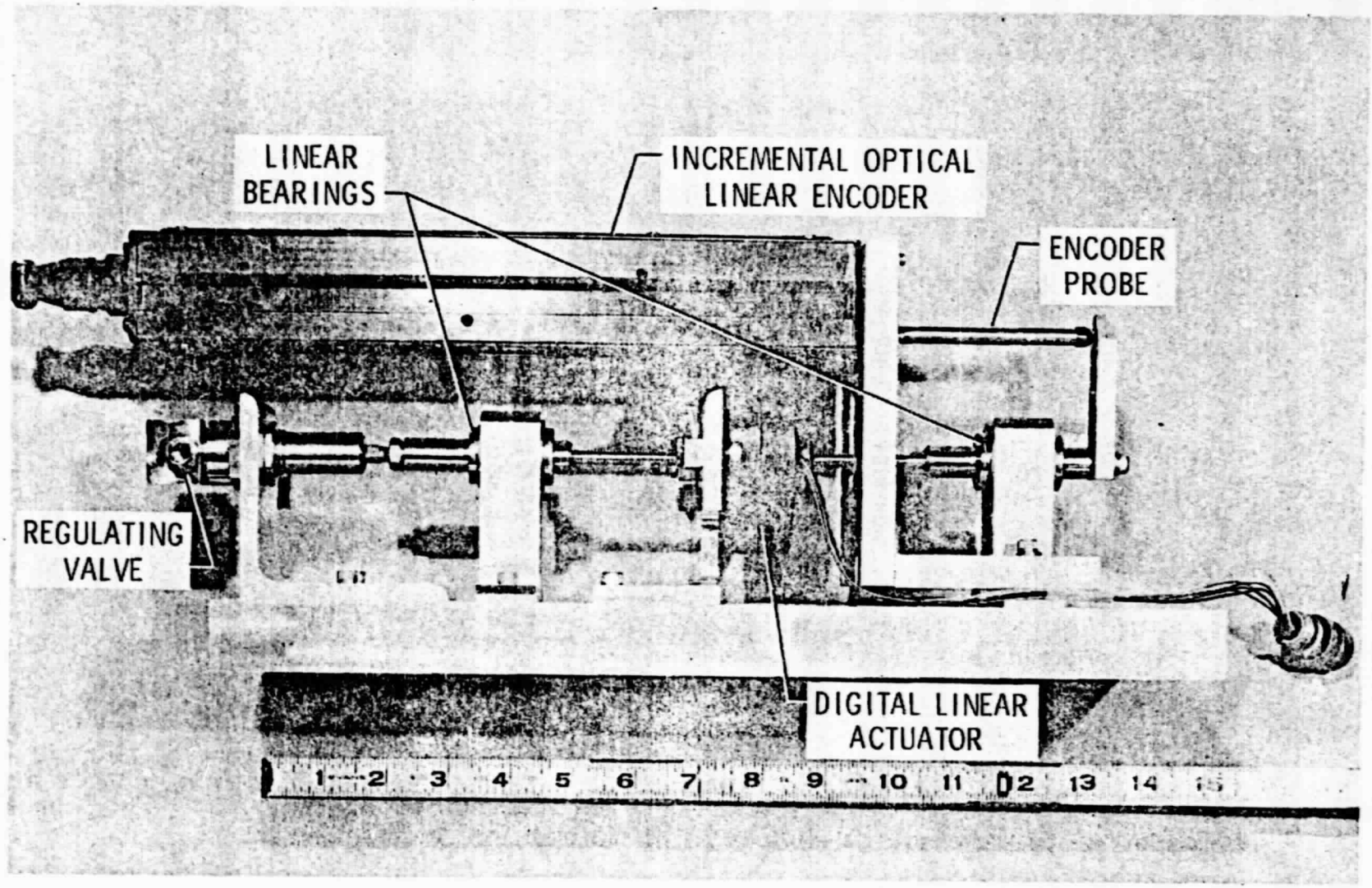

Fig. 11 Photograph of main valve assembly.

the way the LVDT subroutine was written (the subroutine was written to manipulate eight bits not 16), an IOLE Interface Circuit (shown in Fig. A.1, Appendix A), was utilized to reduce the sensitivity of the IOLE by a factor of two. This kept the count-up total to one byte. Testing of the circuit with the IOLE, before the sensor was attached to the assembly, was successful.

With the IOLE Interface Circuit working, the IOLE was sent out to be attached to the valve assembly. Testing of the assembly after return showed the IOLE to be inoperative. Examination of the device revealed it had been damaged during the attachment and that realignment of the internal optical components was required. The manufacturer was contacted in an attempt to get the correct repair procedure. Repairs were completed and testing began to determine the ability of the IOLE to detect the valve stem displacement accurately and repeatably. The IOLE was found to be incapable of accurately measuring the displacement of the stem with the mounting configuration and the stem movement scheme as programmed. Analysis of the assembly revealed the IOLE to be adversely affected by the vibrations of the digital linear actuator during the movement phase. The errors induced by the vibrations were also cumulative, requiring the IOLE Interface Circuit to be reset each time the valve was closed. Rubber gaskets (vibration dampers) were added to the assembly in the locations shown in Fig. 12 to reduce the effect of the vibrations. Also, the Stepping Rate (SR) of linear actuator was reduced from 40.1 steps/sec to 5.41 steps/sec. However, testing showed that even with these modifications, there was still a small error caused by

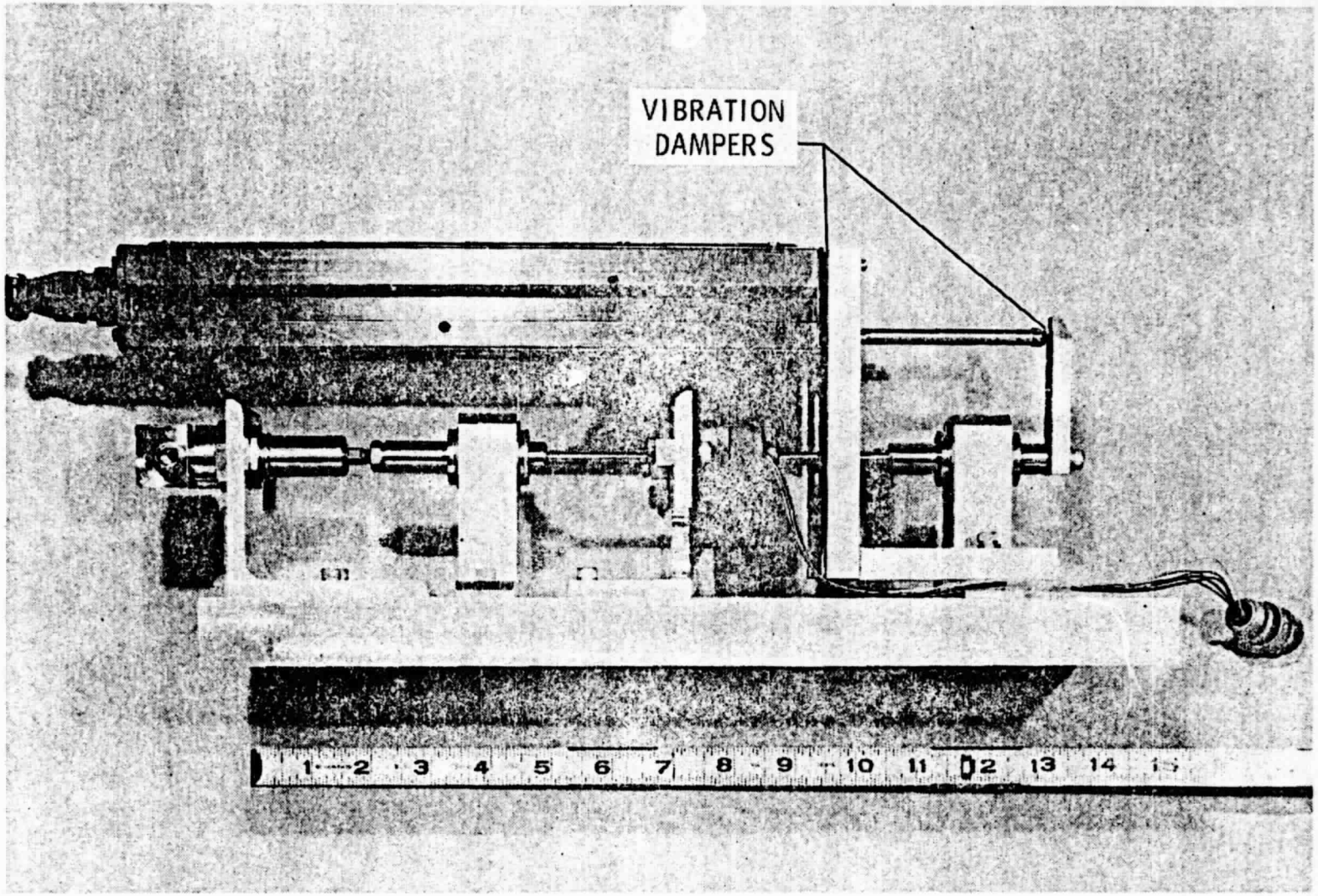

Fig. 12 Photograph of locations of vibration dampers.

vibrations and this error was cumulative in the IOLE Interface Circuit. To eliminate this problem, the Overall Program was modified to close the valve before each repositioning and to reset the Interface Circuit each time the valve was closed. With these modifications in effect, the data table listed in Appendix D was recorded (see Linear encoder calibration data in Appendix D, Table D.2).

With these Overall Program modifications it became necessary to again simulate the assembly. This time simulation was performed using the 1802 simulator on the computer system at NASA Langley Research Center. The simulation revealed minor errors in the program that were corrected and testing began on the complete assembly.

3.8 Testing of Overall Assembly

To allow visual read-out of the binary data supplied by the interface circuits, two arrays of Light Emitting Diodes (LED's) were attached. The outputs from the interface circuits to the LED's were buffered, to prevent loading by the signals being sent to the input ports of the 32 switch bit-setter.

At this point, the overall assembly was tested. A requested volumetric flow rate of 1543 SCCS was entered in the Overall Program via the 32 switch bit-setter. While the system responded, the response was erratic and the test was stopped. Analysis of the situation showed real-time errors in the Overall Program. It was found that the program commands were not synchronized with the timing loop of the MIC. The values of upstream pressure being accepted by the microprocessor system were incorrect after two to three cycles of

the program. The problem was eliminated by altering the circuit so that it didn't begin the four second count cycle until a signal was received from the microprocessor system. After the upstream pressure was read, the microprocessor sent a command to turn the circuit off again. By placing a delay between the time the circuit was turned on and the time the upstream pressure was read by the microprocessor system, the two were synchronized.

Testing of the assembly was repeated. The resulting volumetric flow rate was within $\pm$10% of the requested value, but the assembly would not maintain a steady setting for more than 30 seconds before the valve stem was repositioned. This constant cycling was due to the MIC. The output from the circuit fluctuated within plus and minus one psi of the actual upstream pressure. The reason for this fluctuation was not found. Therefore, to compensate, the Overall Program was altered to cause the microprocessor system to ignore a pressure fluctuation of one psi. When testing began again, the assembly achieved a steady position and compensated for all upstream pressure fluctuations, but the output volumetric flow rate was not the amount requested. The problem arose from the method by which the original flow rate calibration data were taken. The constant upstream pressure value had been taken from the pressure gage attached to the upstream assembly. The pressure drop across the upstream diverting valve plus head loss in the tubing meant the flow rate calibration data in Table 2 was no longer consistent. Therefore, a new set of calibration data were taken. A schematic of the overall system is shown in Fig. 13. The new location of the

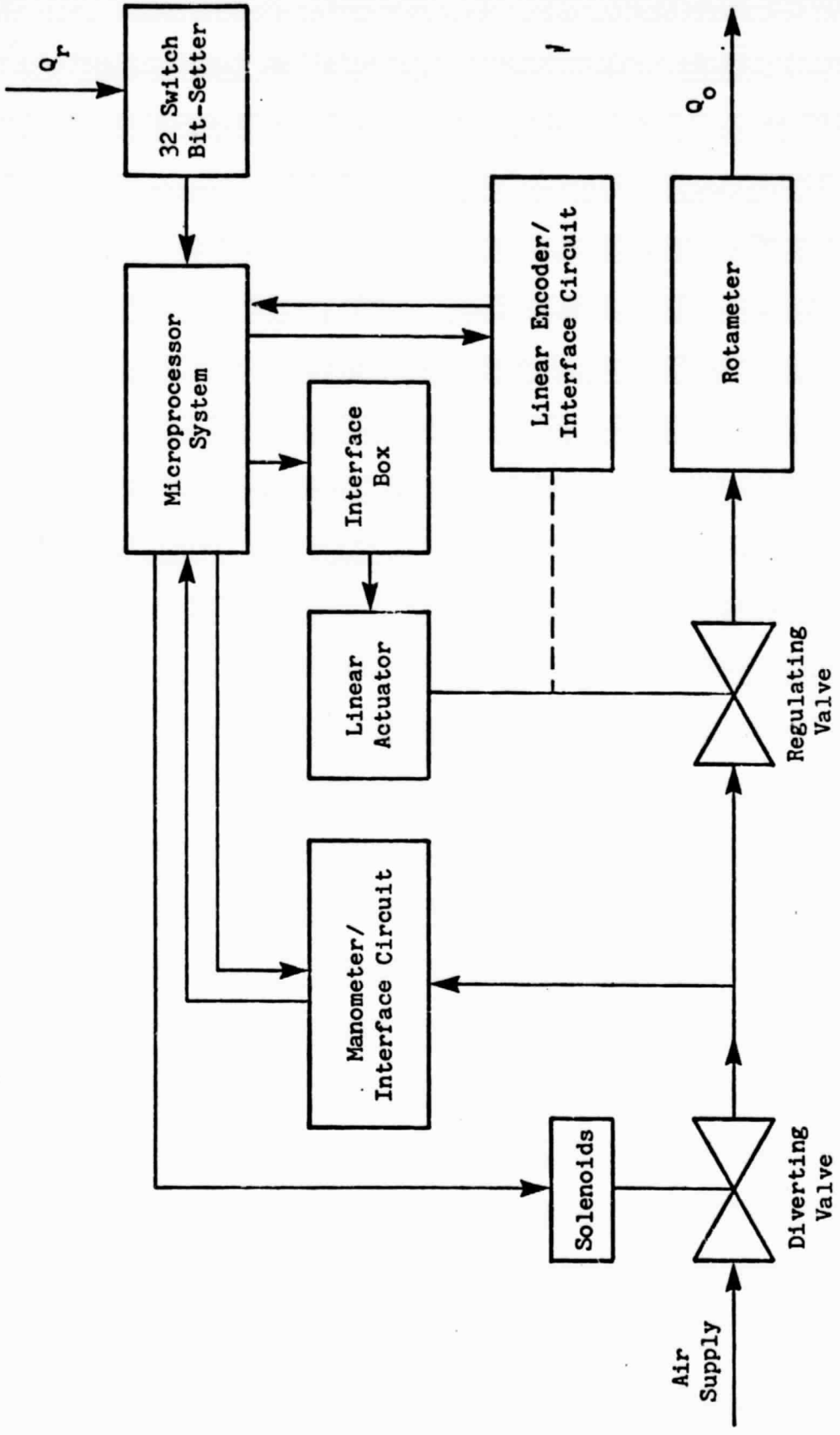

Fig. 13 Schematic of overall assembly.

pressure sensor was approximately 15 cm upstream of the regulating valve. Table 3 shows the averaged flow rate data in the same format used in Table 2. The data is graphically represented in Fig. 14. Note that the shape of the curves in Fig. 14 are the same as those in Fig. 6. Because the interpolation program was written with this curve shape in mind, the interpolation process developed in the Overall Program was still valid.

When the testing of 1543 SCCS was repeated, the resulting volumetric flow rate was within $\pm 5\%$ of the requested value. The upstream pressure was changed to reveal whether the overall assembly could compensate. All output flow rates were within $\pm 5\%$ of the 1543 SCCS requested. Further tests of the system were run with requested flow rates of 708,858, and 1251 SCCS. The results of all these tests can be seen in Figs. 15(a) - 18(b). The data points are numbered to show the sequence of the testing. The primes (') denote the sequence of the testing with the upstream pressure recorded from the Manometer Interface Circuit. The other numbers denote the sequence of testing with the upstream pressure recorded from the digital read-out of the manometer. It should be noted that the microprocessor system accepted input from the interface circuit; therefore, the upstream pressures used in the interpolation process were frequently incorrect.

Figures 15(a) - 18(a) are the delivered flow rate versus the upstream pressure. Figures 15(b) - 18(b) are the IOLE output versus the upstream pressure. As can be seen in Fig. 15(a), the error was more than 5% at two pressure readings during the 708 SCCS testing

Table 3 Volumetric flow rate calibration data

Upstream[a] Pressure	Valve stem displacement (steps from closed position)																				
	00	04	08	12	16	20	24	28	32	36	40	44	48	52	56	60	64	68	72	76	80
	Volumetric flow rate[b]																				
103	0.0	241[c]	463[d]	632	779	901	1024	1114	1208	1270	1331	1373	1435	1482	1515	1543	1543	1543	1543	1543	1543
15	00	70	135	185	230	270	310	340	370	390	410	425	445	460	470	480	480	480	480	480	480
	0000	0046	0087	00B9	00E6	010E	0136	0154	0172	0186	019A	01A9	01BD	01CC	01D6	01E0	01E0	01E0	01E0	01E0	01E0
159	0.0	359[c]	618[d]	840	1052	1237	1373	1515	1638	1727	1822	1883	1973	2015	2058	2105	2133	2147	2147	2147	2147
23	00	105	180	250	320	380	425	470	510	540	570	590	620	635	650	665	675	680	680	680	680
	0000	0069	00B4	00FA	0140	017C	01A9	01D6	01FE	021C	023A	024E	026C	027B	028A	0299	02A3	02A8	02A8	02A8	02A8
214	0.0	481[c]	779	1067	1331	1543	1741	1911	2044	2162	2265	2341	2426	2501	2563	2610	2657	2685	2685	2685	2685
31	00	140	230	325	410	480	545	600	645	685	720	745	775	800	820	835	850	860	860	860	860
	0000	008C	00E6	0145	019A	01E0	0221	0258	0285	02AD	02D0	02E9	0307	0320	0334	0343	0352	035C	035C	035C	035C
269	0.0	599[d]	944	1284	1605	1850	2091	2280	2445	2563	2685	2794	2902	2978	3054	3115	3162	3195	3195	3195	3195
39	00	175	285	395	500	580	660	725	780	820	860	895	930	955	980	1000	1015	1025	1025	1025	1025
	0000	00AF	011D	018B	01F4	0244	0294	02D5	030C	0334	035C	037F	03A2	03BB	03D4	03E8	03F7	0401	0401	0401	0401
324	0.0	732	1100	1496	1836	2147	2398	2638	2822	2978	3115	3242	3370	3464	3563	3625	3691	3705	3705	3705	3705
47	00	215	335	465	575	680	765	845	905	955	1000	1040	1080	1110	1140	1160	1180	1185	1185	1185	1185
	0000	00D7	014F	01D1	023F	02A8	02FD	034D	0389	03BB	03E8	0410	0438	0456	0474	0488	049C	04A1	04A1	04A1	04A1

a. kilo-Pascals gage
psi gage

b. Volumetric Flow Rate (SCCS)
Rotameter_{10}
Rotameter_{16}

c. Reading below Rotameter scale, value extrapolated
d. Air spring phenomena. Steady value extrapolated

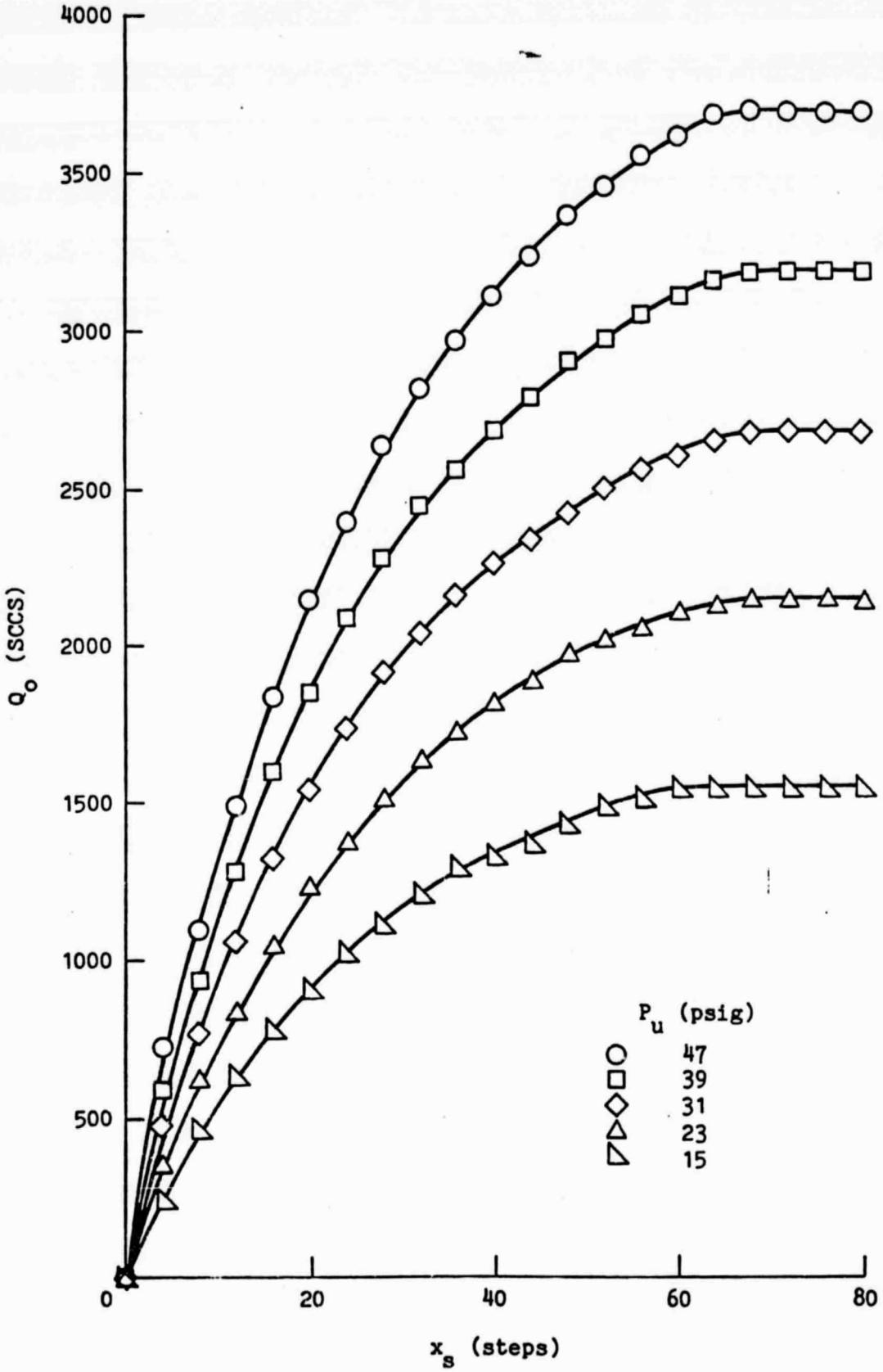

Fig. 14 Volumetric flow rate calibration curves.

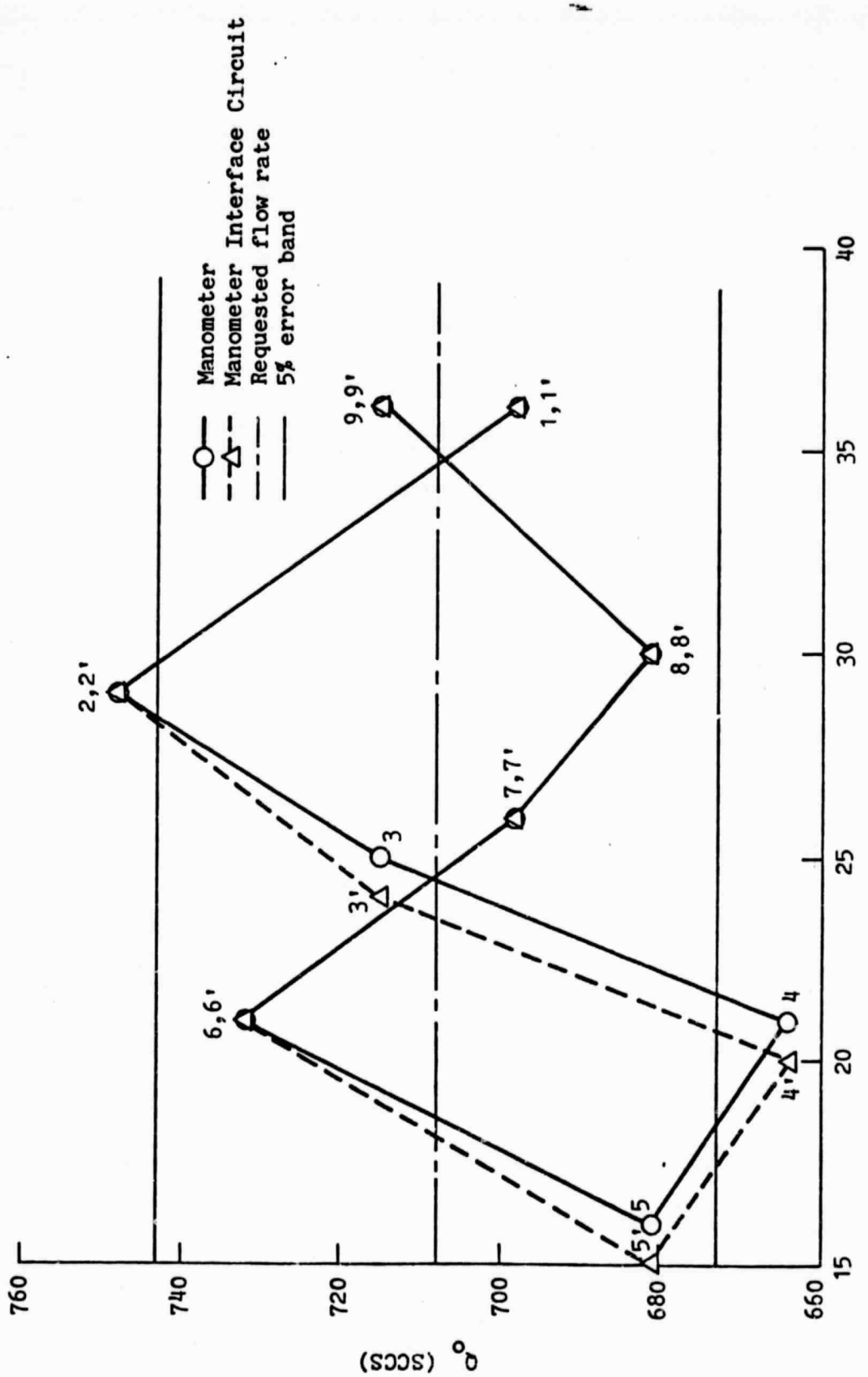

Fig. 15(a) Output volumetric flow rate vs. upstream pressure; Q_r = 708 SCCS.

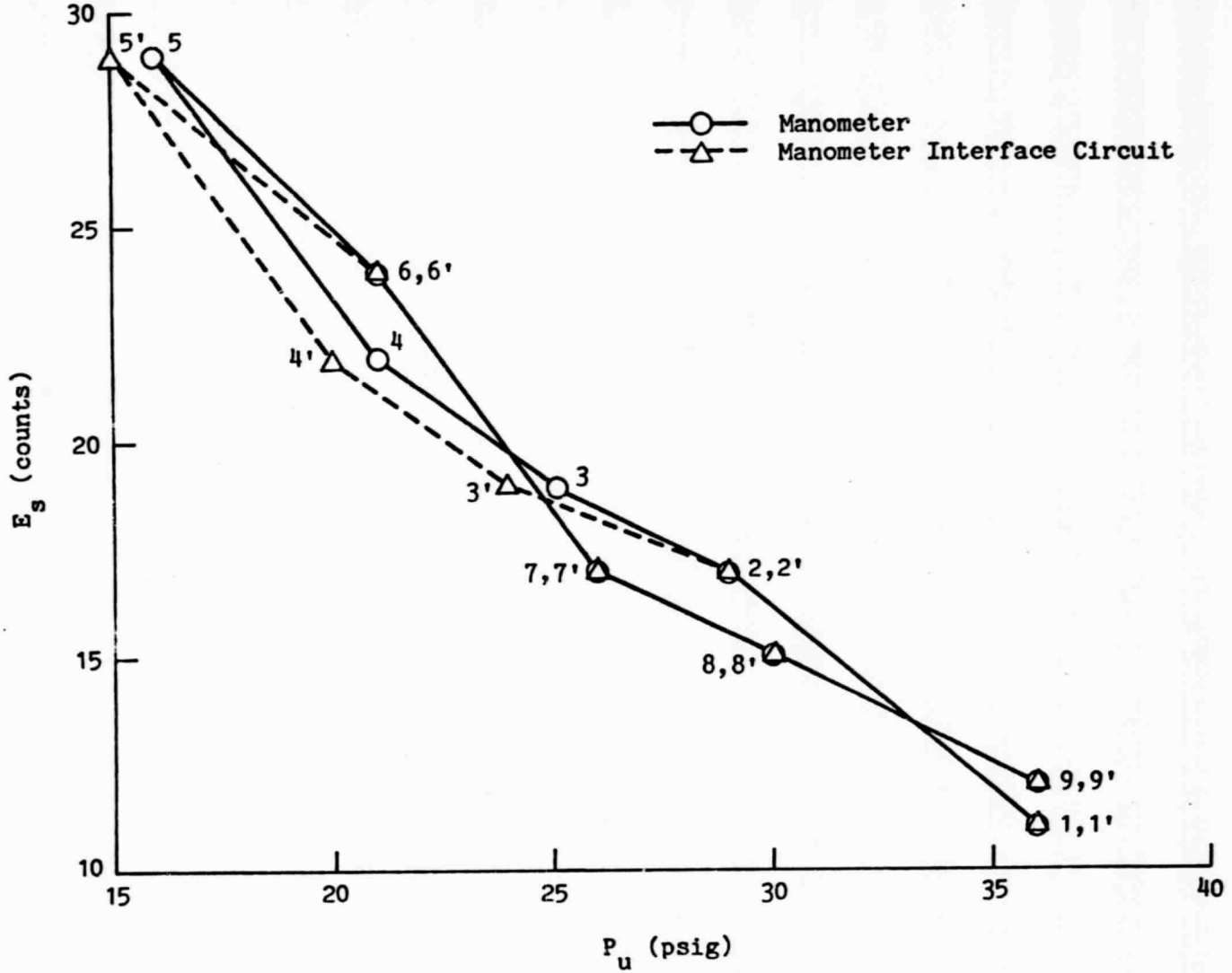

Fig. 15(b) IOLE output vs. upstream pressure; Q_r = 708 SCCS.

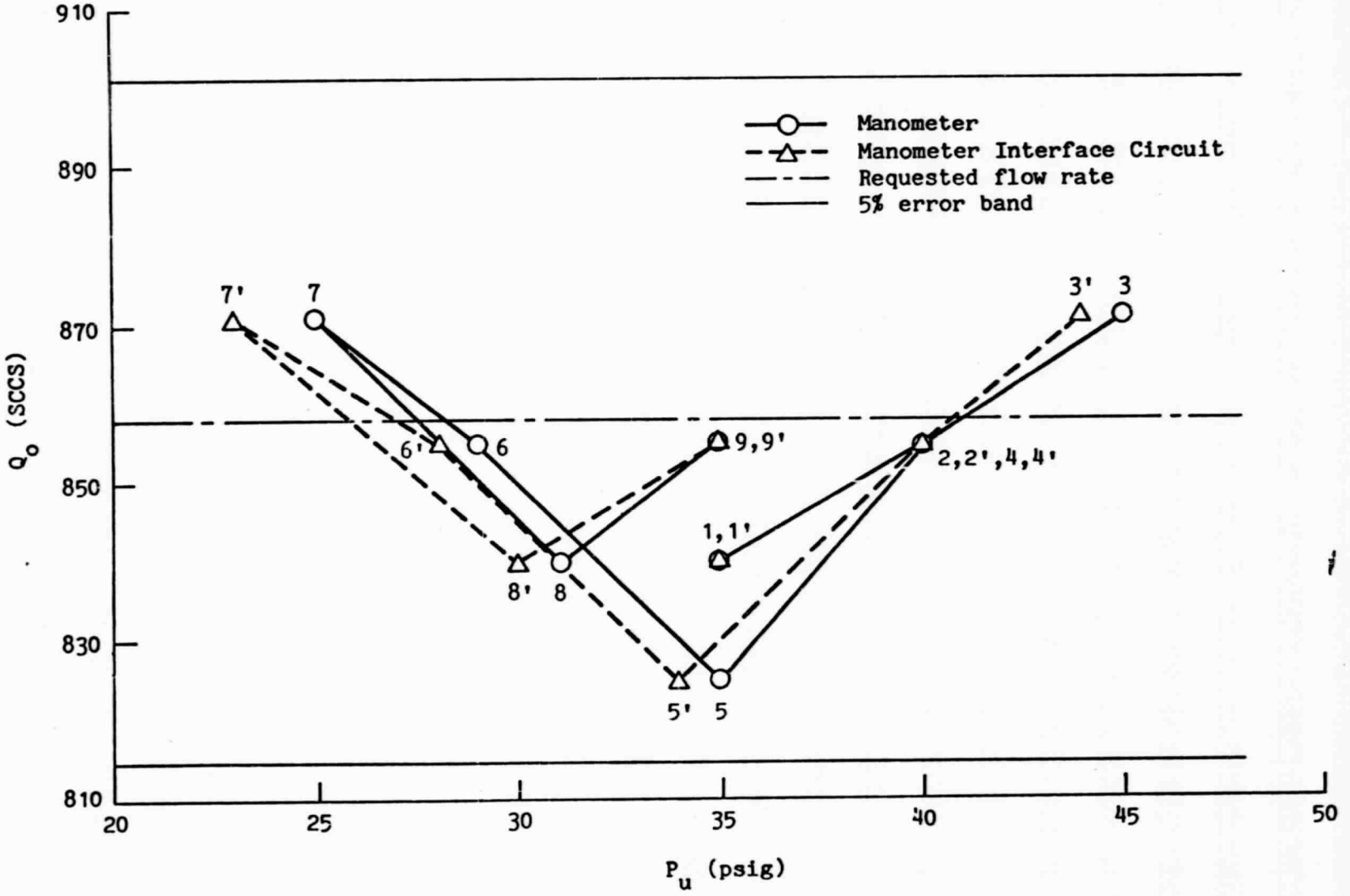

Fig. 16(a) Output volumetric flow rate vs. upstream pressure; Q_r = 858 SCCS.

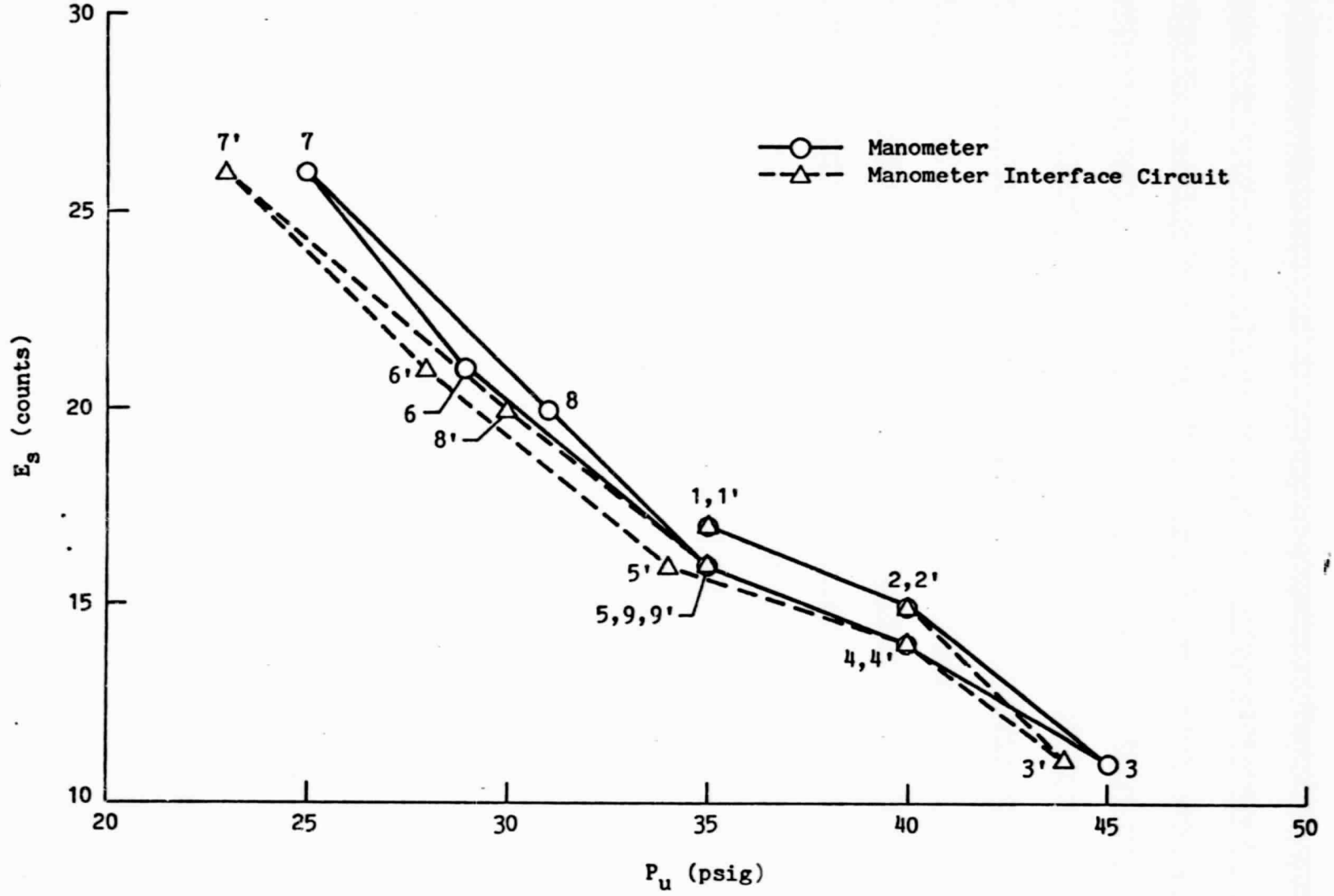

Fig. 16(b) IOLE output vs. upstream pressure; Q_r = 858 SCCS.

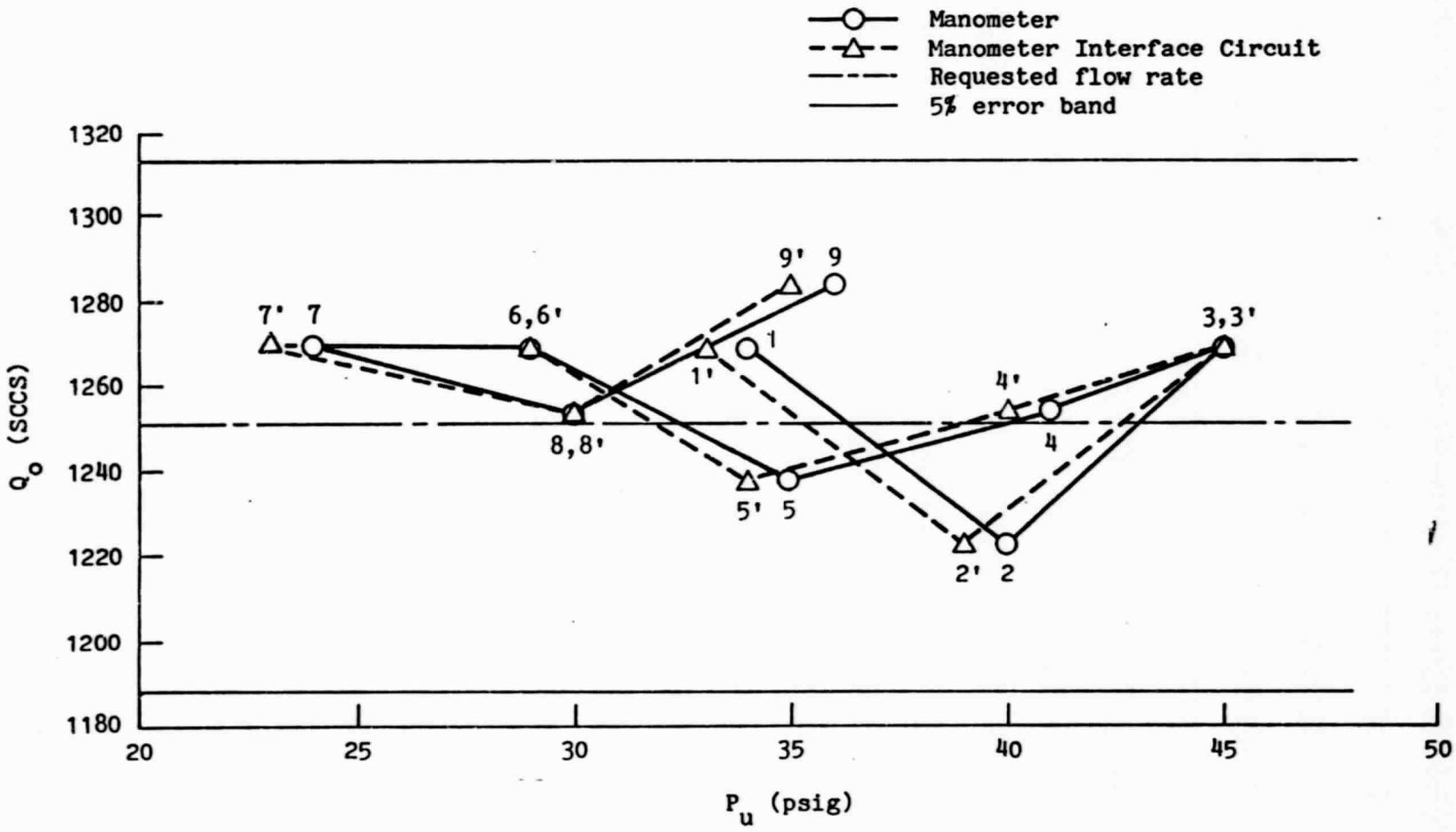

Fig. 17(a) Output volumetric flow rate vs. upstream pressure; Q_r = 1251 SCCS.

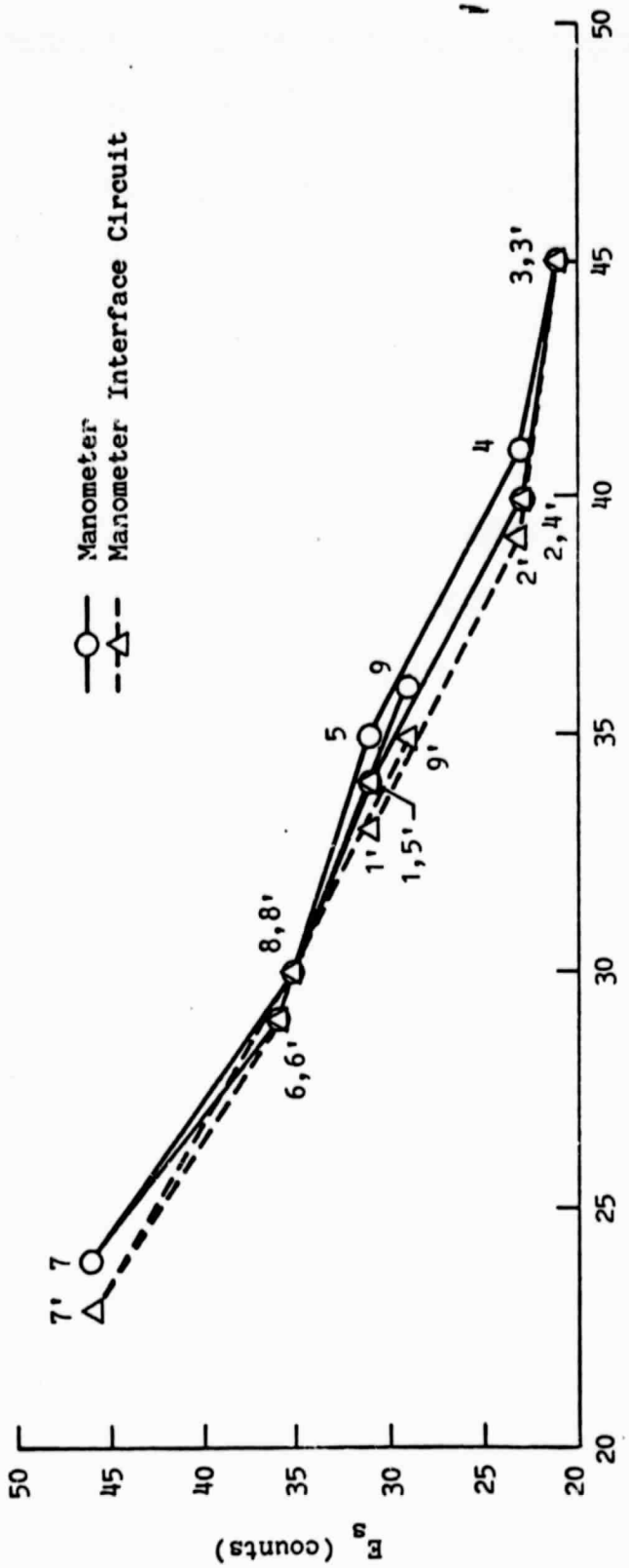

Fig. 17(b) IOLE output vs. upstream pressure; Q_r = 1251 SCCS.

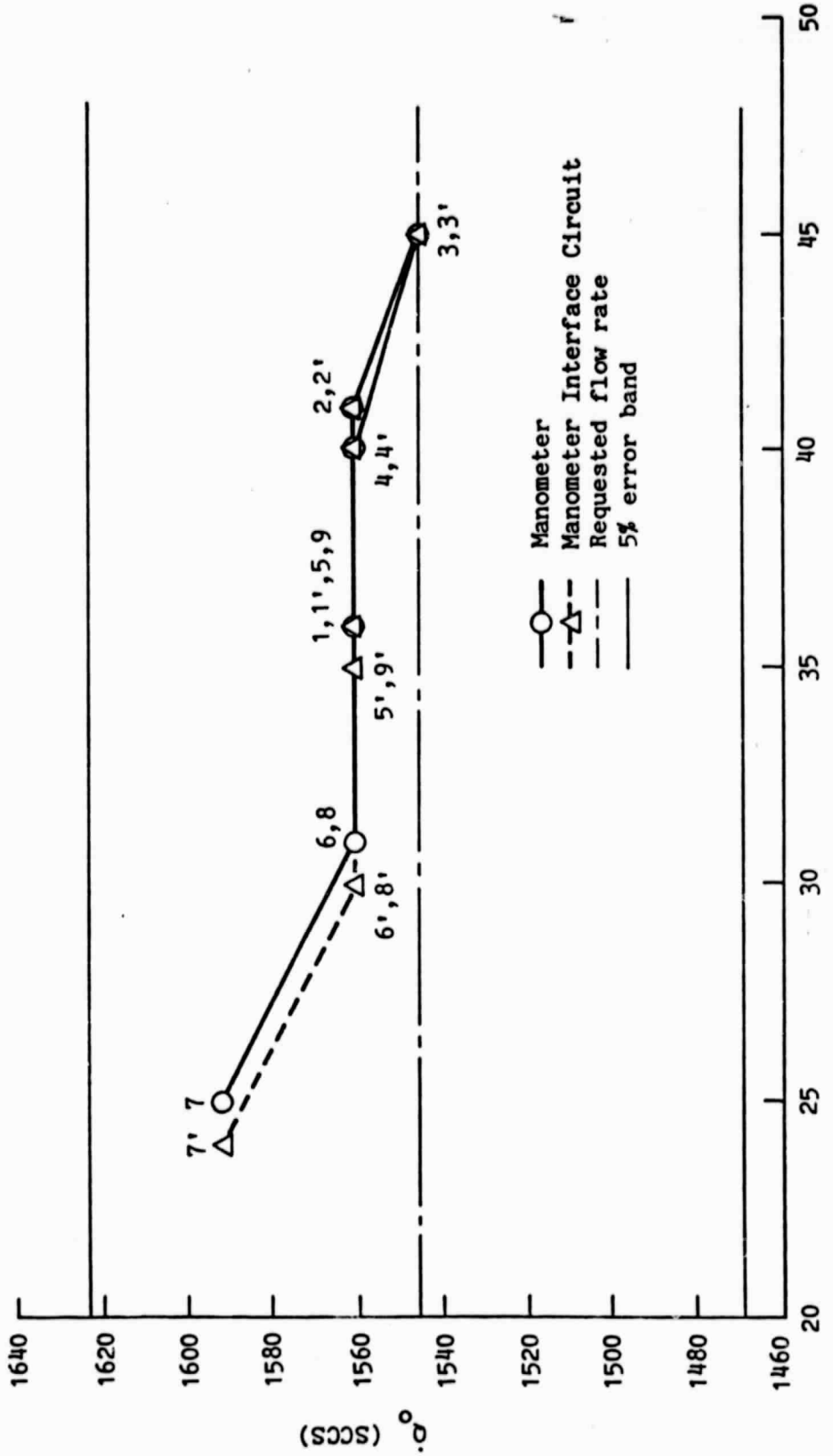

Fig. 18(a) Output volumetric flow rate vs. upstream pressure; Q_r = 1543 SCCS.

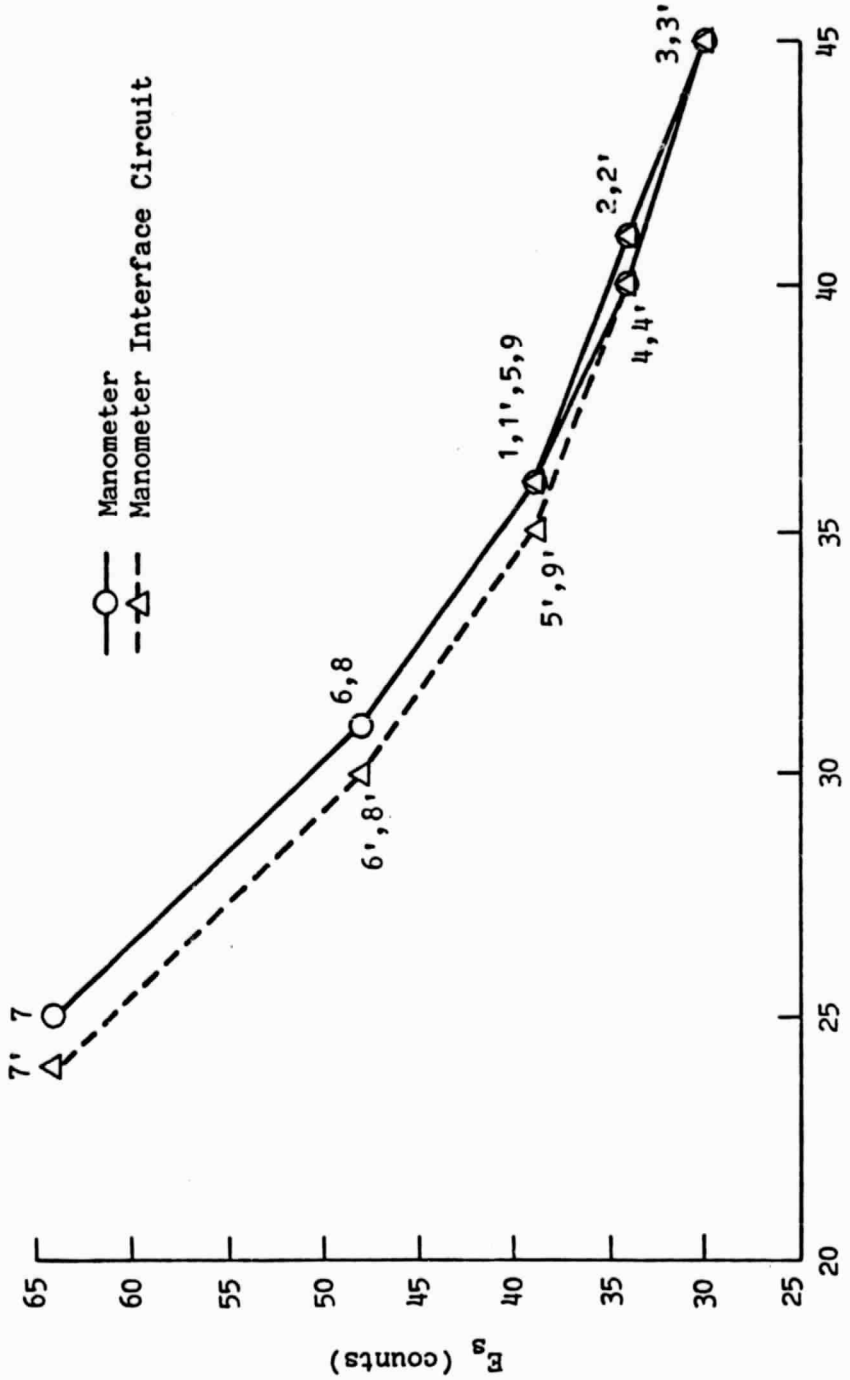

Fig. 18(b) IOLE output vs. upstream pressure; Q_r = 1543 SCCS.

sequence. This was due to the poor resolution for valve stem displacement near the closed position (see Fig. 14). For this reason, the lower limit of volumetric flow rate was set at 779 SCCS. Also, so the Overall Program would be valid over the entire range of possible upstream pressures, the upper limit was set at 1543 SCCS. (This is the value of the maximum flow rate at 15 psig).

3.9 Final Testing

With the testing of the overall assembly successful, the final testing of the assembly for 100 hours began. To monitor the assembly during the testing, an APPLE IIe microcomputer was interfaced with the system through a CYBORG model 91A Isaac Data Acquisition System. Four parameters were monitored by the APPLE/CYBORG System: (1) The output from the Manometer Interface Circuit; (2) the number of steps the linear actuator moved; (3) the actuator direction control bit, output on the data bus; and (4) the signal sent to the solenoid in the upstream assembly that diverted flow back to the regulating valve. Anytime the linear actuator moved, the APPLE IIe printed the date, time, and the measured parameters.

The 100 hour test was divided into two parts. Part A consisted of maintaining the upstream pressure regulator at a constant setting and varying the requested volumetric flow rates. The requested values were changed approximately every half hour during the working day. During the night, the Isaac continued to monitor the system for any changes in parameters. The requested flow rates ranged from 779 SCCS to 1543 SCCS. Part B of the testing consisted of keeping the requested flow rate constant and varying the upstream pressure. The

two requested volumetric flow rates tested were 1071 SCCS and 1251 SCCS. A presentation of the data and the discussion are in Chap. 4.

Since the testing was continuous and the system responded to the requested flow rates to within $\pm 5\%$ at all but three data points, the testing was concluded. A photograph of the complete assembly is shown in Fig. 19. Due to the limitations of certain components of the assembly (to be discussed in Chaps. 4 and 5), it was decided that this was the best possible accuracy to be expected. Recommendations for future developments and research are summarized in Chap. 5.

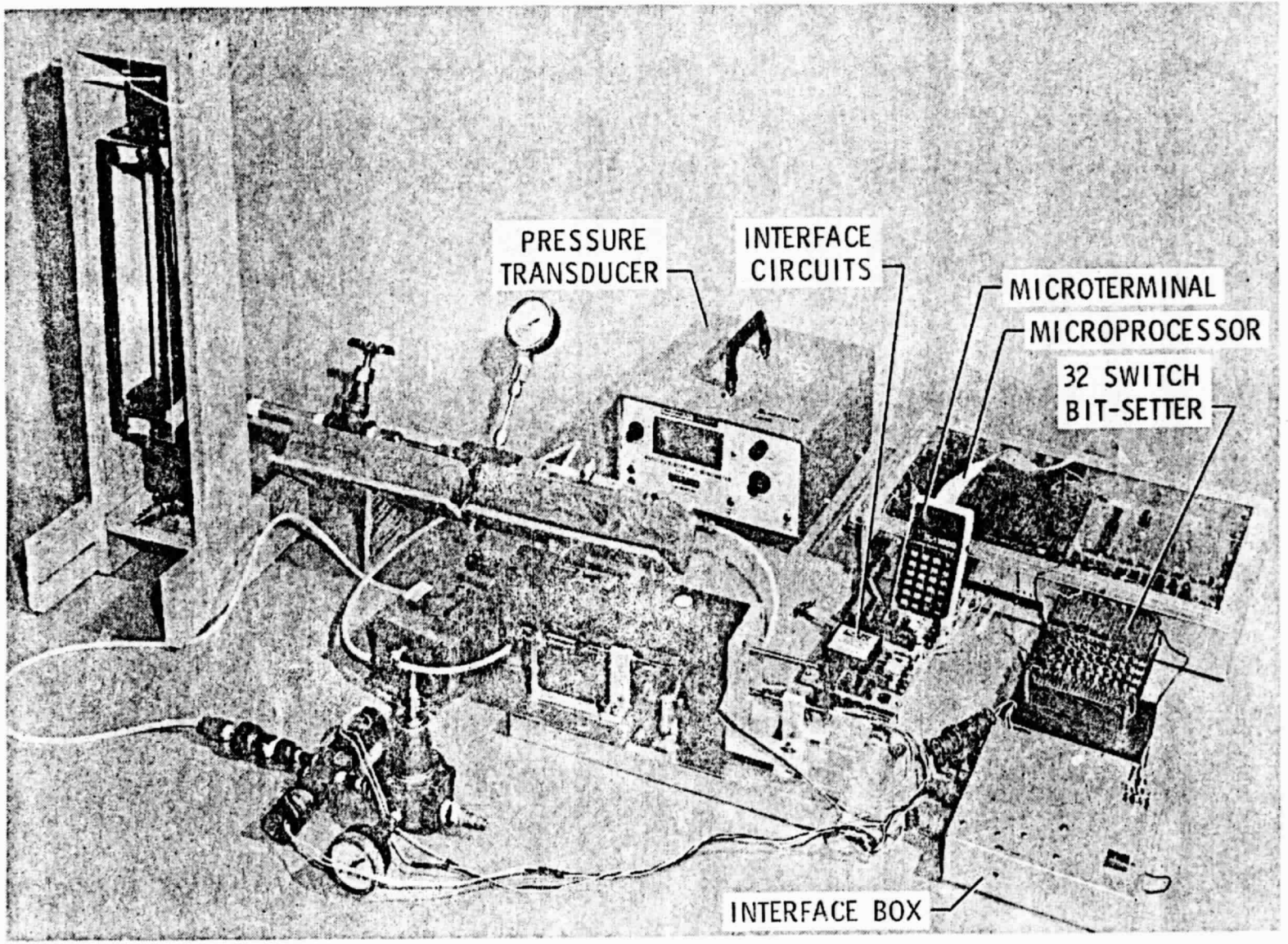

Fig. 19 Photograph of complete assembly.

ORIGINAL PAGE
BLACK AND WHITE PHOTOGRAPH

Chapter 4

RESULTS AND DISCUSSION

The discussion of the 100 hour test results can be divided into two parts: Part A, where the upstream regulating valve setting was held steady with various requested volumetric flow rates entered via the 32 switch bit-setter; and Part B, where the requested flow rate was held constant and the upstream pressure was varied.

4.1 Part A of the 100 Hour Test

The purpose of Part A of the 100 hour test was to verify that the developed assembly was capable of delivering a requested volumetric flow rate. Also, that the system was capable of operating for an extended period of time. Shown in Fig. 20(a) is the delivered volumetric flow rate (Q_o) versus the requested value (Q_r). The delivered flow rates were all within $\pm 5\%$ of the requested values. The scatter of the Q_o values was due to a combination of three factors: (1) The Manometer Interface Circuit performed the ADC of the manometer output with a resolution of only one psi. Since the manometer had an accuracy of 0.01 psi, this resulted in a round-off of the circuit generated reading. This round-off meant the actual upstream pressure was within plus or minus 0.5 psi of the circuit value. Consequently, the delivered flow rate was either higher or lower than that requested, depending on the actual upstream pressure. (2) The Manometer Interface Circuit had a constant fluctuation of plus or minus one psi. Because the source of this error

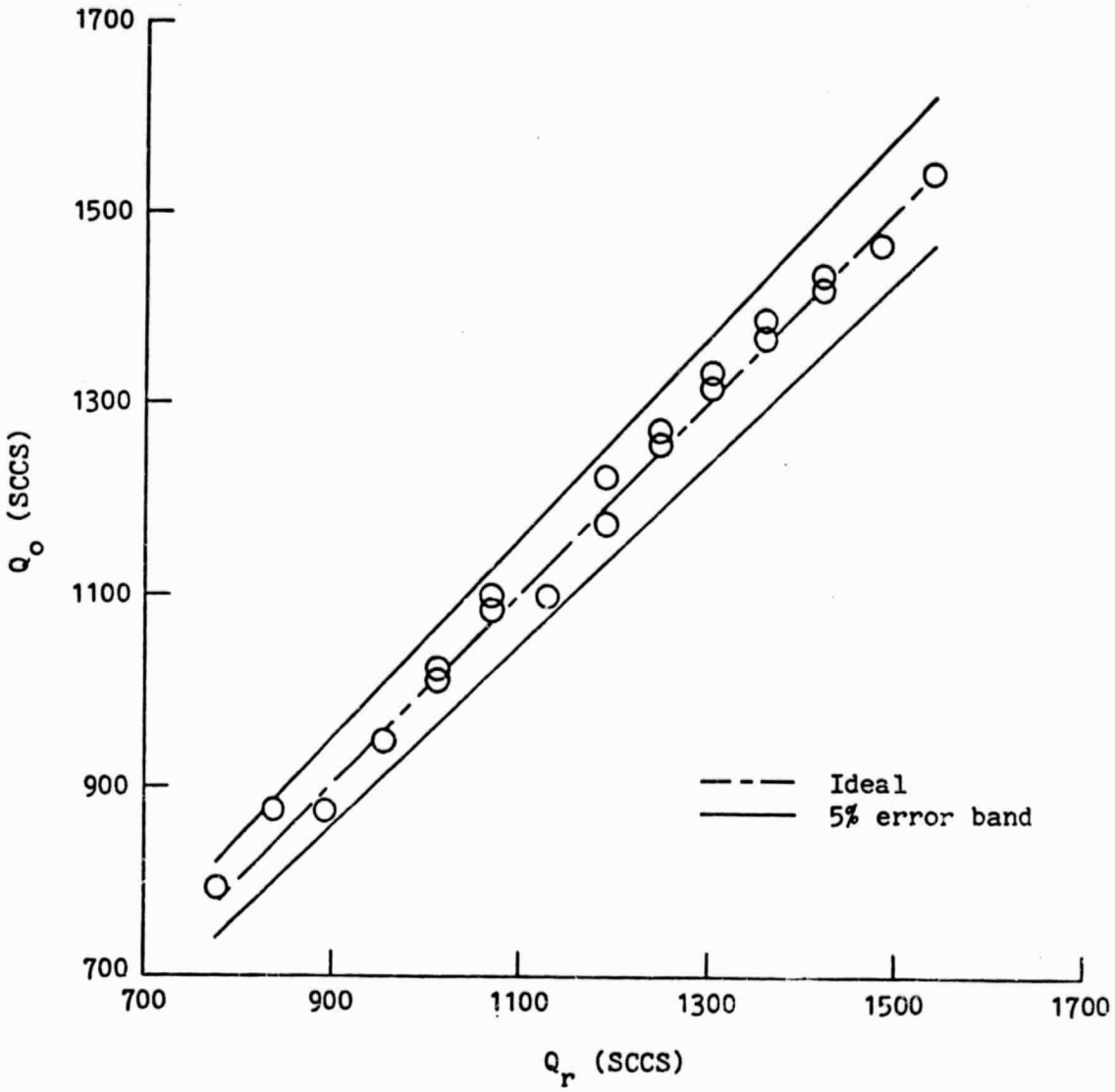

Fig. 20(a) Actual volumetric flow rate vs. requested volumetric flow rate. 100 hour test; Part A.

was never discovered, the Overall Program was altered to compensate. The programming alteration was to prevent this fluctuation from causing the assembly to cycle continuously, thereby making it unstable. The microprocessor system was programmed not to accept any upstream pressure readings, from the circuit, that were within one psi of the value from the previous scan. Because of this, the value of upstream pressure accepted by the microprocessor system could be off by one psi. Since this reading was used to calculate the correct valve stem displacement, the error could cause the stem to be incorrectly positioned. This added to the scatter of delivered flow rates. (3) The linear actuator moved a discrete distance with each step. With only 80 displacements, the number of possible volumetric flow rates was limited. This meant that although a particular flow rate may have been requested, it was not necessarily achievable. In such an instance, the system was designed to deliver the flow rate closest to the requested value. Calculated values of the correct displacements, in steps, are in agreement with these collected by the APPLE IIe microcomputer. It is believed that these errors, individually or in combination, are responsible for the scatter. The results shown in Fig. 20(b) are of the IOLE output versus the requested flow rate (Q_r). Unlike Figs. 15(b) - 18(b), the symbols used represent the direction of the valve stem movement; i.e. opening and closing. While the IOLE output had an error band of plus or minus three counts, the results show that the linear actuator moved in a controlled fashion. Calculations were again made for the correct valve stem displacements (in steps) using the control

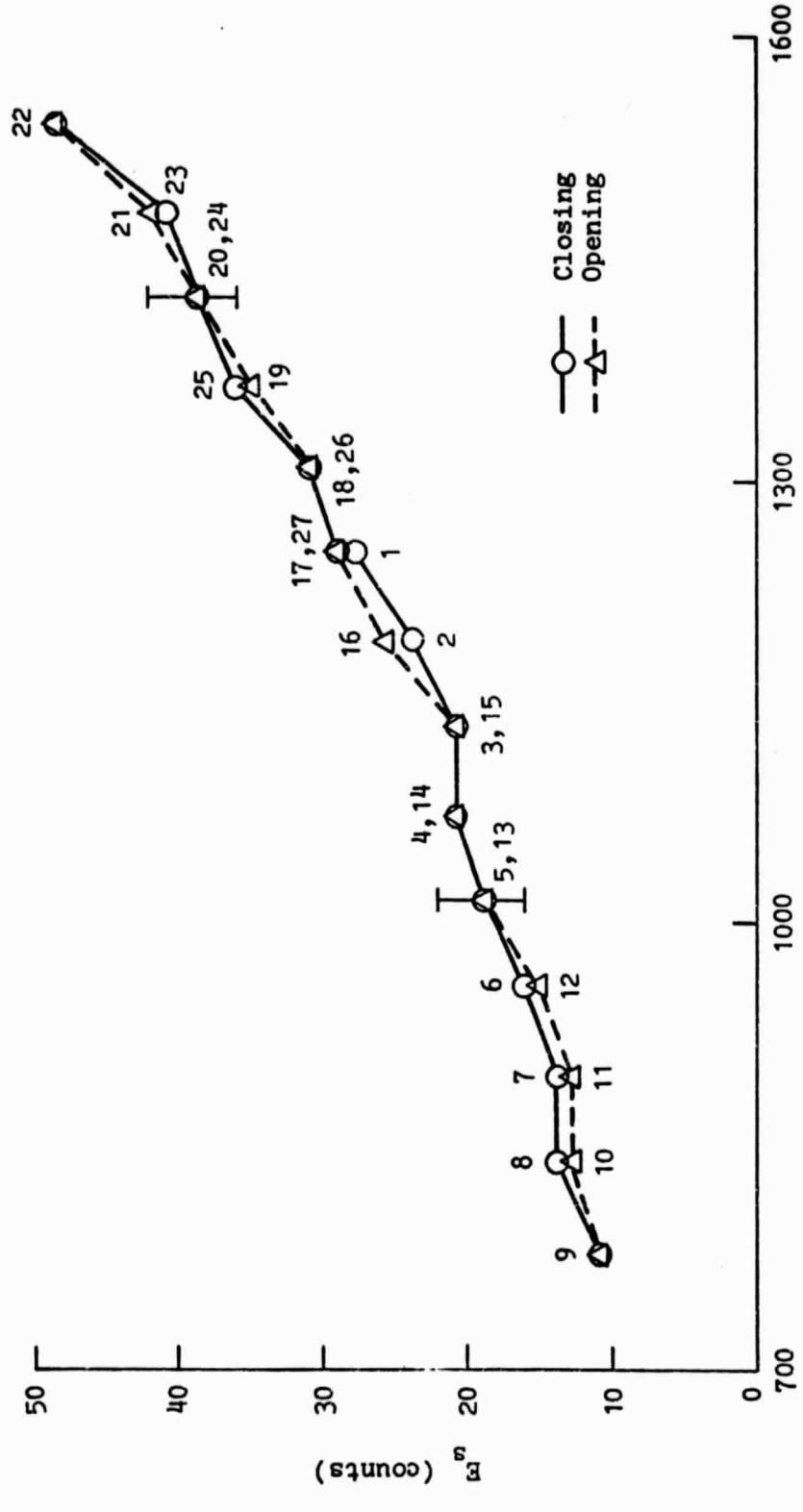

Fig. 20(b) IOLE output vs. requested volumetric flow rate. 100 hour test, Part A.

parameters of upstream pressure and requested flow rate. Comparison of these calculated displacements to those recorded by the APPLE IIe showed the microprocessor system correctly positioned the valve stem.

4.2 Part B of the 100 Hour Test

The purpose of Part B of the 100 hour test was to verify that the developed assembly was capable of maintaining a requested volumetric flow rate in the presence of upstream pressure fluctuations. The two flow rates tested were 1071 SCCS and 1251 SCCS.

Shown in Figs. 21(a) and 21(b) are the results of the 1071 SCCS test. As in Figs. 15-18, the data points are numbered to allow the sequence of testing to be followed. Again, the actual upstream pressure and the manometer signal converted by the Manometer Interface Circuit were not always equal. However, the delivered volumetric flow rates were all within $\pm 5\%$ of the requested value, and the IOLE output showed the linear actuator traveled along a repeatable path. As before, the calculated displacements were in agreement with the values recorded by the APPLE IIe microcomputer.

The results of the 1251 SCCS test are shown in Figs. 22(a) and 22(b). These graphs are similar to the previous ones except for three items: (1) In Fig. 22(a), between points two and three and between points five and six, there are two increases in Q_o without an accompanying change in P_u. This is due to the lack of resolution in the Manometer Interface Circuit. While a change in the upstream pressure of up to one psi would change the delivered flow

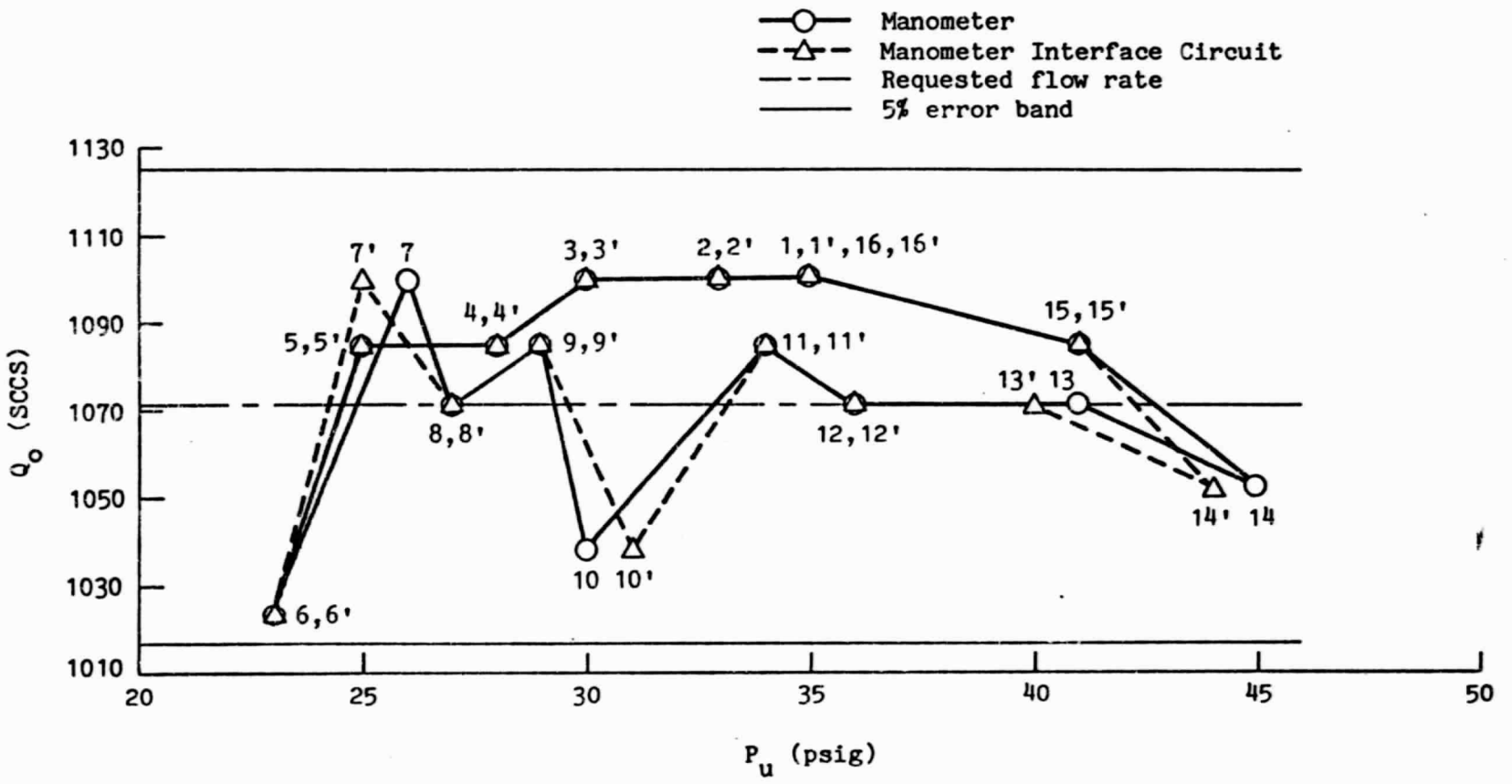

Fig. 21(a) 100 hour test, Part B. Output volumetric flow rate vs. upstream pressure; Q_r = 1071 SCCS.

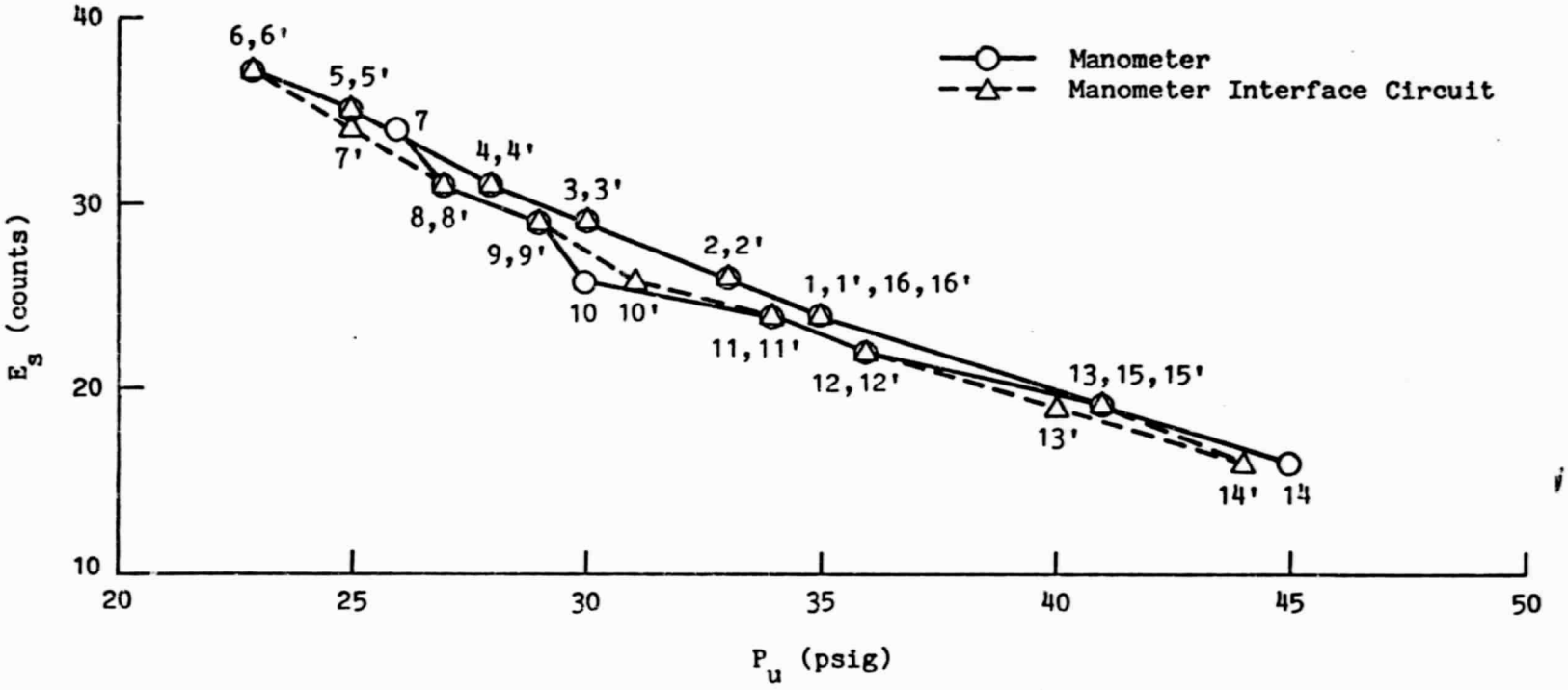

Fig. 21(b) 100 hour test, Part B. IOLE output vs. upstream pressure, Q_r = 1071 SCCS.

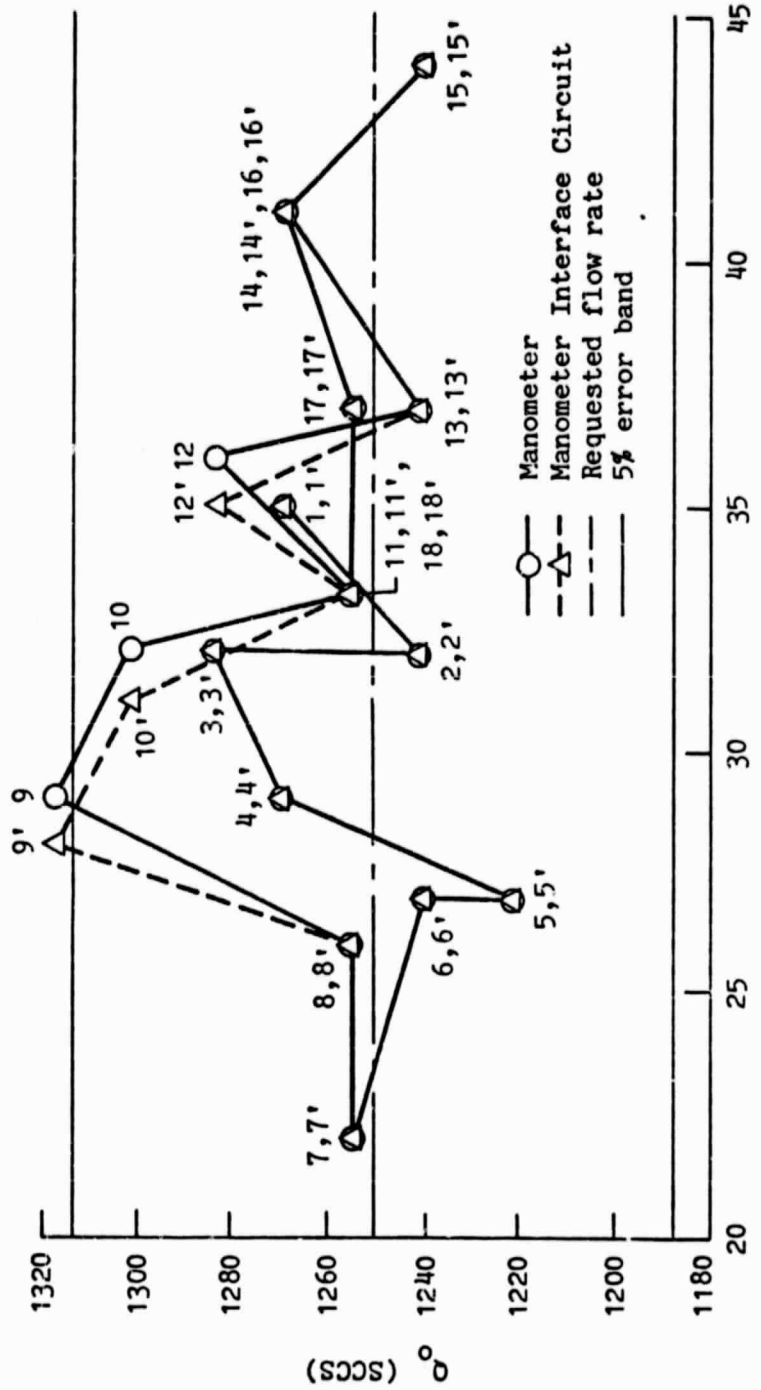

Fig. 22(a) 100 hour test, Part B. Output volumetric flow rate vs. upstream pressure; Q_r = 1251 SCCS.

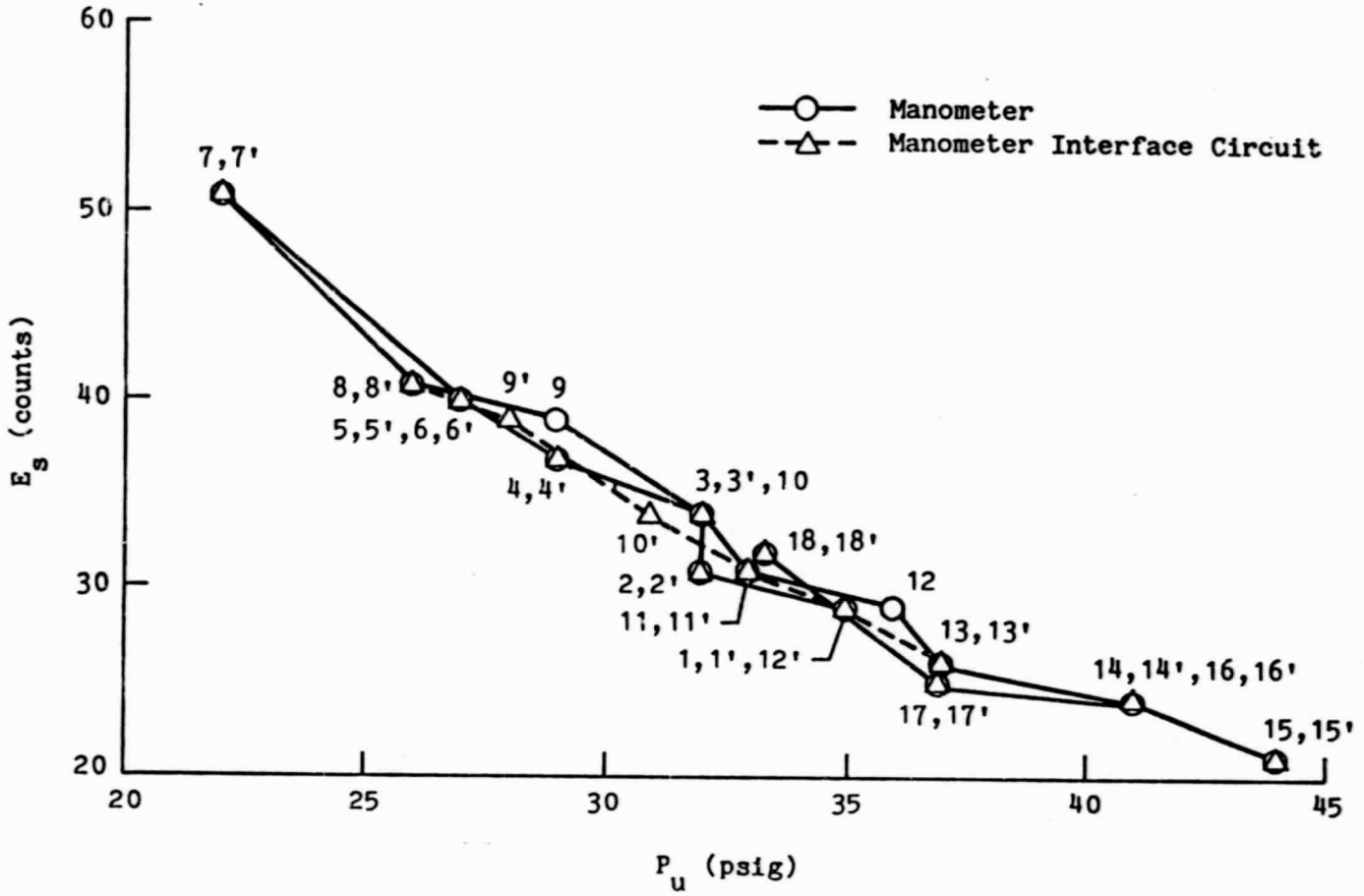

Fig. 22(b) 100 hour test, Part B. IOLE output vs upstream pressure; Q_r = 1251 SCCS.

rate, the upstream pressure reading of the circuit did not reflect this (e.g., a pressure between 37.5-38.4 psig appeared as 38 psig on the Manometer Interface Circuit). (2) Also in Fig. 22(a), point nine is outside the 5% error band. This was again due to the Manometer Interface Circuit. The microprocessor system received an upstream pressure reading of 28 psig while it was actually 29 psig. This one psi difference accounts for an error in the microprocessor calculated displacement of the valve stem and hence, the flow rate is 5.43% greater than the requested value. It should be noted that the value of the closest interpolated flow rate for that upstream pressure is 1275 SCCS; already 1.86% over the requested value. (3) In Fig. 22(b), the vertical rise between points two and three was due to the fluctuation of the Manometer Interface Circuit. To illustrate, assume the correct upstream pressure is 32 psig. The circuit would then read between 31-33 psig. But if the correct upstream pressure fluctuates to 32.5 psi, then round-off error by the circuit will cause the reading to be 33 ± 1 psig. This means when the reading fluctuates to 34 psig, the Overall Program would note the two psi difference between sequential readings. It would then compensate for this change by repositioning the valve stem. As the valve stem moves, the pressure drops due to the greater head loss. The resulting pressure could now drop to the original value 32 psig. Hence, a change in E_s without an accompanying change in P_u.

4.3 Compressibility

Because the flowing fluid is a compressible gas, compressibility effects must be considered. This is necessary because of the

dependency of the rotameter on the viscocity of the fluid being studied [6]. An analysis of the velocity of the air through the regulating valve shows the maximum value to be 59.75 m/sec for the valve fully open and P_u = 47 psig. This equates to a valve orifice Mach number of 0.2; but according to White [7] the effects of compressibility are negligible for velocities under 0.3 Mach. As this is the greatest velocity through the valve, compressibility effects can be neglected.

4.4 Power Requirement

The two solenoids in the upstream assembly and the linear actuator are each rated at 12 W (12V at 1A). The maximum power consumption of 13.3 W occurred during the valve stem movement phase of the operation, because the three 12 W components were never energized simultaneously. A power consumption of 1.3 W was required for normal operation of the system while the actuator was stationary.

Consequently, a single 24 W (12V at 2A) DC power source would be sufficient to drive the entire system. It should be noted that the circuitry for the operational amplifiers used in the system could be modified to operate with a single-sided 12V supply. Additionally, the 5V supply required to operate the RCA 1802 Evalution System can be eliminated since a 12V version of the system exists.

Chapter 5

CONCLUSIONS AND RECOMMENDATIONS

While certain components of this assembly performed below expectations (i.e., the Manometer Interface Circuit and the IOLE), the overall development was considered satisfactory for two reasons. First, experimental results demonstrated that an electro-mechanical proportional flow controller was feasible. Of the 88 delivered volumetric flow rates (Q_o) shown in Figs. 15-22, 98.9% of them were within $\pm$5% of the requested values. The assembly was able to reposition for changes in either upstream pressure or the requested flow rate. Second, this development has shown what areas of such a design need further research. These areas are the bellows spring of the regulating valve, the displacement sensor, and a compact volumetric flow rate measurement device. (It should be noted that the components used in this development were neither state-of-the-art, nor designed for this type of application).

5.1 The Bellows Spring

The regulating valve used in the development was rated to a pressure of 6.89 MPa. Because this was much greater than the system maximum operating pressure of 446 kPa, the bellows spring had a greater stiffness than was necessary. This excessive stiffness was the reason for the addition of the upstream assembly and the alterations to the Overall Program. Even when the valve was unpressurized, a force of 53 N was required to close the valve

completely. This meant only 31 N of linear force was available, from the actuator, to move the valve stem when the system was pressurized. With a bellows cross-sectional area of approximately 1.0 cm^2, the actuator was unable to reposition the valve stem for pressures over 267 kPa gage. Therefore, it is recommended, for future designs, the stiffness of the bellows spring be designed (or selected) to compliment the system maximum operating pressure (i.e., a valve operating at 446 kPa requires a bellows of much less stiffness than a valve operating at 6.89 MPa).

5.2 The Displacement Sensor

The IOLE used in this development was found to be too sensitive to vibrations from the actuator. The accuracy of the displacement sensor in a feedback control system is very important. Because an ADC would already be present to convert the output of the pressure sensor, it is felt that an LVDT, or another displacement sensor with comparable vibration insensitivity, could be used in future designs. Commercially available LVDT's were found that were relatively insensitive to vibration and capable of operating at cryogenic temperatures [3].

5.3 Volumetric Flow Rate Measurement Device

The developed assembly is an example of indirect controls. The delivered volumetric flow rate was monitored as a function of the upstream pressure and the valve stem displacement. This resulted in a delay by the control algorithm, each time a new displacement was calculated. The development of a compact volumetric flow rate measurement device would allow the assembly to be controlled by a

direct feedback signal. This, in turn, would eliminate the need for the complex control algorithm, the displacement sensor, and the pressure sensor. The system would then be able to operate as a quickly responding, continuous system; rather than as a regulator, which has a slower response.

5.4 Other Proportional Flow Controllers

Because the proportional flow controller was shown to be feasible, it is recommended that future research in this area include a mass flow rate controller, and an assembly actuated by a rotary stepping motor. (The concept of rotary motion had been considered in this development, but was not pursued due to constraints on time, manpower, and funds).

REFERENCES

1. Jackson, J.K., "Pulse-Modulated Dual-Gas Control Subsystem for Space Cabin Atmosphere," ASME Paper 74-39119, July/August 1974.

2. Duchaine, R.J. (ed)., Thomas Register of American Manufacturers and Thomas Register Catalog File, 71st ed., Thomas, New York, 1981.

3. Heiserman, D.L., Handbook of Digital IC Application, Prentice-Hall, Englewood Cliffs, New Jersey, 1980, p.4.

4. RCA Corporation, User Manual for the CDP1802 COSMAC Microprocessor, RCA, Manual MPM 201, 1977, pp.7-8.

5. Holman, J.P., Experimental Methods for Engineers, 3rd ed., McGraw-Hill, New York, 1978, p. 219.

6. Ibid., p.232.

7. White, F.M., Fluid Mechanics, McGraw-Hill, New York, 1979, p. 513.

8. Herceg, E.E., Handbook of Measurement and Control, Schaevitz Engineering, Pennsauken, New Jersey, 1976, p. 5.12.

9. Ogata, K., Modern Control Engineering, Prentice-Hall, Englewood Cliffs, New Jersey, 1970, p. 346.

APPENDICES

APPENDIX A

CIRCUIT DIAGRAMS

The figures collected in this appendix are of the interface circuitry, and the control circuitry. The figures listed are:

A.1 IOLE Interface Circuit

A.2 (a) Circuit diagram of Interface Box

(b) Linear actuator power relay

(c) Relay circuit for diverting valve, closing solenoid

(d) Relay circuit for diverting valve, opening solenoid

A.3 Manometer Interface Circuit

A.4 Schematic of the eight switch bit-setter

A.5 Schematic of the 32 switch bit-setter

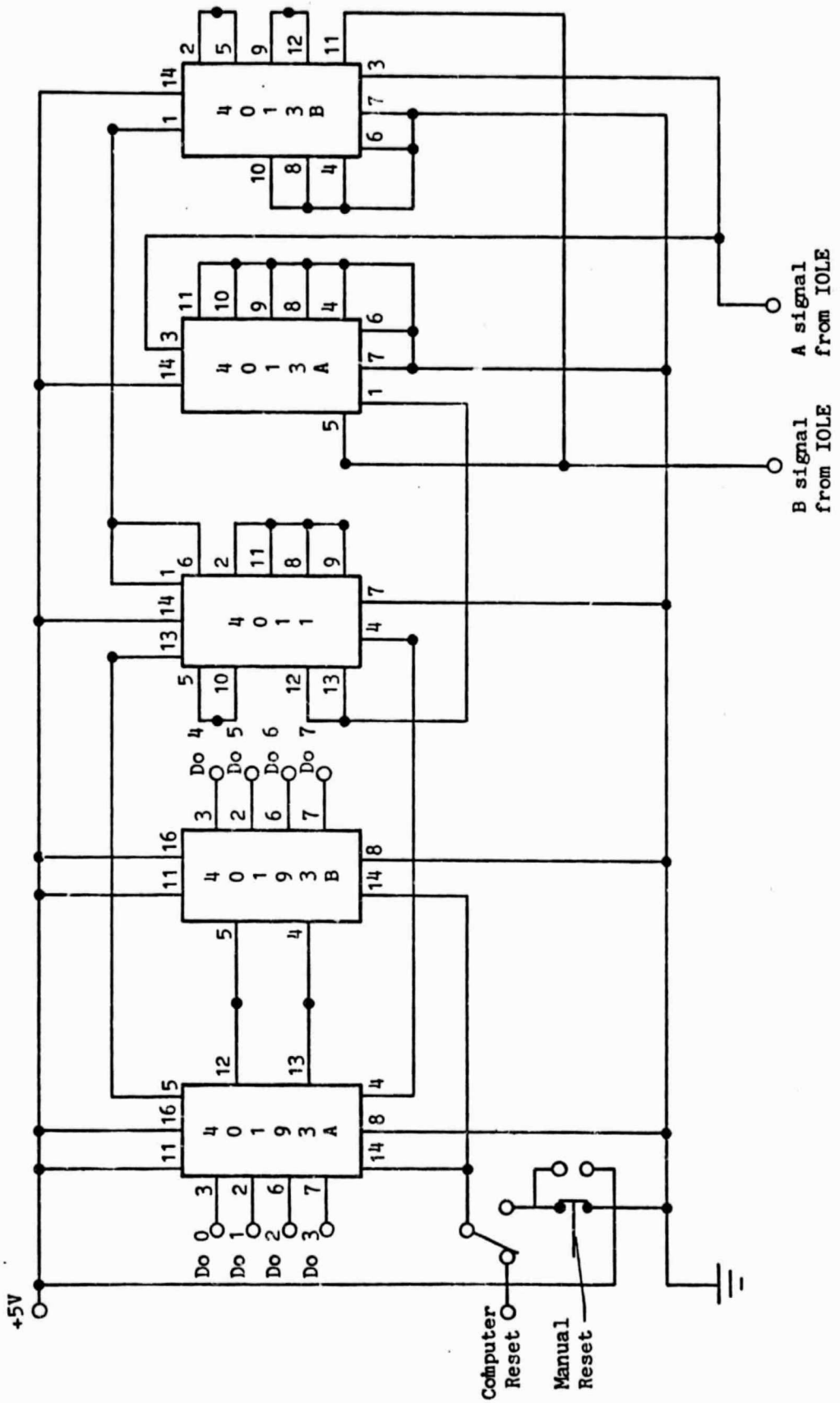

Fig. A.1 IOLE Interface Circuit.

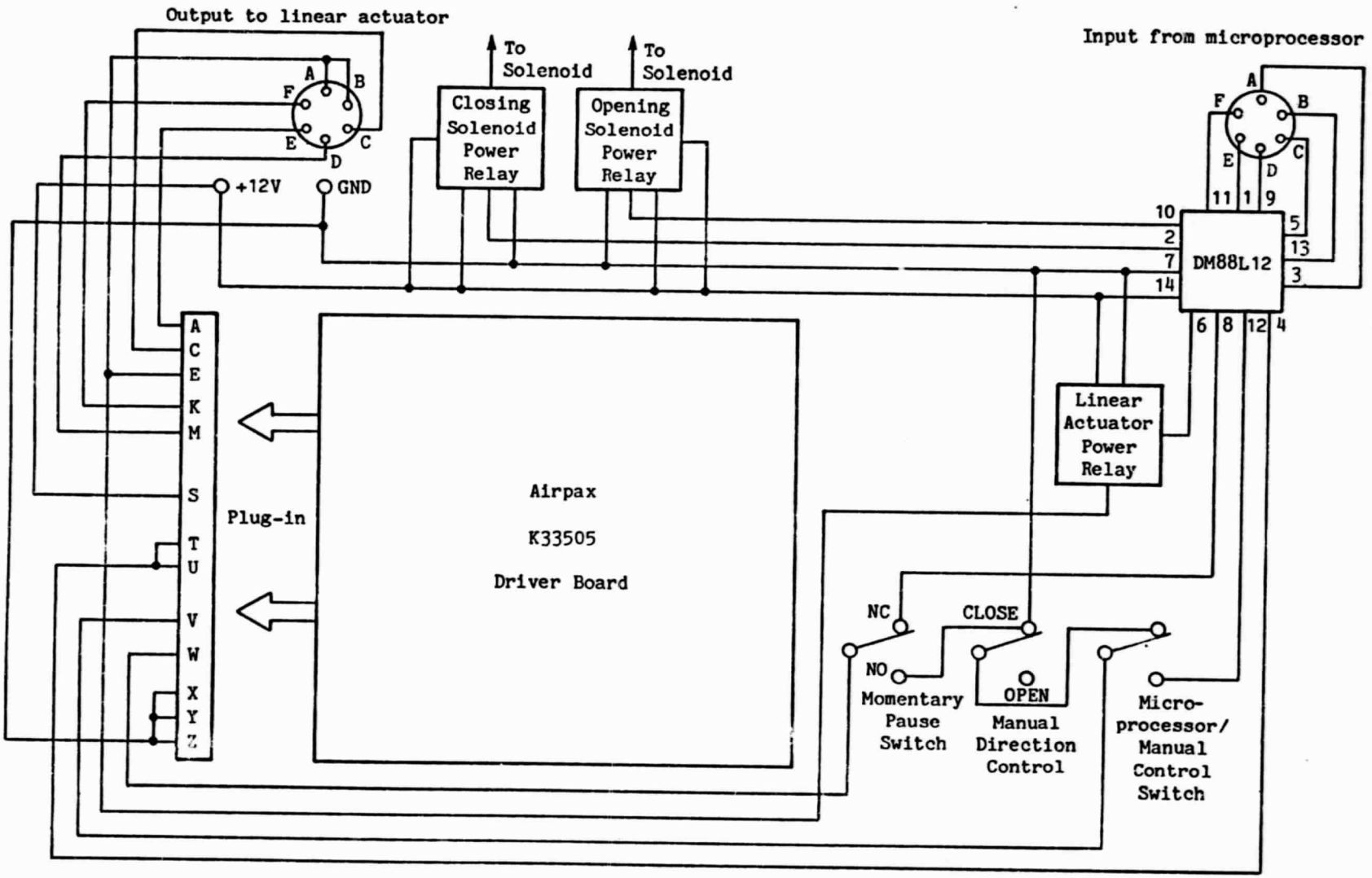

Fig. A.2(a) Circuit diagram of Interface Box.

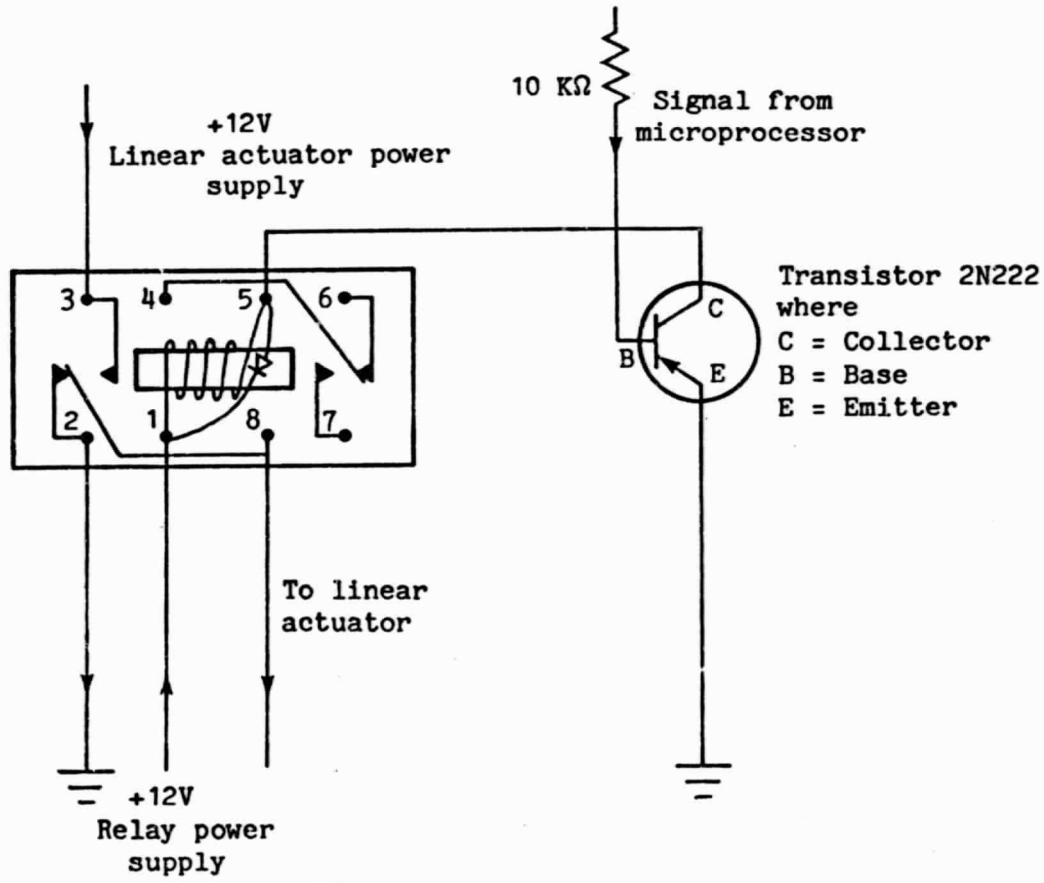

Fig. A.2(b) Linear actuator power relay.

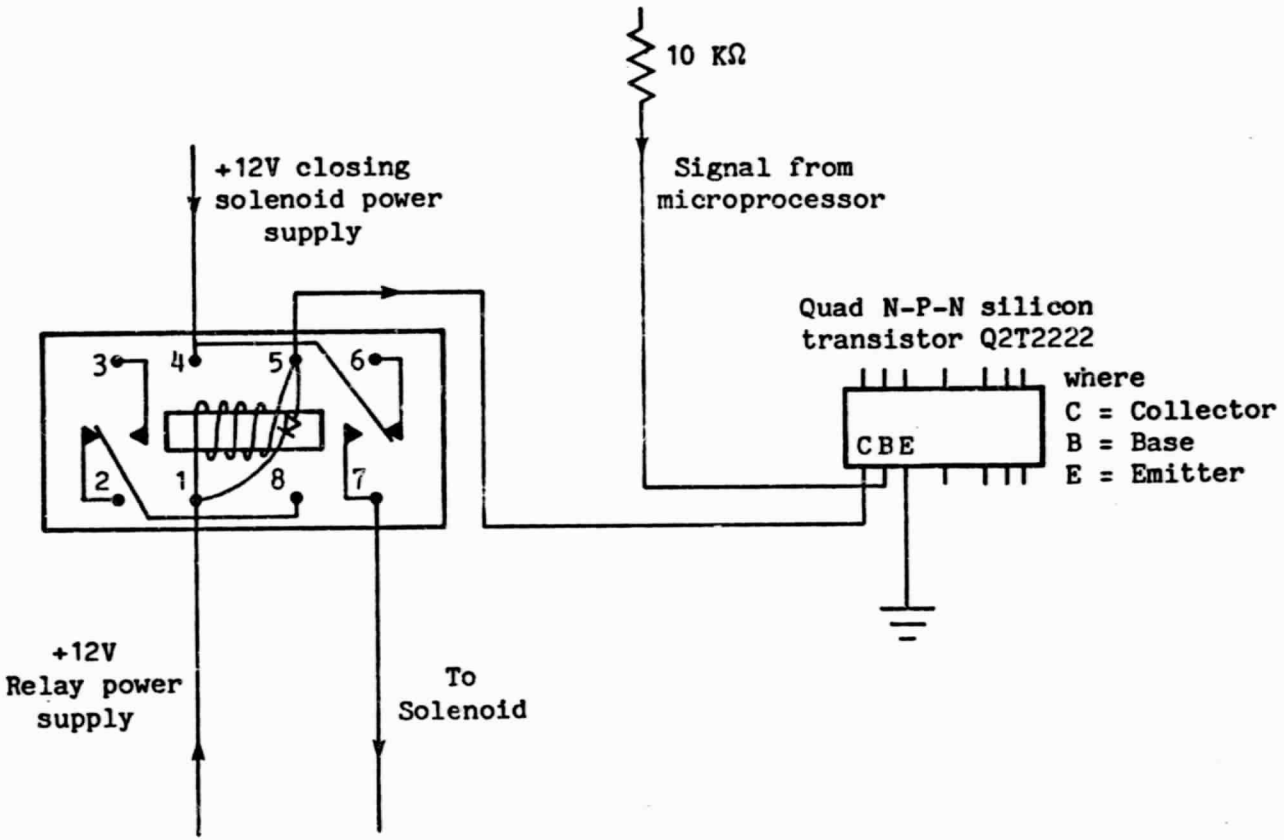

Fig. A.2(c) Relay circuit for diverting valve closing solenoid.

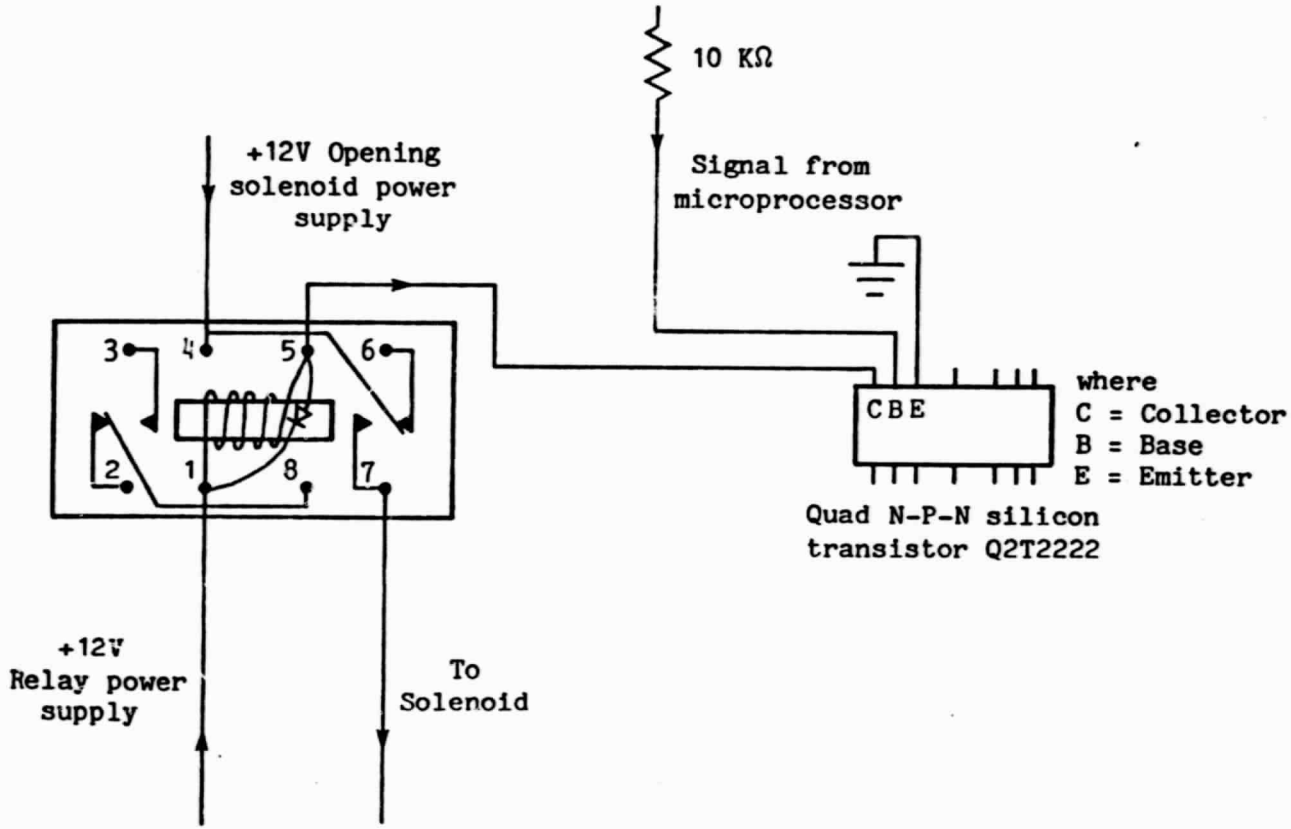

Fig. A.2(d) Relay circuit for diverting valve opening solenoid.

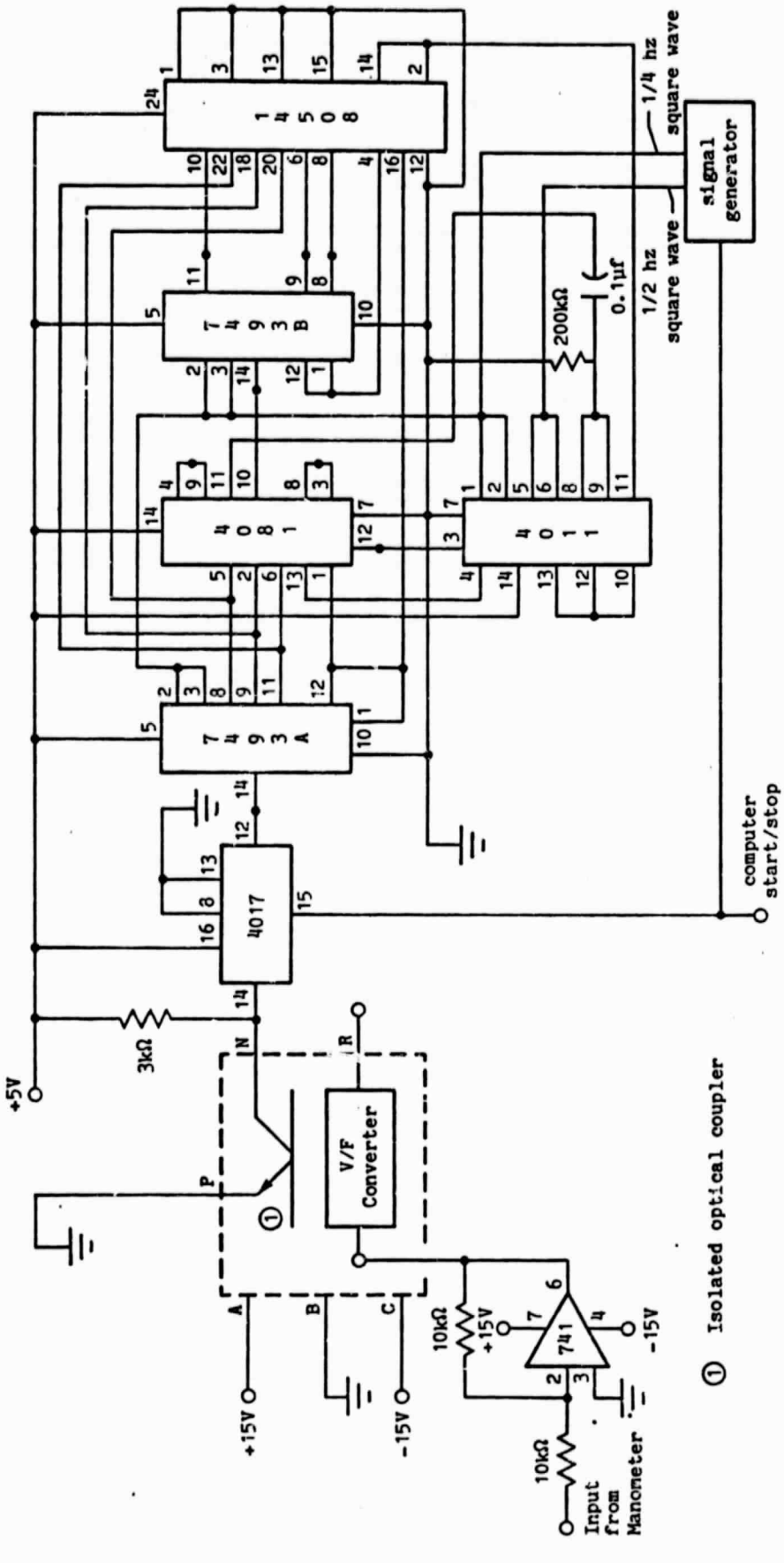

Fig. A.3 Manometer Interface Circuit.

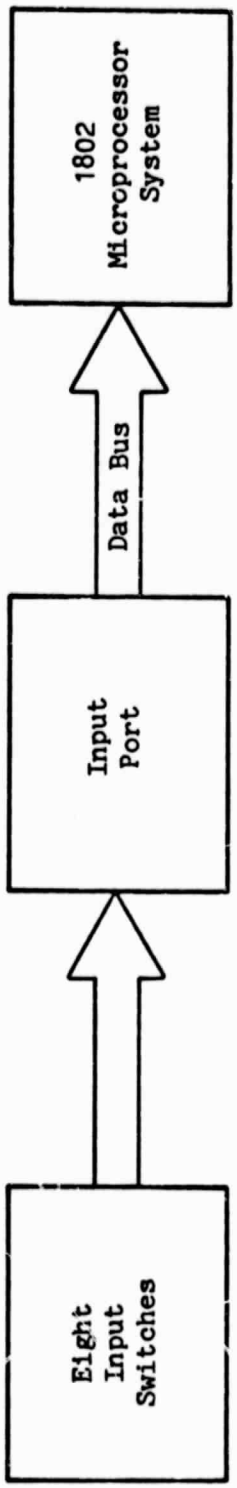

Fig. A.4 Schematic of eight switch bit-setter.

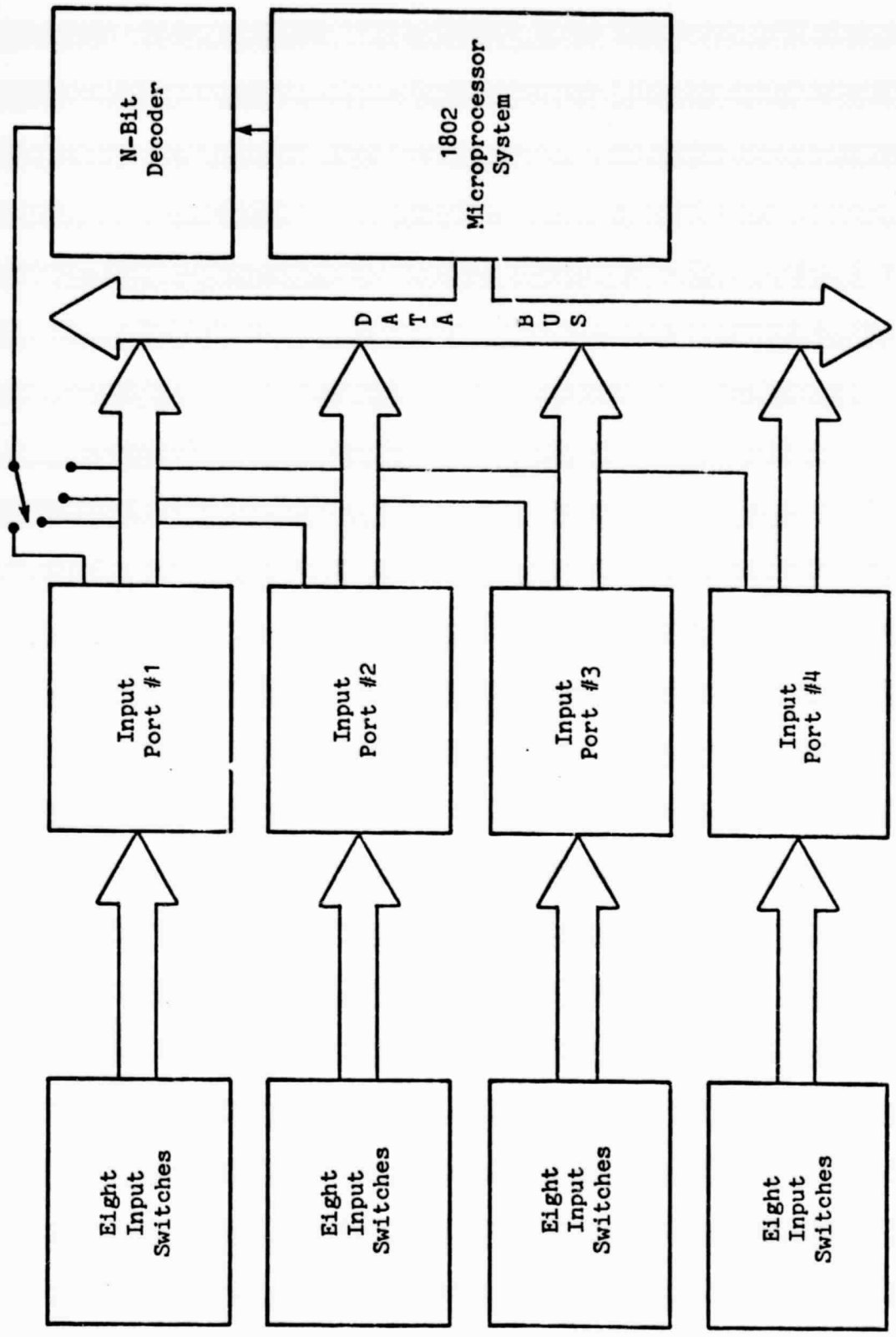

Fig. A.5 Schematic of 32 switch bit-setter.

C-2

APPENDIX B

DEVELOPMENT MICROPROCESSOR PROGRAMS

There were four microprocessor programs developed during the course of this development. They were the Cycling Program, the Actuator Movement Program, the Interpolation Program, and the Overall Program. The first three are listed in the following pages along with comments, to explain the progamming techniques utilized.

CYCLING PROGRAM

LABEL	OBJECT CODE	OPERAND	COMMMENTS
	DIS	#00	.. Disable interrupts
	LBR	ENTRY	..
	LDI	#00	
	STXD		
SET:	LDI	#00	.. Set RB = #0000
	PHI	RB	..
	PLO	RB	..
LOOP:	INP 6		.. Read input switches
	BZ	LOOP	.. Keep reading until input ≠ 0
	PHI	RA	.. Store input into RA hi and lo bytes
	PLO	RA	..
	SHR		.. Divide input by 8 and ..
	SHR		
	SHR		
	SHR		
	PHI	RA	.. Store in RA.1
	GLO	RA	.. Get input from RA.0
	SHL		.. Multiply by 8 and ..
	SHL		
	SHL		
	SHL		
	PLO	RA	.. Store in RA.0
	IRX		
	LDX		
	STXD		
	BNZ	ZY	.. If D ≠ 0 jump to ZY
	SEP	R4	.. Output #00 on data bus
	,A(OUTPUT)	;#00	
	SEP	R4	.. Output #01 on data bus
	,A(OUTPUT)	;#01	
	SEP	R4	
	,A(INCR)		.. Increment RB until RB = $\#0050_{16}$
	BNF	LOOP	.. If DF = 0, jump to LOOP
	IRX		
	LDI	#01	
	STXD		
	BR	SET	.. Go to SET
ZY:	SEP	R4	.. Output #02
	,A(OUTPUT)	;#02	..
	SEP	R4	.. Output #03
	,A(OUTPUT)	;#03	..
	SEP	R4	.. Increment RB until RB = $\#0050_{16}$
	OBJECT		

```
LABEL     CODE          OPERAND      COMMMENTS
          ,A(INCR)                   ..
          BNF           LOOP         .. If DF = 0, jump to LOOP
          IRX
          BR            SET

OUTPUT:   LDA           R6           *** Output Subroutine ***
          STXD                       .. Store number to be output
          IRX
          OUT 5                      .. Output number on data bus
          DEC           R2
          SEP           R4           .. Countdown RA during output for
                                        delay
          ,A(COUNT)
          SEP           R5           .. Return to calling program

INCR:     INC           RB           *** Increment RB Subroutine ***
          GLO           RB
          XRI           #50          .. Compare RB.0 to #50₁₆
          BNZ           XX           .. If RB.0 ≠ #50₁₆, jump to XX
          GHI           RB           .. Compare RB.1 to #00
          XRI           #00
          BNZ           XX           .. If RB.0 ≠ 00, jump to XX
          LDI           #01          .. Set DF = #01
          SHR
          SEP           R5           .. Return to calling program
XX:       LDI           #00          .. Set DF = #00
          SHR
          SEP           R5           .. Return to calling program

COUNT:    GHI           RA           *** Countdown RA Subroutine ***
          STXD                       .. Save RA on stack
          GLO           RA           ..
          STXD                       ..
DOWN:     DEC           RA           .. Decrement RA
          GLO           RA
          BNZ           DOWN         .. If RA.0 ≠ 0, jump to DOWN and
                                        repeat
          GHI           RA
          BZ            OVER         .. If RA.1 = 0, jump to OVER
          DEC           RA           .. Decrement RA
          BR            DOWN         .. Jump to DOWN
OVER:     IRX
          LXDA                       .. Restore RA
          PLO           RA           ..
          LDX                        ..
          PHI           RA           ..
          SEP           R5           .. Return to calling program

          OBJECT
```

```
LABEL     CODE      OPERAND        COMMMENTS

ENTRY:    LDI       #00            ***Standard Call and Return
                                       Technique ***
          PHI       R3
          LDI       #05            .. Load return address into R3
          PLO       R3

INIT:     LDI       A.1(CALL)      .. Load R4 with call address ..
          PHI       R4             ..
          PHI       R5             .. Load R5 with return address
          LDI       A.0(CALL)      ..
          PLO       R4             ..
          LDI       A.0(RETPGM)    ..
          PLO       R5             ..
          LDI       #8C
          PHI       R2             .. Load stack address into R2
          PHI       R7
          LDI       #1F
          PLO       R2
          PLO       RC
          SEX       R2             .. Set stack pointer as R2
          SEP       R3             .. Return to main program

EXITA:    SEP       R3             .. Return to calling program
CALL:     SEX       R2             .. Point to stack
          GHI       R6             .. Save R6 value
          STXD                     ..
          GLO       R6             ..
          STXD                     ..
          GHI       R3             .. Put R3 ..
          PHI       R6             .. Into R6
          GLO       R3             .. Both halves
          PLO       R6
          LDA       R6             .. And put subroutine address ..
          PHI       R3             .. Into R3
          LDA       R6
          PLO       R3
          BR        EXITA          .. Go to calling subroutine

EXITR:    SEP       R3             .. Return to calling program
RETPGM:   GHI       R6             .. Put R6 back ..
          PHI       R3             .. Into R3
          GLO       R6             ..
          PLO       R3             ..
          SEX       R2             .. Point to stack
```

```
LABEL   OBJECT CODE   OPERAND   COMMMENTS
        INC           R2        .. Recover ..
        LDXA                    .. R6
        PLO           R6        .. Lower half
        LDX
        PHI           R6        .. And upper half
        BR            EXITR     .. Go back to calling program
        END
```

ACTUATOR MOVEMENT PROGRAM

```
LABEL    OBJECT CODE    OPERAND      COMMENTS
         DIS            #00          .. Disable interrupts
         LBR            ENTRY        .. Set up standard call and
                                        return technique
MAIN:    LDI            #00          .. Set up registers (R8,RB,RA)
         PLO            RB           .. R8 = #0200 16
         PHI            RB           ..
         PLO            R8           .. RA = #0050 16
         PHI            RA           ..
         LDI            #02          .. RB = #0000
         PHI            R8           ..
         LDI            #50          ..
         PLO            RA           ..
         BR             START
PS:      SEP            R4
         ,A(READ)                    .. Read in displacement (in
                                        steps)
         GHI            RA
         STXD
         IRX
         GLO            RA
         XOR                         .. Compare present input to
                                        previous
         BZ             PR           .. If same, cut off actuator
                                        power
         SEP            R4
         ,A(CLOSE)                   .. Call CLOSE subroutine
         GHI            RA
         PLO            RA
         SEP            R4
         ,A(OPEN)                    .. Call OPEN subroutine
         BR             PS
PR:      SEP            R4
         ,A(OUTPUT)     ;#04         .. Cut off actuator power when ..
         BR             PS           .. it is not in use

START:   SEP            R4           .. Close valve all the way on ..
         ,A(CLOSE)                   .. start up to verify position
         LDI            #00

         PLO            RA

         BR             PS           .. Return to main program
READ:    LDI            #00          *** Data Read Subroutine ***
         SHR
```

LABEL	OBJECT CODE	OPERAND	COMMENTS
	INP 6		.. Read input displacement off data bus
	PHI	RA	
	SMI	#50	.. Compare input to $\#50_{16}$
	BDF	RX	.. If input larger than $\#50_{16}$, jump to RX
	BR	RZ	.. Otherwise, jump to RZ
RX:	LDI	#50	.. Disregard input and store $\#50_{16}$
	PHI	RA	
RZ:	SEP	R5	.. Return to main program
OUTPUT:	LDA	R6	*** Data Output Subroutine ***
	STXD		.. Take number to be output
	IRX		
	OUT 5		.. Output it on data bus
	DEC	R2	.. Continue to output number until ..
	GHI	R8	.. R8 is counted down to #0000
	STXD		..
	GLO	R8	..
	STXD		..
OX:	DEC	R8	..
	GLO	R8	..
	BNZ	OX	..
	GHI	R8	..
	BZ	OY	..
	DEC	R8	..
	BR	OX	..
OY:	IRX		.. When R8 is #0000, restore.
	LDXA		.. The original number in R8 ($\#0200_{16}$)
	PLO	R8	..
	LDX		..
	PHI	R8	..
	SEP	R5	.. Return to calling program
CLOSE:	GLO	RA	*** Valve Closing Subroutine ***
	ADI	#05	.. Add #05 steps to last displacement ..
	PLO	RA	..
CY:	SEP	R4	.. And close down valve before repositioning
	,A(OUTPUT)	;#03	.. Output #03
	SEP	R4	
	,A(OUTPUT)	;#02	.. Output #02
	SEP	R4	
	,A(INCR)		.. Increment RB to RA
	BNF	CY	

```
LABEL    OBJECT CODE    OPERAND      COMMENTS
         GLO            RA           .. Subtract #05 from last
                                        displacement
         SMI            #05
         PLO            RA
         LDI            #00          .. Set RB = #0000
         PLO            RB
         PHI            RB
         SEP            R4
         ,A(OUTPUT)     ;#0A         .. Output #0A16
         SEP            R5           .. Return to calling program

OPEN:    GLO            RA           *** Valve Opening Subroutine ***
         BZ             PT
PB:      SEP            R4
         ,A(OUTPUT)     ;#01         .. Output #01
         SEP            R4
         ,A(OUTPUT)     ;#00         .. Output #00
         SEP            R4
         ,A(INCR)                    .. Increment RB to RA
         BNF            PB
         BR             PU
PT:      SEP            R4
         ,A(OUTPUT)     ;#04         .. Output #04 to cut off actuator
                                        power
PU:      LDI            #00
         PLO            RB           .. Set RB = #0000
         PHI            RB
         SEP            R5           .. Return to calling program
INCR:    INC            RB           *** Incrementing RB to RA
                                        Subroutine ***
         GLO            RA
         STXD
         IRX
         GLO            RB
         XOR                         .. Compare RB to RA
         BNZ            PJ           .. If not equal, jump to PJ
         LDI            #01          .. Set DF = #01
         SHR                         ..
         SEP            R5           .. Return to calling program
PJ:      LDI            #00          .. Set DF = #00
         SHR                         ..
         SEP            R5           .. Return to calling program
ENTRY:   LDI            #00          *** Standard Call and Return
                                        Technique ***
         PHI            R3
         LDI            #05          .. Load return address into R3
         PLO            R3
```

```
LABEL     OBJECT CODE     OPERAND          COMMENTS

INIT:     LDI             A.1(CALL)        .. Load R4 with call address
          PHI             R4               ..
          PHI             R5               .. Load R5 with return address
          LDI             A.0(CALL)        ..
          PLO             R4               ..
          LDI             A.0(RETPGM)      ..
          PLO             R5               ..
          LDI             #8C
          PHI             R2               .. Load Stack address into R2
          PHI             R7
          LDI             #1F
          PLO             R2
          PLO             RC
          SEX             R2               .. Set stack pointer as R2
          SEP             R3               .. Return to main program

EXITA:    SEP             R3               .. Return to calling program
CALL:     SEX             R2               .. Point to stack
          GHI             R6               .. Save R6 value
          STXD                             ..
          GLO             R6               ..
          STXD                             ..
          GHI             R3               .. Put R3 ..
          PHI             R6               .. Into R6
          GLO             R3               .. Both halves
          PLO             R6
          LDA             R6               .. And put subroutine address ..
          PHI             R3               .. Into R3
          LDA             R6
          PLO             R3
          BR              EXITA            .. Go to calling subroutine
EXITR:    SEP             R3               .. Return to calling program
RETPGM:   GHI             R6               .. Put R6 back ..
          PHI             R3               .. Into R3
          GLO             R6               ..
          PLO             R3               ..
          SEX             R2               .. Point to stack
          INC             R2               .. Recover ..
          LDXA                             .. R6
          PLO             R6               .. Lower half
          LDX
          PHI             R6               .. And upper half
          BR              EXITR            .. Go back to calling program END
```

INTERPOLATION PROGRAM

LABEL	OBJECT CODE	OPERAND	COMMENTS
	DIS	#00	.. Disable interrupts
	LBR	ENTRY	.. Set up standard call and return technique
	LDI	#00	.. Set registers R7, RA, RB = #0000
	PLO	R9	..
	PLO	R7	..
	PHI	R7	..
	PLO	RA	..
	PHI	RA	..
	PLO	RB	..
	PHI	RB	..
	LDI	#08	.. Set R9 = $\#0800_{16}$
	PHI	R9	
	LDI	XX	.. Load requested lo byte (XX) ..
	PLO	R8	.. into R8.0
	LDI	YY	.. Load requested hi byte (YY) (YY) ..
	PHI	R8	.. into R8.1
	LDI	ZZ	.. Load system pressure (ZZ) ..
	PHI	R8	.. into RD.0
	GLO	R8	
	STR	R2	.. Put requested flow ..
	GHI	R8	.. rate on stack
	OR		.. Compare hi and lo bytes
	LBZ	VV	.. If both = 0, jump to VV
	LDI	#01	.. Set DF = #01
	SHR		
	GLO	RD	.. Subtract $\#1E_{16}$ from ..
	SMI	#1E	.. system pressure
	BDF	LP	.. If result > or = 0, jump to LP
	LDI	#FF	.. Set RA, RB = $\#FFFF_{16}$
	PLO	RA	.. Branch to VV as ..
	PHI	RA	.. default because system ..
	PLO	RB	.. pressure below table ..
	PHI	RB	.. range
	LBR	VV	..
LP:	GLO	RD	.. Subtract $\#46_{16}$ from ..
	SDI	#46	.. System pressure
	BDF	MM	.. If result > or = 0, jump to MM
	LDI	#FF	.. Set RA, RB = $\#FFFF_{16}$
	PLO	RA	.. Branch to VV as ..
	PHI	RA	.. default because system ..
	PLO	RB	.. pressure above table ..

```
LABEL   OBJECT CODE   OPERAND   COMMENTS
        PHI           RR        .. range
        LBR           VV        ..
MM:     GLO           RD
        STR           R2
        LDI           #01       .. Set DF = #01
        SHR
        LDN           R9        .. Load first table pressure
        SD                      .. Subtract from system pressure
        BNF           JJ        .. If result < = 0, jump to JJ
        BZ            JJ        ..
        GLO           R9
        STR           R2        .. Otherwise, increment table ..
        LDI           #2B       .. pointer to next pressure line
        ADD                     ..
        PLO           R9        ..
        BR            MM        .. Jump to MM
JJ:     LDA           R9        .. Store upper pressure
        STXD                    ..
KK:     GHI           R8        .. Compare hi byte ..
        STR           R2        .. requested flow rate to
        LDN           R9        .. hi byte table flow rates
        SD                      ..
        BNF           RR        .. If result < or =0, jump to PR
        BZ            RR        ..
        GLO           R7
        STR           R2        .. Increment R7.0 by #04
        LDI           #04       ..
        ADD                     ..
        PLO           #R7       ..
        INC           R9        .. Move table pointer to ..
        INC           R9        .. next 2 bytes of data
        BR            KK        .. Jump to KK
RR:     GLO           R8        .. Now search for ..
        STR           R2        .. 2 byte table value ..
        INC           R9        .. > requested value
        LDI           #01       .. Set DF = #01
        SHR                     ..
        LDN           R9
        SD
        STXD
        GHI           R8        .. Compare 2 byte ..
        STR           R2        .. requested flow rate to ..
        DEC           R9        .. 2 byte table flow rate ..
        LDN           R9        ..
        SDB                     ..
        IRX                     ..
        BNF           RZ        .. If result < 0, jump to RZ
        BNZ           TT        .. If result = 0 jump to TT
```

LABEL	OBJECT CODE	OPERAND	COMMENTS
	LDX		
	BNZ	TT	
	BR	RZ	.. Otherwise, jump to RZ
TT:	GLO	R7	
	STR	R2	.. Increment R7.0 by #04
	LDI	#04	..
	ADD		..
	PLO	R7	..
	INC	R9	.. Move table pointer
	INC	R9	.. to next 2 bytes
	BR	RR	
RZ:	LDA	R9	
	STXD		.. Store 2 byte flow ..
	LDN	R9	.. rate as upper limit
	STXD		..
	GLO	R9	..
	STR	R2	..
	GLO	RD	.. Test to see if at ..
	SMI	#1E	.. lowest table pressure
	BZ	RY	.. If so, jump to RY
	LDI	#2B	.. Otherwise, decrement table ..
	SD		.. pointer to next lowest ..
	PLO	R9	.. pressure line
RY:	DEC	R9	
	LDA	R9	.. Begin data collection ..
	STXD		.. at same displacement
	LDN	R9	.. Store flow rate ..
	STXD		.. data of lower limit
	GLO	R7	..
	SHR		..
	STR	R2	..
	LDI	#01	.. Set DF = #01
	SHR		
	GLO	R9	
	SM		
	PLO	R9	
	DEC	R9	
	DEC	R9	
	LDN	R9	.. Store lower pressure
	STXD		..
	LDN	R9	
	ADI	#05	.. Add #05 to lower ..
	STR	R2	.. Pressure
	GLO	RD	
	SD		.. Subtract system pressure
	BDF	TL	.. If result < 0, jump to TL
	SEP	R4	

```
LABEL   OBJECT CODE     OPERAND     COMMENTS

        ,A(HIPR)                    .. Call high pressure
                                       interpolation subroutine
        BR              QQ
TL:     SEP             R4          .. Call low pressure inter-
                                       polation subroutine ..
        ,A(LOPR)                    ..
QQ:     SEP             R4
        ,A(DISP)                    .. Display resulting inter-
                                       polated ..
        BR              QQ          .. flow rate at system pressure
        IRX                         .. Go to bottom of stack
        IRX                         ..
        IRX                         ..
        IRX                         ..
        IRX                         ..
        IRX                         ..
        GLO             R8
        STR             R2
        LDI             #01         .. Set DF = #01
        SHR
        GLO             RA          .. Subtract interpolated flow ..
        SD                          .. rate from requested one
        STXD                        ..
        GHI             R8          ..
        STR             R2          ..
        GHI             RA          ..
        SDB                         ..
        IRX
        BNF             XZ          .. If result < 0, jump to XZ
        BNZ             ZG          .. If result > 0, jump to ZG
        LDX                         ..
        BNZ             ZG          ..
        BR              XZ          .. Jump to XZ
ZG:     GLO             R9
        STR             R2
        GLO             RD
        SMI             #1E
        BZ              ZR
        LDI             #2B         .. Move table pointer
        ADD                         .. to upper pressure line
        PLO             R9          ..
ZR:     LDA             R9
        STXD
        GLO             R7          .. Find location of next ..
        STR             R2          .. 2 byte flow rate ..
        LDI             #04         .. from R7.0 accumulator
        ADD                         ..
        PLO             R7          ..
```

LABEL	OBJECT CODE	OPERAND	COMMENTS
	GLO	R7	
	SHR		
	STR	R2	
	GLO	R9	.. Move table pointer to ..
	ADD		.. next 2 byte flow rate
	PLO	R9	..
	BR	RZ	.. Jump to RZ
XZ:	GLO	R7	
	STR	R2	.. Have upper interpolated ..
	LDI	#04	.. flow rate, now move ..
	SD		.. back 2 data bytes to ..
	PLO	R7	.. get lower limit
	GLO	RD	..
	SMI	#1E	..
	LBZ	HH	..
	GLO	R9	..
	STR	R2	
	LDI	#28	.. Move table pointer ..
	ADD		.. to upper pressure
	PLO	R9	..
HH:	GHI	RA	
	PHI	RF	.. Store old interpolated ..
	GLO	RA	.. flow rate in RF
	PLO	RF	..
	LDA	R9	.. Begin process again ..
	STXD		.. at next lowest displacement
	GLO	R9	..
	STR	R2	
	GLO	R7	
	SHR		
	ADD		
	PLO	R9	
	LDA	R9	
	STXD		
	LDN	R9	
	STXD		
	GLO	RD	
	SMI	#1E	
	BZ	TR	
	GLO	R9	
	STR	R2	
	LDI	#2B	
	SD		
	PLO	R9	
TR:	DEC	R9	
	LDA	R9	
	STXD		
	LDN	R9	

LABEL	OBJECT CODE	OPERAND	COMMENTS
	STXD		
	GLO	R7	
	SHR		
	STR	R2	
	LDI	#01	.. Set DF = #01
	SHR		
	GLO	R9	
	SM		
	PLO	R9	
	DEC	R9	
	DEC	R9	
	LDN	R9	
	STXD		
	LDN	R9	
	ADI	#05	
	STR	R2	
	GLO	RD	
	SD		
	BDF	JN	
	SEP	R4	
	,A(HIPR)		.. Call hi pressure interpolation subroutine
	BR	JP	
JN:	SEP	R4	
	,A(LOPR)		.. Call lo pressure interpolation subroutine
JP:	IRX		.. Go to bottom of stack
	IRX		..
	IRX		..
	IRX		..
	IRX		..
	IRX		..
	GHI	RF	.. Store upper flow rate limit
	STXD		..
	GLO	RF	
	STXD		
	GLO	R7	.. Store upper displacement
	STR	R2	..
	LDI	#04	
	ADD		
	STXD		.. Store lower displacement
	GLO	R7	
	STXD		
	GHI	RA	.. Store lower flow rate limit
	STXD		..
	GLO	RA	..
	STXD		..
	SEP	R4	

```
LABEL   OBJECT CODE     OPERAND     COMMENTS
        ,A(VSD)                     .. Call valve displacement
VV:     SEP             R4
        ,A(DISP)                    .. Display interpolated valve
                                       displacement

        BR              VV          .. Loop back to VV

LOPR:   IRX                         *** Low Pressure Subroutine ***
        IRX
        IRX
        LDXA                        .. Load registers
        PLO             RE          .. RE.0 = lo pressure
        LDXA
        PLO             RA
        LDXA
        PHI             RA          .. RA = lo flow rate
        LDXA
        PLO             RB
        LDXA
        PHI             RB          .. RB = hi flow rate
        LDX
        PHI             RE          .. RE.1 = hi pressure
        GLO             RE
        SD                          .. Subtract lo pressure
        SHR                         .. Divide result by 8
        SHR                         ..
        SHR                         ..
        PHI             RC          .. Store pressure increment in
                                       RC.1
        LDI             #01         .. Set DF = #01
        SHR                         ..
        DEC             R2          .. Move stack to hi flow ..
        DEC             R2          .. rate, lo byte
        GLO             RA          .. Subtract low flow rate, ..
        SD                          .. lo byte
        PLO             RB          .. Store result in RB.0
        IRX                         .. Move stack to hi flow rate, hi
                                       byte
        GHI             RA          .. Subtract lo flow rate, lo byte
        SDB                         ..
        PHI             RB          .. Store result in RB.1
        DEC             R2          .. Go to top of stack
        DEC             R2          ..
        DEC             R2          ..
        DEC             R2          ..
        DEC             R2          ..
        DEC             R2          ..
        DEC             R2          ..
```

LABEL	OBJECT CODE	OPERAND	COMMENTS
	GLO	RB	.. Test RB.0 to see if it is ..
	SHR		.. an odd or even number
	BNF	RJ	.. If even, jump to RJ
	GLO	RB	..
	STR	R2	.. If odd, add #01 to lo byte
	LDI	#01	..
	ADD		..
	PLO	RB	..
	GHI	RB	.. Add #00, with carry, to ..
	STR	R2	.. hi byte in case of ..
	LDI	#00	.. overflow
	ADC		..
	PHI	RB	..
RJ:	GHI	RB	.. Divide RB by 2
	SHR		..
	PHI	RB	..
	GLO	RB	..
	SHRC		..
	PLO	RB	..
	GLO	RB	.. Test RB.0 to see if it is ..
	SHR		.. an odd or even number
	BNF	RK	.. If even, jump to RK
	GLO	RB	
	STR	R2	.. If odd, subtract #01 from ..
	LDI	#01	.. lo byte
	SD		..
	PLO	RB	..
	GHI	RB	.. Subtract #00, with borrow, ..
	STR	R2	.. from hi byte in case of ..
	LDI	#00	.. underflow
	SDB		..
	PHI	RB	..
RK:	GHI	RB	.. Divide RB by 2
	SHR		..
	PHI	RB	..
	GLO	RB	..
	SHRC		..
	PLO	RB	..
	GLO	RB	.. Test RB.0 to see if it is ..
	SHR		.. an odd or even number
	BNF	RL	.. If even, jump to RL
	GLO	RB	
	STR	R2	.. If odd, add #01 to lo byte
	LDI	#01	..
	ADD		..
	PLO	RB	..
	GHI	RB	.. Add #00, with carry, to ..
	STR	R2	.. hi byte in case of ..

```
LABEL   OBJECT CODE   OPERAND   COMMENTS
        LDI           #00       .. overflow
        ADC                     ..
        PHI           RB        ..
        GHI           RB        .. Divide RB by 2
        SHR                     ..
        PHI           RB        ..
        GLO           RB        ..
        SHRC                    ..
        PLO           RB        ..
        GLO           RD
        STR           R2        .. Subtract lower table
                                   pressure ..
        GLO           RE        .. from system pressure
        SD                      ..
        PLO           RC        .. Store result in RC.0
        LDI           #01       .. Set DF = #01
        SHR                     ..
RP:     GLO           RE        .. Get lower table pressure
        STR           R2        ..
        GHI           RC        .. Add pressure increment
        ADD                     ..
        PLO           RE        ..
        GLO           RA        .. Get lower flow rate
        STR           R2        ..
        GLO           RB        .. Add flow rate increment
        ADD                     ..
        PLO           RA        ..
        GHI           RA        ..
        STR           R2        ..
        GHI           RB        ..
        ADC                     ..
        PHI           RA        ..
        GLO           RD
        STR           R2        .. Subtract modified lower
                                   table ..
        GLO           RE        .. pressure from system pressure
        SD                      ..
        PHI           R9        .. Store result in R9.1
        LDI           #01       .. Set DF = #01
        SHR                     ..
        GLO           RC
        STR           R2        .. Subtract new pressure
                                   difference ..
        GHI           R9        .. from old pressure difference
                                   in RC.0
        SD                      ..
        BNF           ZZ        .. If result < 0, jump to ZZ
        BNF           ZZ        .. If result = 0, jump to ZZ
```

```
LABEL    OBJECT CODE    OPERAND    COMMENTS
         GHI            R9         .. Store new pressure
                                      difference ..
         PLO            RC         .. in RC.0
         BR             RP         .. Jump to RP
ZZ:      GLO            RA
         STR            R2         .. Subtract flow rate increment
                                      increment ..
         GLO            RB         .. from RA and store result ..
         SD                        .. back in RA
         PLO            RA         ..
         GHI            RA         ..
         STR            R2         ..
         GHI            RB         ..
         SDB                       ..
         PHI            RA         ..
         LDI            #00        .. Set RB = #0000
         PLO            RB         ..
         PHI            RB         ..
         LDI            #08        .. Set R9.1 = #08
         PHI            R9         ..
         SEP            R5         .. Return to calling program

HIPR:    IRX                       *** High Pressure Subroutine ***
         IRX
         IRX
         LDXA                      .. Load registers
         PLO            RE         .. RE.0 = lo pressure
         LDXA
         PLO            RA
         LDXA
         PHI            RA         .. RA = lo flow rate
         LDXA
         PLO            RB
         LDXA
         PHI            RB         .. RB = hi flow rate
         LDX
         PHI            RE         .. RE.1 = hi pressure
         GLO            RE         .. Subtract lo pressure from ..
         SD                        .. hi pressure
         SHR                       .. Divide result by 8
         SHR                       ..
         SHR                       ..
         PHI            RC         .. Store pressure increment in
                                      RC.1
         LDI            #01        .. Set DF = #01
         SHR                       ..
         DEC            R2         .. Move stack to hi flow ..
         DEC            R2         .. rate, lo byte
```

```
LABEL    OBJECT CODE    OPERAND    COMMENTS
         GLO            RA         .. Subtract lo flow rate, ..
         SD                        .. lo byte
         PLO            RA         .. Store result in RA.0
         IRX                       .. Move stack to hi flow rate, ..
         GHI            RA         .. hi byte
         SDB                       .. Subtract lo flow rate, lo byte
         PHI            RA         .. Store result in RA.1
         DEC            R2         .. Go to top of stack
         DEC            R2         ..
         DEC            R2         ..
         DEC            R2         ..
         DEC            R2         ..
         DEC            R2         ..
         DEC            R2         ..
         GLO            RA         .. Test RA.0 to see if it is ..
         SHR                       .. an odd or even number
         BNF            RM         .. If even, jump to RM
         GLO            RA
         STR            R2         .. If odd, add #01 to lo byte ..
         LDI            #01        ..
         ADD                       ..
         PLO            RA         ..
         GHI            RA         .. Add #00, with carry, to ..
         STR            R2         .. hi byte in case of
         LDI            #00        .. overflow
         ADC                       ..
         PHI            RA         ..
RM:      GHI            RA         .. Divide RA by 2
         SHR                       ..
         PHI            RA         ..
         GLO            RA         ..
         SHRC                      ..
         PLO            RA         ..
         GLO            RA         .. Test RA.0 to see if it is ..
         SHR                       .. an odd or even number
         BNF            RN         .. If even, jump to RN
         GLO            RA
         STR            R2         .. If odd, subtract #01 from ..
         LDI            #01        .. lo byte
         SD                        ..
         PLO            RA         ..
         GHI            RA         ..
         STR            R2         .. Subtract #00, with borrow, ..
         LDI            #00        .. from hi byte in case of ..
         SDB                       .. underflow
         PHI            RA         ..
RN:      GHI            RA         .. Divide RA by 2
         SHR                       ..
```

```
LABEL   OBJECT CODE   OPERAND   COMMENTS
        PHI           RA        ..
        GLO           RA        ..
        SHRC                    ..
        PLO           RA        ..
        GLO           RA        .. Test RA.0 to see if it is ..
        SHR                     .. an odd or even number
        BNF           RQ        .. If even, jump to RQ
        GLO           RA        ..
        STR           R2        .. If odd, add #01 to lo byte
        LDI           #01       ..
        ADD                     ..
        PLO           RA        ..
        GHI           RA        .. Add #00, with cary, to ..
        STR           R2        .. hi byte in case of ..
        LDI           #00       .. overflow
        ADC                     ..
        PHI           RA        ..
RQ:     GHI           RA        .. Divide RA by 2
        SHR                     ..
        PHI           RA        ..
        GLO           RA        ..
        SHRC                    ..
        PLO           RA        ..
        GHI           RE        .. Get hi pressure
        STR           R2        ..
        GLO           RD        .. Subtract system pressure ..
        SD                      .. from hi table pressure
        PLO           RC        .. Store result in RC.0
        LDI           #01       .. Set DF = #01
        SHR                     ..
RG:     GHI           RE        .. Subtract pressure increment ..
        STR           R2        .. from hi pressure
        GHI           RC        ..
        SD                      ..
        PHI           RE        .. Store result in RE.1
        GLO           RB        .. Subtract flow rate increment
        STR           R2        .. from hi flow rate
        GLO           RA        ..
        SD                      ..
        PLO           RB        .. Store result in RB
        GHI           RB        ..
        STR           R2        ..
        GHI           RA        ..
        SDB                     ..
        PHI           RB        ..
        GHI           RE
        STR           R2        .. Subtract system pressure ..
        GLO           RD        .. from modified hi pressure
```

```
LABEL   OBJECT CODE   OPERAND   COMMENTS
        SD
        PHI           R9        .. Store result in R9.1
        LDI           #01       .. Set DF = #01
        SHR
        GLO           RC        .. Subtract old pressure
                                   difference in RC.0 ..
        STR           R2        .. from new pressure difference
        GHI           R9        ..
        SD                      ..
        BNF           ZX        .. If result < 0, jump to ZX
        BZ            ZX        .. If result = 0, jump to ZX
        GHI           R9        .. Store new pressure ..
        PLO           RC        .. difference in RC.0
        BR            RG        .. Jump to RG
ZX:     GLO           RB        .. Add flow rate increment to ..
        STR           R2        .. RB
        GLO           RA        ..
        ADD                     ..
        PLO           RA        ..
        GHI           RB        ..
        STR           R2        ..
        GHI           RA        ..
        ADC                     ..
        PHI           RA        ..
        LDI           #00       .. Set RB = #0000
        PLO           RB        ..
        PHI           RB        ..
        LDI           #08       .. Set R9.1 = #08
        PHI           R9
        SEP           R5        .. Return to calling program

VSD:    IRX                     *** VALVE DISPLACEMENT
                                    SUBROUTINE ***
        IRX
        IRX
        LDXA                    .. Load registers
        PLO           RA        .. RA = Lower flow rate
        LDXA
        PHI           RA
        LDXA
        PLO           RE        .. RE.0 = lower valve stem
                                   displacement (VSD)
        LDXA
        PHI           RE        .. RE.0 = Upper valve stem
                                   displacement
        LDXA                    ..
        PLO           RB        .. RB = Upper flow rate
        LDX
```

LABEL	OBJECT CODE	OPERAND	COMMENTS
	PHI	RB	
	DEC	R2	.. Move stack to upper flow rate, lo byte
	LDI	#01	.. Set DF = #01
	SHR		
	GLO	RA	.. Subtract lower flow rate, lo byte
	SD		..
	PLO	RB	.. Store result in RB.0
	IRX		.. Move stack to upper flow rate, hi byte
	GHI	RA	.. Subtract lower flow rate, hi byte
	SDB		.. with borrow
	PHI	RB	.. Store result in RB.1
	GHI	RB	.. Divide by 2
	SHR		..
	PHI	RB	..
	GLO	RB	..
	SHRC		..
	PLO	RB	..
	GHI	RB	.. Divide RB by 2
	SHR		..
	PHI	RB	..
	GLO	RB	..
	SHRC		..
	PLO	RB	.. RB contains flow rate increment
	DEC	R2	.. Move stack to upper VSD
	DEC	R2	..
	GLO	RE	.. Subtract lower VSD
	SD		..
	SHR		.. Divide result by 4
	SHR		..
	PHI	RE	.. Store final result in RE.0
	DEC	R2	.. Go to top of stack
	DEC	R2	..
	DEC	R2	..
	DEC	R2	..
	DEC	R2	..
	DEC	R2	..
	GLO	R8	.. Get requested flow rate, lo byte
	STR	R2	
	GLO	RA	.. Subtract lower flow rate, lo byte
	SD		..
	PLO	RC	.. Store result in RC.0

```
LABEL   OBJECT CODE   OPERAND   COMMENTS
        GHI           R8        .. Get requested flow rate, hi
                                   byte
        STR           R2        ..
        GHI           RA        .. Subtract, with borrow, lower
                                   flow rate, ..
        SDB                     .. hi byte
        PHI           RC        .. Store result in RC.1
        LDI           #01       .. Set DF = #01
        SHR
PP:     GLO           RA        ..
        STR           R2        .. Add flow rate increment to ..
        GLO           RB        .. lower flow rate
        ADD                     ..
        PLO           RA        .. Store result in RA
        GHI           RA        ..
        STR           R2        ..
        GHI           RB        ..
        ADC                     ..
        PHI           RA        ..
        GLO           RE        .. Add VSD increment to ..
        STR           R2        .. lower VSD
        GHI           RE        ..
        ADD                     ..
        PLO           RE        ..
        GLO           R8        .. Subtract modified lower flow
                                   rate ..
        STR           R2        .. from requested flow rate
        GLO           RA        ..
        SD                      ..
        PLO           R7        .. Store result in R7
        GHI           R8        ..
        STR           R2        ..
        GHI           RA        ..
        SDB                     ..
        PHI           R7        ..
        BDF           XH        .. If result > 0, jump to XH
        GLO           R7        .. Otherwise, take 2's
                                   compliment ..
        XRI           #FF       .. of result and store in R7
        ADI           #01       ..
        PLO           R7        ..
        GHI           R7        ..
        XRI           #FF       ..
        ADCI          #00       ..
        PHI           R7        ..
XH:     LDI           #01       .. Set DF = #01
        SHR                     ..
        GLO           RC
```

```
LABEL   OBJECT CODE   OPERAND          COMMENTS
        STR           R2               .. Subtract new flow rate
                                          difference ..
        GLO           R7               .. from old flow rate difference
        SD                             ..
        STXD                           ..
        GHI           RC               ..
        STR           R2               ..
        GHI           R7               ..
        SDB                            ..
        IRX                            ..
        BNF           GG               .. If result < 0, jump to GG
        BNZ           LJ               .. If result > 0, jump to LJ
        LDX                            .. Move stack to lo byte of
                                          result
        BNZ           LJ               .. If lo byte > 0, jump to LJ
        BR            GG               .. Jump to GG
LJ:     GLO           R7               .. Store R7 value in RC
        PLO           RC               ..
        GHI           R7               ..
        PHI           RC               ..
        BR            PP               .. Jump to PP
GG:     GLO           RE               .. Subtract VSD increment
        STR           R2               .. from modified VSD
        GHI           RE               ..
        SD                             ..
        PLO           RA               .. Store result in RA
        LDI           #00              .. Set RB = #0000
        PLO           RB               ..
        PHI           RB               ..
        PHI           RA               .. Set RA.1 = #00
        LDI           #08              .. Set R9.1 = #08
        PHI           R9               ..
        SEP           R5               .. Return to calling program

DISP:   LDI           A.1(DIGITS)      *** DISPLAY SUBROUTINE ***
        PHI           RF               .. Displays HEX. content of RA ..
        LDI           A.0(DIGITS)      .. and RB in LED display
        PLO           RF               .. Setup pointer
        GHI           RA               .. Digit 1
        SHR
        SHR
        SHR
        SHR
        STR           RF
        INC           RF               .. Digit 2
        GHI           RA
        ANI           #OF
        STR           RF
```

```
LABEL    OBJECT CODE    OPERAND        COMMENTS
         INC            RF             .. Digit 3
         GLO            RA
         SHR
         SHR
         SHR
         SHR
         STR            RF             .. Digits 4
         INC            RF
         GLO            RA
         ANI            #OF
         STR            RF
         INC            RF             .. Digit 5
         GHI            RB
         SHR
         SHR
         SHR
         SHR
         STR            RF
         INC            RF             .. Digit 6
         GHI            RB
         ANI            #OF
         STR            RF
         INC            RF             .. Digit 7
         GLO            RB
         SHR
         SHR
         SHR
         SHR
         STR            RF
         INC            RF             .. Digit 8
         GLO            RB
         ANI            #OF
         STR            RF
         LDI            A.0(DIGITS)
         PLO            RF
         SEP            R4
         ,A(LEDD)
         SEP            R5
         DIGITS =       #8C00          .. Dummy save area

LEDD:    LDI            #80            *** LEDD SUBROUTINE ***
         PLO            RE             .. Refresh display
         LDI            A.1(DTAB)      .. Start with leftmost digit
         PHI            RD             .. Segment table pointer
LOOPD:   LDN            RF             .. Turn off?
         ANI            #80
         BNZ            SKIPD          .. Yes, skip digit
         DEC            R2             .. Get 2 free RAM bytes
```

```
LABEL     OBJECT CODE     OPERAND          COMMENTS

          DEC             R2
          GLO             RE             .. Ready for digit out
          STR             R2
          LDN             RF             .. Tab
          ANI             #0F
          ADI             A.0(DTAB)
          PLO             RD
          LDA             RD             .. Fetch segments
          PHI             RE
          LDN             RF
          ANI             #10            .. Decimal ?
          BZ              DISPD          .. No, show digit only
          GHI             RE
          ANI             #BF
          PHI             RE             .. Else add point
DISPD:    OUT4                           .. Select digit
          GHI             RE             .. Put in segments
          STR             R2
          OUT 3                          .. Turn on
          SEP             R4             .. Delay
          A,(DELSUB)
          SEX             R3
          OUT 3           ,#FF           .. Turn off segment
          SEX             R2
SKIPD:    INC             RF             .. Bump pointer
          GLO             RE             .. Shift
          SHR
          PLO             RE
          BNZ             LOOPD          .. Continue
          DEC             RF             .. Else
          DEC             RF
          DEC             RF
          DEC             RF
          DEC             RF             .. Fix pointer
          DEC             RF
          DEC             RF
          DEC             RF
          SEP             R5             .. And return

DELSUB:   LDI             #20            *** DELAY SUBROUTINE ***
          PLO             RD
DL:       DEC             RD             .. Count done?
          GLO             RD
          BNZ             DL             .. No, keep going
          SEP             R5             .. Return

ENTRY:    LDI             #00            *** Standard Call and Return
                                             Technique ***
```

```
LABEL     OBJECT CODE     OPERAND          COMMENTS

          PHI             R3
          LDI             #05              .. Load return address into R3
          PLO             R3
INIT:     LDI             A.1(CALL)        .. Load R4 with call address
          PHI             R4               ..
          PHI             R5               .. Load R5 with return address
          LDI             A.0(CALL)        ..
          PLO             R4               ..
          LDI             A.0(RETPGM)      ..
          PLO             R5               ..
          LDI             #8C
          PHI             R2               .. Load Stack address into R2
          PHI             R7
          LDI             #1F
          PLO             R2
          PLO             RC
          SEX             R2               .. Set stack pointer as R2
          SEP             R3               .. Return to main program

EXITA:    SEP             R3               .. Return to calling program
CALL:     SEX             R2               .. Point to stack
          GHI             R6               .. Save R6 value
          STXD                             ..
          GLO             R6               ..
          STXD                             ..
          GHI             R3               .. Put R3 ..
          PHI             R6               .. Into R6
          GLO             R3               .. Both halves
          PLO             R6
          LDA             R6               .. And put subroutine address ..

          PHI             R3               .. Into R3
          LDA             R6
          PLO             R3
          BR              EXITA            .. Go to calling subroutine

EXITR:    SEP             R3               .. Return to calling program
RETPGM:   GHI             R6               .. Put R6 back ..
          PHI             R3               .. Into R3
          GLO             R6               ..
          PLO             R3               ..
```

```
LABEL   OBJECT CODE   OPERAND   COMMENTS

        SEX           R2        .. Point to stack
        INC           R2        .. Recover ..
        LDXA                    .. R6
        PLO           R6        .. Lower half
        LDX
        PHI           R6        .. And upper half
        BR            EXITR     .. Go back to calling program
        END
```

APPENDIX C

OVERALL PROGRAM

The Overall Program was the controlling algorithm for the developed assembly. The program listing is from the 1802 assembler at NASA Langley Research Center. To simulate the Standard Call and Return Technique (SCRT) of the actual 1802 microprocessor system, a separate program, entitled ML18, was supplied by Mr. Dave Akey of the Sperry Support Systems. This program was assembled with the Overall Program and is listed as lines 2-232 in the following pages.

```
ER LINE ADDR B1 B2 B3 B4                    1802 ASSEMBLER VER 1.0MR

     2                          NLIST M
     3                        LIST X,A
     4                      ..MACRO LIBRARY
     5                      ..
     6                      ..      REGISTER NAME DEFINITIONS
     7                      ..
     8 0000                 DMA=R0  ..DMA POINTER REGISTER
     9 0001                 INT=R1  ..INTERRUPT HANDLER PC
    10 0002                 SP=R2   ..STACK POINTER
    11 0003                 PC=R3   ..PROGRAM COUNTER
    12 0004                 CC=R4   ..CALL CODE PC
    13 0005                 RT=R5   ..RETURN CODE PC
    14 0007                 S1=R7   ..SCRATCH PAD POINTER
    15 0008                 A0=R8   ..ACCU 0 (MUST BE SAVED BY CALLED ROUTINE)
    16 0009                 A1=R9   ..ACCU 1
    17 000A                 A2=RA   ..ACCU 2
    18 000B                 T0=RB   ..TEMP 0(MUST BE SAVED BY CALLING ROUTINE)
    19 000C                 T1=RC   ..TEMP 1
    20 000D                 T2=RD   ..TEMP 2
    21 000E                 BP=RE   ..BREAK POINT REGISTER
    22 0001                 BIT0=1
    23 0002                 BIT1=2
    24 0004                 BIT2=4
    25 0008                 BIT3=8
    26 0010                 BIT4=16
    27 0020                 BIT5=32
    28 0040                 BIT6=64
    29 0080                 BIT7=128
    30 0100                 BIT8=256
    31 0200                 BIT9=512
    32 0400                 BIT10=1024
    33 0800                 BIT11=2048
    34 1000                 BIT12=4096
    35 2000                 BIT13=8192
    36 4000                 BIT14=16384
    37 8000                 BIT15=32768
```

```
ER LINE ADDR B1 B2 B3 B4                    1802 ASSEMBLER VER 1.0MR

     39                        ..    NLIST S
     40                        ..
     41                        ..
     42                        ..CALL SUBROUTINE CODE
     43                        ..THE LABEL CALLA--CALL ADDRESS MUST BE LOADED INTO
     44                        ..R4 AT SYSTEM INITIALIZATION TIME.  THE OPCODE CALLC
     45                        ..MUST BE PLACED SOMEWHERE IN THE PROGRAM SPACE BUT
     46                        ..NOT WITHIN 15 WORDS OF THE END OF A PAGE.
     47                        ..
     48                        CALLC: MACRO
     49                               LOCAL EXITA
     50                        EXITA: SEP R3  ..R(3) IS POINTING TO THE FIRST INSTRUCTION IN SUBR,
     51                        CALLA: SEX R2  ..POINT TO STACK
     52                               GHI R6  ..PUSH R(6) ONTO STACK
     53                               STXD    TO PREPARE IT FOR POINTING TO ARGUMENTS, AND
     54                               GLO R6  ..DECREMENT TO FREE LOCATION.
     55                               STXD
     56                               GHI R3  ..COPY R(3) INTO R(6) TO SAVE THE RETURN ADDRESS
     57                               PHI R6
     58                               GLO R3
     59                               PLO R6
     60                               LDA R6  ..LOAD THE ADDRESS OF THE SUBROUTINE INTO R(3)
     61                               PHI R3
     62                               LDA R6
     63                               PLO R3
     64                               BR EXITA..BRANCH TO ENTRY POINT OF "CALL" MINUS ONE BYTE
     65                                        ..THIS LEAVES R(4) POINTING TO "CALL" ALLOWING FOLLOWING
     66                                        ..REPEATED CALLS.
     67                               ENDM
```

```
EP LINE ADDR B1 B2 B3 B4                    1802 ASSEMBLER VER 1.0MR

     69                    ..
     70                    ..RETURN SUBROUTINE CODE
     71                    ..THE LABEL RETA--RETURN ADDRESS MUST BE LOADED INTO R5 AT
     72                    ..SYSTEM INITIALIZATION TIME.
     73                    ..THE OPCODE RETCOD MUST BE PLACED SOMEWHERE IN THE PROGRAM SPACE
     74                    ..BUT NOT WITHIN 12 WORDS OF THE END OF A PAGE.
     75                    ..
     76                    RETCOD:MACRO
     77                           LOCAL EXITR
     78                    EXITR: SEP R3  ..RETURN TO MAIN PROGRAM
     79                    RETA:  GHI R6  ..COPY R(6) INTO R(3)
     80                           PHI R3  ..R(3) CONTAINS THE RETURN ADDRESS
     81                           GLO R6
     82                           PLO R3
     83                           SEX R2  ..POINT TO STACK
     84                           INC R2  ..POINT TO SAVED OLD R(6)
     85                           LDXA    ..RESTORE THE CONTENTS
     86                           PLO R6  ..R(6)
     87                           LDX
     88                           PHI R6
     89                           BR EXITR..BRANCH TO ENTRY POINT OF "RETA" MINUS ONE BYTE
     90                                    ..THIS LEAVES R(5) POINTINGTO "RETA" FOR FOLLOWING REPEAT
     91                                    ..CALLS.
     92                    ..
     93                           ENDM
```

```
ER LINE ADDR B1 B2 B3 B4                    1802 ASSEMBLER VER 1.0HR

     95                     PUSH:  MACRO P1,P2
     96                            SEX R2
     97                            GHI P1  ..PUSH P1 ON STACK AND SELECT P2 AS RX
     98                            STXD
     99                            GLO P1
    100                            STXD
    101                            IF P2+0
    102                            SEX P2
    103                            ENDIF
    104                            ENDM
    105                    ..POP
    106                    ..
    107                    POP:   MACRO P1,P2
    108                            SEX R2  ..POP P1 OFF STACK AND SELECT P2 AS RX
    109                            IRX
    110                            LDXA
    111                            PLO P1
    112                            LDX
    113                            PHI P1
    114                            IF P2+0
    115                            SEX P2
    116                            ENDIF
    117                            ENDM
    118                    ..
    119                    ..
    120                    ..
    121                    ..RETURN FROM SUBROUTINE MACRO
    122                    ..
    123                    RETURN:MACRO
    124                            SEP R5
    125                            ENDM
    126                    ..
    127                    ..LOAD REGISTER IMMEDIATE MACRO
    128                            ..LOADS RN WITH VALUE P1
    129                    ..
    130                    LDRI:  MACRO RN,P1
    131                            LDI A.1(P1)
    132                            PHI RN
    133                            LDI A.0(P1)
    134                            PLO RN
    135                            ENDM
    136                    ..
    137                    ..
    138                    ..LOAD REGISTER VIA X AND ADVANCE X
```

```
ER LINE ADDR B1 B2 B3 B4                    1802 ASSEMBLER VER 1.0MR

    139                     LDRVXA:MACRO RN
    140                            LDXA
    141                            PHI RN
    142                            LDXA
    143                            PLO RN
    144                            ENDM
    145                     ..
    146                     ..
```

```
    148                     ;;      THE CALL MACRO AUTOMATICALLY GENERATES THE CODE NECESSARY FOR
    149                     ;;      STANDARD CALL AND RETURN TECHNIQUE.
    150                     CALL:    MACRO   SUBR
    151                              SEP R4
    152                              ACON SUBR
    153                     .       ENDM
```

```
ER LINE ADDR B1 B2 B3 B4                          1802 ASSEMBLER VER 1.0MR

    155                         ..ONE PARAMETER CALL
    156                         CALL1: MACRO SUBR,P1
    157                                SEP R4
    158                                ACON SUBR
    159                                DC P1+0
    160                                ENDM
    161                         ..TWO
    162                         CALL2: MACRO SUBR,P1,P2
    163                                CALL1 SUBR,P1
    164                                ACON P2+0
    165                                ENDM
    166                         ..THREE
    167                         CALL3: MACRO SUBR,P1,P2,P3
    168                                CALL2 SUBR,P1,P2
    169                                ACON P3+0
    170                                ENDM
    171                         ..FOUR
    172                         CALL4: MACRO SUBR,P1,P2,P3,P4
    173                                CALL3 SUBR,P1,P2,P3
    174                                ACON P4+0
    175                                ENDM
```

```
ER LINE ADDR B1 B2 B3 B4                          1802 ASSEMBLER VER 1.0MR

   177                    LRWP:  MACRO REG
   178                           LDA R6  ..LOAD REGISTER REG WITH
   179                           PHI REG ..PARAMETER SUPLIED BY
   180                           LDA R6  ..THE CALLING ROUTINE
   181                           PLO REG
   182                           ENDM
   183                    ..
   184                    ..
   185                    ..
   186                    LRIWP: MACRO REG,REG2
   187                           LRWP REG
   188                           LDA REG
   189                           PHI REG2        ..LOAD REGISTER REG2 WITH PARAMETER POINTED
   190                           LDA REG ..BY THE CALLING ROUTINE SUPPLIED PARAMETER
   191                           PLO REG2..REG IS USED AS AN INTERMEADIATE REG.
   192                           ENDM
```

```
 194                    ..STORE REGISTER VIA REGISTER AND DECREMENT
 195                    ..
 196                    ..RS.0 -> M(RD) RS.1 -> M(RD+1) RD -> RD+1
 197                    ..
 198                    SRVRD: MACRO RS,RD
 199                           GLO RS
 200                           STR RD
 201                           DEC RD
 202                           GHI RS
 203                           STR RD
 204                           ENDM
 205                    ..
 206                    ..STORE REGISTER VIA REGISTER AND INCREMENT
 207                    ..
 208                    ..RS.1 -> M(RD) RS.0 -> M(RD-1) RD -> RD-1
 209                    ..
 210                    SRVRA: MACRO RS,RD
 211                           GHI RS
 212                           STR RD
 213                           INC RD
 214                           GLO RS
 215                           STR RD
 216                           ENDM
 217                    ..STORE REG VIA REG AND INCREMENT PAST
 218                    SRVRAA:MACRO RS,RD
 219                           SRVRA RS,RD
 220                           INC RD
 221                           ENDM
 222                    ..
 223                    ..LOAD REG VIA REG AND DECREMENT PAST
 224                    LRVRD: MACRO RS,RD
 225                           LDN RD
 226                           PLO RS
 227                           DEC RD
 228                           LDN RD
 229                           PHI RS
 230                           DEC RD
 231                           ENDM
 232                        LIST S
 233 0000 71                   DIS
 234 0001 00                   ,00H
 235 0002                      LDRI R2,RAMTOP          ..SETUP STACK
 240 0008                      LDRI R3,MAIN            ..SETUP PROGRAM START ADDRESS
 245 000F                      LDRI R4,CALLA           ..SCRT CALL ADDRESS
```

```
ER LINE ADDR B1 B2 B3 B4                    1802 ASSEMBLER VER 1.0MR

    250 0014                        LDRI R5,RETA        ..RETURN CODE
    255 001A D3                     SEP R3              ..JUMP TO MAIN PROGRAM
    256 001B                        CALLC
    274 002B                        RETCOD              ..SCRT CODE
    290 8C1F                RAMTOP=8C1FH
    291 0038 F8 00          MAIN:   LDI 00H             ..INITIALIZE RA,RB,RC,RF.0,R9.1,RE.0 TO #00
    292 003A AA                     PLO RA              ..
    293 003B BA                     PHI RA              ..
    294 003C B9                     PHI R9              ..
    295 003D A8                     PLO R8              ..
    296 003F B8                     PHI R8              ..
    297 003F AC                     PLO RC              ..
    298 0040 BC                     PHI RC              ..
    299 0041 AF                     PLO RF              ..
    300 0042 AE                     PLO RE              ..SET INITIAL POSITION-SENSOR VALUE
    301 0043 F8 10                  LDI 010H            ..PUT #10HEX. INTO RF.1 FOR COUNTDOWN CYCLE..
    302 0045 BF                     PHI RF              ..TO CLOSE THE SOLENOID
    303 0046 F8 0F                  LDI 0FH             ..SET INITIAL SYSTEM PRESSURE VALUE
    304 0048 AD                     PLO RD
    305 0049 F8 50                  LDI 050H            ..SET NUMBER OF STEPS TO CLOSE VALVE INITIALLY
    306 004B A9                     PLO R9
    307 004C                        CALL1 OUTPUT,044H   ..CLOSE UPSTREAM SOLENOID
    311 0050                        CALL1 OUTPUT,04H    ..TURN OFF POWER TO LINEAR ACTUATOR
    315 0054                        CALL CLOSE          ..CLOSE DOWN VALVE ALL THE WAY
    318 0057 F8 18                  LDI 18H
    319 0059 BF                     PHI RF
    320 005A                        CALL1 OUTPUT,014H   ..RESET LINEAR ENCODER INTERFACE CIRCUIT
    324 005E                        CALL1 OUTPUT,04H    ..CUT OFF POWER TO ACTUATOR
    328 0062 F8 10                  LDI 010H
    329 0064 BF                     PHI RF
    330 0065                        CALL1 OUTPUT,084H   ..OPEN UPSTREAM SOLENOID
    334 0069                        CALL1 OUTPUT,04H    ..TURN OFF POWER TO LINEAR ACTUATOR
    338 0060 F8 FF          RX:     LDI 0FFH
    339 006F BF                     PHI RF
    340 0070                        CALL1 OUTPUT,024H   ..START MANOMETER INTERFACE CIRCUIT
    344 0074                        CALL DELAY          ..CALL DELAY SUBR. LOCATED IN OUTPUT SUBR.
    347 0077                        CALL DELAY          ..CALL DELAY SUBROUTINE AGAIN
    350 007A 69                     INP 1               ..INPUT REQUESTED VOLUMETRIC FLOW RATE HI BYTE
    351 007B B8                     PHI R8
    352 007C 73                     STXD                ..PUT HI BYTE ON STACK AND ADVANCE STACK
    353 007D 6A                     INP 2               ..INPUT REQUESTED VOLUMETRIC FLOW RATE LO BYTE
    354 007F A8                     PLO R8
    355 007F 52                     STR R2              ..PUT LO BYTE ON STACK AND ADVANCE STACK
    356 0080 8C                     GLO RC
    357 0081 F3                     XOR                 ..COMPARE LO BYTE TO PREVIOUS LO BYTE TO CHECK..
```

```
    358 0082 32 8C          BZ DP           ..FOR CHANGE. IF SAME, JUMP TO CHECK HI BYTES
    359 0084 72             LDXA
    360 0085 AC             PLO RC          ..IF DIFFERENT, LOAD NEW VALUES INTO RC
    361 0086 F0             LDX             ..VALUES IN RC HOLDING AREA
    362 0087 BC             PHI RC
    363 0088 F8 01          LDI 01H         ..SET RB.0 = #01 (SET FLAG)
    364 008A AB             PLO RB
    365 008B 38             SKP
    366 008C 60         DP: IRX             ..COMPARE HI BYTE TO PREVIOUS HI BYTE TO CHECK..
    367 008D 9C             GHI RC          ..FOR CHANGE
    368 008E F3             XOR
    369 008F 32 9A          BZ GQ           ..IF SAME, JUMP TO GQ
    370 0091 F0             LDX
    371 0092 BC             PHI RC          ..IF DIFFERENT, LOAD NEW VALUES INTO RC
    372 0093 22             DEC R2
    373 0094 F0             LDX
    374 0095 AC             PLO RC
    375 0096 F8 01          LDI 01H         ..SET RB.0 = #01 (SET FLAG)
    376 0098 AB             PLO RB
    377 0099 60             IRX
    378 009A 68         GQ: INP 3           ..INPUT UPSTREAM PRESSURE
    379 009B BD             PHI RD          ..STORE IN RD.1
    380 009C F8 10          LDI 010H
    381 009E BF             PHI RF
    382 009F                CALL1 OUTPUT,04H ..STOP MANOMETER INTERFACE CIRCUIT
    386 00A3 9D             GHI RD
    387 00A4 FF 0F          SMI 0FH         ..COMPARE INPUT PRESSURE TO TABLE LO VALUE
    388 00A6 33 AB          BDF DD
    389 00A8 F8 0F          LDI 0FH
    390 00AA BD             PHI RD
    391 00AB 9D         DD: GHI RD
    392 00AC 52             STR R2
    393 00AD FD 2F          SDI 02FH        ..COMPARE INPUT PRESSURE TO TABLE HI VALUE
    394 00AF 33 B5          BDF EE
    395 00B1 F8 2F          LDI 02FH
    396 00B3 BD             PHI RD
    397 00B4 52             STR R2
    398 00B5 8D         EE: GLO RD          ..COMPARE PRESS. TO PREVIOUS PRESS. READING
    399 00B6 F3             XOR             ..
    400 00B7 32 CC          BZ PZ           ..IF SAME, JUMP TO PZ
    401 00B9 8D             GLO RD          ..GET OLD SYSTEM PRESSURE
    402 00BA F5             SD              ..SUBT. OLD SYSTEM PRESS. FROM CURRENT READING
    403 00BB 33 C1          BDF RW
    404 00BD FB FF          XRI 0FFH
    405 00BF FC 01          ADI 01H
```

```
ER LINE ADDR B1 B2 B3 B4                    1802 ASSEMBLER VER 1.0MR

    406 00C1 52            RW:  STR R2          ..STORE DIFFERENCE ON STACK
    407 00C2 F8 01              LDI 01H
    408 00C4 F5                 SD              ..SUBTRACT #01 FROM DIFFERENCE
    409 00C5 3B CC              BNF PZ
    410 00C7 9D                 GHI RD          ..GET CURRENT SYSTEM PRESSURE READING
    411 00C8 AD                 PLO RD          ..LOAD NEW PRESS. INTO RD.0 IF DIFFERENT
    412 00C9 F8 01              LDI 01H         ..SET R8.1 = #01 (SET FLAG)
    413 00CB B8                 PHI R8
    414 00CC 88            PZ:  GLO R8          ..CHECK FOR FLAGS IN R8
    415 00CD 52                 STR R2
    416 00CE 98                 GHI R8
    417 00CF F1                 OR
    418 00D0 3A D7              BNZ PY          ..IF A FLAG IS SET, JUMP TO START VALVE MOVEMENT
    419 00D2                    CALL LVDT       ..OTHER WISE, CHECK VALVE FOR CREEP OR SLIPPAGE
    422 00D5 30 6D              BR RX           ..LOOP BACK TO RX FOR DATA INPUT
    423 00D7 8A            PY:  GLO RA          ..STORE INPUT VALUES ON STACK
    424 00D8 73                 STXD            ..RA.0= CALCULATED VSD IN STEPS
    425 00D9 9C                 GHI RC          ..RC.1= HI BYTE OF REQUESTED FLOW RATE
    426 00DA 73                 STXD            ..
    427 00DB 8C                 GLO RC          ..RC.0= LO BYTE OF REQUESTED FLOW RATE
    428 00DC 73                 STXD            ..
    429 00DD 8D                 GLO RD          ..RD.0= UPSTREAM PRESSURE
    430 00DF 73                 STXD            ..
    431 00DF 8F                 GLO RE          ..RE.0= READING FROM VSD SENSOR
    432 00E0 73                 STXD            ..
    433 00E1                    CALL TABRD      ..CALL TABLE READ SUBROUTINE TO CALCULATE..
    436 00E4 F8 10              LDI 010H        ..NECESSARY DISPLACEMENT
    437 00E6 BF                 PHI RF
    438 00E7 F8 00              LDI 00H         ..PUT #00 INTO R9.1,RF.0
    439 00E9 B9                 PHI R9
    440 00EA AF                 PLO RF
    441 00EB 60                 IRX             ..INCREMENT STACK BACK ONE POSITION
    442 00EC 72                 LDXA            ..RELOAD STORED VALUES ON STACK BACK INTO..
    443 00ED AE                 PLO RE          ..THEIR RESPECTIVE REGISTERS
    444 00EE 72                 LDXA
    445 00EF AD                 PLO RD
    446 00F0 72                 LDXA
    447 00F1 AC                 PLO RC
    448 00F2 72                 LDXA
    449 00F3 BC                 PHI RC
    450 00F4 F0                 LDX             ..BUT THIS TIME, PUT STORED OLD VSD IN RA.1
    451 00F5 BA                 PHI RA
    452 00F6 9A                 GHI RA
    453 00F7 52                 STR R2          ..PUT OLD VSD ON STACK
    454 00F8 8A                 GLO RA
```

```
 455 00F9 F5                    SD                      ..SUBTRACT NEW VSD FROM OLD VALUE
 456 00FA C2 01 27              LBZ JB                  ..IF SAME, STAY AT CURRENT DISPLACEMENT
 457 00FD                       CALL1 OUTPUT,044H       ..CLOSE UPSTREAM SOLENOID
 461 0101                       CALL1 OUTPUT,04H        ..TURN OFF POWER TO ACTUATOR
 465 0105 9A                    GHI RA
 466 0106 A9                    PLO R9                  ..STORE VSD IN R9.0
 467 0107 32 0C                 BZ JA                   ..IF VSD = 0, JUMP TO JA
 468 0109                       CALL CLOSE              ..CLOSE VALVE COMPLETELY
 471 010C F8 18           JA:   LDI 18H
 472 010E BF                    PHI RF
 473 010F                       CALL1 OUTPUT,014H       ..RESET LINEAR ENCODER INTERFACE CIRCUIT
 477 0113                       CALL1 OUTPUT,04H        ..CUT OFF POWER TO ACTUATOR
 481 0117 8A                    GLO RA
 482 0118 A9                    PLO R9                  ..STORE VSD IN R9.0
 483 0119 F8 10                 LDI 010H
 484 011B BF                    PHI RF
 485 011C                       CALL OPEN               ..OPEN VALVE TO NEW DISPLACEMENT
 488 011F                       CALL1 OUTPUT,004H       ..OPEN UPSTREAM SOLENOID
 492 0123                       CALL1 OUTPUT,04H        ..TURN OFF POWER TO ACTUATOR
 496 0127                 JB:   CALL LVDT               ..CALL VALVE STEM POSITION SENSING SUBROUTINE..
 499 0128 F8 00                 LDI 00H                 ..SET R8 = #0000
 500 012C A8                    PLO R8                  ..
 501 012D B8                    PHI R8                  ..
 502 012F C0 00 6D              LBR RX                  ..TO CHECK FOR CORRECT POSITIONING
 503 0200                       PAGE
 504 0200                 RET8: RETURN                  ..***TABLE READ SUBROUTINE***
 506 0201                 TABRD: LDRI R9,FLODAT         ..THIS COLLECTS THE DATA FOR INTERPOLATION SUBR.
 511 0207 F8 00                 LDI 00H                 ..LOAD R9 WITH LOCATION OF FLOW RATE TABLE
 512 0209 A7                    PLO R7                  ..PUT #00 INTO R7,RA,RB
 513 020A B7                    PHI R7                  ..
 514 020B AA                    PLO RA                  ..
 515 020C BA                    PHI RA                  ..
 516 020D AB                    PLO RB                  ..
 517 020E BB                    PHI RB                  ..
 518 020F 8B                    GLO RB
 519 0210 52                    STR R2                  ..PUT REQUESTED FLOW RATE ON STACK
 520 0211 98                    GHI R8
 521 0212 F1                    OR                      ..COMPARE HI BYTE TO LO BYTE TO SEE IF INPUT..
 522 0213 C2 03 38              LBZ YY                  ..REQUESTED = #00, IF SO, JUMP TO YY
 523 0216 8D              MM:   GLO RD
 524 0217 52                    STR R2                  ..PUT UPSTREAM PRESSURE ON STACK
 525 0218 F8 01                 LDI 01H                 ..SET DF= #01
 526 021A F6                    SHR
 527 021B 09                    LDN R9                  ..GET PRESS. VALUE OF FIRST DATA LINE IN TABLE
 528 021C F5                    SD                      ..SUBTRACT TABLE VALUE FROM UPSTREAM READING
```

```
    529 021D 3B 29          BNF JJ          ..IF DIFFERENCE < 0, JUMP TO JJ
    530 021F 32 29          BZ JJ           ..IF DIFFERENCE = 0, JUMP TO JJ
    531 0221 89             GLO R9
    532 0222 52             STR R2          ..IF DIFFERENCE >0, GET R9.0(MEMORY ADDRESS)..
    533 0223 F8 2B          LDI 02BH        ..AND ADD #28HEX. TO IT TO JUMP THE POINTER..
    534 0225 F4             ADD             ..TO THE NEXT PRESSURE LINE IN THE TABLE
    535 0226 A9             PLO R9
    536 0227 30 16          BR MM
    537 0229 49         JJ: LDA R9          ..PUT TABLE PRESS. ON STACK FOR C.LC. PURPOSES
    538 022A 73             STXD            ..THIS IS HIGH PRESS. IN INTERPOLATION ROUTINE
    539 022B 98         KK: GHI R8
    540 022C 52             STR R2          ..PUT HI BYTE OF REQUESTED FLOW RATE ON STACK
    541 022D 09             LDN R9          ..GET DATA IN MEMORY SPACE DEFINED BY R9
    542 022E F5             SD              ..SUBTRACT STORED DATA FROM HI BYTE ON STACK
    543 022F 3B 49          BNF RR          ..IF DIFFERENCE < 0, JUMP TO RR
    544 0231 32 49          BZ RR           ..IF DIFFERENCE = 0, JUMP TO RR
    545 0233 87             GLO R7
    546 0234 52             STR R2          ..PUT R7.0 VALUE ON STACK
    547 0235 F8 04          LDI 04H
    548 0237 F4             ADD             ..ADD #04 TO R7.0 VALUE..
    549 0238 A7             PLO R7          ..AND PUT NEW VALUE BACK IN R7.0
    550 0239 87             GLO R7          ..R7 IS USED AS AN ACCUMULATOR TO CALC. ..
    551 023A 52             STR R2          ..# STEPS THE VALVE STEM NEEDS TO BE DISPL.
    552 023B F8 50          XRI 050H        ..SEE IF THE # OF STEPS IN R7 IS = TO MAXIMUM..
    553 023D 3A 45          BNZ KL          ..THE STEM CAN TRAVEL. IF NOT, THEN JUMP TO KL
    554 023F F8 50          LDI 050H        ..IF R7 DOES = #50HEX., THEN PUT #50HEX. INTO..
    555 0241 AA             PLO RA          ..RA.0
    556 0242 60             IRX
    557 0243 30 00          BR RET8         ..RETURN TO MAIN PROGRAM AS A DEFAULT
    558 0245 19         KL: INC R9          ..IF R7 NOT AT MAX.,GO TO NEXT 2 MEMORY BYTES..
    559 0246 19             INC R9          ..POINTED TO BY R9
    560 0247 30 2B          BR KK           ..JUMP TO KK AND START OVER AGAIN
    561 0249 88         RR: GLO R8
    562 024A 52             STR R2          ..PUT LO BYTE OF REQUESTED FLOW RATE ON STACK
    563 024B 19             INC R9          ..INCREMENT TABLE POINTER TO TABLE LO BYTE
    564 024C F8 01          LDI 01H         ..SET DF = #01
    565 024E F6             SHR
    566 024F 09             LDN R9          ..GET TABLE LO BYTE
    567 0250 F5             SD              ..SUBTRACT TABLE LO BYTE FROM REQUESTED LO BYTE
    568 0251 73             STXD            ..PUT DIFFERENCE ON STACK AND DECREMENT STACK
    569 0252 98             GHI R8
    570 0253 52             STR R2          ..COMPARE TABLE HI BYTE TO REQUESTED HI BYTE
    571 0254 29             DEC R9          ..
    572 0255 09             LDN R9          ..
    573 0256 75             SDB             ..SUBTRACT TABLE HI BYTE FROM REQUESTED HI BYTE
```

```
ER LINE ADDR B1 B2 B3 B4                    1802 ASSEMBLER VER 1.0MR

    574 0257 60                    IRX
    575 0258 3B 72                 BNF RZ          ..IF DIFFERENCE < 0, JUMP TO RZ
    576 025A 3A 5F                 BNZ TT          ..IF DIFFERENCE > 0, JUMP TO TT
    577 025C F0                    LDX
    578 025D 32 72                 BZ RZ
    579 025F 87              TT:   GLO R7
    580 0260 FC 04                 ADI 04H         ..ADD #04 TO R7.0
    581 0262 A7                    PLO R7          ..PUT NEW VALUE BACK IN R7.0
    582 0263 52                    STR R2
    583 0264 FB 50                 XRI 050H        ..SEE IF THE # OF STEPS IN R7 IS = TO MAXIMUM..
    584 0266 3A 6E                 BNZ KZ          ..THE STEM CAN TRAVEL. IF NOT, THEN JUMP TO KZ
    585 0268 F8 50                 LDI 050H        ..IF R7 DOES = #50HEX., THEN PUT #50HEX. INTO..
    586 026A AA                    PLO RA          ..RA.0
    587 026B 60                    IRX
    588 026C 30 00                 BR RET8         ..RETURN TO MAIN PROGRAM AS A DEFAULT
    589 026F 19              KZ:   INC R9          ..GO TO THE NEXT 2 MEMORY BYTES DEFINED BY R9
    590 026F 19                    INC R9
    591 0270 30 49                 BR RR
    592 0272 49              RZ:   LDA R9          ..IF NUMBERS IN MEMORY MEET ABOVE TESTS, STORE..
    593 0273 73                    STXD            ..VALUES ON STACK AND DECREMENT STACK
    594 0274 09                    LDN R9
    595 0275 73                    STXD
    596 0276 89                    GLO R9
    597 0277 52                    STR R2
    598 0278 8D                    GLO RD          ..SUBTRACT #0FHEX. FROM SYSTEM PRESS. TO SEE..
    599 0279 FF 0F                 SMI 0FH         ..IF THEY EQUAL. THIS IS TO PREVENT THE TABLE..
    600 027B 32 81                 BZ RY           ..FROM UNDERFLOWING
    601 027D F8 2B                 LDI 02BH        ..SUBTRACT #2BHEX. FROM R9.0
    602 027F F5                    SD              ..THIS WILL JUMP THE TABLE POINTER DOWN..
    603 0280 A9                    PLO R9          ..TO THE NEXT LOWER PRESSURE RANGE
    604 0281 29              RY:   DEC R9          ..STORE FLOW RATE DATA OF LOWER PRESS.
    605 0282 49                    LDA R9          ..
    606 0283 73                    STXD            ..
    607 0284 09                    LDN R9          ..
    608 0285 73                    STXD            ..
    609 0286 87                    GLO R7          ..DIVIDE R7.0 BY 2 TO FIND TRUE VSD
    610 0287 F6                    SHR             ..
    611 0288 52                    STR R2
    612 0289 F8 01                 LDI 01H         ..SET DF = #01
    613 028B F6                    SHR
    614 028C 89                    GLO R9
    615 028D F7                    SM              ..SUBTRACT VSD FROM R9.0 TO GET TO LOWER..
    616 028E A9                    PLO R9          ..PRESSURE VALUE
    617 028F 29                    DEC R9          ..
    618 0290 29                    DEC R9          ..
```

```
ER LINE ADDR B1 B2 B3 B4                    1802 ASSEMBLER VER 1.0MR
    619 0291 09                    LDN R9          ..STORE LOWER PRESSURE
    620 0292 73                    STXD            ..
    621 0293                       CALL INTER      ..CALL INTERPOLATION SUBROUTINE
    624 0296 60                    IRX             ..INCREMENT TO BOTTOM OF STACK
    625 0297 60                    IRX             ..
    626 0298 60                    IRX             ..
    627 0299 60                    IRX             ..
    628 029A 60                    IRX             ..
    629 029B 60                    IRX             ..RA HOLDS INTERPOLATED FLOW RATE
    630 029C 88                    GLO R8          ..GET LO BYTE OF REQUESTED FLOW RATE
    631 029D 52                    STR R2
    632 029F F8 01                 LDI 01H         ..SET DF = #01
    633 02A0 F6                    SHR
    634 02A1 8A                    GLO RA          ..GET LO BYTE OF INTERPOLATED FLOW RATE
    635 02A2 F5                    SD              ..SUBTRACT INTERPOLATED VALUE FROM REQUESTED
    636 02A3 73                    STXD
    637 02A4 98                    GHI R8          ..GET HI BYTE OF REQUESTED FLOW RATE AND STORE
    638 02A5 52                    STR R2
    639 02A6 9A                    GHI RA          ..GET HI BYTE OF INTERPOLATED FLOW RATE
    640 02A7 75                    SDB             ..SUBTRACT INTERPOLATED VALUE FROM REQUESTED..
    641 02A8 60                    IRX             ..WITH BORROW
    642 02A9 3B D4                 BNF XZ
    643 02AB 3A B0                 BNZ ZG
    644 02AD F0                    LDX             ..COMPARE HI AND LO DIFFERENCE AS TEST
    645 02AE 32 D4                 BZ XZ           ..IF DIFFERENCE > 0, WE HAVEN'T UPPER LIMIT..
    646 02B0 89             ZG:    GLO R9          ..ALONG SYSTEM PRESSURE LINE
    647 02B1 52                    STR R2
    648 02B2 8D                    GLO RD
    649 02B3 FF 0F                 SMI 0FH         ..AGAIN, SUBTRACT #0FHEX. FROM SYSTEM PRESS...
    650 02B5 32 BB                 BZ ZR           ..TO PREVENT TABLE UNDERFLOW
    651 02B7 F8 2B                 LDI 02BH
    652 02B9 F4                    ADD             ..JUMP BACK TO UPPER PRESS. LINE
    653 02BA A9                    PLO R9
    654 02BB 49             ZR:    LDA R9          ..STORE UPPER PRESSURE
    655 02BC 73                    STXD
    656 02BD 87                    GLO R7
    657 02BE FC 04                 ADI 04H         ..ADD #04 TO R7.0 ACCUMULATOR
    658 02C0 A7                    PLO R7
    659 02C1 52                    STR R2
    660 02C2 FB 50                 XRI 050H        ..SEE IF THE # OF STEPS IN R7 IS = TO MAXIMUM..
    661 02C4 3A CC                 BNZ KX          ..THE STEM CAN TRAVEL. IF NOT, THEN JUMP TO KX
    662 02C6 F8 50                 LDI 050H        ..IF R7 DOES = #50HEX., THEN PUT #50HEX. INTO..
    663 02C8 AA                    PLO RA          ..RA.0
    664 02C9 60                    IRX
    665 02CA 30 00                 BR RET8         ..RETURN TO MAIN PROGRAM AS A DEFAULT
```

```
    666 02CC 87          KX:   GLO R7        ..DIVIDE R7.0 BY 2
    667 02CD F6                SHR
    668 02CE 52                STR R2        ..STORE RESULT ON STACK
    669 02CF 89                GLO R9
    670 02D0 F4                ADD           ..ADD THIS TO R9.0 TO GET POINTER TO NEXT..
    671 02D1 A9                PLO R9        ..2 BYTE FLOW RATE DATA
    672 02D2 30 72             BR RZ         ..BEGIN DATA GATHERING AGAIN AT RZ
    673 02D4 87          XZ:   GLO R7        ..IF DIFFERENCE < 0 OR = 0,WE HAVE UPPER LIMIT..
    674 02D5 52                STR R2        ..ALONG SYSTEM PRESSURE LINE
    675 02D6 F8 04             LDI 04H       ..SUBTRACT #04 FROM ACCUMULATOR
    676 02D8 F5                SD            ..
    677 02D9 A7                PLO R7        ..
    678 02DA 8D                GLO RD        ..AGAIN, SUBTRACT #0FHEX. FROM SYSTEM PRESS...
    679 02DB FF 0F             SMI 0FH       ..TO PREVENT TABLE UNDERFLOW
    680 02DD 32 E5             BZ HH
    681 02DF 89                GLO R9
    682 02E0 52                STR R2
    683 02E1 F8 2B             LDI 02BH      ..JUMP BACK TO UPPER PRESS. LINE
    684 02E3 F4                ADD
    685 02E4 A9                PLO R9
    686 02E5 9A          HH:   GHI RA        ..STORE NEW INTERPOLATED VALUE IN RF
    687 02E6 BF                PHI RF        ..
    688 02E7 8A                GLO RA        ..
    689 02E8 AF                PLO RF        ..
    690 02E9 49                LDA R9        ..BEGIN BY STORING UPPER PRESS.
    691 02EA 73                STXD
    692 02EB 89                GLO R9
    693 02EC 52                STR R2
    694 02ED 87                GLO R7        ..DIVIDE R7.0 BY 2
    695 02EE F6                SHR           ..
    696 02EF F4                ADD           ..MOVE TABLE POINTER BACK TO 2 BYTE..
    697 02F0 A9                PLO R9        ..TABLE FLOW RATE ENCOUNTERED ON PREVIOUS..
    698 02F1 49                LDA R9        ..INTERPOLATION
    699 02F2 73                STXD          ..STORE FLOW RATE DATA AGAIN
    700 02F3 09                LDN R9
    701 02F4 73                STXD
    702 02F5 8D                GLO RD
    703 02F6 FF 0F             SMI 0FH       ..SUBTRACT #0FHEX. FROM SYSTEM PRESS. TO..
    704 02F8 C2 03 01          LBZ TR        ..PREVENT TABLE UNDERFLOW
    705 02FB 89                GLO R9        ..IF SYSTEM PRESS.= LOWEST TABLE VALUE, JUMP..
    706 02FC 52                STR R2        ..TO TR
    707 02FD F8 2B             LDI 02BH      ..JUMP BACK TO SAME POSITION IN LOWER PRESS...
    708 02FF F5                SD            ..LIMIT
    709 0300 A9                PLO R9        ..STORE FLOW RATE DATA AGAIN
    710 0301 29          TR:   DEC R9        ..
```

```
ER LINE ADDR B1 B2 B3 B4                    1802 ASSEMBLER VER 1.0MR

    711 0302 49                LDA R9          ..
    712 0303 73                STXD            ..
    713 0304 09                LDN R9          ..
    714 0305 73                STXD            ..
    715 0306 87                GLO R7
    716 0307 F6                SHR
    717 0308 52                STR R2
    718 0309 F8 01             LDI 01H         ..SET DF = 01
    719 030B F6                SHR
    720 030C 89                GLO R9
    721 030D F7                SM
    722 030E A9                PLO R9
    723 030F 29                DEC R9
    724 0310 29                DEC R9
    725 0311 09                LDN R9          ..STORE LOWER PRESS. VALUE
    726 0312 73                STXD
    727 0313                   CALL INTER      ..CALL INTERPOLATION SUBROUTINE
    730 0316 60                IRX             ..INCREMENT TO BOTTOM OF STACK
    731 0317 60                IRX             ..
    732 0318 60                IRX             ..
    733 0319 60                IRX             ..
    734 031A 60                IRX             ..
    735 031B 60                IRX             ..
    736 031C 9F                GHI RF
    737 031D 73                STXD            ..STORE PREVIOUS INTERP. FLOW RATE ON STACK
    738 031E 8F                GLO RF          ..
    739 031F 73                STXD            ..
    740 0320 87                GLO R7          ..ADD #04 TO CURRENT VSD VALUE
    741 0321 52                STR R2          ..THIS VALUE CORRESPONDS TO VSD FOR OLD..
    742 0322 F8 04             LDI 04H         ..INTERPOLATED FLOW RATE
    743 0324 F4                ADD
    744 0325 73                STXD            ..STORE OLD ACCUMULATED VSD
    745 0326 87                GLO R7          ..STORE CURRENT VSD
    746 0327 73                STXD
    747 0328 9A                GHI RA          ..STORE CURRENT INTERPOLATED FLOW RATE
    748 0329 73                STXD            ..
    749 032A 8A                GLO RA          ..
    750 032B 73                STXD            ..
    751 032C                   CALL VALDI      ..CALL VALVE DISPLACEMENT SUBROUTINE
    754 032F 60                IRX             ..INCREMENT TO BOTTOM OF STACK
    755 0330 60                IRX             ..
    756 0331 60                IRX             ..
    757 0332 60                IRX             ..
    758 0333 60                IRX             ..
    759 0334 60                IRX             ..
```

```
ER LINE ADDR B1 B2 B3 B4                    1802 ASSEMBLER VER 1.0MP

   760 0335 C0 02 00            LBR RETR          ..RETURN TO CALLING PROGRAM
   761 0338 F8 00         YY:   LDI 00H
   762 033A AA                  PLO RA            ..PUT #00 IN RA.0 AS DEFAULT DISPLACEMENT
   763 033B C0 02 00            LBR RETR          ..RETURN TO CALLING PROGRAM
   764 0400                     PAGE
   765 0400               XY:   RETURN            ..***CLOSE VALVE SUBROUTINE***
   767 0401 R9            CLOSE: GLO R9           ..ADD #02 TO VSD IN R9.0
   768 0402 FC 02               ADI 02H           ..
   769 0404 A9                  PLO R9            ..
   770 0405               XV:   CALL1 OUTPUT,03H  ..OUTPUT #03 ON DATA BUS
   774 0409                     CALL1 OUTPUT,02H  ..OUTPUT #02 ON DATA BUS
   778 040D                     CALL COUNT        ..CALL COUNTDOWN SUBROUTINE
   781 0410 3B 05               BNF XV            ..JUMP TO XV IF COUNTDOWN NOT COMPLETE
   782 0412 R9                  GLO R9            ..SUBTRACT #02 FROM MODIFIED VSD IN R9.0
   783 0413 FF 02               SMI 02H           ..
   784 0415 A9                  PLO R9            ..
   785 0416 F8 00               LDI 00H
   786 0418 AB                  PLO RB            ..RELOAD #0000 INTO RB
   787 0419 BB                  PHI RB
   788 041A                     CALL1 OUTPUT,0AH  ..OUTPUT #0AHEX. ON DATA BUS
   792 041E 30 00               BR XY             ..RETURN TO CALLING PROGRAM
   793 0420               PS:   RETURN            ..***OPEN VALVE SUBROUTINE***
   795 0421 R9            OPEN: GLO R9            ..IF VSD IS = 0, JUMP TO AA
   796 0422 32 31               BZ AA             ..
   797 0424               PO:   CALL1 OUTPUT,01H  ..OUTPUT #01 ON DATA BUS
   801 042B                     CALL1 OUTPUT,00H  ..OUTPUT #00 ON DATA BUS
   805 042C                     CALL COUNT        ..CALL COUNTDOWN SUBROUTINE
   808 042F 3B 24               BNF PO            ..JUMP TO PO IF COUNTDOWN NOT COMPLETE
   809 0431 F8 00         AA:   LDI 00H           ..RESTORE #0000 TO RB
   810 0433 AB                  PLO RB
   811 0434 B9                  PHI R9
   812 0435 30 20               BR PS             ..RETURN TO CALLING PROGRAM
   813 0437               RET4: RETURN            ..***COUNTDOWN SUBROUTINE***
   815 0438 1B            COUNT: INC RB           ..INCREMENT PB
   816 0439 R9                  GLO R9
   817 043A 73                  STXD
   818 043B A0                  IRX
   819 043C PB                  GLO RB            ..COMPARE R9.0 TO PB.0
   820 043D F3                  XOR
   821 043E 3A 45               BNZ PR            ..IF PB <> R9, JUMP TO PR
   822 0440 FB 01               LDI 01H           ..SET DF = #01
   823 0442 F6                  SHR
   824 0443 30 37               BR RET4           ..RETURN TO CALLING PROGRAM
   825 0445 F8 00          PR:  LDI 00H           ..SET DF = #00
   826 0447 F6                  SHR
```

```
ER LINE ADDR B1 B2 B3 B4                    1802 ASSEMBLER VER 1.0MR

    827 0448 30 37          BR RET4         ..RETURN TO CALLING PROGRAM
    828 044A                RET2: RETURN    ..***OUTPUT TO DATA LINES SUBROUTINE***
    830 044A 46             OUTPUT:LDA R6
    831 044C 73                     STXD    ..STORE ON STACK, DATA TO BE OUTPUT
    832 044D 60                     IRX
    833 044E 65                     OUT 5   ..OUTPUT DATA
    834 044F 22                     DEC R2
    835 0450 9F             DELAY: GHI RF   ..**DELAY SUB-SUBROUTINE**
    836 0451 73                     STXD    ..STORE VALUE OF RF ON STACK
    837 0452 FF                     GLO RF  ..
    838 0453 73                     STXD    ..
    839 0454 2F             XX:     DEC RF  ..COUNTDOWN RF UNTIL IT = #0000
    840 0455 8F                     GLO RF  ..
    841 0456 3A 54                  BNZ XX  ..
    842 0458 9F                     GHI RF  ..
    843 0459 32 5F                  BZ XA   ..
    844 045B 2F                     DEC RF  ..
    845 045C 30 54                  BR XX   ..
    846 045E 60             XA:     IRX
    847 045F 72                     LDXA    ..RELOAD ORIGINAL NUMBER INTO RF
    848 0460 AF                     PLO RF  ..
    849 0461 F0                     LDX     ..
    850 0462 BF                     PHI RF  ..
    851 0463 30 4A                  BR RET2 ..RETURN TO CALLING PROGRAM
    852 0500                        PAGE
    853 0500                RET3:  RETURN   ..***INTERPOLATION SUBROUTINE***
    855 0501 60             INTER: IRX      ..MOVE STACK POINTER PAST ADDRESS LOCATIONS..
    856 0502 60                    IRX      ..STORED THERE BY SCRT
    857 0503 60                    IRX      ..
    858 0504 72                    LDXA     ..LOAD STORED DATA INTO REGISTERS
    859 0505 AE                    PLO RE   ..PUT LOW PRESS. INTO RE.0
    860 0506 72                    LDXA
    861 0507 AA                    PLO RA   ..PUT LO BYTE OF LOW FLOW RATE IN RA.0
    862 0508 72                    LDXA
    863 0509 BA                    PHI RA   ..PUT HI BYTE OF LOW FLOW RATE IN RA.1
    864 050A 72                    LDXA
    865 050B AB                    PLO RB   ..PUT LO BYTE OF HIGH FLOW RATE IN RB.0
    866 050C 72                    LDXA
    867 050D BB                    PHI RB   ..PUT HI BYTE OF HIGH FLOW RATE IN RB.1
    868 050F F0                    LDX
    869 050F BE                    PHI RE   ..PUT HIGH PRESS. IN RE.1
    870 0510 8E                    GLO RE
    871 0511 F5                    SD       ..SUBTRACT LOW PRESS. FROM HIGH PRESS.
    872 0512 F6                    SHR      ..DIVIDE DIFFERENCE BY 8
    873 0513 F6                    SHR      ..
```

```
ER LINE ADDR B1 B2 B3 B4                    1802 ASSEMBLER VER 1.0HR

 874 0514 F6                    SHR                ..
 875 0515 BC                    PHI RC             ..PUT PRESS. INCREMENT IN RC.1
 876 0516 F8 01                 LDI 01H            ..SET DF = #01
 877 0518 F6                    SHR
 878 0519 22                    DEC R2             ..DECREMENT STACK TO HIGH FLOW RATE, LO BYTE
 879 051A 22                    DEC R2
 8A0 051B 8A                    GLO RA
 8A1 051C F5                    SD                 ..SUBTRACT FROM THIS, LOW FLOW RATE, LO BYTE
 8A2 051D AB                    PLO RB             ..PUT DIFFERENCE IN RB.0
 8A3 051E 60                    IRX                ..MOVE STACK POINTER TO HI FLOW RATE, HI BYTE
 8A4 051F 9A                    GHI RA
 8A5 0520 75                    SDB                ..SUBTRACT LO FLOW RATE, HI BYTE WITH BORROW
 8A6 0521 BB                    PHI RB             ..PUT DIFFERENCE IN RB.1
 8A7 0522 22                    DEC R2             ..MOVE TO TOP OF STACK
 8A8 0523 22                    DEC R2             ..
 8A9 0524 22                    DEC R2             ..
 8A0 0525 22                    DEC R2             ..
 8A1 0526 22                    DEC R2             ..
 8A2 0527 22                    DEC R2             ..
 8A3 052A 22                    DEC R2             ..
 8A4 0529 F8 00                 LDI 00H            ..SET VALUE OF RD.1 = #00
 8A5 052B BD                    PHI RD             .
 8A6 052C 8B                    GLO RB             ..GET LO BYTE OF DIFFERENCE IN FLOW RATES..
 8A7 052D F6                    SHR                ..AND TEST TO SEE IF LEAST SIGNIFICANT BIT..
 8A8 052E 3B 3B                 BNF RJ             ..(LSB) = 0. IF SO, JUMP TO RJ
 8A9 0530 8B                    GLO RB
 900 0531 FC 01                 ADI 01H            ..ADD #01 TO LO BYTE OF DIFFERENCE
 901 0533 AB                    PLO RB             ..THIS WILL HELP COMPENSATE FOR DIVISION..
 902 0534 9B                    GHI RB             ..TRUNCATION. THIS METHOD REDUCES CUMMULATIVE..
 903 0535 7C 00                 ADCI 00H           ..ERROR BY ALTERNATELY ADDING AND SUBTRACTING..
 904 0537 BB                    PHI RB             ..#01 TO THE FLOW RATE DIFFERENCE
 905 0538 F8 01                 LDI 01H            ..SET RD.1 AS A CHECK FOR ADDITION OR SUBTR.
 906 053A BD                    PHI RD             ..
 907 053B 9B              RJ:   GHI RB             ..DIVIDE HI BYTE OF DIFFERENCE BY 2
 908 053C F6                    SHR                ..
 909 053D BB                    PHI RB
 910 053E 8B                    GLO RB             ..DIVIDE LO BYTE OF DIFFERENCE BY 2 WITH CARRY
 911 053F 76                    SHRC               ..
 912 0540 AB                    PLO RB             ..
 913 0541 8B                    GLO RB
 914 0542 F6                    SHR
 915 0543 3B 50                 BNF RK             ..TEST TO SEE IF NEW DIFFERENCE IS AN EVEN # ..
 916 0545 8B                    GLO RB             ..IF IT IS, JUMP TO RK
 917 0546 FF 01                 SMI 01H            ..SUBTRACT #01 FROM LO BYTE OF DIFFERENCE
 918 0548 AB                    PLO RB             ..
```

```
    910 0549 9R             GHI RB
    920 054A 7F 00          SMBI 00H          ..SUBTRACT #00 FROM HI BYTE OF DIFF. WITH BORROW
    921 054C BB             PHI RB
    922 054D FA 00          LDI 00H           ..SET RD.1 AS A CHECK FOR ADDITION OR SUBTR.
    923 054F BD             PHI RD            ..
    924 0550 9R       RK:   GHI RB            ..DIVIDE HI BYTE OF NEW DIFFERENCE BY 2
    925 0551 F6             SHR               ..
    926 0552 BR             PHI RB            ..
    927 0553 8B             GLO RB            ..DIVIDE LO BYTE OF NEW DIFFERENCE BY 2
    928 0554 76             SHRC              ..
    929 0555 AB             PLO RB            ..
    930 0556 8B             GLO RB
    931 0557 F6             SHR
    932 0558 3B 6F          BNF RL            ..IF NEW DIFF. IS AN EVEN #, JUMP TO RL
    933 055A 9D             GHI RD
    934 055B 3A 67          BNZ PT            ..IF RD.1 <> 0, JUMP TO PT
    935 055D 8R             GLO RB            ..
    936 055E FC 01          ADI 01H           ..ADD #01 TO LO BYTE OF DIFFERENCE
    937 0560 AR             PLO RB            ..
    938 0561 9R             GHI RB            ..
    939 0562 7C 00          ADCI 00H          ..ADD #00 TO HI BYTE OF DIFF. WITH CARRY
    940 0564 BR             PHI RB            ..
    941 0565 30 6F          BR RL             ..JUMP TO RL
    942 0567 8R       PT:   GLO RB            ..THIS SECTION IS TO ALTERNATE THE ADDITION..
    943 0568 FF 01          SMI 01H           ..AND SUBTRACTION IN CASE THE LAST TEST..
    944 056A AR             PLO RB            ..SHOWED AN EVEN NUMBER IN THE LSB. BY DOING..
    945 056B 9B             GHI RB            ..THIS, THE ADDITION OF #01 AGAIN, BEFORE..
    946 056C 7F 00          SMBI 00H          ..#01 IS SUBTRACTED, IS AVOIDED.
    947 056E BB             PHI RB
    948 056F 9B       RL:   GHI RB            ..DIVIDE HI BYTE OF NEW DIFFERENCE BY 2
    949 0570 F6             SHR               ..
    950 0571 BB             PHI RB            ..
    951 0572 8R             GLO RB            ..DIVIDE LO BYTE OF NEW DIFF. BY 2 WITH CARRY
    952 0573 76             SHRC              ..
    953 0574 AB             PLO RB            ..
    954 0575 8D             GLO RD
    955 0576 52             STR R2
    956 0577 8E             GLO RE            ..SUBTRACT LOWER PRESS. FROM SYSTEM PRESS.
    957 0578 F5             SD
    958 0579 33 7F          BDF ZB            ..IF RESULT > OR = 0, JUMP TO ZB
    959 057B FB FF          XRI 0FFH          ..TAKE THE 2'S COMPLIMENT OF RESULT
    960 057D FC 01          ADI 01H           ..
    961 057F AC       ZB:   PLO RC            ..STORE RESULT
    962 0580 FB 01          LDI 01H           ..SET DF = #01
    963 0582 F6             SHR
```

```
 964 0583 8E          RP:   GLO RE
 965 0584 52                STR R2
 966 0585 9C                GHI RC
 967 0586 F4                ADD                 ..ADD PRESS. INCREMENT TO LOWER PRESS.
 968 0587 AF                PLO RE              ..
 969 0588 8A                GLO RA
 970 0589 52                STR R2
 971 058A 8B                GLO RB
 972 058B F4                ADD                 ..ADD LO FLOW RATE INCREMENT TO LOW FLOW..
 973 058C AA                PLO RA              ..RATE,LO BYTE
 974 058D 9A                GHI RA
 975 058E 52                STR R2
 976 058F 9B                GHI RB
 977 0590 74                ADC                 ..ADD HI FLOW RATE INCREMENT TO LOW FLOW..
 978 0591 BA                PHI RA              ..RATE, HI BYTE
 979 0592 8D                GLO RD
 980 0593 52                STR R2
 981 0594 8E                GLO RE
 982 0595 F5                SD                  ..SUBTR. MODIFIED LOWER PRESS.FROM SYSTEM PRESS.
 983 0596 33 9C             BDF ZC              ..IF RESULT IS > OR = 0, JUMP TO ZC
 984 0598 FB FF             XRI 0FFH            ..TAKE THE 2'S COMPLIMENT OF RESULT
 985 059A FC 01             ADI 01H
 986 059C B9          ZC:   PHI R9              ..PUT RESULT IN R9.1
 987 059D F8 01             LDI 01H             ..SET DF = #01
 988 059F F6                SHR
 989 05A0 8C                GLO RC
 990 05A1 52                STR R2
 991 05A2 99                GHI R9
 992 05A3 F5                SD                  ..SUBTRACT NEW DIFF. IN PRESS. FROM OLD DIFF.
 993 05A4 3B AC             BNF ZZ              ..IF RESULT IS < OR = 0, JUMP TO ZZ
 994 05A6 3A AC             BZ ZZ               ..
 995 05A8 99                GHI R9
 996 05A9 AC                PLO RC              ..STORE NEW DIFF.IN PRESS.IN OLD DIFF.HOLDING..
 997 05AA 30 83             BR RP               ..AREA. JUMP TO RP
 998 05AC 8A          ZZ:   GLO RA
 999 05AD 52                STR R2
1000 05AE 8B                GLO RB
1001 05AF F5                SD                  ..SUBTRACT FLOW RATE INCREMENT FROM LATEST ..
1002 05B0 AA                PLO RA              ..MODIFIED LO FLOW RATE
1003 05B1 9A                GHI RA              ..
1004 05B2 52                STR R2              ..
1005 05B3 9B                GHI RB              ..
1006 05B4 75                SDB                 ..RESULTING INTERPOLATED FLOW RATE STORED IN RA
1007 05B5 BA                PHI RA
1008 05B6 F8 00             LDI 00H
```

```
1009 0588 AB                    PLO RB                  ..SET RB = #0000
1010 0589 BB                    PHI RB
1011 058A F8 08                 LDI A.1(FLODAT)         ..RELOAD ADDRESS OF FLOW RATE TABLE BACK..
1012 058C B9                    PHI R9                  ..INTO R9.1
1013 058D 30 00                 BR RET5                 ..RETURN TO CALLING PROGRAM
1014 0600                       PAGE
1015 0600                RET3:  RETURN
1017 0601 60             VALDI: IRX                     ..***VALVE DISPLACEMENT SUBROUTINE***
1018 0602 60                    IRX                     ..INTERPOLATES THE CORRECT VALVE DISPLACEMENT
1019 0603 60                    IRX
1020 0604 72                    LDXA                    ..LOAD NECESSARY DATA INTO REGISTERS
1021 0605 AA                    PLO RA                  ..PUT LOWER FLOW RATE, LO BYTE, IN RA.0
1022 0606 72                    LDXA
1023 0607 BA                    PHI RA                  ..PUT LOWER FLOW RATE , HI BYTE, IN RA.1
1024 0608 72                    LDXA
1025 0609 AE                    PLO RE                  ..PUT LOWER VSD IN RE.0
1026 060A 72                    LDXA
1027 060B BE                    PHI RE                  ..PUT UPPER VSD IN RE.1
1028 060C 72                    LDXA
1029 060D A8                    PLO R8                  ..PUT UPPER FLOW RATE, LO BYTE, IN R8.0
1030 060F F0                    LDX
1031 060F B8                    PHI R8                  ..PUT UPPER FLOW RATE, HI BYTE, IN R8.1
1032 0610 22                    DEC R2                  ..MOVE STK.POINTER TO LO BYTE OF UPPER FLOW RATE
1033 0611 FB 01                 LDI 01H                 ..SET DF = #01
1034 0613 F6                    SHR                     ..
1035 0614 8A                    GLO RA
1036 0615 F5                    SD                      ..SUBTR. FROM THIS THE LOWER FLOW RATE, LO BYTE
1037 0616 AB                    PLO RB
1038 0617 60                    IRX                     ..MOVE STK.POINTER TO HI BYTE OF UPPER FLOW RATE
1039 0618 9A                    GHI RA
1040 0619 75                    SDB                     ..SUBTR. FROM THIS THE LOWER FLOW RATE, HI BYTE
1041 061A BB                    PHI RB
1042 061B 8B                    GLO RB                  ..CHECK THE LSB IN RB FOR #00
1043 061C F6                    SHR
1044 061D 3B 27                 BNF AJ                  ..IF LSB IN RB = #00, JUMP TO AJ
1045 061F 8B                    GLO RB
1046 0620 FC 01                 ADI 01H                 ..OTHERWISE, ADD #01 TO RB TO PREVENT..
1047 0622 AB                    PLO RB                  ..DIVISION TRUNCATION.
1048 0623 9B                    GHI RB                  ..
1049 0624 7C 00                 ADCI 00H                ..
1050 0626 BB                    PHI RB                  ..
1051 0627 9B             AJ:    GHI RB
1052 0628 F6                    SHR                     ..DIVIDE FLOW RATE DIFFERENCE, HI BYTE, BY 2
1053 0629 BB                    PHI RB                  ..
1054 062A 8B                    GLO RB
```

```
FK LINE ADDR B1 B2 B3 B4                    1802 ASSEMBLER VER 1.0MR

1055 062B 76                SHRC            ..DIVIDE FLOW RATE DIFF., LO BYTE, BY 2 W/ CARRY
1056 062C AB                PLO RB
1057 062D 8B                GLO RB
1058 062E F6                SHR
1059 062F 3B 39             BNF AK          ..IF LSB IN RB = #00, JUMP TO AK
1060 0631 8B                GLO RB
1061 0632 FF 01             SMI 01H         ..OTHERWISE, SUBTRACT #01 FROM RB TO PREVENT..
1062 0634 AB                PLO RB          ..DIVISION TRUNCATION.
1063 0635 9B                GHI RB          ..
1064 0636 7F 00             SMBI 00H        ..
1065 0638 BB                PHI RB          ..
1066 0639 9B            AK: GHI RB
1067 063A F6                SHR             ..DIVIDE HI BYTE OF FLOW RATE DIFFERENCE BY 2
1068 063B BB                PHI RB          ..
1069 063C 8B                GLO RB
1070 063D 76                SHRC            ..DIVIDE LO BYTE OF FLOW RATE DIFF.BY 2 W/ CARRY
1071 063E AB                PLO RB
1072 063F 22                DEC R2          ..MOVE STACK POINTER TO UPPER VSD
1073 0640 22                DEC R2          ..
1074 0641 8E                GLO RE
1075 0642 F5                SD              ..SUBTRACT FROM THIS THE LOWER VSD
1076 0643 F6                SHR             ..DIVIDE DIFFERENCE BY 4
1077 0644 F6                SHR             ..
1078 0645 BE                PHI RE
1079 0646 22                DEC R2          ..GO TO TOP OF STACK
1080 0647 22                DEC R2          ..
1081 0648 22                DEC R2          ..
1082 0649 22                DEC R2          ..
1083 064A 22                DEC R2          ..
1084 064B 22                DEC R2          ..
1085 064C 88                GLO R8          ..GET LO BYTE OF REQUESTED FLOW RATE
1086 064D 52                STR R2
1087 064E 8A                GLO RA          ..SUBTR. FROM THIS THE LOWER FLOW RATE, LO BYTE
1088 064F F5                SD              ..
1089 0650 AC                PLO RC          ..STORE RESULT IN RC.0
1090 0651 98                GHI R8          ..GET HI BYTE OF REQUESTED FLOW RATE
1091 0652 52                STR R2
1092 0653 9A                GHI RA
1093 0654 75                SDB             ..SUBTR. FROM THIS THE LOWER FLOW RATE, HI BYTE
1094 0655 BC                PHI RC          ..STORE RESULT IN RC.1
1095 0656 FB 01             LDI 01H         ..SET DF = #01
1096 0658 F6                SHR
1097 0659 8A            PP: GLO RA
1098 065A 52                STR R2
1099 065B 8B                GLO RB          ..ADD LOWER FLOW RATE INCREMENT TO LOWER..
```

```
ER LINE ADDR B1 B2 B3 B4                    1802 ASSEMBLER VER 1.0MR

   1100 065C F4                 ADD          ..FLOW RATE, LO BYTE
   1101 065D AA                 PLO RA       ..
   1102 065E 9A                 GHI RA
   1103 065F 52                 STR R2
   1104 0660 98                 GHI R8       ..ADD UPPER FLOW RATE INCREMENT TO LOWER..
   1105 0661 74                 ADC          ..FLOW RATE, HI BYTE
   1106 0662 BA                 PHI RA       ..
   1107 0663 8E                 GLO RE
   1108 0664 52                 STR R2
   1109 0665 9E                 GHI RE       ..ADD VSD INCREMENT TO LOWER VSD
   1110 0666 F4                 ADD          ..
   1111 0667 AE                 PLO RE
   1112 0668 88                 GLO R8
   1113 0669 52                 STR R2
   1114 066A 8A                 GLO RA
   1115 066B F5                 SD           ..SUBTRACT LO BYTE OF MODIFIED LOWER FLOW..
   1116 066C A7                 PLO R7       ..RATE FROM REQUESTED LO BYTE
   1117 066D 98                 GHI R8
   1118 066E 52                 STR R2
   1119 066F 9A                 GHI RA       ..SUBTRACT HI BYTE OF MODIFIED LOWER FLOW..
   1120 0670 75                 SDB          ..RATE FROM REQUESTED HI BYTE, WITH BORROW
   1121 0671 B7                 PHI R7
   1122 0672 33 80              BDF XH       ..IF DIFFERENCE > OR = 0, JUMP TO XH
   1123 0674 87                 GLO R7
   1124 0675 FB FF              XRI 0FFH     ..TAKE THE 2'S COMPLIMENT OF RESULT LO BYTE
   1125 0677 FC 01              ADI 01H      ..
   1126 0679 A7                 PLO R7
   1127 067A 97                 GHI R7
   1128 067B FB FF              XRI 0FFH     ..TAKE THE 2'S COMPLIMENT OF RESULT HI BYTE
   1129 067D 7C 00              ADCI 00H     ..
   1130 067F B7                 PHI R7
   1131 0680 F8 01          XH: LDI 01H      ..SET DF = #01
   1132 0682 F6                 SHR
   1133 0683 8C                 GLO RC
   1134 0684 52                 STR R2
   1135 0685 87                 GLO R7       ..SUBTRACT NEW DIFF. FROM OLD DIFF., LO BYTE
   1136 0686 F5                 SD           ..
   1137 0687 73                 STXD
   1138 0688 9C                 GHI RC
   1139 0689 52                 STR R2       ..SUBTRACT NEW DIFF. FROM OLD DIFF., HI BYTE..
   1140 068A 97                 GHI R7       ..WITH BORROW
   1141 068B 75                 SDB          ..
   1142 068C 60                 TRX
   1143 068D 3B 9A              BNF GG       ..IF DIFFERENCE < 0, JUMP TO GG
   1144 068F 3A 94              BNZ LJ       ..IF DIFFERENCE > 0, JUMP TO LJ
```

```
1145 0691 F0                LDX
1146 0692 32 9A             BZ GG                   ..IF DIFFERENCE = 0, JUMP TO GG
1147 0694 87          LJ:   GLO R7
1148 0695 AC                PLO RC                  ..STORE NEW DIFF. IN OLD DIFF. HOLDING AREA
1149 0696 97                GHI R7                  ..
1150 0697 BC                PHI RC                  ..
1151 0698 30 59             BR PP                   ..JUMP TO PP
1152 069A 8E          GG:   GLO RE
1153 069B 52                STR R2
1154 069C 9E                GHI RE                  ..SUBTRACT VSD INCREMENT FROM LATEST MODIFIED..
1155 069D F5                SD                      ..LOWER VSD
1156 069E AA                PLO RA                  ..STORE INTERPOLATED VSD IN RA.0
1157 069F F8 00             LDI 00H                 ..PUT #00 IN R8 AND RA.1
1158 06A1 A8                PLO R8                  ..
1159 06A2 B8                PHI R8                  ..
1160 06A3 BA                PHI RA                  ..
1161 06A4 F8 08             LDI A.1(FLODAT)         ..LOAD ADDRESS OF FLOW RATE TABLE..
1162 06A6 B9                PHI R9                  ..BACK INTO R9
1163 06A7 30 00             BR RET3                 ..RETURN TO CALLING PROGRAM
1164 0700                   PAGE
1165 0700             RET7: RETURN                  ..***VALVE STEM POSITION SENSING SUBROUTINE***
1167 0701 6F          LVDT: INP 7                   ..GET INPUT OF POSITION SENSOR
1168 0702 A7                PLO R7                  ..STORE IN R7.0
1169 0703 8E                GLO RE                  ..GET STORED VALUE OF PREVIOUS POSITION
1170 0704 F3                XOR                     ..COMPARE CURRENT VALUE TO PREVIOUS VALUE
1171 0705 32 00             BZ RET7                 ..IF EQUAL, RETURN TO CALLING PROGRAM
1172 0707 8A                GLO RA                  ..STORE INTERPOLATED VSD IN ..
1173 0708 A8                PLO R8                  ..R8.0
1174 0709 F8 01             LDI 01H                 ..SET DF = #01
1175 070B F6                SHR
1176 070C F8 09             LDI A.1(VTDAT)          ..LOAD POSITION SENSING DATA TABLE ADDRESS..
1177 070E B8                PHI R8                  ..INTO R8
1178 070F 08                LDN R8
1179 0710 52                STR R2
1180 0711 AE                PLO RE                  ..LOAD TABLE READING OF CURRENT VSD INTO RE.0
1181 0712 87                GLO R7
1182 0713 F5                SD                      ..SUBTRACT FROM THIS ACTUAL POSITION READING
1183 0714 33 1A             BDF LM                  ..IF DIFFERENCE > OR = 0, JUMP TO LM
1184 0716 F8 FF             XRI 0FFH                ..TAKE THE 2'S COMPLIMENT OF THE RESULT
1185 0718 FC 01             ADI 01H                 ..
1186 071A 52          LM:   STR R2                  ..STORE DIFFERENCE ON STACK
1187 071B F8 03             LDI 03H
1188 071D F5                SD                      ..SUBTRACT #03 FROM DIFFERENCE
1189 071E 3B 00             BNF RET7                ..IF RESULT < 0, RETURN TO CALLING PROGRAM
1190 0720 32 00             BZ RET7                 ..IF RESULT = 0, RETURN TO CALLING PROGRAM
```

```
ER LINE ADDR B1 B2 B3 B4                    1802 ASSEMBLER VER 1.0MR

   1191 0722 8A                     GLO RA
   1192 0723 A9                     PLO R9                  ..STORE VSD IN R9.0
   1193 0724 FF 10                  LDI 010H                ..LOAD #10HEX. INTO RF.1 FOR COUNTDOWN
   1194 0726 BF                     PHI RF                  ..
   1195 0727                        CALL1 OUTPUT,044H       ..CLOSE UPSTREAM SOLENOID
   1199 072A                        CALL1 OUTPUT,04H        ..CUT OFF POWER TO ACTUATOR
   1203 072F F8 00                  LDI 00H                 ..SET RB = #0000
   1204 0731 B8                     PHI RB                  ..
   1205 0732 A8                     PLO RB                  ..
   1206 0733 89                     GLO R9                  ..CHECK TO SEE IF VSD = 0
   1207 0734 32 39                  BZ LK                   ..IF SO, JUMP TO LK
   1208 0736                        CALL CLOSE              ..CLOSE VALVE COMPLETELY
   1211 0739 F8 18          LK:     LDI 18H                 ..LOAD #18HEX. INTO RF.1 FOR COUNTDOWN
   1212 073B BF                     PHI RF                  ..
   1213 073C                        CALL1 OUTPUT,014H       ..RESET LINEAR ENCODER INTERFACE CIRCUIT
   1217 0740                        CALL1 OUTPUT,04H        ..CUT OFF POWER TO ACTUATOR
   1221 0744 FF 10                  LDI 010H                ..LOAD #10HEX. INTO RF.1 FOR COUNTDOWN
   1222 0746 BF                     PHI RF                  ..
   1223 0747                        CALL OPEN               ..OPEN VALVE TO CORRECT DISPLACEMENT
   1226 074A                        CALL1 OUTPUT,084H       ..OPEN UPSTREAM SOLENOID
   1230 074E                        CALL1 OUTPUT,04H        ..CUT OFF POWER TO ACTUATOR
   1234 0752 30 00                  BR RET7                 ..RETURN TO CALLING PROGRAM
   1235 0800                        PAGE
   1236 0800 0F 00 00 00    FLODAT: DC 0FH,00H,00H,00H,46H,00H,87H      ..FLOW RATE DATA TABLE..15 PSI LINE
   1237 0804 46 00 87
   1238 0807 00 B9 00 E6            DC 00H,0B9H,00H,0E6H,01H,0EH
   1239 080B 01 0E
   1240 080D 01 36 01 54            DC 01H,36H,01H,54H,01H,72H
   1241 0811 01 72
   1242 0813 01 86 01 9A            DC 01H,86H,01H,9AH,01H,0A9H
   1243 0817 01 A9
   1244 0819 01 BD 01 CC            DC 01H,0BDH,01H,0CCH,01H,0D6H
   1245 081D 01 D6
   1246 081F 01 E0 01 E0            DC 01H,0E0H,01H,0E0H,01H,0E0H
   1247 0823 01 F0
   1248 0825 01 E0 01 E0            DC 01H,0E0H,01H,0E0H,01H,0E0H
   1249 0829 01 E0
   1250 082B 17 00 00 00            DC 17H,00H,00H,00H,69H,00H,084H     ..23 PSI LINE
   1251 082F 69 00 84
   1252 0832 00 FA 01 40            DC 00H,0FAH,01H,40H,01H,7CH
   1253 0836 01 7C
   1254 0838 01 A9 01 D6            DC 01H,0A9H,01H,0D6H,01H,0FEH
   1255 083C 01 FE
   1256 083E 02 1C 02 3A            DC 02H,1CH,02H,3AH,02H,4EH
   1257 0842 02 4F
```

```
ER LINE ADDR B1 B2 B3 B4                    1802 ASSEMBLER VER 1.0MR

   125A 0844 02 6C 02 7B          DC 02H,6CH,02H,7BH,02H,8AH
   1259 0848 02 8A
   1260 084A 02 99 02 A3          DC 02H,99H,02H,0A3H,02H,0A8H
   1261 084F 02 A8
   1262 0850 02 A8 02 A8          DC 02H,0A8H,02H,0A8H,02H,0A8H
   1263 0854 02 A8
   1264 0856 1F 00 00 00          DC 1FH,00H,00H,00H,8CH,00H,0E6H     ..31 PSI LINE
   1265 085A 8C 00 E6
   1266 085D 01.45 01 9A          DC 01H,45H,01H,9AH,01H,0E0H
   1267 0861 01 E0
   1268 0863 02 21 02 5A          DC 02H,21H,02H,5AH,02H,85H
   1269 0867 02 85
   1270 0869 02 AD 02 D0          DC 02H,0ADH,02H,0D0H,02H,0E9H
   1271 086D 02 E9
   1272 086F 03 07 03 20          DC 03H,07H,03H,20H,03H,34H
   1273 0873 03 34
   1274 0875 03 43 03 52          DC 03H,43H,03H,52H,03H,5CH
   1275 0879 03 5C
   1276 087A 03 5C 03 5C          DC 03H,5CH,03H,5CH,03H,5CH
   1277 087F 03 5C
   1278 0881 27 00 00 00          DC 27H,00H,00H,00H,0AFH,01H,1DH     ..39 PSI LINE
   1279 0885 AF 01 1D
   1280 0888 01 8B 01 F4          DC 01H,8BH,01H,0F4H,02H,44H
   1281 088C 02 44
   1282 088F 02 94 02 D5          DC 02H,94H,02H,0D5H,03H,0CH
   1283 0893 03 0C
   1284 0894 03 34 03 5C          DC 03H,34H,03H,5CH,03H,7FH
   1285 0898 03 7F
   1286 089A 03 A2 03 BB          DC 03H,0A2H,03H,0BBH,03H,0D4H
   1287 089E 03 D4
   1288 08A0 03 E8 03 F7          DC 03H,0E8H,03H,0F7H,04H,01H
   1289 08A4 04 01
   1290 08A6 04 01 04 01          DC 04H,01H,04H,01H,04H,01H
   1291 08AA 04 01
   1292 08AC 2F 00 00 00          DC 2FH,00H,00H,00H,0D7H,01H,4FH     ..47 PSI LINE
   1293 08B0 D7 01 4F
   1294 08B3 01 D1 02 3F          DC 01H,0D1H,02H,3FH,02H,0A8H
   1295 08B7 02 A8
   1296 08B9 02 FD 03 4D          DC 02H,0FDH,03H,4DH,03H,89H
   1297 08BD 03 89
   1298 08BF 03 BB 03 E8          DC 03H,0BBH,03H,0E8H,04H,10H
   1299 08C3 04 10
   1300 08C5 04 38 04 56          DC 04H,38H,04H,56H,04H,74H
   1301 08C9 04 74
   1302 08CB 04 B8 04 9C          DC 04H,88H,04H,9CH,04H,0A1H
```

```
ER LINE ADDR B1 B2 B3 B4                    1802 ASSEMBLER VER 1.0MR

 1303 08CE 04 A1
 1304 08D1 04 A1 04 A1          DC 04H,0A1H,04H,0A1H,04H,0A1H
 1305 08D5 04 A1
 1306 0900                      PAGE
 1307 0900 00 02 04 05  VTDAT: DC 00H,02H,04H,05H,07H,09H,0CH,0EH    ..DISPLACEMENT DATA TABLE
 1308 0904 07 09 0C 0E
 1309 0908 10 13 15 18          DC 10H,13H,15H,18H,1AH,1DH,20H,22H
 1310 090C 1A 1D 20 22
 1311 0910 25 27 2A 2C          DC 25H,27H,2AH,2CH,2EH,31H,33H,36H
 1312 0914 2E 31 33 36
 1313 0918 38 38 3D 40          DC 38H,38H,3DH,40H,42H,45H,47H,4AH
 1314 091C 42 45 47 4A
 1315 0920 4C 4E 51 53          DC 4CH,4EH,51H,53H,55H,58H,5AH,5DH
 1316 0924 55 58 5A 5D
 1317 0928 5F 61 64 66          DC 5FH,61H,64H,66H,68H,6AH,6CH,6EH
 1318 092C 68 6A 6C 6E
 1319 0930 70 72 75 77          DC 70H,72H,75H,77H,79H,7BH,7EH,80H
 1320 0934 79 7B 7F 80
 1321 0938 82 85 88 8A          DC 82H,85H,88H,8AH,8DH,8FH,92H,94H
 1322 093C 8D 8F 92 94
 1323 0940 96 98 9A 9C          DC 96H,98H,9AH,9CH,9EH,9FH,0A0H,0A0H
 1324 0944 9E 9F A0 A0
 1325 0948 A1 A2 A3 A3          DC 0A1H,0A2H,0A3H,0A3H,0A4H,0A5H,0A6H,0A6H,0A7H
 1326 094C A4 A5 A6 A6
 1327 0950 A7
 1328 0951                      END

 ASSEMBLER ERRORS =    0
```

APPENDIX D

DATA

Contained in the following pages are:

Table D.1 Preliminary calibration data

Table D.2 Linear encoder calibration table

Table D.1 Preliminary calibration data

	P_u = 30 psig			P_u = 40 psig			P_u = 50 psig			P_u = 60 psig			P_u = 70 psig		
x_s (steps)	Δp (psig)	Q_o (SCCS)	L_s (volts)	Δp (psig)	Q_o (SCCS)	L_s (volts)	Δp (psig)	Q_o (SCCS)	L_s (volts)	Δp (psig)	Q_o (SCCS)	L_s (volts)	Δp (psig)	Q_o (SCCS)	L_s (volts)
0	28	0	11.28	38	0	11.27	49	0	11.27	59	0	11.33	69	0	11.28
4	28	241	11.32	38	293	11.31	47	359	11.32	57	411	11.37	67	496	11.32
8	26	481	11.39	35	599	11.38	44	732	11.38	54	901	11.44	62	1038	11.39
12	24	698	11.46	32	883	11.45	41	1067	11.45	49	1270	11.51	57	1430	11.46
16	21	883	11.53	29	1128	11.52	36	1359	11.52	44	1576	11.57	51	1784	11.53
20	19	1005	11.59	26	1298	11.58	33	1562	11.59	39	1812	11.64	45	2058	11.59
24	17	1114	11.66	23	1416	11.66	29	1713	11.66	35	1996	11.71	40	2270	11.67
28	15	1189	11.73	20	1506	11.72	26	1798	11.73	31	2133	11.78	36	2426	11.73
32	13	1237	11.80	18	1576	11.79	23	1888	11.80	28	2228	11.84	33	2534	11.80
36	12	1284	11.86	17	1624	11.86	21	1949	11.86	26	2289	11.90	30	2596	11.87
40	12	1312	11.92	16	1652	11.92	20	1996	11.92	24	2331	11.96	28	2657	11.93
44	11	1326	11.98	15	1680	11.98	19	2025	11.98	23	2364	12.02	26	2700	11.99
48	10	1340	12.04	14	1713	12.04	18	2044	12.04	22	2412	12.09	25	2733	12.05
52	10	1359	12.10	14	1713	12.10	17	2072	12.10	21	2440	12.15	24	2761	12.11
56	10	1373	12.16	13	1727	12.15	16	2086	12.16	20	2440	12.20	23	2794	12.17
60	9	1373	12.21	13	1741	12.21	16	2086	12.22	19	2440	12.25	22	2808	12.22
64	9	1388	12.27	12	1756	12.26	15	2105	12.26	19	2473	12.30	21	2822	12.28
68	9	1388	12.32	12	1770	12.32	15	2119	12.32	18	2487	12.35	21	2841	12.33
72	8	1388	12.37	12	1770	12.37	15	2133	12.37	18	2501	12.40	20	2841	12.38
76	8	1388	12.38	12	1770	12.38	15	2133	12.39	18	2501	12.41	20	2841	12.38
80	8	1388	12.39	12	1770	12.39	15	2133	12.39	18	2501	12.42	20	2855	12.39

Table D.2 Linear encoder calibration table

x_s (steps)	E_s (counts)	x_s (steps)	E_s (counts)
00	00	41	97
01	02	42	100
02	04	43	102
03	05	44	104
04	07	45	106
05	09	46	108
06	12	47	110
07	14	48	112
08	16	49	114
09	19	50	117
10	21	51	119
11	24	52	121
12	26	53	123
13	29	54	126
14	32	55	128
15	34	56	130
16	37	57	133
17	39	58	136
18	42	59	138
19	44	60	141
20	46	61	143
21	49	62	146
22	51	63	148
23	54	64	150
24	56	65	152
25	59	66	154
26	61	67	156
27	64	68	158
28	66	69	159
29	69	70	160
30	71	71	160
31	74	72	161
32	76	73	162
33	78	74	163
34	81	75	163
35	83	76	164
36	85	77	165
37	88	78	166
38	90	79	166
39	93	80	167
40	95		

APPENDIX E

SYSTEM STABILITY

The developed assembly was a static system and therefore inherently stable in response to changes in the requested volumetric flow rate. There were two reasons for this design:

(1) Because of the indirect control techniques used, a finite amount of computation time was required by the microprocessor system to calculate the valve stem displacement for each command change. In classical controls, such a delay in the response could cause a system to become unstable [9]. Therefore, the developed system was designed to operate in a non-continuous mode. The control algorithm was written such that during computation, the actuator position was held stationary and the microprocessor system would not accept any more inputs. Thus, the developed system is considered static.

(2) As mentioned in Chap. 3, an upstream assembly was added to interrupt the flow while the valve stem was repositioned. This was because the actuator was not capable of supplying sufficient force to move the valve stem as long as there was flow through the valve. This was the major reason for designing the assembly as a static system.

It should be noted that in cases where the requested volumetric flow rate remained unchanged, the system would reposition in the presence of upstream pressure fluctuations. Due to the long computational time requirement, the assembly would not be able to respond to any fast changing pressure fluctuation. However, it is to be expected that the source pressure fluctuation would be minimal. It was noticed that perturbations in the source pressure, due to the repositioning, were present in the system; but were damped out before the end of the control algorithm cycle.