# NEW ASPECTS OF DATA ACQUISITION AND REDUCTION IN GEL PERMEATION CHROMATOGRAPHY

by

William G. Walther, Bs. Che.

A Thesis

Submitted to the Faculty of Graduate Studies
in Partial Fulfilment of the Requirements
for the Degree of Master of Engineering

McMaster University

January 1972

MASTER OF ENGINEERING (1972) McMASTER UNIVERSITY
(Chemical Engineering) Hamilton, Ontario

TITLE: NEW ASPECTS OF DATA ACQUISITION AND REDUCTION IN GEL PERMEATION CHROMATOGRAPHY

AUTHOR: William G. Walther, Bs. Che. (University of Delaware)

SUPERVISORS: Dr. A.E. Hamielec
Dr. J.D. Wright

NUMBER OF PAGES: 203

SCOPE AND CONTENTS:

The work of this study is divided into two parts. Part I reports on the development of a dedicated minicomputer, data acquisition, and reduction system for GPC. The hardware, software, and operating performance of the system is discussed in some detail.

Part II, reports on an experimental study design to determine whether axial dispersion corrections are universal in the sense of being independent of polymer composition. Results for poly(vinyl-chloride), polystyrene, polybutadiene, and poly(methy-methacrylate) are discussed.

## ACKNOWLEDGEMENTS

The author wishes to thank his supervisors Dr. A.E. Hamielec and Dr. J.D. Wright for their enthusiasm and guidance during this project. In addition, he is indebted to Mr. Toshi Ishige for his patient assistance in testing the minicomputer system, and to Miss Charlotte Traplin for typing the thesis. Finally, he would like to thank his wife Joyce for her forebearance and assistance.

# TABLE OF CONTENTS

## NEW ASPECTS OF DATA ACQUISITION AND REDUCTION IN GEL PERMEATION CHROMATOGRAPHY


| | | PAGE |
|---|---|---|
| 1. | INTRODUCTION | 1 |
| | PART I - The Minicomputer System | 4 |
| 2. | The GPC Minicomputer System: Hardware | 5 |
| | 2:1 Minicomputers | 5 |
| | 2:2 Minicomputer Systems | 5 |
| | 2:3 Central Processing Unit | 10 |
| | 2:4 Interface | 11 |
| | 2:5 Real Time Clock | 12 |
| | 2:6 Teletype | 12 |
| 3. | The GPC Minicomputer System: Software | 14 |
| | 3:1 Basic Formulation | 14 |
| | 3:2 Programming a Minicomputer | 15 |
| | 3:3 Interrupts | 18 |
| | 3:4 Operating System | 19 |
| | 3:5 Calculation Programs | 21 |
| | 3:6 Storage Buffers | 25 |
| 4. | The GPC Minicomputer System: Performance and Observations | 26 |
| | 4:1 Hardware | 26 |
| | 4:2 Software - Calculation Routines | 27 |
| | 4:3 Software - As a System | 29 |
| | PART II - Experimental Studies | 33 |
| 5. | Gel Permeation Chromatography | 34 |
| | 5:1 Introduction | 34 |
| | 5:2 Characterization of a Molecular Weight Distribution | 35 |
| | 5:3 Calibration | 35 |
| | 5:4 Interpreting a Chromatogram | 41 |
| 6. | Experimental | 47 |
| | 6:1 Operating Conditions | 47 |
| | 6:2 Standards | 48 |

# TABLE OF CONTENTS

| | | PAGE |
|---|---|---|
| 7. | Results and Discussion | 50 |
| | 7:1 Calibration Curves | 50 |
| | 7:2 Application of the Correction Curves | 58 |
| 8. | Conclusions | 65 |
| 9. | Recommendations | 66 |
| 10. | Nomenclature | 67 |
| 11. | References | 69 |
| 12. | Appendices | 71 |

## TABLE INDEX


| | | PAGE |
|---|---|---|
| 2-1 | Hardware Costs of the GPC Minicomputer System | 9 |
| 3-1 | Operating Commands | 22 |
| 6-1 | Standards | 49 |
| 7-1 | Spreading Parameters for ASTM Standards | 55 |
| 7-2 | Molecular Weight Averages for NBS Polystyrene Standards | 59 |
| 7-3 | Molecular Weight Averages for PVC | 61 |
| 7-4 | Molecular Weight Averages for PMMA | 62 |
| 7-5 | Molecular Weight Average for PBD | 63 |


## FIGURE INDEX


| | | |
|---|---|---|
| 2-1 | General Purpose Minicomputer System | 7 |
| 2-2 | GPC Minicomputer System | 8 |
| 3-1 | Interrupt Structure and Priority Levels | 20 |
| 4-1 | Core Maps | 31 |
| 7-1 | Polystyrene Calibration Curve | 51 |
| 7-2 | Universal Calibration Curve | 52 |
| 7-3 | Axial Dispersion Calibration Curves | 56 |
| 7-4 | Axial Dispersion Correction Curves | 57 |

# 1. INTRODUCTION

Since its commercial introduction in 1963, gel permeation chromatography (GPC) has found increasing popularity amongst polymer chemists and chemical engineers interested in polymer science. Several practical and theoretical problems have arisen due to the many new applications that have been found for GPC. The work described in this thesis considers two aspects of data acquisition and reduction in GPC. The first part reports on the development of a dedicated minicomputer system for data acquisition and reduction. The second part examines the feasibility of applying universal axial dispersion corrections in GPC.

The dedicated minicomputer system that will be described represents a third and major advance in data acquisition and reduction in GPC. Unlike gas and liquid chromatography, gel permeation chromatography requires several integrations of the detector response to obtain the desired molecular weight averages. A digital computer is normally used for the calculations which are otherwise tedious and time consuming. The initial approach to data acquisition and reduction involved the measurement of chromatogram heights manually from a recorder trace, punching the data onto computer cards, and finally processing them on a large digital computer. This approach is costly in manpower, some detector accuracy is lost, and processed data may not be available for up to one day. The next development was to reduce manpower with the automation of data acquisition. Waters Associates marketed a digital translator that electronically reads and outputs the chromatogram signal and event markers on paper tape. The tape is then used as input for later data processing. The digital translator reduced manpower and increased accuracy, however, it was still necessary to wait up to one day for processed data. The cost of data processing on a large digital computer is also a consideration. Several laboratories have reported using a minicomputer for data acquisition,[3,4,5,24] however, once again the raw data was processed by a large computer.

Another approach to the problem would be to use a large on-line process control computer, such as IBM 1800. This approach has been used at Ohio State University with a PDP-15.[26] These systems are economically justifiable only when the GPC application is a small part of a larger on-line operation. Our alternative was to develop a dedicated minicomputer system.

The minicomputer system developed and described in this thesis acts as both a data acquisition and data reduction system. The minicomputer system samples and stores chromatogram heights and event markers. When this sampling process has been completed, the minicomputer calculates and outputs the molecular weight averages and the molecular weight distribution on a teletype. Usually the molecular weight averages and molecular weight distributions are available a few minutes after the polymer sample has passed through the GPC detector. No further processing by another computer is normally required.

A specific objective of this investigation was to assess the feasibility of constructing a useful system using a minimum configuration minicomputer, containing only 4K words of memory. The obvious reason for doing this was to minimize memory cost. To accomplish this goal it was necessary to minimize the length of each subroutine. Also, many decisions had to be made as to what should, and should not, be included in the software package with respect to labels, error messages, and complexity of the calculation routines.

The objective of the second part of this study was to determine whether axial dispersion corrections in GPC are universal in character. Recent investigations have demonstrated the necessity of correcting the GPC response for axial dispersion and skewing.[1,2] Balke and Hamielec[1] have shown that axial dispersion correction curves can be constructed by calibrating with known standards. The correction curves are then used to correct the molecular weight averages of unknown samples of the same polymer. The experimental work described in Part II examines the possibility of applying axial dispersion curves determined with polystyrene standards, to polybutadiene, poly (vinyl-chloride), and poly(methyl-methacrylate) samples. The extrapolation of correction factors for polystyrene to other polymers is important since too few standards for other polymers are available. If the full advantage of axial dispersion corrections is to be realized, the correction methods must be easy to apply. The availability of a universal correction method would go a long way towards this goal.

# PART I

## The GPC Minicomputer System

## 2. The GPC Minicomputer System: Hardware

In this chapter the hardware comprising the GPC minicomputer system is discussed in some detail. The lay-out and construction of the interface was done by Mr. Ivan Taylor, of Datagen of Canada Ltd., Hull, P.Q. A few references will be made in this chapter to program interrupts. The reader, unfamiliar with this term, is referred to section 3:3 for a detailed discussion of the meaning of an interrupt, and the steps to be followed when one has occurred in a computer system.

### 2:1 Minicomputers

A computer is classified as a minicomputer on the basis of purchase price, word size, and to a certain extent, memory size.[23] The purchase price of the central processing unit with the minimum memory will usually not exceed $10,000.[23] Minicomputers generally have 16 or fewer binary bits per word. Some, including one of the most widely used machines, have a 12-bit word. The minimum memory used is normally 4K words (4096). The maximum memory size varies from machine to machine, but a common figure would be 32K words.

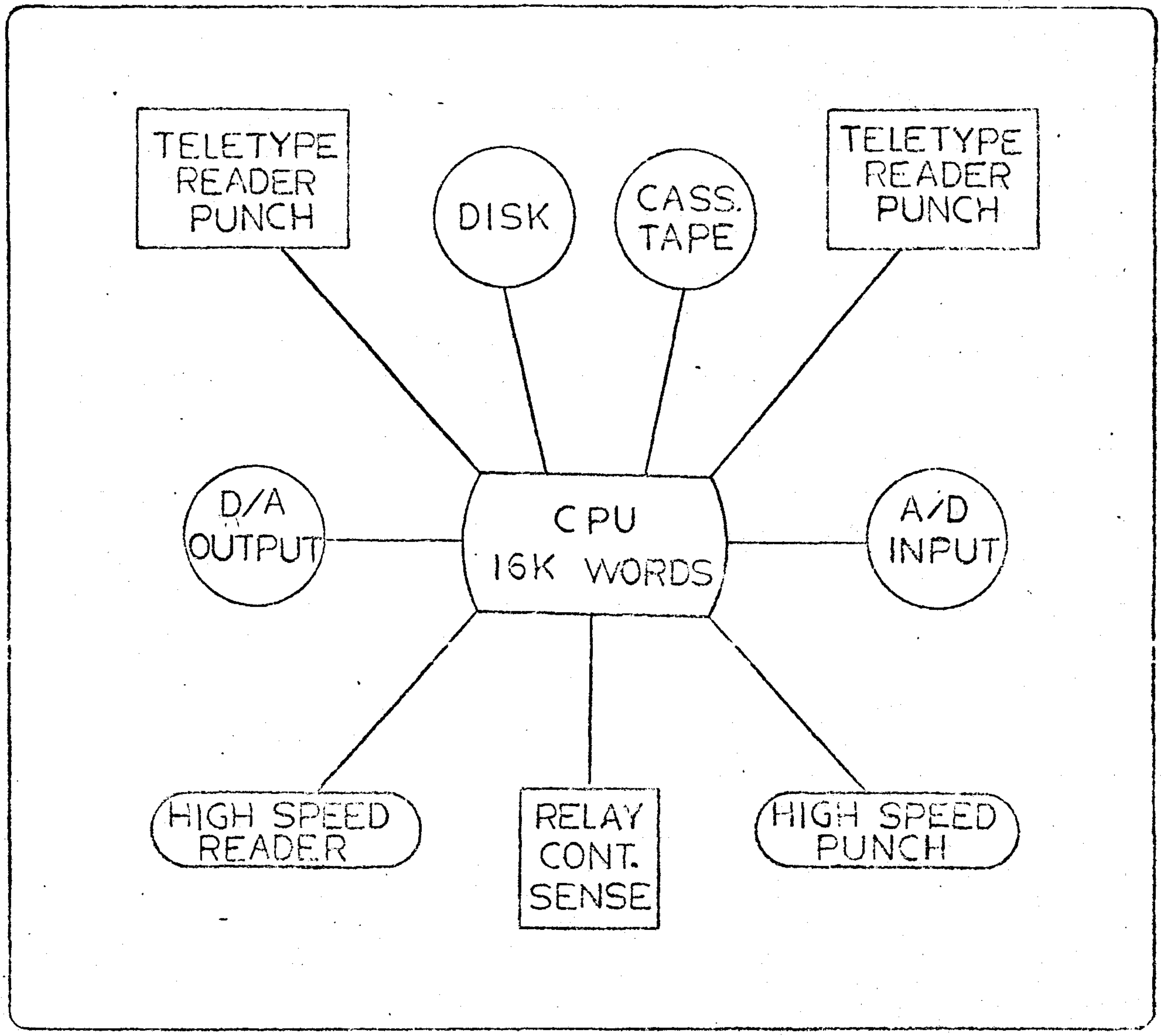
### 2:2 Minicomputer Systems

A minicomputer alone is of limited use without additional input-output (I/O) devices which are required for data and program communication. In a minimum configuration system, a teletype is usually sufficient. More efficient and varied input-output facilities are required for development work and general purpose applications. The programming and I/O efficiency of the minimum configuration system can be increased by adding several other peripheral devices. Most minicomputers may address a number of peripheral devices of which the most commonly used are high speed paper tape readers, high speed paper tape punches, analog to digital (A/D) converters, digital to analog (D/A) convertors, line printers, real-time clocks, cassette tape units, and disks.

A typical general purpose minicomputer system is shown in Figure 2-1. This system is primarily used for control studies, program development, and general purpose computations. All GPC software was developed on the system pictured in Figure 2-1. The system includes a 256K disk, cassette tape unit, 16 channels of A/D input, 6 D/A converters, 16K words of memory, high speed paper tape reader, high speed paper tape punch, 16 relay outputs, 16 contact sense inputs, and two teletypes.

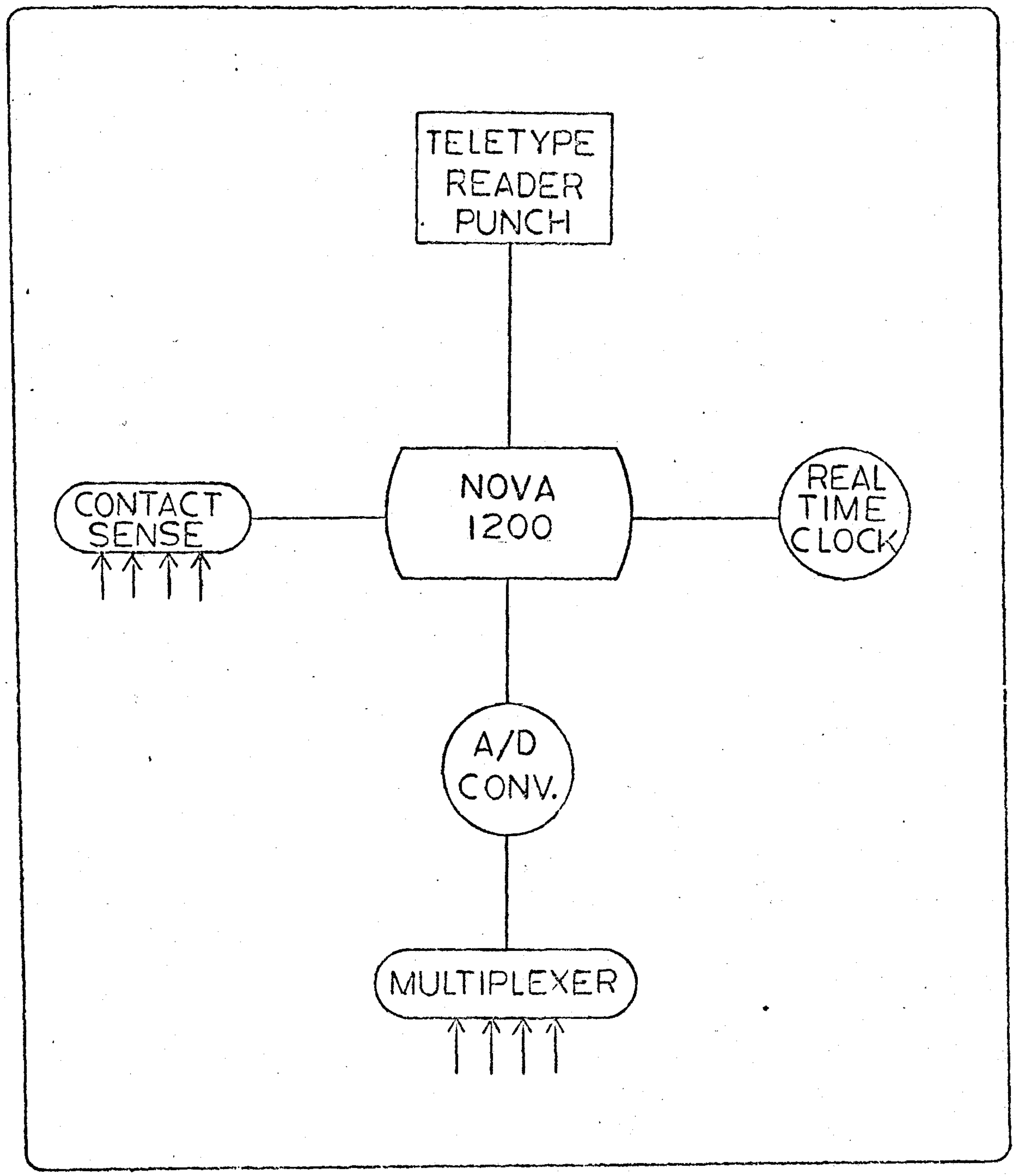
The minicomputer system, designed to be used with several GPC's simultaneously, is shown in Figure 2-2. Design specifications and features will be discussed in subsequent sections. The hardware costs are outlined in Table 2-1. The basic system, costing $10,500, has hardware facilities for operation with four GPC's. With 4K words of memory, the system will operate with only one GPC because of software storage requirements (see section 4:3). An additional 4K words of memory costing $3,000 is sufficient to increase the operating capacity to the design maximum.

The price of this system should drop as cheaper memory units become available. An 8K memory module for a Nova 1210, costing




GENERAL PURPOSE
MINICOMPUTER SYSTEM

FIGURE 2-1




GPC MINICOMPUTER SYSTEM

FIGURE 2-2

| | |
|---|---|
| NOVA 1200 CPU | $2,665 |
| 4K Words 16-bit Memory | 3,000 |
| Real Time Clock | 445 |
| Power Monitor and Autorestart | 445 |
| Teletype Interface | 220 |
| ASR 33 Teletype | 1,300 |
| Interface | 2,500 |
| Total (4K system) | $10,575 * |
| Additional 4K memory | $3,000 |

*Not Including Provincial or Federal Tax

Hardware Costs of the GPC Minicomputer System

Table 2-1

approximately $4,000, has recently been announced. An 8K Nova 1210 GPC system would cost $2,000 less than a 8K Nova 1200 GPC system with two 4K memory modules.

## 2:3 Central Processing Unit (CPU)

The CPU for the GPC minicomputer system is a Nova 1200 made by Datagen of Canada. The Nova 1200 is a 16-bit word machine with a 1.2 microsecond cycle time. It has four accumulators and I/O facilities for 64 separate devices. Memory may be expanded up to 32K words. The Nova 1200 was chosen over other minicomputers for several reasons.

Firstly, the Chemical Engineering Department had a general purpose Datagen minicomputer system that could be used for development work. The efficient input-output fascilities of the general purpose system reduced loading, editing and testing times. Secondly, a 16-bit machine was preferred over machines with smaller words sizes. A 16-bit machine requires only two words for a floating point number, while a 12-bit machine requires three for similar accuracy. Also, a more powerful instruction set is possible with a 16-bit machine than with a 12-bit machine because of the extra word length. Finally, Datagen was willing to assist us in the development of our system, which substantially reduced development time and cost.

There are limitations, however, on the hardware manipulations that can be performed by a minicomputer because of the small word size. In the Nova 1200 there are no hardware floating point operations. Hardware fixed point multiply and divide is optionally available but is not included in the current system. All floating point manipulations and multiply/divide operations, therefore, are done by subroutines. The increased processing time for these operations is not a serious constraint for GPC because calculation times do not interfere with real-time operation.

## 2:4 Interface

The interface includes a 10-bit analog to digital converter (A/D), a four channel multiplexer, four constant gain amplifiers, and eight contact sense lines. It was designed and built by Datagen to operate with four GPC's simultaneously. The original unit is external to the computer and requires a separate power supply.

The interface was designed to accept 10 millivolt or 100 millivolt full scale analog inputs. The Burr-Brown amplifiers, multiplexer and A/D converter transmit an instantaneous digital value of the GPC chromatogram to the CPU. The amplifiers raise the input signal to the level required by the converter. The multiplexer selects which of the four analog input channels will be read. The A/D converter converts the analog signal to a 10-bit binary number which may be read by the CPU. The converter was scaled to read 1/10 millivolt per bit on the high range or 0-1000 over the full scale input. The converter will operate at rates up to 20,000 conversions per second with a precision of 1/1000. The chromatogram sampling system includes a low pass filter to remove 50 cycle and higher frequency noise. This is discussed in more detail in section 4:1.

The contact sense lines are used as event markers. Two are required for each GPC. When a retention volume dump or sample injection occurs, a relay closes in the GPC. The contact sense lines communicate this to the CPU by a corresponding change in state of a buffer register. (see Appendix A:5) The change can be monitored by the CPU or it can be used to cause a hardware interrupt. The contact sense lines operate directly on signal levels greater than 1.3 volts (DC). In cases where this is not available, a 5 volt supply is provided by the interface which may be applied to the contact sense register through the closed external contacts.

### 2:5 Real-Time Clock

The real-time clock (RTC) produces a program interrupt at one of four frequencies to provide a means by which periodic data collection or events may be implemented. In the GPC system the RTC is used to implement sampling of the chromatogram signal at a rate specified by the user. The RTC places one restriction on the programmer. The clock must be serviced (i.e., a pulse must be counted and the clock restarted) before the next clock pulse is due, or a pulse could be missed. In the GPC system the critical servicing time is 100 milliseconds. Actual servicing time was estimated at 0.1 millisecond.

### 2:6 Teletype

The teletype used in the GPC system is an ASR-33 modified by Datagen to interface with the Nova 1200. The ASR-33 has a printing and reading speed of ten characters a second. In the GPC system the teletype is the main channel of communication between the user and computer. Through it, the user initiates all system software commands and receives all results from data collection or calculations. The teletype includes a paper tape reader and punch. The reader is used to load binary programs into the computer and to input data stored on paper tape. The paper tape punch will reproduce on tape any eight bit character transmitted to the teletype.

## 3. The GPC Minicomputer Systems: Software

Chapter 3 discusses the software specifically developed for a minicomputer system containing 4K words of memory and operating with one GPC. The software was designed to minimize the modifications required for expansion at a later date to an 8K system handling several GPCs. Details, flow diagrams and program listings for each subroutine are included in the Appendix.

### 3:1 Basic Formulation

The software for the GPC system was designed to perform two distinct operations. Firstly, it was to control data acquisition through an interactive mode of operation with the user. This meant programming an operating system to collect, store and output data on command. Secondly, the software was to perform certain calculations on the data and output appropriate averages, graphs and tables.

To date, most systems developed for the GPC have performed the two functions independently. Minicomputers and digital translators have been used for data acquisition [3,4,24] and for data acquisition with minor data reduction [5]. The output from the data acquisition system was used as input data for further processing by a large computer either inhouse or via a time-sharing terminal. The GPC system described here combines both functions to permit simultaneous data acquisition and reduction. The on-line system, operating in a dedicated mode, offers several advantages over separate acquisition and reduction operations.

Firstly, total processing time is reduced to approximately five minutes after the sample has passed through the GPC detector. Data reduction begins immediately upon completion of a sample run. Secondly, any errors that arise in preparing data for processing are eliminated, because in most cases data can be store and processed internally. Thirdly, the dedicated minicomputer requires no systems support, and will operate in most laboratory environments without additional protective housing. Finally, the minicomputer system could pay for itself by eliminating the computer time costs associated with large computer systems or time-sharing. Assuming processing costs to be $2 a sample, it would take approximately 2-3 years of normal operation to pay for the minicomputer system's hardware.

## 3:2 Programming A Minicomputer

Most minicomputers may be programmed in several languages including Fortran, Algol, Assembly and Basic. Unfortunately, compiler level languages such as Fortran and Algol requires extensive memory and peripheral hardware to operate efficiently. In addition, most minicomputer Fortran and Algol compiler level languages do not include real-time commands. Assembly language, on the other hand, is suited for real-time work because of specific hardware instructions and the memory savings made possible by its less general nature.[25] The penalty for this power, however, is that Assembly language programs will generally not operate in computers made by different manufacturers. All programming for the GPC system was done in Data General Assembly language despite the disadvantage of being restricted to Datagen computers.

Data General Assembly language is a set of instruction mnemonics that are directly convertible to a 16-bit binary word. A one to one correspondence exists between each instruction and one memory word. The mnemonic instructions are used to implement very basic types of operations including moving data, performing basic arithmetic operations, and controlling input-output.

Data can be moved directly from memory to the accumulators in 16-bit word form, and vice versa. Although hardware arithmetic and logical operations are limited to eight basic instructions including add, subtract, and increment, a powerful set of skip instructions can be combined with the eight basic instructions to allow several logical steps to be performed as a result of one instruction word. All multiplications and divisions are programmed as subroutines. Similarly, there are no hardware floating point manipulations. Floating point operations require a special floating point interpretive subroutine. Every time a floating point manipulation is required, the user must transfer control to the floating point interpreter, which interprets each floating point instruction and performs the necessary manipulations. After all the floating point instructions are completed, control is returned to the user's routine. Assembly language is format free. All teletype input-output is under the direct control of the user. Characters are passed one at a time to the teletype, when it is not busy. Similarly, all data transfers from other peripheral devices are under the user's control. Efficient input-output is usually associated with priority interrupt programming (see 3:3).

Developing a program for the minicomputer in Assembly language is not unlike programming in Fortran. A logic flow diagram is constructed for the over-all program and each subroutine. Each routine is then written and punched on paper tape. An editor program, supplied with the computer, is used at this stage to correct obvious programming errors. The next step is to assemble the program using the Assembler program. Assembling converts each instruction and numerical constant to its binary equivalent. This stage corresponds to one of the latter operations in compiling a Fortran program. Any errors in program structure or mnemonics are detected and flagged. The output from the assembler stage is a program listing and either an absolute binary, or relocatable binary paper tape. The program is now ready for loading, execution and final debugging.

Absolute binary tapes are loaded with the binary loader that resides in core. Relocatable programs are loaded with the relocatable loader. After loading, the program is executed by entering the starting address in the switches and depressing the RESET and START switches on the computer. Once each subroutine is tested, corrected and working properly, the entire program can be loaded and tested.

Development time can be kept to a minimum by working with a disk operating system. The disk operating system eliminates reading and punching programs on paper tape by creating and storing programs as files on the disk. The Debugger program was also extremely useful in tracking and eliminating logic and addressing errors. The Debugger program is used for on-line monitoring and altering of accumulators, memory and device flags.

## 3:3 Interrupts

Basically, each hardware device is capable of signalling the CPU that it requires attention. For example, this might occur either because some data which had been previously requested is now available, or because a real-time event such as clock pulse has occurred. The process of a device requesting attention is called an interrupt. Provided the interrupt switch on the CPU is on, certain hardware instructions are automatically executed to take control from the currently operating program to a master program which determines the action to be taken. The attention a device requires depends on which device has caused the interrupt. The process of executing a specific set of software instructions for a device that caused an interrupt is commonly referred to as servicing an interrupt.

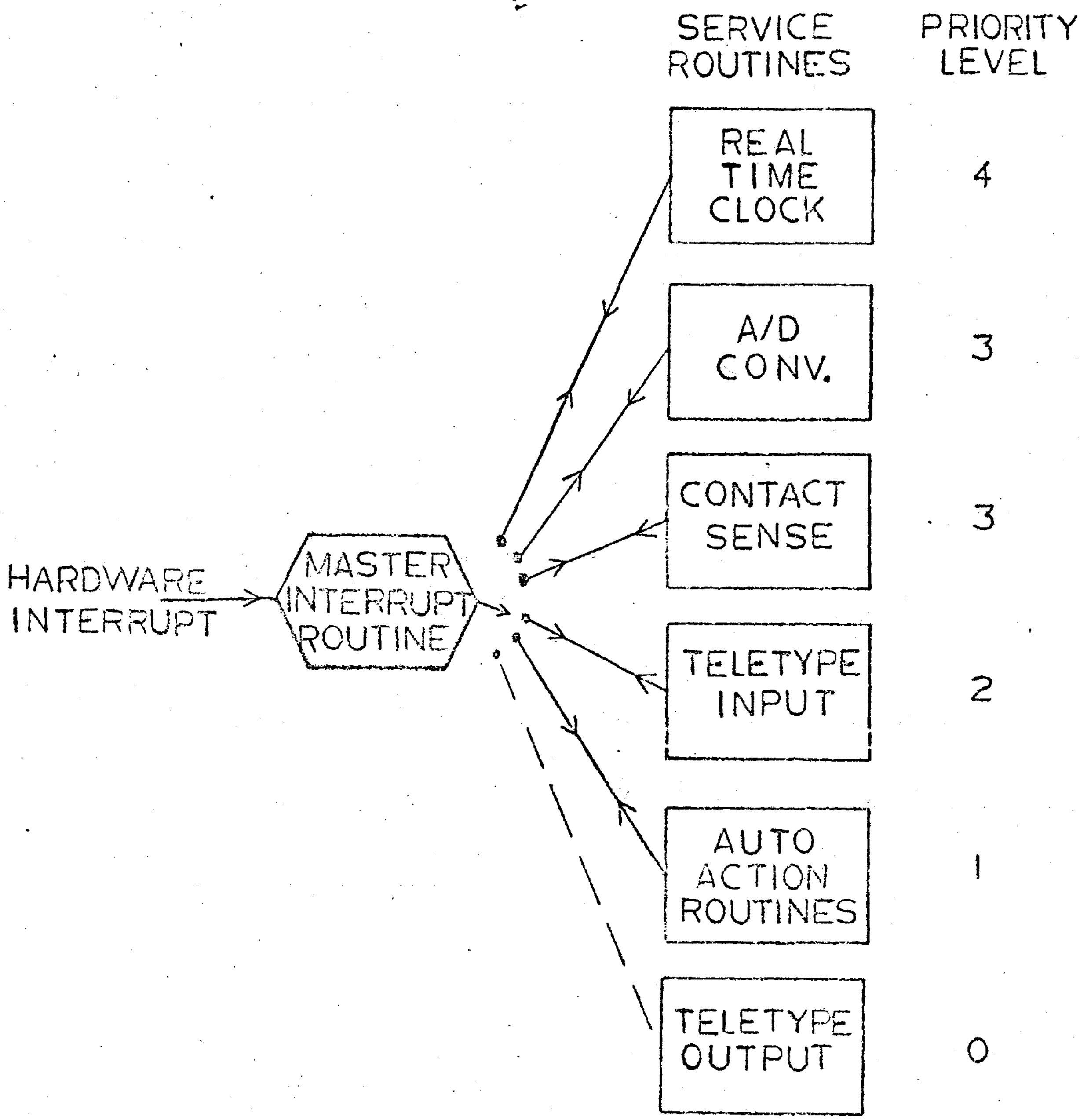
Furthermore, it is possible to assign priorities to the devices, so that a device with a high priority will receive preferential attention. Any device of higher priority may interrupt a device or servicing routine of lower priority. After the higher priority device is serviced, control is returned to the point at which the lower priority device was interrupted. All interrupts from lower priority devices are deferred (masked) until all higher priority devices have been serviced. A priority interrupt system allows the programmer to insure that each real-time event or interrupt is acknowledged in its proper sequence with respect to its relative importance.

## 3:4 The GPC Operating System

The GPC operating system is comprised of a series of subroutines that control data sampling, storage, and output on command. It is structured around the priority interrupt system illustrated in Figure 3-1. The real-time clock was given the highest priority to avoid a clock pulse being missed. The A/D converter and contact sense device were assigned a higher priority than the teletype input because their input data is time dependent. Teletype output was given a very low priority, and in fact is not really operating as a separate device in the interrupt system. There are two reasons for this. Firstly, there is no need to speed up teletype output because the computer spends less than 5% of its time outputting information. Secondly, there was no room for the output buffers required for efficient interrupt-type teletype output.

The GPC interrupt software operates as follows (refer to Figure 3-1). When an interrupt occurs, control is transferred to the master interrupt routine. The master interrupt routine stores all the information required to return to the interrupted routine. This includes the contents of all the accumulators, the carry bit, and the return address. It then determines which device caused the interrupt, disables its interrupt capability and that of all devices of equal and lower priority, and transfers control to the appropriate service routine. After the device is serviced the master interrupt routine restores the original status of the hardware and returns control to the interrupted routine.

The master interrupt routine transfers control to the appropriate device servicing routine. The action taken in all the servicing routines




GPC INTERRUPT STRUCTURE AND PRIORITY LEVELS

FIGURE 3-1

is controlled by the user through nine operating system commands. The purpose of each command is summarized in Table 3-1. New operating information is entered by typing a command followed by an escape (esc key). The LOOK, MON and ATD commands monitor and test the system status and operations. The BEGIN, INJ, STOP, TYPE and CALC commands control data acquisition and reduction. The DATA command is used to initiate automatic sampling and/or processing. The reader is referred to the Appendix Section A:4 for detailed explanations of how the operating system commands may be used to implement and control data acquisition and reduction.

## 3:5 Calculation Programs

Three sets of calculational subroutines were created to perform the calculations normally used in GPC. Program 1 reduces the raw data to the desired molecular weight averages and molecular weight distribution, and is included in the 4K on-line GPC system. Program 2 and Program 3 are used for special purpose calculations and could not be included in the 4K system because of memory limitations. They were therefore programmed to be used off-line (i.e., when the data collection system is not being used). Both Program 2 and Program 3 were designed so that they could be easily modified to operate in an 8K on-line system.

### Program 1

Program 1 is used to characterize the chromatograms of unknown polymer samples. It is included in the basic 4K GPC software package to permit simultaneous operation with the data acquisition system. It is called with the teletype command CALC at priority level 2, or automatically following the completion of a sample run at priority level 1.

| Command | Purpose |
| --- | --- |
| LOOK | Output current operating status |
| MON | Output all information as it is collected |
| ATD | Test analog to digital channel |
| BEGIN | Enter manual data collection parameters |
| INJ | Note manual sample injection |
| STOP | Stop storing data |
| TYPE | Output data for a particular sample |
| CALC | Compute and output the molecular weight averages and the molecular weight distribution |
| DATA | Enter automatic data collection and/or processing parameters |

TABLE 3-1

Program 1 accepts data from papertape or memory and:

1. decodes and outputs raw data,
2. interpolates for flow variations and reduces the raw data to 60 points,
3. subtracts the baseline and outputs the results in graphical form,
4. calculates and outputs the chromatogram area and mean,
5. calculates and outputs the molecular weight averages, and
6. calculates and outputs the differential molecular weight distribution.

The molecular weight averages and differential molecular weight distribution are calculated by the methods outlined in Chapter 5. The total processing time is approximately 5 minutes. Of this 5 minutes, about 30 seconds are spent actually doing the calculations. The remaining 4½ minutes are required to output the results.

Program 2

Program 2 uses a golden section optimization algorithm to search for an effective calibration curve. An effective calibration curve is sometimes required when it becomes necessary to calibrate with one or more broad MWD standards, or when a corrected differential distribution is desired, as outlined in Section 5:4. Program 3, which includes most of the routines of Program 1, is also run off-line because it would not fit in the basic 4K system. In addition, it includes a golden section search program which performs a single variable search to minimize the objective function OB, where:

$$OB = abs(P(t) - P(D_2')) \qquad 3:4:1$$

P(t) - true polydispersity

$P(D_2')$ - polydispersity determined with calibration curve slope $D_2'$.

The result is an effective linear calibration curve of the form:

$$M = D_1' \exp(-D_2'v) \qquad 3:4:2$$

$$(\text{linear form } \ln M = \ln D_1 - D_2 v)$$

The effective calibration curve can be used to compute the corrected differential distribution from Equation 5:4:2.

Program 3

Program 3 is used to calculate instrument spreading parameters from the chromatograms of standard samples. It can be run when the basic 4K system is not being used on-line with a GPC.

On-line processing with Program 3 would be convenient, but is not necessary because the spreading parameters cannot usually be calculated immediately.

Program 3 includes most of the subroutines of Program 1. In addition 100 words of programming are required to input additional information and to implement equations 5:4:9 and 5:4:10. Program 3 outputs the computed spreading parameters in addition to all the information outlined in Program 1. Total calculation time is also about 5 minutes.

## 3:6 Storage Buffers

Three storage buffers were required. One is for fixed point data collected by the operating system, another is for floating point data produced by the calculational routines, and the third is used by the floating point interpretive package.

The operating system buffer stores operating information and the raw chromatogram data. The raw chromatogram data is stored in a continuous loop. In other words, when the buffer is filled, the storage pointer recycles to the top of the buffer and new data overwrites the old. The user is therefore restricted to sampling at a rate that will not fill the buffer before a sample run is complete. For efficient GPC operation, the buffer should be large enough to hold two data sets to permit overlapping samples. The minimum requirement is 400 storage words for each GPC.

The second storage buffer contains the 60 floating point adjusted chromatogram heights produced by the calculation routines. (see Appendix section A:2) Floating point numbers require two locations each, therefore, a total of 120 words are necessary.

The floating point interpreter package requires 60 locations for temporary storage. A pointer to this storage buffer is stored at memory location seven.

# 4. The GPC Minicomputer System: Performance and Observations

Chapter 4. discusses the performance of the GPC minicomputer system's hardware and software. Operating experience and refinements to the original system are discussed in some detail.

## 4.1 Hardware

The GPC minicomputer system was interfaced to a Waters Model 200 GPC for a period of eight months. During this period the CPU, interface, and real-time clock performed without a failure. The least reliable component in the hardware system was the teletype which was serviced twice. Proper preventative maintenance at regular intervals should minimize teletype breakdowns.

A histogram program was used to collate 10,000 consecutive readings of the analog input to determine the precision of the A/D converter. During the early stages of development, difficulties in ground potentials caused noise problems. The ground loops were eliminated by grounding the GPC, interface, teletype and CPU at the same point. Also, an additional inductance filter was required to eliminate both the very high frequency noise (100-1000 Kc) caused by the GPC strip chart recorder and some 60 cycle noise which was primarily caused by ground loops. With a typical chromatogram signal the precision of one conversion based on 10,000 consecutive conversions in 0.50 seconds was 1/1000.

The linearity of the convertor was found to be 1/1000 over the range of permissible inputs. Interference effects due to a load on an adjacent channel were not measureable. For normal GPC operation, no further filtering was necessary. Operating the GPC at high temperatures could introduce low cycle noise that would not be removed by the hardware filters. A time-averaged software filter in the A/D interrupt routine might be beneficial in such cases.

The contact sense lines also operated reliably. However, occasionally the relay in the GPC tripped twice for one event. The cause was a detection problem in the GPC photo-electric syphon circuit and not relay bounce. Multiple recognitions which occur in a very short period of time (relay bounce) were avoided by including a 1 millisecond hold circuit in the contact sense circuits of the interface. The multiple recognitions due to the GPC photo-electric syphon circuit occurred over longer periods of time. To avoid this problem it became necessary to ignore retention volume interrupts for a period of 10 seconds after a retention volume interrupt was first acknowledged. A software flag was created for this purpose.

## 4:2 Software: Calculational Routines

To check the calculational methods employed on the minicomputer, molecular weight averages were computed and compared with those obtained using identical data, and the standard Fortran data reduction programs of McMaster University [(4)] for a CDC6400. Using raw data collected with a digital translator, the number average molecular weights, $\bar{M}_n(\infty)$, and the z-average molecular weight, $\bar{M}_z(\infty)$, computed by the minicomputer and by the CDC6400 agreed within ±2%, while the weight average molecular weight, $\bar{M}_w(\infty)$, agreed within ±1%. Using raw data collected by the minicomputer, the molecular weight averages ($\bar{M}_n(\infty)$, $\bar{M}_w(\infty)$ and $\bar{M}_z(\infty)$) computed by the minicomputer and by the CDC6400 all differed by less than ±1%. The use of different interpolation processes in each method and the accuracy in representing a floating point number were found to be the sources of these differences.

The major source of the differences was attributed to the interpolation routine used to produce flow adjusted, baseline corrected, heights from the raw chromatogram data. When identical interpolated data, not raw data were used, the molecular weight averages computed by the minicomputer and the CDC 6400 were exact to five significant figures. The difference in the sixth figure was attributed to cumulative round off error. The accuracy in representing a floating point number in a computer is limited by the number of bits used to represent its exponent and mantissa. A floating point is exact to seven significant figures in a Nova 1200 and exact to fourteen significant figures (single precision) in a CDC 6400. Because errors introduced by the interpolation process were significantly greater, round off errors were not investigated further.

The interpolation process used in the minicomputer differed from the standard Fortran routines in two respects. Firstly, the minicomputer system used only linear interpolation while the standard Fortran routines included three point interpolation around the peak. Linear interpolation was used in the minicomputer because of memory limitations. Secondly, all baseline corrected and interpolated heights smaller than the basic uncertainty in the data (1/1000), were set equal to zero in the minicomputer system. This was not done in the Fortran routines.

The improved agreement in the molecular weight averages calculated with raw data collected by the minicomputer, as compared to raw data collected by the digital translator, was attributed to electrical interference effects. Chromatogram heights recorded by the digital translator less than 10 seconds after the retention volume dump are influenced by that dump.[3] The interference is produced in both the GPC amplifiers and in the translator. In the minicomputer system, the retention volume sense circuit did not in any way affect the A/D converter circuit. The interference effects from the GPC circuits produce a slight error in a height that is requested less than five seconds after a retention volume dump. The error produced in that one point is at most 5%. The error contribution to the molecular weight averages by that one point is insignificant. The point is only one of many used to produce interpolated heights which in turn are used to characterize the chromatogram.

### 4:3 Software As a System

The basic 4K system, containing the operating system and Program 1 calculation routines, was operated for eight months by T. Ishige, S.K. Vig and B. Bakova. Several modifications were made to the software package based on this experience. Every attempt was made to eliminate all complicated operating instructions. As a result, the only precaution the user must observe is not to interrupt an automatic calculation in order to perform a second calculation. Otherwise, potential problems in interrupting non-reentrant routines (i.e., interrupting a specific routine and then using that routine a second time) were avoided by deferring user interrupts at all critical points. Every reasonable attempt was also taken to eliminate all other user errors by programming the system to recognize and flag unusual events or commands with error messages.

Figure 4-1 is a core diagram showing the memory required for the operating system, Program 1 calculational routines, the storage buffers and the floating point interpreter package for the existing 4K system and a proposed 8K system. As Figure 4-1 indicates, the entire core was required for the 4K system. The operating system storage buffer of 400 locations is the minimum required to permit overlapping samples. Because of the limited memory, every subroutine was designed and written to minimize the program length rather than the execution time. The standard floating point package was used rather than the extended version. The exponential function included in the extended version, but not in the standard version, was rewritten in compact form. Lengthy messages and labels were kept to a minimum because normally every two letters require one memory location. (Complex packing can reduce storage requirement to two locations for five letters).

The 4K software package of Figure 4-1 contains only Program 1 calculational subroutines. There was no room left to include Program 2 or Program 3. Program 2 and Program 3 were therefore written to be used off-line. However, they were designed to be easily adapted to an on-line 8K system. The core diagram of a proposed 8K system, shown in Figure 4-1, is based on the subroutines of the existing 4K system. The operating

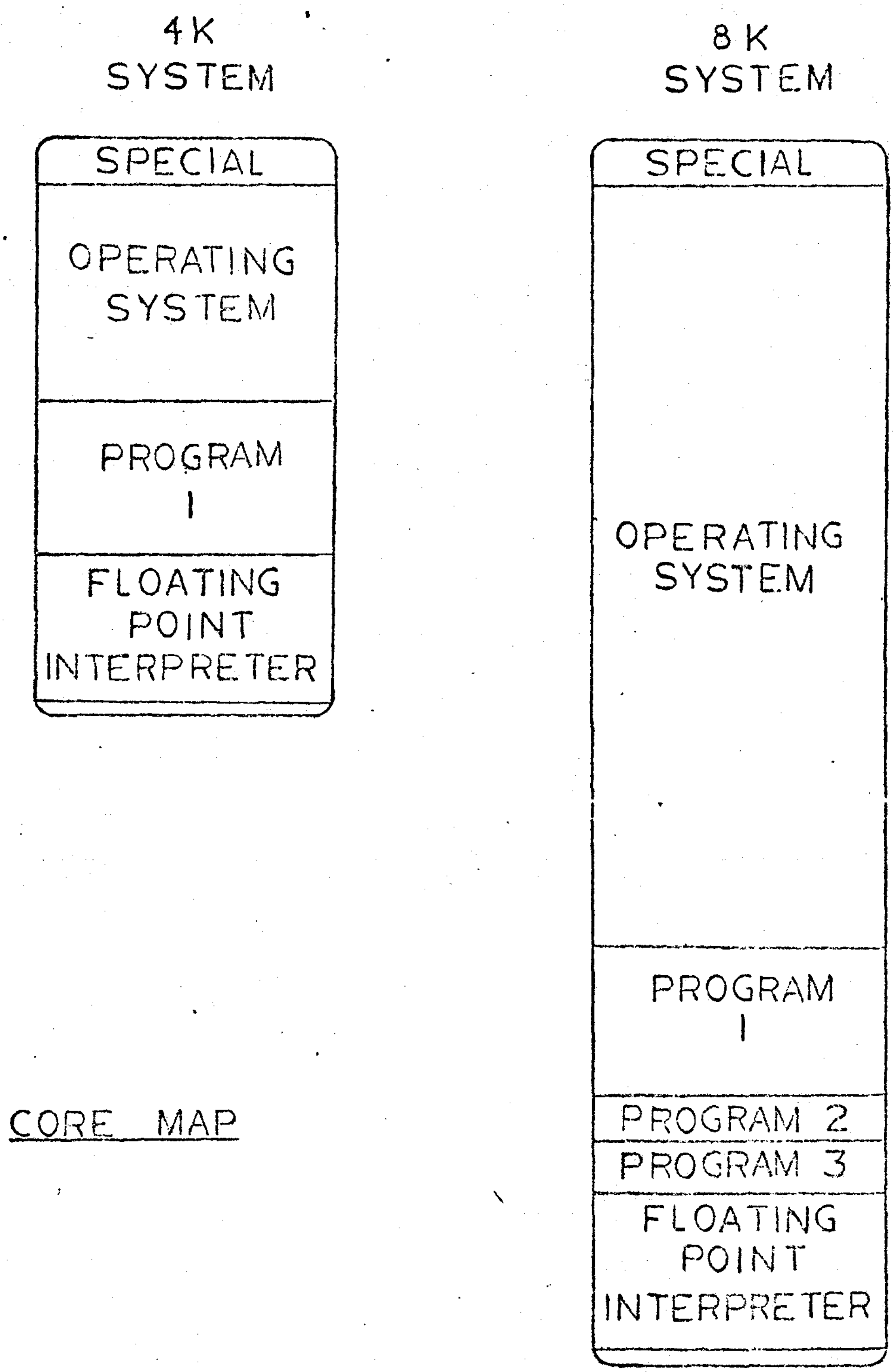

4K SYSTEM
SPECIAL
OPERATING SYSTEM
PROGRAM 1
FLOATING POINT INTERPRETER
8K SYSTEM
SPECIAL
OPERATING SYSTEM
PROGRAM 1
PROGRAM 2
PROGRAM 3
FLOATING POINT INTERPRETER
CORE MAP

FIGURE 4-1

system was also designed to operate with more than one GPC. Essentially the only requirement for expanding the operating capacity is an additional storage buffer for each GPC. The 8K system shown in Figure 4-1 should operate with 4 GPC's simultaneously and include Program 1 and Program 2 calculation routines.

# PART II

## Experimental Studies

## 5. Gel Permeation Chromatography

### 5:1 Introduction

A gel permeation chromatograph (GPC) is an analytical tool for measuring the molecular weight distribtuion (MWD) of polymers. The MWD of a sample is obtained by affecting a separation according to molecular size in solution and relating the molecular size to molecular weight. The separation in GPC occurs in the liquid phase in columns packed with porous gel or glass beads. As the polymer molecules flow through the columns, they permeate into the pores, size permitting. Large molecules are therfore least held up and elute first, followed by successively smaller molecules. The length of time a molecule has spent in the column is characterized by the amount of solvent that has passed through the columns since the sample was introduced. This quantity, the amount of solvent that has passed through the columns, is called the retention volume, v. The amount of material eluting at a particular retention volume can be determined by comparing physical properties of the polymer containing carrier solvent with pure carrier solvent. The most commonly used detector in GPC is the differential refractometer. At low concentrations, the change in refractive index is directly proportional to the mass of polymer/unit volume and independent of molecular weight (except for very small molecular weights or oligomers). If the relationship between the retention volume and molecular weight is known, it is then possible to transform the distribution of detector response versus retention volume to weight fraction of polymer versus molecular weight.

## 5:2 Characterization of a Molecular Weight Distribution

If the molecular weight distribution W(M) describing a polymer sample is normalized, such that

$$\int_0^\infty W(M)\, dM = 1 \qquad 5:2:1$$

then the k<sup>th</sup> molecular weight average, $\bar{M}k$, is

$$\bar{M}k = \frac{\int_0^\infty W(M)\, M^{k-1}\, dM}{\int_0^\infty W(M)\, M^{k-2}\, dM} \qquad 5:2:2$$

where k = 1, 2, 3 corresponds to the number, weight and z average molecular weights, respectively. For discrete data, the values of the integrals on the right hand side of 5:2:2 are conveniently found by numerical integration techniques, such as Simpson's Rule. Higher order integration techniques are not warranted because of the low experimental precision in measuring W(M).

## 5:3 Calibration

Because the GPC is not an absolute instrument, it is first necessary to establish a relationship between retention volume and molecular weight. This is done by injecting a series of narrow molecular weight distribution standards with known $\bar{M}_n$ and $\bar{M}_w$, and determining the retention volume at the peak of the chromatogram for each standard. The log of the root mean squared molecular weight RMS(M), defined by

$$RMS(M) = \sqrt{\bar{M}_W \bar{M}_N} \qquad 5:3:1$$

is then plotted versus the peak retention volume, PRV. Often the molecular weight retention volume relationship takes the form

$$M = D_1 \exp(- D_2 v) \qquad 5:3:2$$

or

$$M = D_1 \exp(-(D_2 v + D_3 v^2)) \qquad 5:3:3$$

where $D_1$, $D_2$, $D_3$ are constants.
Each polymer will have its own set of constants $D_1$, $D_2$, $D_3$. Chemically different polymers will in general have different calibration curves because molecular size depends on both molecular weight and chemical composition. The establishment of a molecular weight calibration curve often presents a problem because a wide range of standards is generally not available for all polymers. Two methods have been suggested to overcome this difficulty. These methods will now be briefly discussed.

Benoit's Approach

Benoit was the first to suggest the possibility of a universal calibration curve. He found that a plot of the product of the polymer intrinsic viscosity $[\eta]$ and molecular weight (M) versus retention volume (v) was a single curve for several polymer types. His results are not surprising since the separation mechanism in GPC depends on size in solution. According to Einstein's viscosity relationship,

$$[\eta] = 0.25\ N\ Vh/M \qquad 5:3:4$$

N - Avagadro's number

Vh - hydrodynamic volume

or

$$M\ [\eta]\ \alpha\ Vh \qquad 5:3:5$$

That is, the product of the intrinsic viscosity and molecular weight is proportional to the size in solution (the hydrodynamic volume). If the separation is indeed by size in solution, and equation 5:3:5 is valid, then a plot of $[\eta]M$ versus retention volume should be valid for all polymer types.

Using Benoit's universal parameter ($[\eta]M$), it is possible to determine a calibration curve for one polymer using standards of more readily available polymers. If the intrinsic viscosity-molecular weight relationship for polymer x follows the Mark-Houwink relationship,[21]

$$[\eta]_x = K_x M^{a_x} \qquad 5:3:6$$

$K_x a_x$ constants for polymer x

and the universal calibration curve is known over a range of retention volumes,

$$[\eta]M = D_1'\ \exp(-D_2'v) \qquad 5:3:7$$

$D_1'$, $D_2'$ - universal calibration curve constants.

Then the calibration curve for polymer X in terms of the universal constants becomes:

$$(M)_x = \left(\frac{D_1'}{K_x}\right)^{\frac{1}{1+a_x}} \exp\left(\frac{-D_2' v}{1+a_x}\right) \qquad 5:3:8$$

Equation 5:3:8 predicts calibration curves with different slopes and intercepts for polymers with different K's and a's. Before using 5:3:8 to predict a calibration curve, the values of the Mark-Houwink constants K and a must be known. In general, K and a are functions of both temperature and solvent as well as polymer type. Handbook values are available for the most common polymers and solvents[21], generally of room temperature. Relying on literature values for K and a places severe limitations on both the use and accuracy of results obtained with equation 5:3:8.

Dawkins' Approach[8]

A second method of universal calibration was proposed by Dawkins in an attempt to explain a few cases where Benoit's approach did not appear to work. Dawkins suggested using the unperturbed dimensions rather than the hydrodynamic volume for universal calibration. His approach assumes that the separation mechanism depends more on the rigid dimensions than the swollen size in solution. To see the differences in the two approaches, equation 5:3:6 is expanded in a form suggested by Flory[22];

$$[\eta] = \phi \left[\frac{\langle Lo^2 \rangle}{M}\right]^{3/2} M^{\frac{1}{2}} \alpha^3 \qquad 5:3:9$$

$\phi$ - universal constant

$\alpha$ - linear deformation due to Solvent polymer interaction

Lo - unperturbed end-to-end dimension

The ratio $<\frac{Lo^2}{M}>$ has been shown to be independent of chain length for a specific polymer. As equation 5:3:9 indicates, if the linear deformation due to polymer-solvent interaction, $\alpha$, is similar for a series of polymers, than either $[\eta]M$ or $<Lo^2>$ may be used for universal calibration. In cases where $\alpha$ is different, it is again possible to calibrate for one polymer, using different polymers if the ratios $<Lo^2>$ are known. The calibration curve for polymer x, in terms of a calibration curve for polymer y described by,

$$M_y = (D_1)_y \exp((-D_2)_y v) \qquad 5:3:10$$

is found by assuming the universal parameter is $<Lo^2>$, or:

$$<Lo^2>_x = <Lo^2>_y$$

expanding,

$$M_x \left(\frac{<Lo^2>}{M_x}\right)_x = M_y \left(\frac{<Lo^2>}{M_y}\right)_y$$

$$M_x = M_y \left[\left(\frac{<Lo^2>}{M}\right)_y \Big/ \left(\frac{<Lo^2>}{M}\right)_x\right]$$

and substituting 5:3:10.

$$M_x = D_{1_y} \exp(-D_{2y} v) \left[\left(\frac{<Lo^2>}{M}\right)_y \Big/ \left(\frac{<Lo^2>}{M}\right)_x\right] \qquad 5:3:11$$

Equation 5:3:11 predicts calibration curves with similar slopes, but different intercepts for different polymers. It differs significantly from Benoit's approach in that respect. Because of experimental error in GPC, it would seem difficult to firmly establish which method, Benoit's or Dawkins', is the most accurate until the separation mechanism has been established by some other means.

## 5:4 Interpreting A GPC Chromatogram

The two conveniently measured experimental parameters in GPC are the retention volume and concentration of polymer in the eluting stream. As a sample elutes from the columns, there is an inverse relationship between molecular size and retention volume. The large molecules elute first followed by successively smaller and smaller molecules. The distribution in the retention volume space W(V) is related to the desired molecular weight distribution W(M), such that

$$W(M)\ dM = -W(v)\ dv \qquad 5{:}4{:}1$$

or

$$W(M) = -\ W(v)\ /\ (dM/dv) \qquad 5{:}4{:}2$$

When a polymer sample with distribution W(v ) is injected into the GPC, the detector response F(v) is given by Tung's integral equation:[9]

$$F(v) = \int_0^{\infty} W(v)\ G(v,y)\ dy \qquad 5{:}4{:}3$$

The function G(v,y) accounts for the spreading and skewing which occurs in the columns and connecting tubing [10], and in general will be different for each column set.

When the effect of axial dispersion is negligible, the detector response may be used directly to find the molecular weight distribution. The molecular weight averages obtained, $\bar{M}_n(\infty)$, $\bar{M}_w(\infty)$, $\bar{M}_z(\infty)$, are commonly referred to as the infinite resolution molecular weight averages. If dispersion cannot be neglected, equation 5:4:3 must be solved for W(v). To do this we must measure a G(v,y) by calibration.

Gaussian Instrumental Spreading

When the response of a single polymer species is Gaussian, G(v,y) can be represented as:

$$G(v,y) = \left(\frac{h}{\pi}\right)^{0.5} \exp(-h(v-y)^2) \qquad 5:4:4$$

where in general h = h(y).

Several methods have been proposed to solve Tung's equation, 5:4:3, numerically for W(v) by substituting the right-hand side of Equation 5:4:4 for G(v,y).[9,11,12] Because the methods are limited to symmetrical instrument spreading, they have not found wide spread use.

An analytical solution to Tung's equation exists if h in equation 5:4:4 is assumed constant. Hamielec and Ray[13] have shown that if the molecular weight calibration curve is linear, then, regardless of the complexity of the sample, the ratio of the $k^{th}$ corrected molecular weight averages $\bar{M}k(h)$ to the $k^{th}$ infinite resolution molecular weight average is:

$$\frac{\bar{M}k(h)}{\bar{M}k(\infty)} = \exp\left((3 - 2k)\, D_2^{\,2}/4h\right) \qquad 5:4:5$$

Tung and Runyon[11], using reverse flow techniques, were able to measure and correlate h with retention volume for a series of polymer standards. They also reported that h calculated for one PVC sample and one polybutadiene sample fell on the h curve determined with polystyrene samples, suggesting the possible universal nature of h. That is, it seems possible that instrument spreading due to axial dispersion depends on retention volume only, and not chemical composition. Smith and Ehulich[14] recently arrived at similar conclusions also using reverse flow techniques.

Asymmetrical Instrumental Spreading

Unfortunately, the instrumental spreading function is very often skewed (asymmetrical) and cannot be adequately described by equation 5:4:4[1]. Skewed Chromatograms are often produced under conditions of high flow rate (desirable operating condition) as well as by overloading and loss of resolution at the high and low ends of the molecular weight calibration curve[1,2]. Provder and Rosen[2,15] proposed a general shape function

$$G(v-y) = \phi(v-y) + \sum_{n=3}^{\infty} \{-1^{n} \frac{A_n}{n} U_2^{n/2} \phi^{n} (v-y)\} \quad 5:4:6$$

$$\phi(v-y) = (\frac{\pi U_2}{2})^{0.5} \exp[-(v-y)^2/2U_2]$$

$\phi^{n}(v-y)$ - $n^{th}$ order derivitives of (v-y)

$A_1$, $A_2$, $A_3$ - shape parameters

to describe instrumental spreading. Practical limitations of calibration limit the number of terms that may be kept in 5:4:6. It is the author's experience that truncation of 5:4:6 at practical limits produces unrealistic G(v-y) functions. Similar attempts[16,17,18] to solve Tung's integral equation with asymmetrical instrument spreading functions have been only partially successful because of the difficulty in describing G(v-y).

Rather than solving Tung's equation, Balke and Hamielec[1] have defined an empirical skewing factor SK, to correct for skewing. The corrected molecular weight averages $\bar{M}_n(t)$ and $\bar{M}_w(t)$ are in terms of the infinite resolution averages $\bar{M}_n(\infty)$ and $\bar{M}_w(\infty)$, the axial dispersion parameter h, and the skewing parameter Sk:

$$\bar{M}_n(h,Sk) = \bar{M}_n(\infty)\ (1 + Sk/2)\ \exp(D_2^{\,2}/4h) \qquad 5:4:7$$

$$\bar{M}_w(h,Sk) = \bar{M}_w(\infty)\ (1 + Sk/2)\ \exp(-D_2^{\,2}/4h) \qquad 5:4:8$$

Equations 5:4:7 and 5:4:8 are similar to Hamielec and Ray's analytical solution for symmetrical spreading, except for the skewing correction (1 + Sk/2).

The relationship between retention volume and the spreading parameters, h and Sk, can be determined in a manner similar to the methods employed to establish the molecular weight calibration curve outlined in 5:3. The experimental infinite resolution molecular weight averages and the absolute molecular weight averages of the standards are used to calculate local h and Sk's. In terms of known values $\bar{M}_n(t)$, $\bar{M}_w(t)$, $\bar{M}_n(\infty)$ $M_w(\infty)$, and $D_2$, h and Sk for standards are

$$h = 2\ A/[\ell n\ (\frac{\bar{M}_w(\infty)}{M_n(\infty)} - \ell n\ \frac{\bar{M}_w(t)}{M_n(t)}] \qquad 5:4:9$$

$$Sk = \frac{\bar{M}_w(t)}{M_w(\infty)} + \frac{M_n(t)}{M_n(\infty)} - (e^{A/h} + e^{-A/h}) \qquad 5:4:10$$

where $A = \frac{D_2^{\,2}}{4}$

The local values of h and Sk can be used with the peak retention volume of the sample to construct spreading parameter calibration curves. Infinite resolution values for any sample may be corrected for skewing and axial dispersion by picking appropriate values for h and Sk and applying equation 5:4:7-8. The corrected differential molecular weight distribution may be found by searching for an effective calibration curve which fits the data to the corrected molecular weight averages of 5:4:7-8.[1]

The empirical correction method of Balke and Hamielec has limitations however. Attempting to pick an appropriate value of h and Sk raises some important questions. In the cases where h and Sk change with retention volume, how are the average h and Sk determined? Is the peak or mean elution volume used to determine the spreading parameters from the calibration curves? Sk has been found to be concentration dependent. If the Sk curve is established with narrow standards it tends to over-correct broad samples. Therefore, how is Sk adjusted for changing poly-dispersity? The answers to these questions are not yet known. They will probably remain so until an extensive series of standards becomes available for various polymers. One question that can be examined is, how do the spreading parameter calibration curves change with chemical species? In other words, are instrumental spreading correction curves universal? Do they depend on retention volume alone? Chan[19] has recently reported using skewing and axial dispersion correction curves determined with polystyrene to correct the infinite resolution molecular weight averages of eleven PVC samples. He found that corrected number averages agreed within 25% of the absolute number averages and that corrected weight averages agreed within 15% of the absoluted weight averages. Although the error is quite large, it is not inconsistent with the experimental error associated with the absolute averages and GPC.

The possible existance of universal skewing and axial dispersion correction curves is an important question. For most polymers other than polystyrene, there are insufficient standards available which could be used to construct individual instrument spreading calibration curves. If universal spreading correction curves cannot be constructed, GPC users are forced to rely on infinite resolution values which can be significantly in error.

# 6. Experimental

## 6.1 Experimental Conditions

The present study was done in conjunction with an ASTM D-20,70.04 Task Force round robin test. The purpose of the ASTM test was to compare molecular weight averages determined by several different laboratories for identical polystyrene samples and GPC operating conditions. Because operating conditions can affect the accuracy of the results, the ASTM committee set specific conditions. The operating conditions they chose are:

| | |
|---|---|
| Solvent: | THF (Tetrahydrofurane) |
| Columns: | 4 each 4' x 3/8" |
| Packing: | Styragel |
| Gel Porosity : | $10^6$, $10^5$, $10^3$, $10^2$ (waters designations) |
| Temperature : | 25° C. |
| Flow : | 1 ml/min |
| Concentration: | 0.25% (wt) |

A Water's Associates GPC Model 200, with two minor modifications was used for this study. A vapor feedback loop was installed on the syphon bottle to prevent solvent evaporation. Also, an additional 1000 ml surge tank was employed downstream from the pump to further reduce flow variations. The columns were supplied by the ASTM committee to meet their specifications. The solvent was DuPont THF. Samples were dissolved in degassed solvent from the reservoir and filtered through a 0.2μ millipore filter. The samples were injected manually with a hypodermic syringe.

All data acquisition and reduction was done with the minicomputer system. The chromatogram heights were recorded every 20 seconds over the elution range of the columns. Molecular weight averages and distributions were calculated according to the methods outlined in Chapter 5.

## 6.2 Standards

The standards used were linear polystyrene (PS) supplied by the ASTM committee, linear polybutadiene (PBD) from Phillips Petroleum Co., linear poly(vinyl-chloride)(PVC) from Pressure Chemical, and poly(methyl methacrylate) (PMMA) from Imperial Chemical, Rohm and Hass, and S. Balke of McMaster University. The absolute number averages, $\bar{M}_n(t)$, and weight averages $\bar{M}_w(t)$ supplied by the vendor are summarized in Table 6-1.

| Designation | Source | $\bar{M}_n(t)$ | $\bar{M}_w(t)$ |
|---|---|---|---|
| BFG1 | ASTM | 1670 | 2030 |
| BFG2 | " | 4000 | 3600 |
| BFG3 | " | 16300 | 9700 |
| BFC4 | " | 19650 | 19850 |
| BFC4 | " | 49000 | 51000 |
| BFC6 | " | 96200 | 98200 |
| BFC7 | " | 164000 | 173000 |
| BFC8 | " | 392000 | 411000 |
| BFC9 | " | 773000 | 867000 |
| BFC10 | " | 780000 | 2145000 |
| NBS706 | National Bureau of Standards | 136000 | 257000 |
| NBS705 | " | 170000 | 179000 |
| PUC2 | Pressure Chemical | 25000 | 68000 |
| PUC3 | " | 41000 | 118000 |
| PUC4 | " | 54000 | 137000 |
| PMMA1 | Rohm & Hass | 50000 | 290000 |
| PMMA2 | Imperial Chemical | 33400 | 78000 |
| PMMA3 | McMaster University | 60000 | 115000 |
| PBD1 | Philps Petroleum | 16100 | 17000 |
| PBD2 | " | 135000 | 170000 |
| PBD3 | " | 206000 | 272000 |
| PBD4 | " | 226000 | 332000 |
| PBD5 | " | 286000 | 423000 |

Table 6-1

STANDARDS

# 7. Results and Discussion

## 7.1 Calibration Curves

The polystyrene samples supplied by the ASTM committee were injected using the conditions outlined in Chapter 6. Figure 7-1 is a plot of the root mean square molecular weight, RMS(M), versus the experimental peak retention volume. The experimental points were fitted with a third order polynomial, using a least squares technique, to give the following calibration curve for polystyrene:

$$M_{PS} = 1.567 \text{ E5 } \exp(.6386 \text{ v} - 2.255 \text{ E-2 v}^2) \qquad 7:1:1$$

The break in the calibration curve at retention volume 28 was due to the porosities of the columns. Loss of resolution at the low molecular weight end is characterized by the increasing absolute value of the slope.

### The Benoit Universal Calibration Curve

A universal calibration curve based on Benoit's method was generated from the polystyrene data. The results are illustrated in Figure 7-2. The following third order polynomial approximation, determined by the least squares method, was used to fit the universal relationship,

$$[\eta]M = 1.170 \text{ E5 } \exp(+1.098 \text{ v} - 3.848 \text{ E-02 v}^2) \qquad 7:1:2$$

The universal curve shows the same loss in resolution at the low molecular weight end as the polystyrene curve.

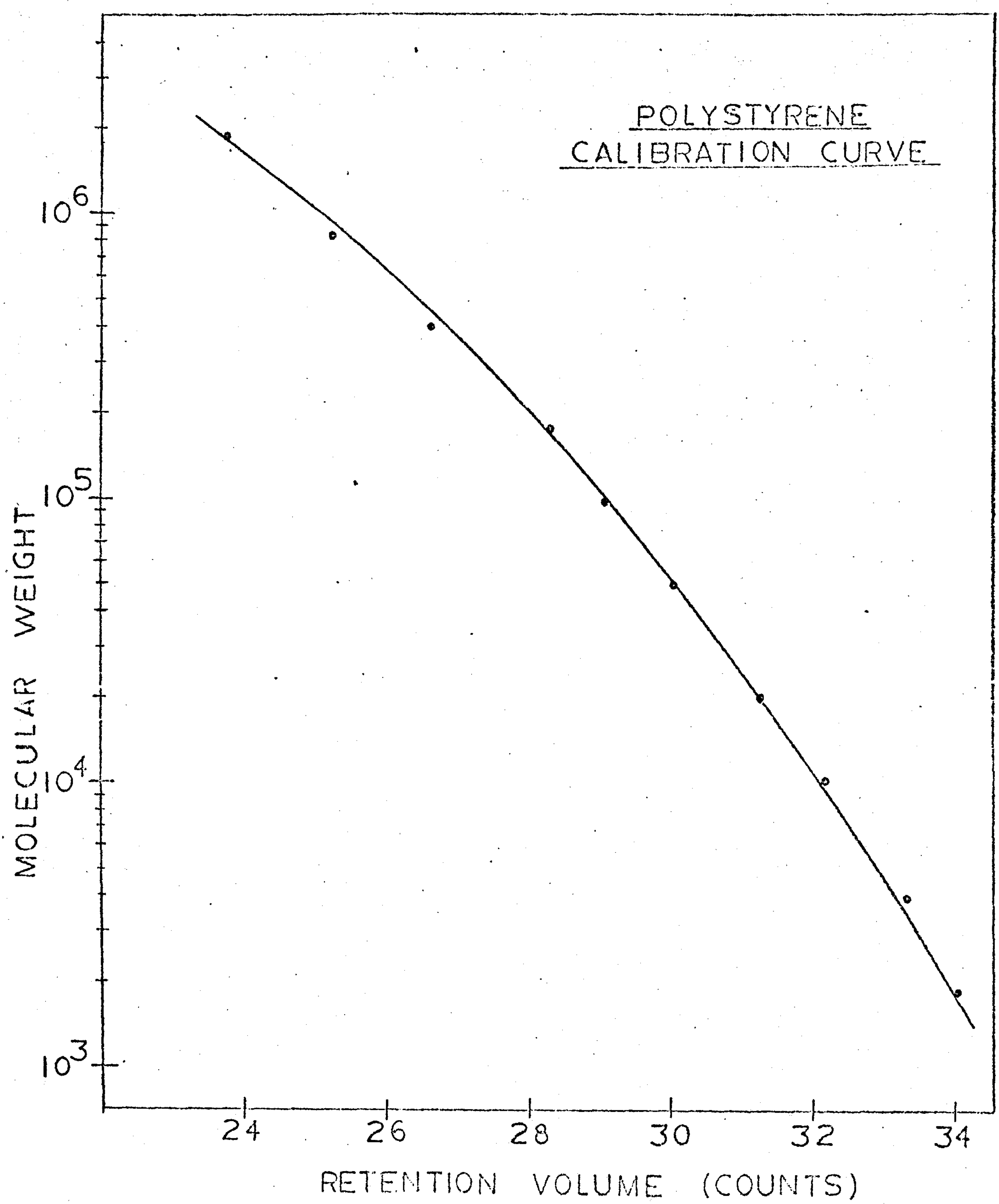

POLYSTYRENE
CALIBRATION CURVE
$10^6$
$10^5$
$10^4$
$10^3$
MOLECULAR WEIGHT
24
26
28
30
32
34
RETENTION VOLUME (COUNTS)

FIGURE 7-1

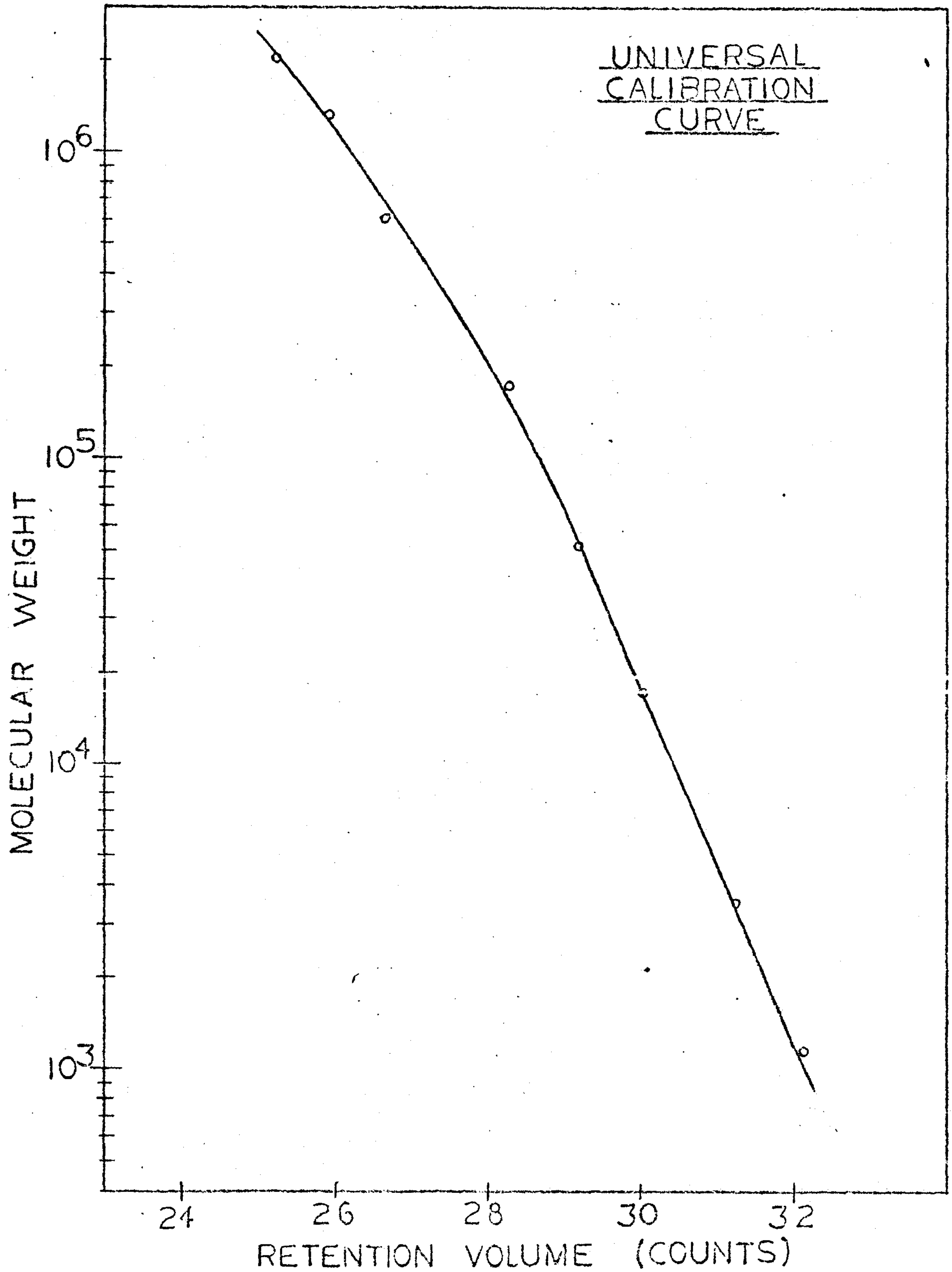

UNIVERSAL CALIBRATION CURVE
MOLECULAR WEIGHT
$10^6$
$10^5$
$10^4$
$10^3$
24
26
28
30
32
RETENTION VOLUME (COUNTS)

FIGURE 7-2

The PVC, PBD and PMMA calibration curves were computed from 7:1:2 according to 5:3:8. The following Mark-Houwink constants were used:

$$[\eta_{PS}]_{THF} = 1.60 \times 10^{-4} M^{.706} \qquad (2)$$

$$[\eta_{PVC}]_{THF} = 1.63 \times 10^{4} M^{.766} \qquad (2)$$

$$[\eta_{PMMA}]_{THF} = 9.5 \times 10^{-4} M^{.525} \qquad (6)$$

Unfortunately, the Mark Houwink constants for polybutadiene in THF were not available in the literature. The intrinsic viscosity-molecular weight relationship was estimated by assuming the Benoit universal calibration curve was valid. The intrinsic viscosities which fit the PBD samples to the universal curve were determined and plotted versus molecular weight. A least squares fit gave the following Mark-Houwink type relationship:

$$[\eta] = 2.47 \times 10^{-4} M^{.562}$$

for PBD in THF at 25°C.

Dawkins' Universal Calibration Curves

The calibration curves for PVC, PMMA, and PBD were generated by multiplying the polystyrene calibration curve, equation 7:1:2, by the following ratios:

$$(\langle Lo^2\rangle/M)_{PS}/(\langle Lo\rangle/M)_{PVC} = 0.54 \quad (20)$$

$$(\langle Lo^2\rangle/M)_{PS}/(\langle Lo^2\rangle/M)_{PMMA} = 1.08 \quad (8)$$

$$(\langle Lo^2\rangle/M)_{PS}/(\langle Lo^2\rangle/M)_{PBD} = 0.66 \quad (8)$$

Instrumental Spreading Parameter Calibration Curves

The absolute and infinite resolution number average and weight average molecular weights of the ASTM standards were used to determine the spreading parameters shown in Table 7-1. The magnitudes of the corrections are better visualized by defining:

$$B = 1 + SK/2 \qquad 7:1:4$$

$$H = \exp(-D_2^{\,2}/4h) \qquad 7:1:5$$

where:

$$\bar{M}_n(h,SK) = \bar{M}_n(\infty) \quad B/H \qquad 7:1:6$$

$$\bar{M}_w(h,SK) = \bar{M}_w(\infty) \quad B\ H \qquad 7:1:7$$

A plot of the spreading parameters h, SK and the magnitudes of the corrections are graphically illustration in Figure 7-3 and 7-4. The scatter in Figure 7-3 and 7-4 is reasonable and consistent with similar plots in the literature.[1,2,11]

The net effect of the corrections, for $26 < v < 32$, is to raise $\bar{M}_n(\infty)$ and slightly raise $\bar{M}_w(\infty)$. Normally, the correction factor for $\bar{M}_w(\infty)$ should be less than 1.0; however, in this case, the correction

| Sample | PRV | h | SK | H | B |
|---|---|---|---|---|---|
| ASTM-1 | 34.0 | 3.29 | .02 | .94 | 1.01 |
| ASTM-2 | 33.3 | 2.62 | .24 | .93 | 1.12 |
| ASTM-3 | 32.1 | 1.58 | .24 | .90 | 1.12 |
| ASTM-4 | 31.2 | 1.19 | .28 | .88 | 1.14 |
| ASTM-5 | 30.0 | 1.06 | .44 | .89 | 1.22 |
| ASTM-6 | 29.2 | 1.06 | .36 | .90 | 1.18 |
| ASTM-7 | 28.3 | .89 | .60 | .89 | 1.30 |
| ASTM-8 | 26.6 | .48 | .62 | .85 | 1.31 |
| ASTM-9 | 25.2 | .43 | .46 | .87 | 1.23 |
| ASTM-10 | 23.7 | .20 | 2.30 | .80 | 2.15 |

Spreading Parameters for ASTM Standards

Table 7-1

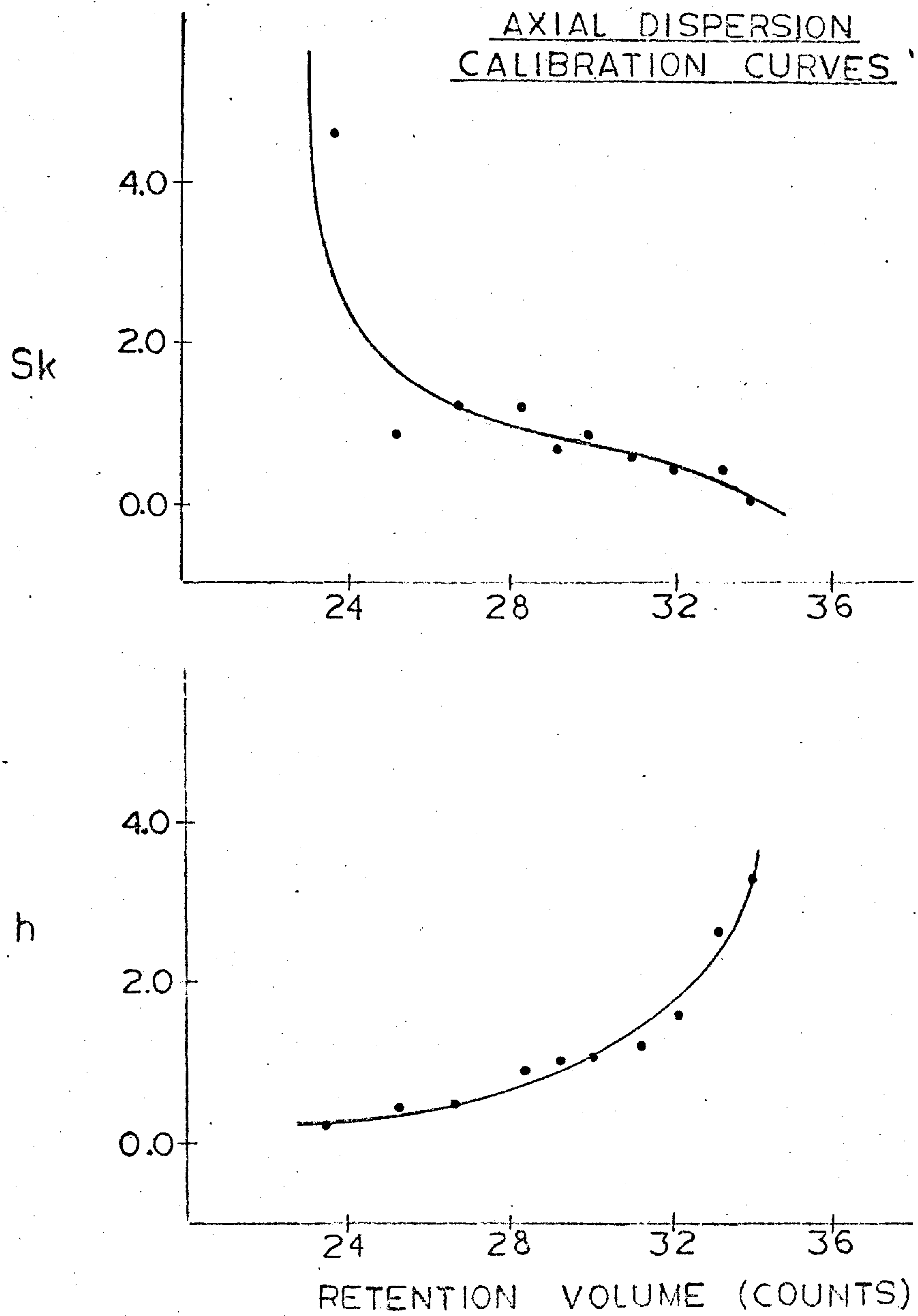

AXIAL DISPERSION
CALIBRATION CURVES
Sk
4.0
2.0
0.0
24
28
32
36
h
4.0
2.0
0.0
24
28
32
36
RETENTION VOLUME (COUNTS)

FIGURE 7-3

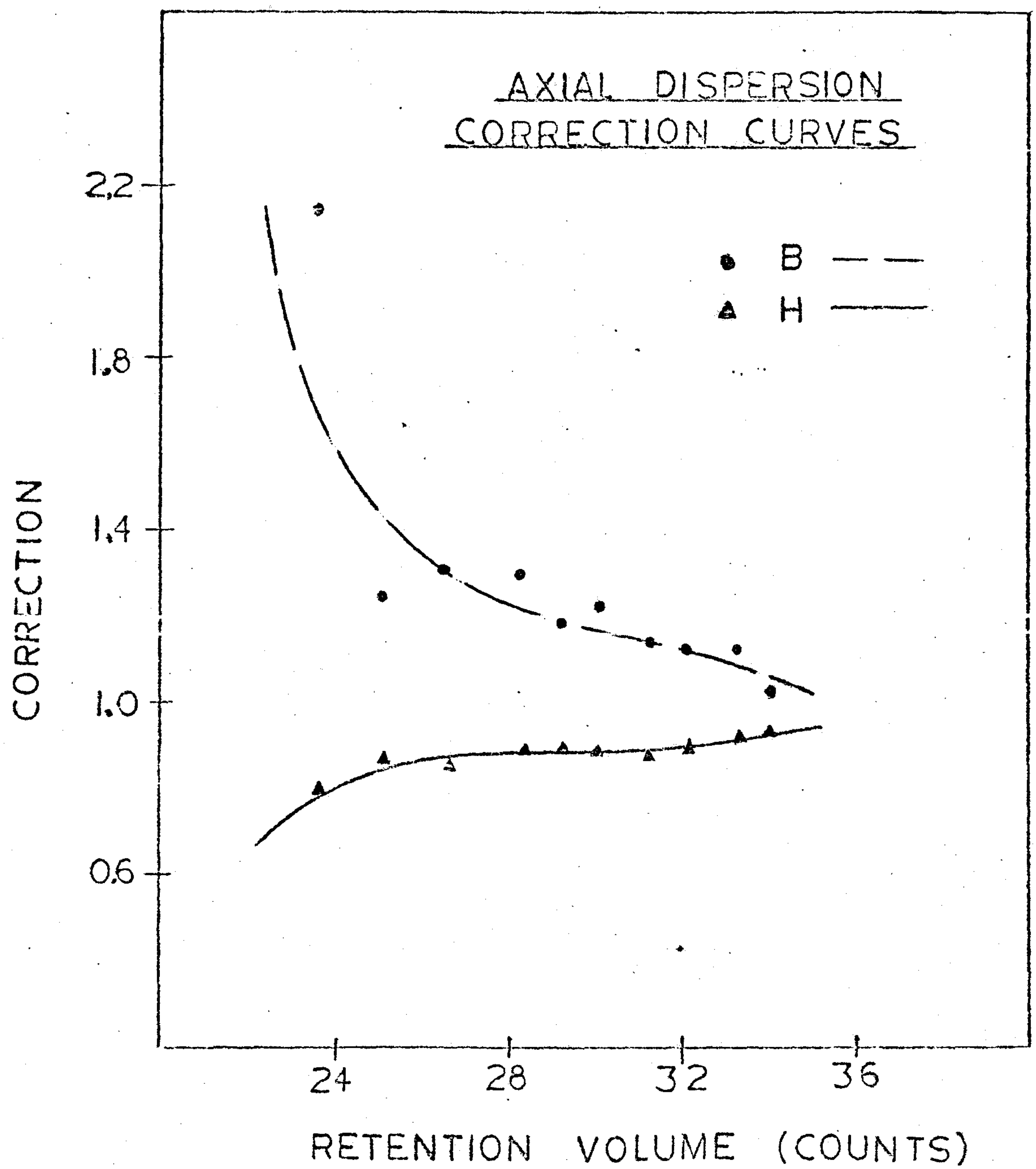

AXIAL DISPERSION
CORRECTION CURVES
B
H
2.2
1.8
1.4
1.0
0.6
CORRECTION
24
28
32
36
RETENTION VOLUME (COUNTS)

FIGURE 7-4

for skewing is larger than the correction for axial dispersion. The magnitude of the skewing correction depends on the concentration of the polymer injected (see Section 5:4). The concentration of polymer for injection, chosen by the ASTM committee, was 0.25 wt%, quite a high value[19]. As a result there was significant skewing due to overloading. Chan[19] has recently published data which supports this conclusion. He found at 0.25% concentration the skewing corrections necessary were much larger than at 0.06% concentration.

## 7:2 Application of the Correction Curves

Two National Bureau of Standards (NBS 705 and 706) polystyrene standards were injected to test the validity of the skewing corrections. The infinite resolution molecular weight averages were corrected and compared with the absolute values. The results are summarized in Table 7-2. The corrected number average molecular weights, $\bar{M}_n(h,SK)$, agree within 7% of the absolute number average molecule weights, $\bar{M}_n(t)$. The corrected weight average molecular weights, $\bar{M}_w(h,SK)$, agree within 13% of the absolute weight average molecular weights, $\bar{M}_w(t)$. The corrections improve $\bar{M}_n(\infty)$ but do not improve $\bar{M}_w(\infty)$. Two observations suggest the source of these inconsistances.

First, the difference in the absolute and corrected weight average molecular weights for the broad standard NBS-706 is greater than the difference for the narrow standard, NBS-705. This is expected. The skewing correction determined with narrow standards will over correct broad standards because of concentration effects as noted in 5:4. The

| Sample | $\overline{M_n}(t)$ | $\overline{M_w}(t)$ | $\overline{M_n}(\infty)$ | $\overline{M_w}(\infty)$ | $\overline{M_n}(h,SK)$ | $\overline{M_w}(h,SK)$ |
|---|---|---|---|---|---|---|
| NBS705 | 170,000 | 179,000 | 132,000 | 190,000 | 177,000 | 194,000 |
| NBS706 | 136,000 | 257,000 | 105,000 | 290,000 | 145,000 | 295,000 |

Molecular Weight Averages For

NBS Polystyrene Standards

Table 7-2

weight average molecular weight for NBS-706 is therefore over corrected for skewing. The second observation was that the corrected averages are all greater than the reported values. It is possible that the corrected molecular weight averages of the ASTM standards, used to establish the calibration curve, were biased because of experimental error in determining the absolute averages. It is also possible that the calibration curve is biased in the retention volume range of the NBS standards. It might be better to approximate the polystyrene calibration curve with two linear segments rather than a third order polynomial.

Infinite resolution molecular weight averages were also computed for the PVC, PMMA and PBD standards. The infinite resolution molecular weight averages were corrected using the polystyrene instrument spreading curves, with the assumption that the instrument spreading correction curves are universal and depend on retention volume only. The results are found in Tables 7-3, 7-4 and 7-5. $\bar{M}_n$(h,SK) and $\bar{M}_n$(t) differed by 25%, while $\bar{M}_w$(h,SK) and $\bar{M}_w$(t) differed by 10%. In most cases the corrections gave results in better agreement with classical measurements. There are several complicating factors, however.

First, the errors in establishing a molecular weight calibration curve using a universal curve make it difficult in some cases to assess the true effect of the axial dispersion corrections. The corrections increased the accuracy of the molecular weight averages predicted, when Dawkins' universal calibration is used. On the other hand, using Benoit's approach for universal calibration, the corrections improved the molecular weight averages of the PVC and PBD samples, but not the PMMA samples.

| Sample | Calibration* Method | $\bar{M}_n(t)$ | $\bar{M}_w(t)$ | $\bar{M}_n(\infty)$ | $\bar{M}_w(\infty)$ | $\bar{M}_n(h,SK)$ | $\bar{M}_w(h,SK)$ |
|---|---|---|---|---|---|---|---|
| PVC-2 | B | 25,000 | 68,000 | 30,000 | 66,000 | 40,000 | 69,500 |
| | D | " | " | 23,000 | 49,600 | 31,000 | 57,000 |
| PVC-3 | B | 41,100 | 119,000 | 50,000 | 121,000 | 66,000 | 129,000 |
| | D | " | " | 38,000 | 96,000 | 51,000 | 109,000 |
| PVC-4 | B | 54,000 | 137,000 | 64,000 | 147,000 | 87,000 | 154,000 |
| | D | " | " | 52,000 | 122,000 | 74,000 | 130,000 |

*Calibration Method

B - Benoit's Method

D - Dawkins' Method

Table 7-3

Molecular Weight Averages for PVC

| Sample | Calibration* Method | $\bar{M}_n(t)$ | $\bar{M}_w(t)$ | $\bar{M}_n(\infty)$ | $\bar{M}_w(\infty)$ | $\bar{M}_w(h,SK)$ | $\bar{M}_w(h,SK)$ |
|---|---|---|---|---|---|---|---|
| PMMA-1 | B | 50,000 | 290,000 | 51,000 | 384,000 | 67,800 | 410,000 |
| | D | " | " | 58,500 | 279,000 | 77,000 | 310,000 |
| PMMA-2 | B | 33,400 | 78,000 | 37,000 | 87,000 | 50,000 | 94,000 |
| | D | " | " | 38,000 | 76,500 | 51,000 | 82,000 |
| PMMA-3 | B | 60,000 | 115,000 | 57,600 | 113,000 | 77,000 | 122,000 |
| | D | " | " | 57,000 | 95,000 | 81,000 | 104,000 |

*Calibration Method

B - Benoit's Method

D - Dawkins' Method

Table 7-4

Molecular Weight Averages of PMMA

| Sample | Calibration* Method | $\bar{M}_n(t)$ | $\bar{M}_w(t)$ | $\bar{M}_n(\infty)$ | $\bar{M}_w(\infty)$ | $\bar{M}_n(h,SK)$ | $\bar{M}_w(h,SK)$ |
|---|---|---|---|---|---|---|---|
| PBD-1 | B | 16,100 | 17,000 | 13,800 | 18,900 | 18,500 | 19,200 |
| | D | " | " | 9,000 | 16,400 | 13,000 | 17,500 |
| PBD-2 | B | 135,000 | 170,000 | 75,000 | 144,000 | 100,000 | 155,000 |
| | D | " | " | 92,000 | 164,000 | 124,000 | 175,000 |
| PBD-3 | B | 206,000 | 272,000 | 120,000 | 232,000 | 160,000 | 242,000 |
| | D | " | " | 150,000 | 254,000 | 204,000 | 272,000 |
| PBD-4 | B | 226,000 | 332,000 | 133,000 | 245,000 | 180,000 | 305,000 |
| | D | " | " | 163,000 | 317,000 | 222,000 | 340,000 |
| PBD-5 | B | 286,000 | 423,000 | 183,000 | 389,000 | 248,000 | 405,000 |
| | D | " | " | 220,000 | 410,000 | 300,000 | 436,000 |

*Calibration Method

B - Benoit's Method

D - Dawkins' Method

Table 7-5

Molecular Weight Averages for PBD

The uncertainty in the parameters a, K and $(\langle Lo^2 \rangle/M)$ used to predict the calibration curves for non-polystyrene standards, tended to mask the improved accuracy of the corrected averages.

A second complicating factor was the fact that axial dispersion correction curves for narrow standards were used to correct broad standards. The corrected number and weight average molecular weights for the narrow PBD samples differed by 15% and 5% from the absolute number and weight average, as compared to 25% and 10% for the broad PMMA and PVC samples. The skewing correction over corrected broad samples as predicted.

Finally, the third complicating factor was the assumption that the absolute molecular weight averages given by classical analytical techniques are correct. It is possible the absolute averages are significantly different than the real averages because of the experimental error associated with any method of measuring molecular weights such as osmometry and light scattering.

## 8. Conclusions

1. The minicomputer system proved suitable for simultaneous data acquisition and reduction in gel permeation chromatography. Molecular weight averages can be computed and made available within minutes of the completion of a sample run.

2. A minimum configuration minicomputer system containing 4K words of memory, which costs $10,500, is adequate for operation with one GPC. However, an 8K system is required if the system is used with more than one GPC simultaneously, more than one detector is used,or if the on-line calculational routines are to include more than the standard molecular weight average computations.

3. The assumption that axial dispersion correction curves are universal appears reasonable. Correction curves were found to be dependent on retention volume and independent of polymer composition. This was found to be reasonable for PVC, PS, PMMA AND PBD. It is therefore recommended that any reliable standards may be used to construct axial dispersion correction curves and that these curves may then be applied in a universal manner.

## 9. Recommendations

1. As a result of the initial success of the GPC minicomputer system, it is suggested that an additional 4K words of memory be added to the system. This would permit maximum utilization of the existing hardware by increasing the operating scope and program contents of the system.

2. Future studies are also suggested to determine the effect of concentration and polydispersity on the axial dispersion corrections. In particular, it would be very useful to establish a method to adjust axial disperion correction curves determined with narrow standards, for unknown samples with varying polydispersities.

## 10. Nomenclature

| | |
|---|---|
| a | Mark-Houwink constant |
| A/D | Analog to digital converter |
| B | 1 + Sk/2 |
| CPU | Central Processing Unit |
| $D_1$, $D_2$, $D_3$ | Molecular weight calibration curve constants |
| $D_1'$, $D_2'$, $D_3'$ | Universal calibration curve constants |
| F(v) | GPC detector response |
| G(v,y) | Instrument spreading function |
| h | symmetrical axial dispersion factor |
| H | $D_2^2/h$ |
| K | Mark-Houwink constant |
| k | molecular weight index, 1 = number, 2 = weight, etc. |
| Lo | unperterbed end-to-end dimmension |
| M | molecular weight |
| $\bar{M}k$ | $k^{th}$ molecular weight |
| $\bar{M}k(h)$ | $k^{th}$ molecular weight corrected for symmetrical axial dispersion |
| $\bar{M}_n(t)$ | absolute number average molecular weight |
| $\bar{M}_w(\infty)$ | infinite resolution number average molecular weight |
| $\bar{M}_n(h,SK)$ | corrected number average molecular weight |
| $\bar{M}_w(t)$ | absolute weight average molecular weight |
| $\bar{M}_w(\infty)$ | infinite resolution weight average molecular weight |
| $\bar{M}_w(h,SK)$ | corrected weight average molecular weight |

| | |
|---|---|
| PBD | Polybutadiene |
| PMMA | Poly(methyl-methacrylate) |
| PS | Polystyrene |
| PVC | Poly(vinyl-chloride) |
| $P(\infty)$ | infinite resolution polydispersity |
| $P(t)$ | absolute polydispersity |
| $P(D_2')$ | polydispersity compute with $D_2'$ |
| RMS(M) | Root mean squared molecular weight |
| RTC | Real Time Clock |
| SK | Skewing Factor |
| v | retention volume |
| W(M) | Molecular weight distribution molecular weight space |
| W(v) | Molecular weight distribution, retention volume space |

Greek Symbols

| | |
|---|---|
| $\alpha$ | linear deformation due to solvent-polymer interaction |
| $\eta$ | intrinsic viscosity |

## 11. References


1. S.T. Balke and A.E. Hamielec, J. Appl. Polym Sci., 13, 1381 (1969).

2. E.M. Rosen and T. Provder, Separ. Sci., 5, 485 (1970).

3. P.J. Brussan, J. Chrom. Sci., 9, 1 (1971).

4. A. Hui, Ph.D. dissertation, "Kinetics of Free Radical Polymerization of Styrene to Complete Conversion", McMaster University.

5. L.D. Moore and J.R. Overton, Ninth Int. GPC Seminar Preprints, 399 (1970).

6. Grubisic, Rempp, Benoit, Polymer Letters, 5, 753 (1967).

7. H. Benoit and Z. Grubisic, Fifth Int. GPC Seminar Preprints, 119, (1968).

8. J.V. Dawkins, J. Macrmol. Sci.-Phys., B2(4), 623 (1968).

9. L.H. Tung, J. Appl. Polym. Sci., 10, 375 (1966).

10. A.C. Ouano and J.A. Biesenberger, Ninth Int. GPC Seminar Preprints, 465 (1970).

11. L.H. Tung and J.R. Runyon, J. Appl. Polym. Sci., 13, 2397, (1969).

12. P.E. Pierce and J.E. Armonus, J. Appl. Polym. Sci. C, 21, 23 (1968).

13. A.E. Hamielec and W.H. Roy, J. Appl. Poly. Sci., 13, 1319 (1969).

14. B.S. Ehrlich and W.V. Smith, Polymer Preprints, ACS Div. Polymer Chem., 12(2), 847 (1971).

15. E.M. Rosen and T. Provder, J. Apply. Polym. Sci., 15, 1687 (1971).

16. T. Ishige, S.I. Lee and A.E. Hamielec, J. Apply. Polym Sci., 15, 1607, (1971).

17. M. Hess and R.F. Kratz, J. Polym. Sci., A-2, 4, 731 (1966).

18. K.S. Chang and Y.M. Harms, J. Appl. Polym. Sci., 13, 1459 (1969).

19. R.K.S. Chan, Polym. Eng. Sci., 11, 152, (1971).

20. R.K.S. Chan, Polym. Eng. Sci., 11, 187, (1971).

21. J. Brandrup and E.H. Immergut, Polymer Handbook, John Wiley & Sons, New York (1966).

22. P.J. Flory, Principals of Polymer Chemistry, Cornell University Press, Ithaca, N.Y. (1953).

23. J.L. Butler, Instrumentation Technology, 17, October, 67 (1970).

24. Hooker Chemical private communication.

25. W. Kipiniak and P. Quint, Control Engineering, Reprint No. 940, Reuben H. Donnelly Corporation (1968).

# APPENDIX

A:1 Details of the On-line Data Acquisition and Reduction System

A:2 Details of Program 2, Calibration Curve Search Program

A:3 Details of Program 3, Axial Dispersion Calibration Program for Standards

A:4 Operating Instructions

A:5 The Contact Sense Device

## A:1 Details of the On-line Data Acquisition and Reduction System

The data acquisition and reduction software was programmed to operate in a Nova 1200 minicomputer with 4K words of memory. It is adequate for on-line data acquisition and reduction from one GPC. The total data acquisition and reduction software package includes the basic floating point interpreter, the Operating System and Program 1 calculation routines. To fascilitate discussion, the remainder of this section has been subdivided as follows:

A:1-1 Data Coding

A:1-2 Operating System - Logic

A:1-3 Operating System - Symbols

A:1-4 Operating System - Program Summaries and Listings

A:1-5 Program 1 - Logic

A:1-6 Program 1 - Symbols

A:1-7 Program 1 - Program Summaries and Listings

### A:1-1 Data Coding

The data coding method used in the GPC minicomputer system is summarized below:

| Number | Meaning |
|---|---|
| 0 - 1023 | Chromatogram height |
| 1100 - 2000 | Sample numbers |
| 2100 - 3000 | Sample stop, (sample number + 1000) |
| 4000 | Sample injection |
| 7000 - 7100 | Retention volume dump (7000) plus clock reading (0 - 100) |

A:1-2 Operating System - Logic

The operating system controls data acquisition, storage and output. It includes eleven subroutines and one storage buffer, structured around a priority interrupt system. A flow diagram of the operating system is illustrated in Figure A-1. As Figure A-1 indicates, the operating system is conviently divided into four parts, the initialization and auto-action routine, the interrupt routines, the teletype input-output (I/O) service routines, and the storage buffer. In discussing the interaction of these four parts, reference will be made to specific subroutines using an abbreviated form of the subroutine name. The reader is directed to the program listings of A:1-4 for specific details of these individual routines.

The system begins in the initialization routine, IANDM, which clears the storage buffer, starts the real-time clock, starts device 40 (contact sense lines), and enables the interrupts. Control is then transferred to the auto-action routine with a zero (low) priority. The auto-action routine monitors software flags. If a flag is set to non-zero, control is transferred from the auto-action routine to the appropriate auto-routine. That auto-routine outputs a message, outputs data, executes a calculation, or rings the teletype bell to indicate a retention volume dump.

When a device requests attention, the auto-action routine is interrupted. Control is transferred (hardware transfer) to the master interrupt routine, MIR. The master interrupt routine stores the pre-interrupt state of the CPU and transfers control to an interrupt servicing

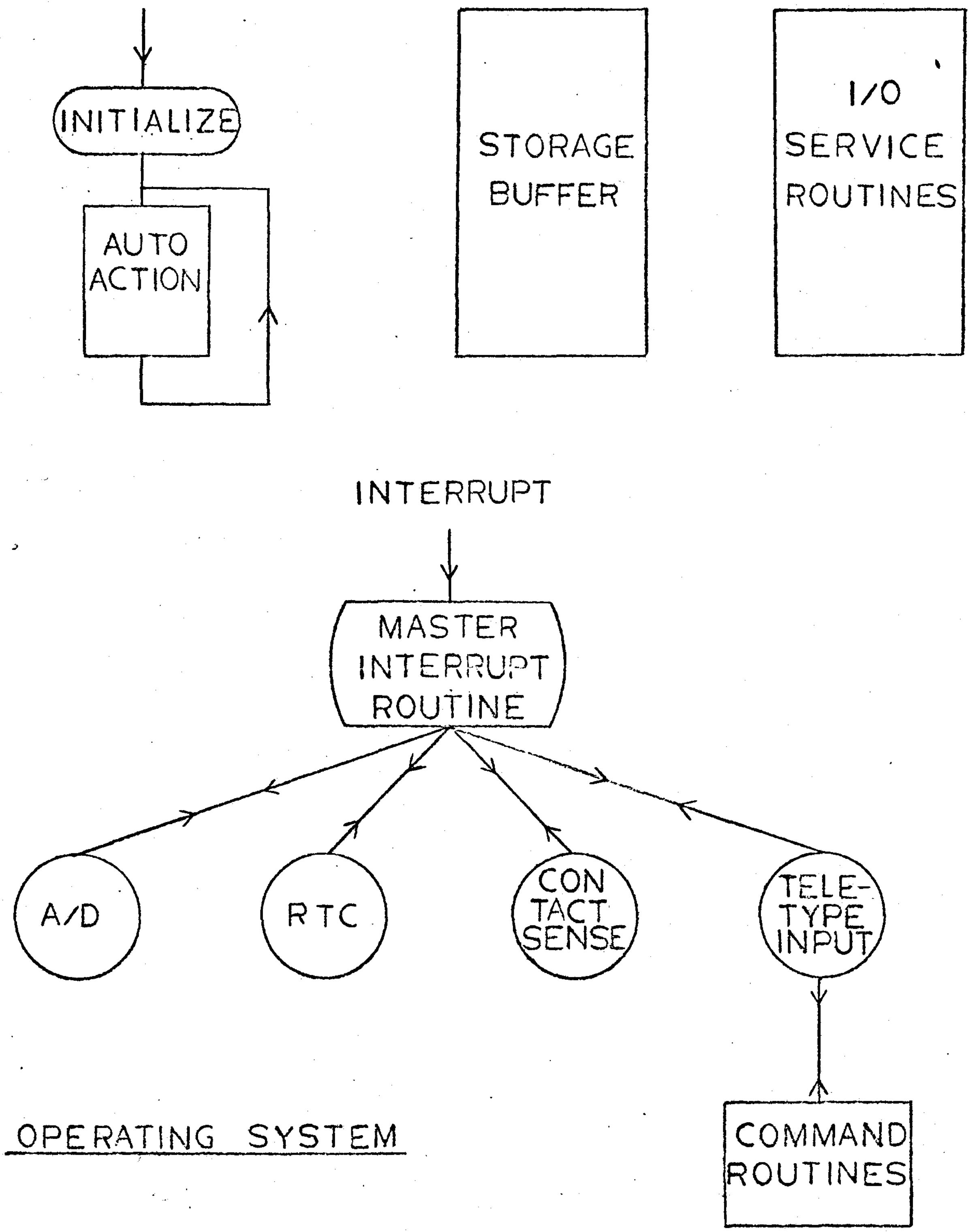

INITIALIZE
AUTO ACTION
STORAGE BUFFER
I/O SERVICE ROUTINES
INTERRUPT
MASTER INTERRUPT ROUTINE
A/D
RTC
CON TACT SENSE
TELE-TYPE INPUT
COMMAND ROUTINES
OPERATING SYSTEM

FIGURE A-1

routine (see 3:4). The interrupt servicing routine acknowledges the device which caused the interrupt in a manner specified by the user through a series of operating commands. When the user types an operating command the teletype input interrupt servicing routine, TELIN, decodes the input and transfers control to one of the nine command routines. The command routines request new operating information or perform a specific task, such as outputting current operating information. Any new operating information is stored in the operating system storage buffer and used by the interrupt servicing routines of program SERV1 to determine what, if any, action is to be taken when the real-time clock, device 40, or the A/D converter causes an interrupt.

The operating system storage buffer (GST) contains, in addition to the operating parameters, the auto-action flags and chromatogram heights. Each storage location is defined as a displacement from a storage pointer, making it easy to expand the system to operate with additional GPC's.

The teletype I/O service routines IOSER, and MESSOP, are used for teletype communication outside the interrupt scheme. After control is transferred to a command routine, teletype communication is no longer a part of the interrupt system. The I/O servicing routines, therefore, include basic alphameric, arithmetic and paper control input-output subroutines. Some of these subroutines change the current mask of the master interrupt routine to disable teletype input interrupts. Changing, the mask avoids potential reentrance problems which could arise by a device causing an interrupt before the CPU had completed servicing a prior one.

## A:1-3 Operating System - Symbols

The following symbols are defined for interprogram communication:

| | | |
|---|---|---|
| ATDX | - | address of A/D test routine |
| BEGIN | - | address of BEGIN command routine |
| BIND | - | address of binary to decimal output routine |
| CMASK | - | current interrupt priority mask |
| CRLF | - | address of carriage return-line feed routine |
| DATA | - | address of data command routine |
| DBIN | - | address of decimal to binary input routine |
| ERROT | - | address of error message routine |
| FEED | - | address of blank tape output routine |
| GETC | - | address of character input I/O routine |
| INJ | - | address of injection acknowledge routine |
| ISR | - | address of master interrupt routine |
| LOOK | - | address of LOOK command routine |
| MNPL | - | monitor output - numbers per line |
| MON | - | address of monitor command routine |
| MSP | - | monitor output - spaces between numbers |
| NUPL | - | table output - numbers per line |
| PUTC | - | address of character output I/O routine |
| RETN | - | return address of master interrupt routine |
| SPACE | - | address of space output I/O routine |
| SPPL | - | table output - spaces between numbers |
| STOP | - | address of STOP routine |
| TABLE | - | address of TABLE output I/O routine |

| | | |
|---|---|---|
| .A40 | - | address of device 40 interrupt service routine |
| .ATYPE | - | address of auto-type routine |
| .ATD | - | address of A/D interrupt service routine |
| .CLOCK | - | address of real-time clock interrupt service routine |
| .EV | - | address of elution volume interrupt service routine |
| .IINIT | - | address of interrupt initialization routine |
| .STO1 | - | address of auto-stop routine |
| .TELIN | - | address of teletype input interrupt service routine |

A:1-4 Operating System - Listings

MIR - Master Interrupt routine

IANDM - Initialization add auto-action

SERV1 - Interrupt service routine

TELIN - Teletype input service routine

DATA - Data and Begin command

MONIT - Monitor Command

OUTPUT- Type command

LKATD - LOOK and ATD commands

STOP - Manual and auto stop

IOSER - Input-output routines

MESSOP- messages for operating system

GST - operating system storage buffer

**Program** Initialization and Auto-Action Routines

**Synopsis** This program includes the initialization routine for the operating system and the auto action monitor routine.

**Method**

1. Clear storage buffer
2. Exit to external initialization routines
3. Check auto-action software flags and exit to appropriate routines if necessary
4. Repeat step 3.

**Notes** This program contains the starting address of the system GPC.SV.

Location 2 - start with buffer clear

Location 3 - start without buffer clear

```
;***********************************************
;INITIALIZATION AND AUTO-ACTION ROUTINES
;REVISED NOV 1, 1971
;***********************************************


;MAIN PROGRAM BEGINS IN THIS ROUTINE AT
;LOCATION        2       START WITH STORAGE BUFFER CLEAR
;                3       START WITHOUT BUFFER CLEAR


                .TITL   IANDM
                .ENT     MNPL MSP
                .EXTD   GETC .CLEAR  AM12  .GINIT
                .EXTD   .IINIT SPACE CRLF TYPE TABLE DBIN BIND
                .EXTD   .ATYPE .ACAL .G1 NUM    ADDRS

      000002    .LOC 2
00002 006002$           JSR@ .CLEAR
00003 006000-           JSR@ .IN3
      000007    .LOC 7
00007 000150'           WA               ;POINTER FOR FP WORK AREA

                .ZREL
00000-000000'   .IN3:   INIT2
                .NREL

                ;INITIALIZE SYSTEM

00000'062677    INIT2:  IORST
00001'020001$           LDA 0 GETC
00002'040040            STA 0 40         ;STORE INPUT ROUTINE IN 40
00003'006007$           JSR@ CRLF
00004'006005$           JSR @.IINIT      ;EXIT TO INTERUPT INITIAL
00005'006004$           JSR@ .GINIT      ;EXIT TO GPC INITIALIZE
00006'006005            JSR@ 5           ;INITIALIZE FLOT PT
00007'020534            LDA 0 MNPL
00010'040532            STA 0 MCOUNT     ;SET UP MONITOR OUTPUT PAR
00011'102520            SUBZL 0 0        ;FREQ=1
00012'061114            DOAS 0 RTC       ;START CLOCK
00013'060140            NIOS 40          ;TURN ON SPECIAL DEVICE
00014'060177            INTEN            ;TURN ON INTERUPTS
00015'006007$           JSR@ CRLF
00016'020531            LDA 0 T52
00017'006041            JSR@ 41          ;"*"
00020'006007$           JSR@ CRLF


                ;MAIN ROUTINE RECYCLES HERE

                ;AUTO TYPE ?

00021'030016$   TEST1:  LDA 2 .G1
00022'021030            LDA 0 30 2       ;AUTO TYPE
00023'101235            MOVZR# 0 0 SNR   ;SAMPLE TO BE TYPE
00024'000412            JMP M1A
```

```
00025'035027            LDA 3 27 2
00026'117000            ADD 0 3
00027'025777            LDA 1 -1 3         ;PICK UP SAMPLE NO.
00030'044405            STA 1 AT1
00031'021024            LDA 0 24 2         ;MINIMIUM
00032'025022            LDA 1 22 2         ;MAXIMIUM
00033'015030            DSZ 30 2           ;DECREMENT AT
00034'006014$           JSR@ .ATYPE
00035'000000  AT1:      0

              ;AUTO CALCULATION ?

00036'030016$ M1A:      LDA 2 .G1
00037'021034            LDA 0 34 2         ;AUTO ADJ
00040'101235            MOVZR# 0 0 SNR     ;CAL TO BE DONE
00041'000406            JMP M2             ;NO
00042'035033            LDA 3 33 2         ;YES
00043'117000            ADD 0 3
00044'025777            LDA 1 -1 3         ;PICK UP SAM NO.
00045'015034            DSZ 34 2           ;DECREMENT COUNTER
00046'006015$           JSR@ .ACAL         ;DO CALC

              ;BELL OUTPUT ON EV ?

00047'030016$ M2:       LDA 2 .G1
00050'021011            LDA 0 11 2         ;BELL COUNT
00051'101005            MOV 0 0 SNR
00052'000407            JMP M2A            ;TRY MONITOR
00053'015011            DSZ 11 2
00054'000401            JMP .+1
00055'020403            LDA 0 BELL
00056'006041            JSR@ 41            ;OUTPUT BELL
00057'000770            JMP M2
00060'000007  BELL:     7

              ;MESSAGE OUTPUT ?

00061'020017$ M2A:      LDA 0 NUM
00062'101005            MOV 0 0 SNR        ;IS NUM ZERO
00063'000420            JMP M3             ;YES
00064'006007$           JSR@ CRLF
00065'014017$           DSZ NUM            ;DECREMENT NUMBER
00066'000401            JMP .+1
00067'060277            INTDS
00070'014020$           DSZ ADDRS
00071'034020$           LDA 3 ADDRS
00072'014020$           DSZ ADDRS
00073'060177            INTEN
00074'021777            LDA 0 -1 3         ;MESSAGE BYTE POINTER
00075'031400            LDA 2 0 3          ;NUMBER
00076'006010$           JSR@ TYPE          ;OUTPUT MESSAGE
00077'145000            MOV 2 1
00100'006013$           JSR@ BIND
00101'006007$           JSR@ CRLF
00102'000717            JMP TEST1          ;TRY AGAIN
```

```
                    ;MONITOR OUTPUT ?

00103'030016$ M3:      LDA 2 .G1
00104'025013           LDA 1 13 2           ;MONITOR STATUS
00105'125005           MOV 1 1 SNR
00106'000741           JMP M2               ;REPEAT IF ZERO
00107'035014           LDA 3 14 2           ;LAST MONITOR PRINT OUT
00110'021000           LDA 0 0 2            ;CURRENT LOCATION
00111'162033           ADCZ# 3 0 SNC        ;CURRENT>LAST MONITOR
00112'000422           JMP M3C              ;NO
00113'025400  M3D:     LDA 1 0 3            ;PICK UP NUMBER
00114'006013$          JSR@ BIND            ;OUTPUT IT
00115'021014           LDA 0 14 2           ;MONITOR LOC
00116'025022           LDA 1 22 2           ;MAX
00117'115400           INC 0 3              ;MON+1
00120'122415           SUB# 1 0 SNR         ;MON=MAX
00121'035024           LDA 3 24 2           ;START
00122'055014           STA 3 14 2           ;REPLACE MON LOC
00123'014417           DSZ MCOUNT
00124'000405           JMP M3B
00125'020416  M3A:     LDA 0 MNPL
00126'040414           STA 0 MCOUNT
00127'006007$          JSR@ CRLF
00130'000717           JMP M2
00131'024413  M3B:     LDA 1 MSP
00132'006006$          JSR@ SPACE           ;OUTPUT SPACES
00133'000714           JMP M2
00134'025023  M3C:     LDA 1 23 2           ;WARNING
00135'106433           SUBZ# 0 1 SNC        ;WARN>= CURR
00136'000711           JMP M2  ;NO
00137'136033           ADCZ# 1 3 SNC        ;MON>= WARN
00140'000707           JMP M2  ;NO
00141'000752           JMP M3D              ;YES

00142'000000  MCOUNT:  0           ;OUTPUT COUNTER
00143'000010  MNPL:    10          ;NUMBERS PER LINE
00144'000004  MSP:     4           ;SPACE BETWEEN NUMBERS
00145'000003  T3:      3
00146'177757  TN21:    -21
00147'000052  T52:     52
      000100  WA:      .BLK 64.             ;FP WORK AREA

              .END
```

| | |
|---|---|
| Program | Master Interrupt Routine |
| Synopsis | This Master Interrupt Routine is to be used with the A/D convertor, real-time clock, device 40, and teletype input. |
| Method | 1. Store accumulators, carry, program counter, current mask<br>2. Determine which device caused interrupt<br>3. Issue new mask and exit to service routine<br>4. Restore accumulators, carry, program counter, and mask<br>5. Return to interrupted point. |
| Notes | Priority levels (4-highest)<br>4. Real time clock<br>3. A/D convertor and device 40<br>2. Teletype input<br>1. Auto-action<br>- Error HALT on seven consecutive interrupts<br>- Error HALT on unknown interrupt |

```
;*************************************************
;MASTER INTERRUPT ROUTINE

;REVISED NOV 2, 1971

;*************************************************


;INITIALIZATION
;CALL SEQUENCE  -          JSR@ .IINIT
;                          RETURN


; INTERRUPT ROUTINE
;HARDWARE CALL  -          @1
;                          RETURN TO INTERRUPT POINT

;NOTE
;SERVICE ROUTINES MAY RETURN        -      @RETN
;OR                                 -      @ 3


            .TITL   MIR
            .EXTD   .INJ .G1 .A40 .ATD .TELIN .CLOCK
            .ENT    ISR .IINIT RETN CMASK


      000001 .LOC 1
00001 000006' ISR                    ;POINTER TO INTERUPT ROUTINE


            .ZREL

00000-000000  CMASK:  0              ;CURRENT MASK
00001-000000' .IINIT: IINIT          ;POINTER TO INITIALIZATION
00002-000064' RETN:   XRETN          ;RETURN AFTER SERVICING


            .NREL

            ;INITIALIZATION ROUTINE

00000'024516  IINIT:  LDA 1 ASTK     ;INITIALIZE POINTER
00001'044516          STA 1 ADSTK
00002'126520          SUBZL 1 1      ;MASK EQUAL TO ONE
00003'066077          DOB 1 CPU      ;ISSUE MASK
00004'044000-         STA 1 CMASK
00005'001400          JMP 0 3


            ;MAIN INTERUPT ROUTINE
            ;SAVE ACCUMULATORS,PC,CARRY AND CMASK

00006'056511  ISR:    STA 3 @ADSTK
00007'034510          LDA 3 ADSTK    ;AC3=ADDRESS OF STACK
00010'041401          STA 0 1 3      ;SAVE ACC
00011'045402          STA 1 2 3
00012'051403          STA 2 3 3
00013'102560          SUBCL 0 0
```

```
00016'041405           STA 0 5 3
00017'020000-          LDA 0 CMASK
00020'041406           STA 0 6 3
00021'030477           LDA 2 SIZE        ;PUSH STACK
00022'157000           ADD 2 3
00023'030476           LDA 2 CHECK
00024'172433           SUBZ# 3 2 SNC     ;CHECK>= NEXT
00025'063077           HALT              ;NO
00026'054471           STA 3 ADSTK       ;YES
00027'063777           SKPDZ CPU         ;POWER FAIL
00030'000457           JMP PWRFL


                 ;DETERMINE WHICH DEVICE CAUSED INTERUPT

00031'063714  ISR1:    SKPDZ RTC
00032'000422           JMP IRTC
00033'063721           SKPDZ ADCV
00034'000415           JMP IADCV
00035'063740           SKPDZ 40
00036'000407           JMP I40
00037'063710           SKPDZ TTI
00040'000417           JMP ITTI
00041'000401           JMP FAULT

                 ;ERROR, NO SUCH DEVICE

00042'061477  FAULT:   INTA 0
00043'063077           HALT
00044'000420           JMP XRETN


                 ;PICK UP MASK

00045'061477  I40:     INTA 0            ;BRING IN CODE
00046'024446           LDA 1 M40
00047'034003S          LDA 3 .A40
00050'000411           JMP OUT
00051'024441  IADCV:   LDA 1 MAD
00052'034004S          LDA 3 .ATD
00053'000406           JMP OUT
00054'024437  IRTC:    LDA 1 MRTC
00055'034006S          LDA 3 .CLOCK
00056'000403           JMP OUT
00057'024436  ITTI:    LDA 1 MTTI
00060'034005S          LDA 3 .TELIN


                 ;EXIT TO ROUTINE

00061'044000- OUT:     STA 1 CMASK
00062'066177           DOBS 1 CPU
00063'005400           JSR 0 3


                 ;SERVICE ROUTINES RETURN HERE
                 ;RESTORE INFORMATION AND RETURN TO
```

```
00065'034432            LDA 3 ADSTK         ;STACK
00066'030432            LDA 2 SIZE
00067'156400            SUB 2 3
00070'031406            LDA 2 6 3           ;AC2=OLD MASK
00071'072077            MSKO 2              ;ISSUE OFD MASK
00072'054425            STA 3 ADSTK         ;UPDATE POINTER
00073'050000-           STA 2 CMASK         ;UPDATE MASK
00074'021405            LDA 0 5 3
00075'040000            STA 0 0
00076'021404            LDA 0 4 3           ;RESTORE CARRY
00077'101220            MOVZR 0 0
00100'021401            LDA 0 1 3           ;RESTORE ACC
00101'025402            LDA 1 2 3
00102'031403            LDA 2 3 3
00103'036414            LDA 3 @ADSTK
00104'060177            INTEN
00105'002000            JMP @0

                ;POWERFAIL INTERUPT

00106'002002-           JMP@ RETN
00107'020777    PWRFL:  LDA 0 .-1
00110'040000            STA 0 0
00111'063077            HALT

                ;MASKS
00112'000773    MAD:    773                 ;A/D
00113'000777    MRTC:   777                 ;REAL TIME CLOCK
00114'000773    M40:    773                 ;DEVICE 40
00115'000003    MTTI:   3                   ;TELETYPE INPUT

                ;STORAGE POINTERS
00116'000122'   ASTK:   STACK
00117'000122'   ADSTK:  STACK
00120'000007    SIZE:   7
00121'000172'   CHECK:  STACK+50            ;IN LAST BLOCK
      000052    STACK:  .BLK 6*7            ;STORAGE BLOCK

                .END
```

**Program** Interrupt Service Routines

**Synopsis** This program includes the interrupt service routines for the real-time clock, the A/D converter, a sample injection, and retention volume dump.

**Methods**

Sample Injection

1. Increment sample counter
2. If it is first injection, reset clock counters
3. Store sample number in memory
4. Store injection indicator in memory
5. Activate injection acknowledge message
6. Return

Elution Volume Dump

1. Increment elution volume counter
2. Mask elution volume for 10 seconds
3. Store elution volume indicator plus clock reading in memory
4. Turn chromatogram sampling flag on, if ev= start
5. Turn chromatogram sampling flag to off, if ev = stop and transfer control to STOP routine
6. Return

Real-time Clock

1. Decrement counter one, return if non-zero
2. Reset counter one
3. Decrement counter two, return if non-zero
4. Rest counter two
5. Turn off A/D converter
6. Return

A/D Converter

1. Bring in converted signal
2. Store in operating system buffer
3. Return

Note - If "RV Sense" is set to zero with Begin or Data command, the real-time clock routine will simulate a retention volume dump every minute for reference purposes.

```
                ;*********************************************

                ;INTERRUPT SERVICE ROUTINES FOR
                ;       CLOCK
                ;       ATD
                ;       DEVICE 40 (CONTACT SENSE)


                ;REVISED NOV 1, 1971

                ;*********************************************



                .TITL   SERV1
                .ENT    INJ .A40 .CLOCK
                .ENT     PRESENT .ATD
                .EXTD   AM30   .G1  .STO1 ADDRS NUM RETN


                .ZREL
00000-000000'   .A40:   A40
00001-000133'   .CLOCK: CLOCK
00002-000210'   PRESENT:XPRE
00003-000204'   .ATD:   ATD

                .NREL

                ;DEVICE 40 CONTACT SENSE LINES
                ;CALL SEQUENCE  -        JSR@ .A40
                ;                        RETURN
                ;INPUT  - AC0 HAS INTA


00000'060140    A40:    NIOS 40
00001'101113            MOVL# 0 0 SNC    ;BIT 0 SET
00002'001400            JMP 0 3          ;RETURN, CONTACT OPEN
00003'024406            LDA 1 T7
00004'034407            LDA 3 .EV
00005'123404            AND 1 0 SZR      ;RV OR INJ
00006'034404            LDA 3 .INJ
00007'005400            JSR 0 3
00010'002006$           JMP @RETN        ;RETURN TO MIR

00011'000007    T7:     7
00012'000014'   .INJ:   INJ
00013'000072'   .EV:    EV


                ;INJECTION RECOGNITION ROUTINE
                ;CALL SEQUENCE  -        JSR@ .INJ
                ;                        RETURN

00014'054450    INJ:    STA 3 ISAVE
00015'030002$           LDA 2 .G1
00016'021026            LDA 0 26 2       ;AUTO INJ
00017'101005            MOV 0 0 SNR
00020'000404            JMP INJ3
00021'021015            LDA 0 15 2
00022'101400            INC 0 0
```

```
00023'041016             STA 0 16 2          ;SAM NO. +1
00024'021001    INJ3:    LDA 0 1 2           ;NUMBER OF SAMPLES
00025'101005             MOV 0 0 SNR         ;FIRST OR SECOND SAMPLE - ?
00026'000405             JMP INJ1            ;FIRST SAMPLE
00027'025006    INJ2:    LDA 1 6 2           ;CURRENT EV
00030'045004             STA 1 4 2           ;EV AT SECOND INJ
00031'025016             LDA 1 16 2          ;LOAD SAMPLE NUMBER
00032'000412             JMP IREC
00033'020432    INJ1:    LDA 0 T100
00034'041010             STA 0 10 2          ;RESET CLOCK 2 TO 100
00035'126400             SUB 1 1
00036'045006             STA 1 6 2           ;SET EV TO 0
00037'024427             LDA 1 T599
00040'045020             STA 1 20 2          ;SET MINUTE COUNTER TO 600
00041'021005             LDA 0 5 2
00042'041007             STA 0 7 2           ;RESET CLOCK 1
00043'025015             LDA 1 15 2          ;LOAD SAMPLE NUMBER
00044'121000    IREC:    MOV 1 0
00045'006002-            JSR@ PRESENT
00046'020001$            LDA 0 AM30          ;"INJECTION NOTED FOR"
00047'042004$            STA@ 0 ADDRS        ;STORE MESSAGE BYTE POINTER
00050'010004$            ISZ ADDRS           ;INCREMENT ADDRESS
00051'046004$            STA 1 @ADDRS        ;STORE SAMPLE NUMBER
00052'010004$            ISZ ADDRS
00053'010005$            ISZ NUM
00054'011001             ISZ 1 2             ;INCREMENT SAMPLE COUNTER
00055'024413             LDA 1 T4000
00056'020407             LDA 0 T100
00057'035010             LDA 3 10 2          ;CLOCK 2
00060'162400             SUB 3 0             ;100 - CLOCK2
00061'123000             ADD 1 0             ;ADD 4000+ EITHER MINUTE OR 100B
00062'004526             JSR XPRE            ;OUTPUT INJECTION NUMBER
00063'002401             JMP@ ISAVE
00064'000000    ISAVE:   0
00065'000144    T100:    100.
00066'001127    T599:    599.
00067'001130    T600:    600.
00070'007640    T4000:   4000.
00071'015530    T7000:   7000.


                ;ELUTION VOLUME ROUTINE
                ;CALL SEQUENCE  -    JSR@ .EV
                ;                    RETURN

00072'054440    EV:      STA 3 ESAVE
00073'030002$            LDA 2 .G1
00074'021001             LDA 0 1 2           ;NUM OF SAM
00075'101005             MOV 0 0 SNR
00076'000425             JMP READ
00077'020432             LDA 0 TIME
00100'101004             MOV 0 0 SZR         ;TIME = 0
00101'002431             JMP@ ESAVE          ;NO, BAD CONTACT
00102'020426             LDA 0 .TIME
00103'040426             STA 0 TIME

00104'011006    EV0:     ISZ 6 2             ;INCREMENT EV COUNTER
00105'021006             LDA 0 6 2           ;EV COUNTER
00106'025002             LDA 1 2 2           ;START ING EV
00107'122415             SUB# 1 0 SNR        ;EV>= STARTING EV
```

```
00110'011021             ISZ 21 2           ;TURN ON HEIGHT POINTER
00111'025003   EV1:      LDA 1 3 2          ;LAST EV
00112'034003$            LDA 3 .STO1        ;AUTOMATIC STOP ADDRESS
00113'106415             SUB# 0 1 SNR       ;CURRENT EV = LAST EV ?
00114'054416             STA 3 ESAVE        ;SET RETURN TO STOP
00115'020754   EREC:     LDA 0 T7000        ;NO
00116'025010             LDA 1 10 2         ;CLOCK 2
00117'034746             LDA 3 T100
00120'136400             SUB 1 3            ;100 - CLOCK 2
00121'163000             ADD 3 0            ;YES - ADD CLOCK OR MINUTE
00122'004466             JSR XPRE           ;OUTPUT ELUTION VOLUME CODE
00123'060477   READ:     READS 0
00124'101112             MOVL# 0 0 SZC      ;BIT 0 UP
00125'011011             ISZ 11 2           ;INC BELL COUNTER
00126'006404   EOUT:     JSR@ ESAVE
00127'002006$            JMP@ RETN

00130'177324   .TIME:    -300.
00131'000000   TIME:     0
00132'000000   ESAVE:    0


               ;CLOCK SERVICE ROUTINES
               ;CLOCK PATH IS SET BY BEGIN ROUTINE
               ;AND STORED IN G1+17


00133'060114   CLOCK:    NIOS RTC
00134'030002$            LDA 2 .G1
00135'025017             LDA 1 17 2         ;EV ON
00136'034444             LDA 3 .CLOA
00137'125004             MOV 1 1 SZR        ;EV ON?
00140'034443             LDA 3 .CLOB        ;YES
00141'021001             LDA 0 1 2          ;NUM OF SAM
00142'101004             MOV 0 0 SZR        ;SAMPLES?
00143'005400             JSR 0 3            ;YES
00144'002006$            JMP@ RETN          ;NO

               ;ELUTION VOLUME OFF

00145'054433   CLOA:     STA 3 BSAVE        ;SAVE RETURN
00146'004407             JSR CLOB           ;JUMP TO NORMAL CLOCK ROUTINE
00147'015020             DSZ 20 2           ;DECREMENT CLOCK 4
00150'002430             JMP@ BSAVE         ;RETURN IF NOT EQUAL TO ZERO
00151'020716             LDA 0 T600         ;LOAD 600
00152'041020             STA 0 20 2         ;RESTORE MINUTE COUNTER
00153'004717             JSR EV             ;NORMAL EV
00154'002424             JMP@ BSAVE

               ;ELUTION VOLUME ON

00155'054424   CLOB:     STA 3 CSAVE        ;STORE RETURN
00156'020753             LDA 0 TIME
00157'101004             MOV 0 0 SZR
00160'010751             ISZ TIME
00161'000401             JMP .+1
```

```
00162'015007            DSZ 7 2          ;CLOCK 1
00163'002416            JMP@ CSAVE       ;RETURN FOR NON-ZERO
00164'021005            LDA 0 5 2        ;RATE
00165'041007            STA 0 7 2        ;RESTORE RATE COUNTER
00166'015010            DSZ 10 2         ;CLOCK 2 =
00167'002412            JMP @ CSAVE      ;RETURN IF CLOCK 2 IS NOT ZERO
00170'020675            LDA 0 T100       ;=0
00171'041010            STA 0 10 2       ;RESTORE CLOCK 2 TO 100
00172'021021            LDA 0 21 2       ;HEIGHT POINTER 0-NO, 1 YES
00173'101005            MOV 0 0 SNR      ;CHECH HEIGHT POINTER
00174'006405            JSR@ CSAVE       ;RETURN IF ZERO
00175'021025            LDA 0 25 2       ;LOAD CHANNEL NUMBER
00176'061121            DOAS 0 ADCV
00177'002402            JMP@ CSAVE
00200'000000  BSAVE:    0
00201'000000  CSAVE:    0
00202'000145' .CLOA:    CLOA
00203'000155' .CLOB:    CLOB


              ;ATD SERVICE ROUTINE
00204'030002$ ATD:      LDA 2 .G1
00205'062621            DICC 0 ADCV      ;BRING IN DATA AND CLEAR
00206'004402            JSR XPRE         ;STORE RESULT IN DATA BLOCK
00207'002006$           JMP@ RETN


              ;STORE AC0 IN STORAGE BUFFER


00210'060277  XPRE:     INTDS
00211'054415            STA 3 PSAVE
00212'044415            STA 1 PS1
00213'035000            LDA 3 0 2        ;CURRENT ADDRESS
00214'041400            STA 0 0 3        ;STORE IN MEMORY
00215'175400            INC 3 3          ;CURRENT +1
00216'025022            LDA 1 22 2       ;MAX
00217'166033            ADCZ# 3 1 SNC    ;MAX>CURR
00220'035024            LDA 3 24 2       ;NO
00221'055000            STA 3 0 2        ;YES
00222'034404            LDA 3 PSAVE
00223'024404            LDA 1 PS1
00224'060177            INTEN
00225'001400            JMP 0 3          ;RETN
00226'000000  PSAVE:    0
00227'000000  PS1:      0

              .END
```

Program    Teletype Input Interrupt Service Routine

Synopsis    This routine interprets teletype input and transfers control to a command service routines

Method

1. Store first input
2. Decodes input if subsequent input is an escape and transfers control to the appropriate command routine
3. Clear input on carriage return
4. Return to master interrupt routine

Notes -    error 10 for illegal input.

```
;**************************************************

;TELETYPE INPUT INTERRUPT SERVICE ROUTINE

;REVISED SEPT 2, 1971

;**************************************************

;CALL SEQUENCE  -         JSR@ .TELIN
;                         RETURN

                .TITL   TELIN
                .ENT    .TELIN
                .EXTN   BEGIN MCAL MON STOP TYPE1 LOOK ATDX INJ
                .EXTN   DATA CPAR
                .EXTD   CRLF PUTC ERROT

                .ZREL
00000-000000'   .TELIN: TELIN           ;POINTER TO ROUTINE
                .NREL

                ;BRING IN FIRST CHATRACTER AND WAIT FOR ESCAPE

00000'054502    TELIN:  STA 3 SAVE
00001'060610            DIAC 0 TTI      ;BRING IN CHARACTER
00002'024501            LDA 1 MSK
00003'123400            AND 1 0         ;MASK WITH 7 BIT MASK
00004'024474            LDA 1 ESC
00005'106415            SUB# 0 1 SNR    ;CHECH FOR ESCAPE
00006'000413            JMP TEL1        ;YES -
00007'024475            LDA 1 CR
00010'122415            SUB# 1 0 SNR    ;CARRIAGE RETURN ?
00011'000430            JMP TEL2        ;YES
00012'024467            LDA 1 INPUT
00013'125005            MOV 1 1 SNR     ;FIRST INPUT FOLLOWING CR - ?
00014'040465            STA 0 INPUT     ;YES -STORE CHARACTER
00015'006002$           JSR@ PUTC       ;NO - ECHO CHARACTER
00016'063610            SKPDN TTI
00017'000777            JMP .-1         ;WAIT FOR NEXT INPUT
00020'000761            JMP TELIN+1     ;GO TO START

                ;DECODE LETTER

00021'006002$   TEL1:   JSR@ PUTC
00022'006001$           JSR@ CRLF
00023'024456            LDA 1 INPUT     ;LOAD INPUT CHARACTER
00024'102400            SUB 0 0
00025'040454            STA 0 INPUT     ;CLEAR INPUT
00026'034423            LDA 3 .LETTER
00027'031400    TEL3:   LDA 2 0 3       ;LETTER
00030'151005            MOV 2 2 SNR     ;CHECK LETTER
00031'000405            JMP TERR        ;JUMP TO ERROR ROUTINE
00032'146415            SUB# 2 1 SNR    ;CHECK INPUT
00033'000412            JMP TEL4
00034'175400            INC 3 3         ;INCREMENT ROUTINE POINTER
00035'000772            JMP TEL3        ;TRY AGAIN
```

```
                ;LETTER NOT DECODABLE

00036'020447    TERR:   LDA 0 TERR1
00037'006003$           JSR@ ERROT
00040'002442            JMP @SAVE

                ;CLEAR INPUT STORAGE ON RETURN

00041'126420    TEL2:   SUBZ 1 1
00042'044437            STA 1 INPUT     ;CLEAR INPUT
00043'006002$           JSR@ PUTC
00044'002436            JMP@ SAVE

                ;EXIT TO APPROPRIATE ROUTINE

00045'024432    TEL4:   LDA 1 NUML      ;NUM OF LETT +1
00046'137000            ADD 1 3
00047'007400            JSR@ 0 3
00050'002432            JMP@ SAVE

00051'000052'   .LETTER:LETTER
00052'000102    LETTER: 102             ;B
00053'000103            103             ;C
00054'000115            115             ;M
00055'000123            123             ;S
00056'000124            124             ;T
00057'000114            114             ;L
00060'000101            101             ;A
00061'000111            111             ;I
00062'000104            104             ;D
00063'000120            120             ;P
00064'000000            0               ;END OF LETTERS
00065'177777    BEGIN
00066'177777    MCAL
00067'177777    MON
00070'177777    STOP
00071'177777    TYPE1
00072'177777    LOOK
00073'177777    ATDX
00074'177777    INJ
00075'177777    DATA
00076'177777    CPAR

00077'000013    NUML:   11.             ;LETT+1
00100'000033    ESC:    33              ;ASCII CODE
00101'000000    INPUT:  0               ;HOLD FOR FIRST LETTER
00102'000000    SAVE:   0               ;RETURN
00103'000177    MSK:    177
00104'000015    CR:     15
00105'000012    TERR1:  10.             ;ERROR NUMBER FOR UNDECODABLE
                                        ;INPUT

                .END
```

Program    Data and Begin Commands

Synopsis    These routines input and store sample collection and processing parameters for automatic operation (DATA) and manual operation (BEGIN)

Method
1. Input operating parameters
2. Store parameters in operating system storage buffer
3. Return

```
;**********************************************

;DATA AND BEGIN COMMAND ROUTINES

;REVISED SEPT 2, 1971

;**********************************************


;DATA COMMAND
;CALL SEQUENCE  -        JSR DATA
;                        RETURN


;BEGIN COMMAND
;CALL SEQUENCE  -        JSR BEGIN
;                        RETURN


.TITL   DATA
.ENT    DATA BEGIN
.EXTD   .CPAR  AM37 .G1 TYPE DBIN CRLF
.EXTD   AM13 AM14 AM16 AM12 AM15 AM11
.EXTD   AM35 AM36


.NREL

;ENTRY FOR DATA COMMAND

00000'054435  DATA:   STA 3 DR1
00001'030003$         LDA 2 .G1
00002'020015$         LDA 0 AM35
00003'006004$         JSR@ TYPE        ;"AUTO INJ"
00004'006005$         JSR@ DBIN
00005'125004          MOV 1 1 SZR      ;INPUT=0
00006'126520          SUBZL 1 1        ;NO
00007'045026          STA 1 26 2

00010'020016$         LDA 0 AM36
00011'006004$         JSR@ TYPE        ;"AUTO TYPE"
00012'006005$         JSR@ DBIN
00013'125004          MOV 1 1 SZR      ;INPUT=0
00014'126520          SUBZL 1 1        ;NO
00015'045030          STA 1 30 2

00016'020002$         LDA 0 AM37
00017'006004$         JSR@ TYPE        ;"AUTO CALC
00020'006005$         JSR@ DBIN
00021'125004          MOV 1 1 SZR
00022'126520          SUBZL 1 1        ;SET = 1
00023'045034          STA 1 34 2

00024'021026          LDA 0 26 2
00025'101005          MOV 0 0 SNR
00026'000403          JMP DAI
00027'006006$         JSR@ CRLF
00030'004411          JSR BEG1

;INPUT STORAGE PARAMETERS
```

```
00031'021034  DAI:     LDA 0 34 2
00032'101004           MOV 0 0 SZR      ;AC ON
00033'006001$          JSR@ .CPAR       ;YES
00034'002401           JMP@ DR1         ;RETURN
00035'000000  DR1:     0

               ;ENTRY FOR BEGIN COMMAND

00036'030003$ BEGIN:   LDA 2 .G1
00037'102400           SUB 0 0
00040'041026           STA 0 26 2       ;TURN OFF AUTO INJ
00041'054451  BEG1:    STA 3 BSAVE      ;SAVE RETURN
00042'006006$          JSR@ CRLF
00043'030003$          LDA 2 .G1        ;RETRIVE GPC DATA BLOCK IN AC2
00044'020007$          LDA 0 AM13
00045'006004$          JSR@ TYPE        ;"SAMPLE NO. "
00046'006005$          JSR@ DBIN        ;INPUT SAMPLE NO.
00047'021001           LDA 0 1 2        ;NUMBER OF SAMPLES
00050'101004           MOV 0 0 SZR      ;NUMBER OF SAMPLES = ?
00051'000424           JMP B1A
00052'045015           STA 1 15 2       ;STORE SAMPLE NUMBER IN FIRST NO
00053'020010$          LDA 0 AM14
00054'006004$          JSR@ TYPE        ;"EV SENSE "
00055'006005$          JSR@ DBIN        ;INPUT TYPE OF GPC
00056'045017  B3:      STA 1 17 2       ;STORE CLOCK PATH
00057'020011$          LDA 0 AM16
00060'006004$          JSR@ TYPE        ;"RATE
00061'006005$          JSR@ DBIN        ;GET RATE CODE
00062'102520           SUBZL 0 0        ;AC0=1
00063'106433           SUBZ# 0 1 SNC    ;RATE>=1 ?
00064'105000           MOV 0 1          ;NO SET = 1
00065'045005           STA 1 5 2        ;YES STORE
00066'020012$          LDA 0 AM12
00067'006004$          JSR@ TYPE        ;"ON "
00070'020013$          LDA 0 AM15
00071'006004$          JSR@ TYPE        ;"AT EV "
00072'006005$          JSR @DBIN        ;INPUT STARTING EV
00073'045002           STA 1 2 2        ;STORE START
00074'000402           JMP B1
00075'045016  B1A:     STA 1 16 2       ;SECOND SAM
00076'020014$ B1:      LDA 0 AM11
00077'006004$          JSR@ TYPE        ;"OFF "
00100'020013$          LDA 0 AM15
00101'006004$          JSR@ TYPE        ;"AT EV "
00102'006005$          JSR@ DBIN        ;INPUT END EV
00103'021001           LDA 0 1 2        ;AC0 NUMBER OF SAMPLES
00104'101005           MOV 0 0 SNR      ;FISRT OR SECOND SAMPLE
00105'045003           STA 1 3 2        ;STORE RESULT
00106'045012           STA 1 12 2       ;ALSO IN SECOND
00107'006006$          JSR@ CRLF        ;OUTPUT CARRIAGE RETURN
00110'006006$          JSR@ CRLF
00111'002401           JMP@ BSAVE

00112'000000  BSAVE:   0

               .END
```

<u>Program</u> Monitor Command

<u>Synopsis</u> This routine is used to change the status of the monitor

<u>Method</u>

1. Request new monitor status
2. Store new status in storage buffer
3. Return

```
;**************************************************

;MONITOR COMMAND

;REVISED SEPT 1, 1971

;****************************************************

;CALL SEQUENCE  -          JSR MON
;                          RETURN


        .TITL   MONIT
        .ENT    MON
        .EXTD   CRLF .G1 TYPE DBIN SPACE  AM23

        .NREL

00000'054414  MON:    STA 3 SAVE      ;RETURN
00001'030002$         LDA 2 .G1       ;RETRIVE GPC DATA POINTER IN AC2
00002'020006$         LDA 0 AM23
00003'006003$         JSR@ TYPE       ;"MONINTOR "
00004'024411          LDA 1 T4
00005'006005$         JSR@ SPACE
00006'006004$         JSR@ DBIN       ;INPUT NEW STATUS
00007'045013          STA 1 13 2      ;STORE NEW STATUS
00010'021000          LDA 0 0 2       ;CURRENT LOCATION
00011'125004          MOV 1 1 SZR     ;CHECK NEW STATUS
00012'041014          STA 0 14 2      ;SET MONITOR PRINT OUT TO CURREN
00013'002401          JMP@ SAVE

00014'000000  SAVE:   0
00015'000004  T4:     4
        .END
```

<u>Program</u> Type Command

<u>Synopsis</u> This routine outputs data from memory in an automatic mode (auto-type) or manually as a result of the type command.

<u>Methods</u>

<u>Auto-Type</u>

1. Search for data
2. If data is not found type "DATA NOT FOUND" otherwise output data in table form
3. Return

<u>Command Type</u>

1. Input sample number
2. Search for data
3. If data is found, output it in table form
4. If data is not found, type "DATA NOT FOUND" "STORAGE DUMP". If input is non-zero, output entire storage area
5. Return

```
;********************************************

;TYPE COMMAND ROUTINE

;REVISED SEPT 1, 1971

;************************************************


;TYPE COMMAND
;CALL SEQUENCE   -        JSR TYPE1
;                         RETURN


;AUTO TYPE
;CALL SEQUENCE   -        JSR@ .ATYPE
;                         SAMPLE NUMBER
;                         RETURN
;INPUT    -       AC0   MINIMIUM MEMORY ADDRESS
;         -       AC1   MAXIMIUM MEMORY ADDRESS


.TITL    OUTPUT
.ENT     TYPE1 .ATYPE
.EXTD    BIND CRLF TYPE AM21 AM13  GETC DBIN AM25 AM33
.EXTD    .G1 ERROT  FEED TABLE

.ZREL

00000-000000' .ATYPE: ATYPE          ;POINTER TO ROUTINE


.NREL

;AUTO TYPE ENTRY

00000'040543  ATYPE:  STA 0 MIN
00001'044543          STA 1 MAX
00002'111000          MOV 0 2        ;AC2= START
00003'021400          LDA 0 0 3
00004'040541          STA 0 HOLD
00005'175400          INC 3 3
00006'054531          STA 3 TSAVE    ;RETURN
00007'006014$         JSR@ FEED
00010'006002$         JSR@ CRLF
00011'020005$         LDA 0 AM13
00012'006003$         JSR@ TYPE
00013'024532          LDA 1 HOLD
00014'006001$         JSR@ BIND
00015'006002$         JSR@ CRLF
00016'006002$         JSR@ CRLF
00017'176520          SUBZL 3 3      ;AC3=1
00020'054522          STA 3 AT
00021'000424          JMP TT1


;ENTRY FOR MANUAL TYPE

00022'054515  TYPE1:  STA 3 TSAVE    ;RETURN
```

```
00023'020004$          LDA 0 AM21
00024'006003$          JSR@ TYPE           ;"TURN ON TAPE  "
00025'063610           SKPDN TTI
00026'000777           JMP .-1             ;WAIT FOR KEY TO BE HIT
00027'060210           NIOC TTI
00030'006002$          JSR@ CRLF
00031'006014$          JSR@ FEED
00032'020005$          LDA 0 AM13
00033'006003$          JSR@ TYPE           ;SAMPLE NO. "
00034'006007$          JSR@ DBIN           ;INPUT SAMPLE NO.
00035'044510           STA 1 HOLD          ;STORE SAMPLE NUMBER
00036'102400           SUB 0 0
00037'040503           STA 0 AT            ;SET UP AT
00040'030012$          LDA 2 .G1
00041'021022           LDA 0 22 2          ;MAX
00042'040502           STA 0 MAX
00043'031024           LDA 2 24 2          ;MIN
00044'050477           STA 2 MIN


                 ;SEARCH FOR DATA

00045'020477   TT1:    LDA 0 MAX
00046'176400           SUB 3 3
00047'054472           STA 3 COUNT
00050'054432           STA 3 TH2
00051'054430           STA 3 TH1           ;CLEAR TH1
00052'024473           LDA 1 HOLD          ;SAMPLE NO.

                 ;SEARCH FOR SAMPLE NUMBER

00053'035000   TT2:    LDA 3 0 2           ;DATA VALUE
00054'136415           SUB# 1 3 SNR        ;IS IT SAMPLE NO.
00055'000405           JMP F1              ;YES
00056'151400           INC 2 2             ;NO, INCREMENT LOCATION
00057'142433           SUBZ# 2 0 SNC       ;MAX>= CURRENT
00060'004426           JSR ER01
00061'000772           JMP TT2             ;YES TRY AGAIN

                 ;FOUND SAMPLE NUMBER

00062'050417   F1:     STA 2 TH1           ;STORE SAMPLE NO. ADDRESS
00063'034463           LDA 3 T1000
00064'167000           ADD 3 1             ;SAMPLE NO. + 1000
00065'044460           STA 1 HOLD

                 ;SEARCH FOR STOP

00066'035000   TT3:    LDA 3 0 2
00067'010413           ISZ TH2             ;INCREMENT POINTER
00070'136415           SUB# 1 3 SNR        ;IS IT STOP
00071'000405           JMP F2              ;YES
00072'151400           INC 2 2             ;NO INCREMENT LOCATION
00073'142433           SUBZ# 2 0 SNC       ;MAX>=CURR
00074'004412           JSR ER01            ;NO
00075'000771           JMP TT3             ;YES TRY AGAIN
```

```
                   ;DATA FOUND, OUTPUT IT

00076'020445  F2:     LDA 0 MIN        ;LOAD MINIMIUM
00077'024445          LDA 1 MAX        ;LOAD MAXIMIUM
00100'006015$ TOUT:   JSR@ TABLE       ;OUTPUT NUMBERS
00101'000000  TH1:    0
00102'000000  TH2:    0


                   ;RETURN

00103'006002$         JSR@ CRLF
00104'006014$         JSR@ FEED        ;FEEDER TAPE
00105'002432          JMP@ TSAVE

                   ;END OF MEMORY CHECK

00106'020433  ERO1:   LDA 0 COUNT      ;RECYCLE
00107'101004          MOV 0 0 SZR      ;FIRST TIME
00110'000405          JMP ERO3         ;NO
00111'010430          ISZ COUNT        ;YES
00112'030431          LDA 2 MIN        ;LOWEST LOC
00113'020431          LDA 0 MAX        ;LARGEST
00114'001400          JMP 0 3          ;RETN


                   ;DATA NOT FOUND

00115'020010$ ERO3:   LDA 0 AM25
00116'006003$         JSR@ TYPE        ;"DATA NOT FOUND "
00117'006002$         JSR@ CRLF
00120'020422          LDA 0 AT
00121'101004          MOV 0 0 SZR      ;AUTO TYPE
00122'002415          JMP @TSAVE       ;YES
00123'020011$         LDA 0 AM33
00124'006003$         JSR@ TYPE        ;"STORAGE DUMP "
00125'006007$         JSR@ DBIN
00126'125005          MOV 1 1 SNR
00127'002410          JMP@ TSAVE       ;NO
00130'020413          LDA 0 MIN
00131'040750          STA 0 TH1        ;STORE STARTING ADDRESS
00132'024412          LDA 1 MAX        ;MAXIMIUM
00133'106400          SUB 0 1          ;NUMBER OF POINTS
00134'044746          STA 1 TH2
00135'024407          LDA 1 MAX        ;MAXIMIUM
00136'000742          JMP TOUT         ;OUTPUT BUFFER AND RETURN

00137'000000  TSAVE:  0
00140'000033  ESC:    33
00141'000000  COUNT:  0
00142'000000  AT:     0
00143'000000  MIN:    0
00144'000000  MAX:    0
00145'000000  HOLD:   0
00146'001750  T1000:  1000.

              .END
```

<u>Program</u> LOOK and ATD Command Routines

<u>Synopsis</u> Routines to output current operating information and to test the A/D convertor

<u>Method</u>

<u>ATD Command</u>

1. The analog signal is read and outputted each time a key is struck
2. Inputting an escape returns control to the operating system

<u>LOOK Command</u>

1. Outputs current information stored in the operating system storage buffer
2. Return

```
;*********************************************

;LOOK AND ATD COMMAND ROUTINES

;REVISED SEPT 2, 1971

;*********************************************


;LOOK COMMAND
;CALL SEQUENCE  -          JSR LOOK
;                          RETURN


;A/D TEST ROUTINE
;CALL SEQUENCE  -          JSR ATDX
;                          RETURN


.TITL   LKATD
.ENT    LOOK ATDX
.EXTD   SPACE CRLF TYPE .G1 BIND
.EXTD   AM10 AM11 AM12   AM15 AM16
.EXTD   AM21  AM23   AM26 AM34
.EXTD   RETN

.NREL

;ENTRY FOR LOOK COMMAND

00000'006002$ LOOK:   JSR@ CRLF
00001'030004$         LDA 2 .G1         ;RETRIVE DATA STORAGE LOCATION
00002'020006$         LDA 0 AM10
00003'006003$         JSR@ TYPE         ;"GPC "
00004'024472          LDA 1 T1
00005'006001$         JSR@ SPACE        ;OUTPUT TWO SPACES
00006'020010$         LDA 0 AM12
00007'025001          LDA 1 1 2         ;SAMPLE NUMBER
00010'125005          MOV 1 1 SNR       ;IS THERE A SAMPLE
00011'020007$         LDA 0 AM11        ;NO LOAD OFF
00012'006003$         JSR@ TYPE         ;"ON " OR "OFF "
00013'024462          LDA 1 T3
00014'006001$         JSR@ SPACE
00015'020011$         LDA 0 AM15
00016'006003$         JSR@ TYPE         ;"AT EV "
00017'025006          LDA 1 6 2         ;EV NUMBER
00020'006005$         JSR@ BIND
00021'024454          LDA 1 T3
00022'006001$         JSR@ SPACE        ;OUTPUT FIVE SPACES
00023'020014$         LDA 0 AM23
00024'006003$         JSR@ TYPE         ;"MONITOR "
00025'025013          LDA 1 13 2        ;MONITORE STATUS
00026'020010$         LDA 0 AM12
00027'125005          MOV 1 1 SNR
00030'020007$         LDA 0 AM11
00031'006003$         JSR@ TYPE         ;"ON " OR "OFF "
00032'024443          LDA 1 T3
00033'006001$         JSR@ SPACE
00034'021001          LDA 0 1 2         ;NUMBER OF SAMPLES
00035'101005          MOV 0 0 SNR       ;SAMPLE INJECTED
```

```
00036'000413            JMP LO1             ;NO
00037'020012$           LDA 0 AM16
00040'006003$           JSR@ TYPE           ;"RATE CODE "
00041'025005            LDA 1 5 2           ;RATE
00042'006005$           JSR@ BIND
00043'024432            LDA 1 T3
00044'006001$           JSR @SPACE
00045'020016$           LDA 0 AM34
00046'006003$           JSR@ TYPE           ;"NS "
00047'025001            LDA 1 1 2           ;NUMBER OF SAMPLES
00050'006005$           JSR@ BIND           ;OUTPUT NUMBER
00051'006002$ LO1:      JSR@ CRLF
00052'002017$           JMP @RETN

              ;ENTRY FOR A/D TEST ROUTINE

00053'030004$ ATDX:     LDA 2 .G1           ;WHICH GPC
00054'006002$           JSR@ CRLF
00055'063610            SKPDN TTI
00056'000777            JMP .-1
00057'060610            DIAC 0 TTI          ;CLEAR TTI
00060'024414            LDA 1 ESC
00061'106415            SUB# 0 1 SNR        ;ESC?
00062'002017$           JMP@ RETN           ;YES
00063'021025            LDA 0 25 2          ;LOAD CHANNEL NUMBER
00064'060277            INTDS
00065'061121            DOAS 0 ADCV
00066'063621            SKPDN ADCV
00067'000777            JMP .-1
00070'066621            DICC 1 ADCV         ;BRING IN CONVERTED NUMBER
00071'060177            INTEN
00072'006005$           JSR@ BIND
00073'000761            JMP ATDX+1

00074'000033  ESC:      33
00075'000003  T3:       3
00076'000001  T1:       1

              .END
```

**Program** Manual and Auto-Stop Routines

**Synopsis** This program includes the command STOP routine and auto-stop.

**Method**

Auto-Stop or STOP Command

1. Decrement sample counter
2. Enter stop data pointer in memory
3. Initialize stop message to be outputted by auto-action
4. Return

**Notes** - Errors 11 for stop not preceeded by an injection.

```
;***********************************************

;MANUAL AND AUTO STOP ROUTINES

;REVISED SEPT 2, 1971

;************************************************

;MANUAL STOP
;CALL SEQUENCE  -        JSR STOP
;                        RETURN


;AUTO STOP
;CALL SEQUENCE  -        JSR@ .STO1
;                        RETURN


           .TITL   STOP
           .ENT    STOP .STO1
           .EXTD   NUM ADDRS PRESENT ERROT AM26 RETN CRLF
           .EXTD   .G1

           .ZREL
00000-000003' .STO1:  STO1            ;POINTER TO AUTO STOP

           .NREL

;MANUAL STOP ENTRY

00000'054476  STOP:   STA 3 SAVE      ;RETURN
00001'006007$         JSR@ CRLF
00002'000402          JMP S1A


;ENTRY FOR AUTO STOP

00003'054473  STO1:   STA 3 SAVE
00004'030010$ S1A:    LDA 2 .G1
00005'015001          DSZ 1 2         ;DECREMENT SAMPLE COUNTER=
00006'000414          JMP S2          ;NON-ZERO MEANS A SAMPLE REMAINS

;ONLY ONE SAMPLE INJECTED

00007'025015  S1:     LDA 1 15 2      ;GET SAMPLE NUMBER
00010'176400          SUB 3 3
00011'055021          STA 3 21 2      ;TURN OFF HEIGHT POINTER
00012'021026          LDA 0 26 2
00013'101005          MOV 0 0 SNR
00014'000424          JMP S3
00015'135400          INC 1 3
00016'055015          STA 3 15 2      ;INC SAM NO AND STORE
00017'021012          LDA 0 12 2
00020'041003          STA 0 3 2       ;SET UP NEXT STOP FOR AUT
00021'000417          JMP S3

;ONE SAMPLE REMAINS

00022'021004  S2:     LDA 0 4 2       ;READING AT INJ
00023'025006          LDA 1 6 2       ;CURRENT
```

```
00024'106400          SUB 0 1              ;CURRENT-INJ
00025'045006          STA 1 6 2            ;SAVE AS CURRENT
00026'176400          SUB 3 3
00027'021002          LDA 0 2 2            ;HEIGHT STORE
00030'106433          SUBZ# 0 1 SNC        ;CURRENT>=START OF STORE
00031'055021          STA 3 21 2           ;NO
00032'025001   S2A:   LDA 1 1 2
00033'125112          MOVL# 1 1 SZC        ;IS SAMPLE COUNTER NEGATIVE
00034'000434          JMP SERR             ;YES
00035'025015          LDA 1 15 2           ;GET SAMPLE NI.
00036'021016          LDA 0 16 2           ;GET SECOND SAMPLE NUMBER
00037'041015          STA 0 15 2           ;STORE SECOND NUMBER IN FIRST NB

               ;NOTE STOP

00040'020005$  S3:    LDA 0 AM26
00041'042002$         STA@ 0 ADDRS         ;STORE MESSAGE BYTE
00042'010002$         ISZ ADDRS            ;INCREMENT POINTER
00043'046002$         STA@ 1 ADDRS         ;STORE NUMBER
00044'010002$         ISZ ADDRS            ;INCREMENT POINTER
00045'010001$         ISZ NUM              ;INCREMENT NUMBER
00046'021030          LDA 0 30 2           ;AUTO TYPE ON
00047'101005          MOV 0 0 SNR
00050'000405          JMP S4               ;OFF
00051'035027          LDA 3 27 2           ;ADD OF STORAGE
00052'117000          ADD 0 3
00053'045400          STA 1 0 3            ;STORE SAMPLE NO.
00054'011030          ISZ 30 2
00055'021034   S4:    LDA 0 34 2           ;AUTO CALC
00056'101005          MOV 0 0 SNR          ;ON OR OFF
00057'000405          JMP S5               ;OFF
00060'035033          LDA 3 33 2           ;ADD OF STORE
00061'117000          ADD 0 3
00062'045400          STA 1 0 3            ;STORE SAMPLE NO
00063'011034          ISZ 34 2

               ;STORE STOP IN MEMORY
00064'020413   S5:    LDA 0 T1000
00065'123000          ADD 1 0              ;1000+SAMPLE NUMBER
00066'006003$         JSR@ PRESENT
00067'002407          JMP@ SAVE


               ;ERROR FOR STOP THAT WAS NOT PRECEEDED
               ;BY AN INJECTION

00070'011001   SERR:  ISZ 1 2              ;INCREMENT SAMPLE COUNTER TO ZEB
00071'000777          JMP .-1
00072'020403          LDA 0 SER1           ;ERROR NUMBER
00073'006004$         JSR@ ERROT           ;JUMP TO ERROR ROUTINE
00074'002402          JMP@ SAVE            ;RETN

00075'000013   SER1:  11.                  ;ERROR NUMBER
00076'000000   SAVE:  0
00077'001750   T1000: 1000.

               .END
```

Program — Input-Output Routines

Synopsis — This program contains I/O service routines to:

GETC - input and echo one character

PUTC - output one character

SPACE - output n spaces

TYPE - type a alphameric message

TABLE - output a table of number

DBIN - input a decimal number

BIND - output a decimal number

ERROT - output error number

FEED - output n null characters

Note - Control A input interrupts input requist and returns control to master interrupt routine

- Rubout clears input (for DBIN only)

- All output routines mask teletype input to avoid reentrant problems

```
;*******************************************

;INPUT-OUTPUT ROUTINES
;GETC, PUTC, SPACE, CRLF, TYPE, TABLE,
;DBIN, BIND, ERROT, FEED

;REVISED SEPT 2, 1971

;*******************************************

;SPECIAL ACTION INPUT
;CONTROL A-      INTERRUPTS INPUT TO JMP@ RETN

;RUBOUT       -    FOR DBIN ONLY, CLEARS INPUT


                .TITL   IOSER
                .ENT    SPACE CRLF TYPE DBIN BIND TABLE
                .ENT    FEED ERROT  GETC PUTC FELED NOPL SPPL
                .EXTD   RETN CMASK

       000040   .LOC 40
00040 000000'   XGETC
00041 000014'   XPUTC

                .ZREL
00000-000000'   GETC:   XGETC
00001-000014'   PUTC:   XPUTC
00002-000026'   SPACE:  XSPACE
00003-000013'   CRLF:   XCRLF
00004-000064'   TYPE:   XTYPE
00005-000117'   TABLE:  XTABLE
00006-000206'   DBIN:   XDBIN
00007-000273'   BIND:   XBIND
00010-000053'   ERROT:  XERROT
00011-000353'   FEED:   XFEED
00012-000003    TMSK:   3               ;MASK ON OUTPUT

                .NREL

                ;ROUTINE TO INPUT CHARACTER INTO AC0
                ;WITH ECHO

                ;CALL SEQUENCE  -       JSR@ 40  (OR GETC)
                ;                       RETURN

00000'060110    XGETC:  NIOS TTI        ;START TELETYPE
00001'063610            SKPDN TTI       ;WAIT FOR INPUT
00002'000777            JMP .-1
00003'060610            DIAC 0 TTI      ;GET CHARACTER
00004'024571            LDA 1 MSK
00005'123400            AND 1 0         ;7 BIT MASK
00006'024565            LDA 1 CONA
00007'106414            SUB# 0 1 SZR    ;CONRTOL A ?
00010'000404            JMP XPUTC
00011'004402            JSR XCRLF
00012'002001$           JMP@ RETN       ;RETURN TO OP SYS


                ;ROUTINE TO OUTPUT CARRIAGE RETURN AND LINE FEED
```

```
                 ;CALL SEQUENCE   -        JSR@ CRLF
                 ;                         RETURN

00013'020556     XCRLF:  LDA 0 CR

                 ;ROUTINE TO OUTPUT CHARACTER IN AC0
                 ;CARIAGE RETURN OUTPUTS LINE FEED

                 ;CALL SEQUENCE   -        JSR@ 41  (OR PUTC)
                 ;                         RETURN

00014'101005     XPUTC:  MOV 0 0 SNR
00015'001400             JMP 0 3           ;RETN IF NULL
00016'063511             SKPBZ TTO         ;WAIT UNTIL DONE
00017'000777             JMP .-1
00020'061111             DOAS 0 TTO        ;OUTPUT CHARACTER
00021'024550             LDA 1 CR
00022'106414             SUB# 0 1 SZR      ;CARRIAGE RETURN ?
00023'001400             JMP 0 3           ;NO - RETURN
00024'020544             LDA 0 LF
00025'000767             JMP XPUTC         ;OUTPUT LINE FEED


                 ;SPACE ROUTINE
                 ;CALL SEQUENCE   -        JSR@ SPACE
                 ;                         RETURN
                 ;INPUT-AC1 NUMBER OF SPACES

00026'125005     XSPACE: MOV 1 1 SNR       ;CHECK FOR ZERO INPUT
00027'001400             JMP 0 3           ;RETURN
00030'020002$            LDA 0 CMASK
00031'040421             STA 0 S3          ;STORE CURRENT MASK
00032'020012-            LDA 0 TMSK        ;TTI MASK
00033'040002$            STA 0 CMASK
00034'062077             DOB 0 CPU
00035'054413             STA 3 S1          ;SAVE RETURN
00036'044413             STA 1 S2          ;NO. OF OUTPUTS
00037'020533             LDA 0 SP
00040'004754             JSR XPUTC
00041'014410             DSZ S2
00042'000776             JMP .-2           ;REPEAT
00043'020407             LDA 0 S3
00044'040002$            STA 0 CMASK
00045'034403             LDA 3 S1
00046'062077             DOB 0 CPU
00047'001400             JMP 0 3           ;RETURN
00050'000000     S1:     0
00051'000000     S2:     0
00052'000000     S3:     0


                 ;ROUTINE TO OUTPUT ERROR MESSAGE

                 ;CALL SEQUENCE   -        JSR@ ERROT
                 ;                         RETURN

00053'054410     XERROT: STA 3 ERROR
00054'111000             MOV 0 2
00055'020523             LDA 0 AMER        ;LOAD ERROR MESSAGE BYTE POINTER
```

```
00056'004406          JSR XTYPE
00057'145000          MOV 2 1
00060'006007-         JSR @BIND        ;OUTPUT ERROR NUMBER
00061'004732          JSR XCRLF        ;OUTPUT CARRIAGE RETURN AND LF
00062'002401          JMP@ ERROR       ;RETURN THROUGH ERROR
00063'000000  ERROR:  0

              ;ROUTINE TO TYPE A MESSAGE

              ;CALL SEQUENCE  -       JSR@ TYPE
              ;                       RETURN
              ;BYTE POINTER IS PASSED IN ACØ


00064'024002$ XTYPE:  LDA 1 CMASK
00065'044431          STA 1 TY3         ;SAVE MASK
00066'024012-         LDA 1 TMSK
00067'044002$         STA 1 CMASK      ;NEW MASK
00070'066077          DOB 1 CPU        ;ISSUE NEW MASK
00071'054423          STA 3 TY1        ;STORE RETURN
00072'050423          STA 2 TY2        ;SAVE TY2

00073'024501  XTY1:   LDA 1 T377       ;LOAD MASK
00074'111220          MOVZR Ø 2
00075'031000          LDA 2 Ø 2
00076'101402          INC Ø Ø SZC
00077'151300          MOVS 2 2
00100'133404          AND 1 2 SZR
00101'000407          JMP XTY2
00102'020414          LDA Ø TY3        ;GET OLD MASK
00103'040002$         STA Ø CMASK      ;RESTORE MASK
00104'034410          LDA 3 TY1
00105'030410          LDA 2 TY2
00106'062077          DOB Ø CPU        ;ISSUE MASK
00107'001400          JMP Ø 3
00110'063511  XTY2:   SKPBZ TTO
00111'000777          JMP .-1          ;WAIT
00112'071111          DOAS 2 TTO
00113'000760          JMP XTY1

00114'000000  TY1:    Ø
00115'000000  TY2:    Ø
00116'000000  TY3:    Ø


              ;ROUTINE TO OUTPUT A SERIES OF NUMBERS IN TABLE FORM
              ;NOPL NUMBERS PER LINE, SPPL SPACES
              ;FROM A STORAGE LOCATION WITH LOWER AND UPPER BOUNDS
              ;WITH RECYCLE TO LOWEST LOCATION IF NECESSARY

              ;CALL SEQUENCE  -       JSR@ TABLE
              ;                       STARTING ADDRESS OF DATA
              ;                       NUMBER OF NUMBERS TO BE OUTPUTTE
              ;                       RETURN
              ;INPUT   -      ACØ -MINIMIUM ALLOWABLE ADDRESS
              ;               AC1 -MAXIMIUM ALLOWABLE ADDRESS

00117'040443  XTABLE: STA Ø TA1
00120'021401          LDA Ø 1 3        ;NO OF OUTPUTS
```

```
00121'101005            MOV 0 0 SNR
00122'001400            JMP 0 3          ;RETURN FOR ZERO OUTPUTS
00123'040443            STA 0 TA5        ;NO OF OUTPUTS
00124'044437            STA 1 TA2
00125'054437            STA 3 TA3        ;RETURN
00126'020433            LDA 0 NOPL
00127'040436            STA 0 TA4        ;OUTPUT PER LINE
00130'050437            STA 2 TA6
00131'031400            LDA 2 0 3        ;STARTING ADDRESS
00132'025000    TAB1:   LDA 1 0 2        ;PICK UP NUMBER
00133'004540            JSR XBIND
00134'014431            DSZ TA4          ;LINE CHECK
00135'000407            JMP TAB2
00136'006003-           JSR@ CRLF
00137'020422            LDA 0 NOPL
00140'040425            STA 0 TA4
00141'102400            SUB 0 0
00142'006041            JSR@ 41          ;OUTPUT NULL
00143'000403            JMP TAB2+2
00144'024414    TAB2:   LDA 1 SPPL
00145'004661            JSR XSPACE       ;SPACES
00146'151400            INC 2 2          ;INCREMENT POINTER
00147'024414            LDA 1 TA2        ;MAX
00150'146433            SUBZ# 2 1 SNC    ;MAX>=CURR
00151'030411            LDA 2 TA1        ;NO
00152'014414            DSZ TA5          ;OUTPUT-1
00153'000757            JMP TAB1
00154'006003-           JSR@ CRLF
00155'030412            LDA 2 TA6
00156'034406            LDA 3 TA3
00157'001402            JMP 2 3          ;RETURN
00160'000004    SPPL:   4
00161'000010    NOPL:   10
00162'000000    TA1:    0
00163'000000    TA2:    0
00164'000000    TA3:    0
00165'000000    TA4:    0
00166'000000    TA5:    0
00167'000000    TA6:    0
00170'000012    LF:     12
00171'000015    CR:     15
00172'000040    SP:     40
00173'000001    CONA:   1
00174'000377    T377:   377
00175'000177    MSK:    177
00176'000004    T4:     4
00177'000010    T10:    10
00200'000402"   AMER:   MER*2
                MER:    .TXT " ERROR  "
00201'042440
00202'051122
00203'051117
00204'020040
00205'000000

                ;DECIMAL TO BINARY INPUT

                ;CALL SEQUENCE  -         JSR@ DBIN
                ;                         RETURN
```

```
                       ;NUMBER RETURNS IN AC1

00206'054455  XDBIN:   STA 3,.EC03       ; SAVE AC3
00207'050453           STA 2,.EC02       ; SAVE AC2
00210'102400  .XD1:    SUB 0,0
00211'040453           STA 0,.EC10       ; CLEAR SIGN WORD
00212'040453           STA 0,.EC11       ; CLEAR SUM WORD
00213'006040           JSR @.EC40        ; GET A CHARACTER
00214'024452           LDA 1,.EC20       ; TEST FOR "+"
00215'106405           SUB 0,1,SNR
00216'000405           JMP .EC97         ; YES
00217'024450           LDA 1,.EC21       ; NO, TEST FOR "-"
00220'106404           SUB 0,1,SZR
00221'000403           JMP .EC96         ; NO EXPLICIT SIGN
00222'010442           ISZ .EC10         ; SET FLAG WORD FOR NEGATIVE
                                         ; NUMBER
00223'006040  .EC97:   JSR @.EC40        ; GET ANOTHER CHARACTER
00224'024751  .EC96:   LDA 1 MSK
00225'122415           SUB# 1 0 SNR
00226'000430           JMP .XD2          ;REPEAT IF RUBOUT
00227'024441           LDA 1,.EC22       ; ASCII "0"
00230'030441           LDA 2,.EC23       ; ASCII "9"
00231'142033           ADCZ# 2,0,SNC     ; SKIP IF > 9
00232'106032           ADCZ# 0,1,SZC     ; SKIP IF >= 0
00233'000406           JMP .EC95         ; NOT A DIGIT, THERFORE A BREAK
                                         ; CHARACTER
00234'122400           SUB 1,0           ; REDUCE DIGIT TO 0-9 BINARY
                                         ; RANGE
00235'024430           LDA 1,.EC11       ; SUM WORD
00236'004412           JSR .EC50         ; MULTIPLY BY 10 AND ADD
00237'044426           STA 1,.EC11       ; SAVE SUM
00240'000763           JMP .EC97         ; GET NEXT CHARACTER



00241'024424  .EC95:   LDA 1,.EC11       ; RESULT TO AC1
00242'125120           MOVZL 1,1
00243'014421           DSZ .EC10         ; TEST SIGN
00244'125221           MOVZR 1,1,SKP     ; POSITIVE
00245'124640           NEGOR 1,1         ; NEGATIVE
00246'030414           LDA 2,.EC02       ; RESTORE AC2
00247'002414           JMP @.EC03


                       ; ROUTINE TO MULTIPLY AC1 BY 10 AND ADD AC0


00250'131120  .EC50:   MOVZL 1,2         ; N*2
00251'151120           MOVZL 2,2         ; N*4
00252'147000           ADD 2,1           ; N*5
00253'125120           MOVZL 1,1         ; N*5*2 = N*10
00254'107000           ADD 0,1           ; ADD AC0
00255'001400           JMP 0,3           ; SUCCESS RETURN


00256'020403  .XD2:    LDA 0 T100
00257'006001-          JSR@ PUTC
00260'000730           JMP .XD1
00261'000100  T100:    100
```

```
00262'000000  .EC02:  0                    ; SAVE AC2
00263'000000  .EC03:  0                    ; SAVE AC3

00264'000000  .EC10:  0                    ; FLAG WORD FOR SIGN OF RESULT
00265'000000  .EC11:  0                    ; RUNNING SUM WORD

00266'000053  .EC20:  "+                   ; ASCII "+"
00267'000055  .EC21:  "-                   ; ASCII "-"
00270'000060  .EC22:  "0                   ; ASCII 0"
00271'000071  .EC23:  "9                   ; ASCII "9"
00272'000123  .EC24:  "S                   ; ASCII "S" FOR INDICATION
                                           ; ENTRY

      000040  .EC40=40                     ; ADDRESS OF GET CHARACTER
                                           ; ROUTINE
      000041  .EC41=41                     ; ADDRESS OF PUT CHARACTER
                                           ; ROUTINE


              ;BINARY TO DECIMAL OUTPUT

              ;CALL SEQUENCE  -       JSR@ BIND
              ;                       RETURN
              ;INPUT  -       NUMBER IN AC1

00273'020002$ XBIND:  LDA 0 CMASK
00274'040441          STA 0 .ED00
00275'020012-         LDA 0 TMSK
00276'040002$         STA 0 CMASK
00277'062077          DOB 0 CPU
00300'054437          STA 3,.ED03          ; SAVE RETURN
00301'050435          STA 2,.ED02          ; SAVE AC2
00302'034450          LDA 3,.ED30          ; ADDRESS OF POWER OF TEN TABLE
00303'054443          STA 3,.ED10          ; INITIALIZE POINTER
00304'020444          LDA 0,.ED20          ; ASSUME NEGATIVE
00305'044442          STA 1 .ED11
00306'125112          MOVL# 1,1,SZC
00307'124401          NEG 1,1,SKP
00310'000403          JMP .+3              ; NO, IT IS POSITIVE; GET PLUS
00311'044436  .ED97:  STA 1,.ED11          ; SAVE N
00312'006041          JSR @.ED40           ; PUT OUT SIGN OR DIGIT
00313'024434          LDA 1,.ED11          ; GET CURRENT VALUE OF N
00314'036432          LDA 3,@.ED10         ; GET CURRENT  POWER OF TEN
00315'010431          ISZ .ED10            ; BUMP POINTER
00316'161005          MOV 3,0,SNR
00317'000407          JMP .ED98            ; PUT OUT NULL
00320'020431          LDA 0,.ED22          ; GET ASCII "0"
00321'166422  .ED99:  SUBZ 3,1,SZC         ; DOES POWER OF TEN GO IN?
00322'101401          INC 0,0,SKP          ; YES, BUMP RESULT DIGIT
00323'167001          ADD 3,1,SKP          ; NO, RESTORE PREVIOUS VALUE
00324'000775          JMP .ED99            ; CONTINUE SUBTRACTING
00325'000764          JMP .ED97            ; PUT OUT DIGIT

00326'006041  .ED98:  JSR @.ED40           ; PUT OUT NULL WORD
00327'030407          LDA 2,.ED02          ; RESTORE AC2
00330'020405          LDA 0 .ED00          ;GET OLD MASK
00331'040002$         STA 0 CMASK
00332'034405          LDA 3 .ED03
00333'062077          DOB 0 CPU
```

```
00334'001400            JMP 0 3          ; RETURN


00335'000000   .ED00:   0                ;
00336'000000   .ED02:   0                ; SAVE AC2
00337'000000   .ED03:   0                ; SAVE AC3

      000012            .RDX 10
00340'023420   .ED05:   10000            ; POWER OF TEN TABLE
00341'001750            1000             ; 10**3
00342'000144            100              ; 10**2
00343'000012            10               ; 10**1
00344'000001            1                ; 10**0
00345'000000            0                ; END OF TABLE INDICATION
      000010            .RDX 8

00346'000000   .ED10:   0                ; ADDRESS OF CURRENT POWER OF
                                         ; TEN ENTRY
00347'000000   .ED11:   0                ; RUNNING SUM WORD

00350'000055   .ED20:   "-               ; ASCII "-"
00351'000060   .ED22:   60               ; ASCII "0"

00352'000340'  .ED30:   .ED05            ; ADDRESS OF POWER OF TEN TABLE

      000041   .ED40=41                  ; PAGE ZERO PUT CHARACTER

               ;ROUTINE TO OUTPUT BLANK TAPE

               ;CALL SEQUENCE  -         JSR@ FEED
               ;                         RETURN

00353'024414   XFEED:   LDA 1 FELED
00354'044412            STA 1 FTEM2
00355'102400            SUB 0 0
00356'063511   XF1:     SKPBZ TTO
00357'000777            JMP .-1
00360'061111            DOAS 0 TTO
00361'014405            DSZ FTEM2
00362'000774            JMP XF1
00363'060211            NIOC TTO
00364'001400            JMP 0 3

00365'000000   FTEM1:   0
00366'000000   FTEM2:   0
00367'000150   FELED:   150
                        .END
```

Program Messages for Operating System

Synopsis This routine contains all the messages for the operating system.

Method Byte pointers are stored on page zero.

```
;**************************************************

;MESSAGES FOR OPERATING SYSTEM

;REVISED SEPT 1, 1971

;**************************************************


.TITL   MESSOP
.ENT    AM10 AM11 AM12 AM13 AM14 AM15 AM16
.ENT    AM21 AM23 AM25 AM26 AM30
.ENT    AM33 AM34 AM35 AM36 AM37

.ZREL

;BYTE POINTERS

AM10:   M10*2
AM11:   M11*2
AM12:   M12*2
AM13:   M13*2
AM14:   M14*2
AM15:   M15*2
AM16:   M16*2
AM21:   M21*2
AM23:   M23*2
AM25:   M25*2
AM26:   M26*2
AM30:   M30*2
AM33:   M33*2
AM34:   M34*2
AM35:   M35*2
AM36:   M36*2
AM37:   M37*2

.NREL

M10:    .TXT "GPC "
M11:    .TXT "OFF "
M12:    .TXT "ON  "
M13:    .TXT "SAMPLE NO. "
M14:    .TXT "RV SENSE "
M15:    .TXT "AT RV "
M16:    .TXT "RATE  "
M21:    .TXT "TURN ON TAPE "
M23:    .TXT "MONITOR "
M25:    .TXT "DATA NOT FOUND "
M26:    .TXT "STOP NOTED FOR "
M30:    .TXT "INJECTION NOTED FOR "
M33:    .TXT " STORAGE DUMP "
M34:    .TXT " NS "
M35:    .TXT "AUTO INJ  "
M36:    .TXT "AUTO TYPE "
M37:    .TXT "AUTO CALC "
.END
```

Program — Operating System Storage Buffer and Buffer Initialization

Synopsis — This routine clears storage when initialized and then provides storage areas for operating parameters and chromatogram heights

Method

Initialization

1. Clear height storage area
2. Clear parameter storage area
3. Return

Notes - This program must be loaded last, as chromatogram heights are stored from the end of this program to location 5577 (start of floating point interpreter.)

```
                ;**********************************************
                ;OPERATING SYSTEM STORAGE BUFFER
                ;AND BUFFER INITIALIZATION

                ;REVISED SEPT 2, 1971

                ;**********************************************

                ;NOTE    - THIS PROGRAM MUST BE LOADED LAST
                ;          HEIGHTS ARE STORED FROM THE END OF THIS PROGRM
                ;          UP TO LOCATION 5577
                .TITL   GST
                .ENT    .CLEAR .GINIT NUM ADDRS .G1
                .ENT    PC .PC PD .PD
                .EXTD   RETN

                .ZREL
00000-000000'   .CLEAR: CLEAR
00001-000010'   .GINIT: GINIT
00002-000044'   .G1:    G1
00003-000000    NUM:    0
00004-000031'   ADDRS:  .A
00005-000115'   .PC:    PC
00006-000103'   .PD:    PD

                .NREL
                ;CLEAR CHROMATOGRAM STORAGE AREA

00000'030521    CLEAR:  LDA 2 .G1A
00001'024465            LDA 1 G1C
00002'102400            SUB 0 0
00003'041000            STA 0 0 2
00004'151400            INC 2 2
00005'132033            ADCZ# 1 2 SNC   ;.MAX
00006'000775            JMP .-3
00007'001400            JMP 0 3

                ;CLEAR OPERATING PARAMETER STORAGE AREA

00010'030002-   GINIT:  LDA 2 .G1
00011'102400            SUB 0 0
00012'040003-           STA 0 NUM       ;CLEAR NUM
00013'024416            LDA 1 .A
00014'044004-           STA 1 ADDRS
00015'020504            LDA 0 .G1A
00016'040426            STA 0 G1
00017'041024            STA 0 24 2      ;START = CURRENT
00020'020410            LDA 0 TN21      ;-21
00021'126400            SUB 1 1
00022'151400    GI1:    INC 2 2
00023'045000            STA 1 0 2
00024'101404            INC 0 0 SZR     ;THROUGH
00025'000775            JMP GI1         ;NO
00026'030002-           LDA 2 .G1
00027'001400            JMP 0 3         ;RETURN


00030'177757    TN21:   -21
00031'000032'   .A:     A
```

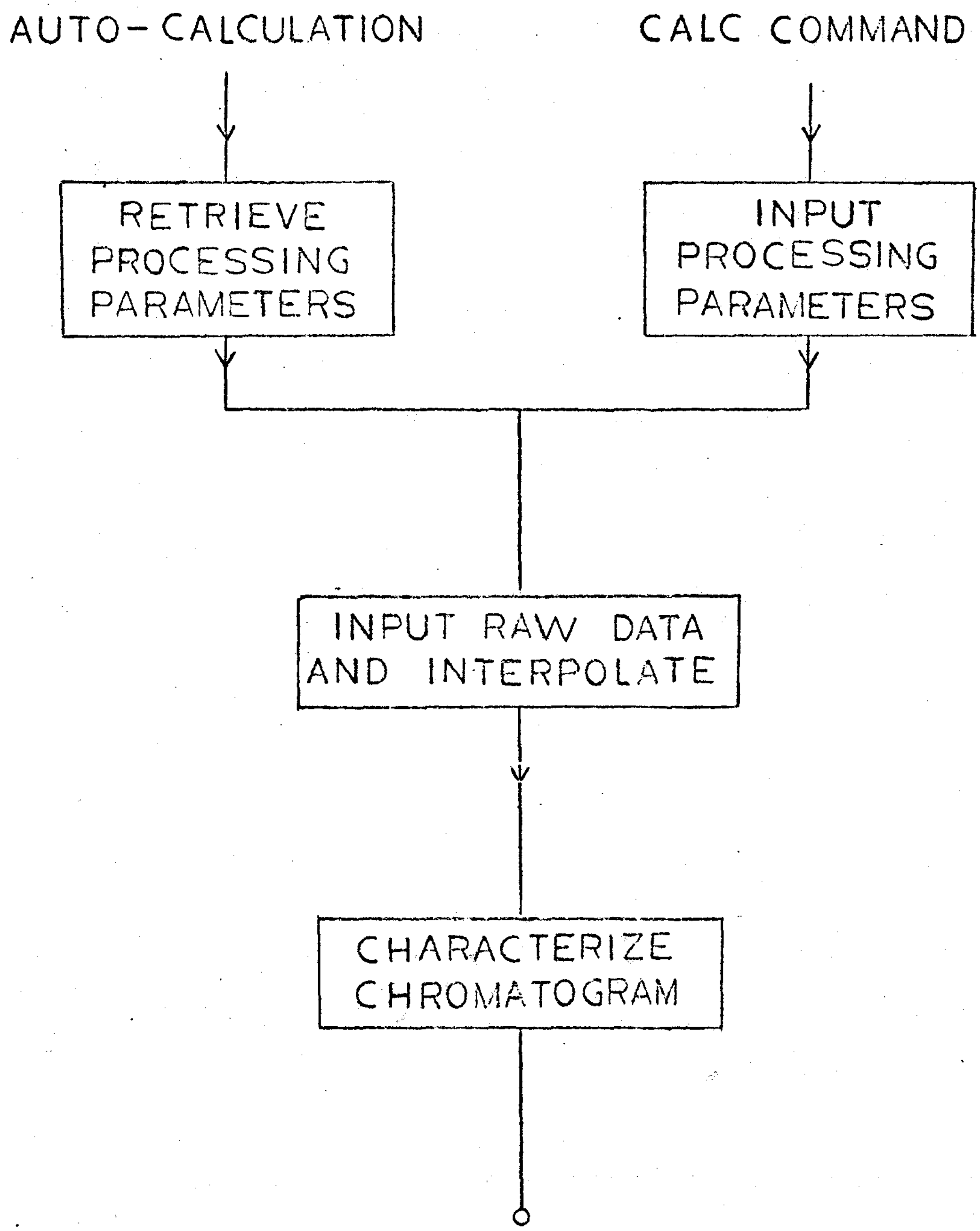
```
      000012  A:       .BLK 10.
00044'000121' G1:      .G1A      ;0 -CURRENT STORAGE LOCATION
00045'000000           0         ;1 - NUMBER OF SAMPLES
00046'000000           0         ;2 - STARTING ELUTION VOLUME
00047'000000           0         ;3 - END ELUTION VOLUME FIRST SAMPLE
00050'000000           0         ;4 - EV AT SECOND INJ
00051'000000           0         ;5 - RATE CODE 1=10 SEC., 2-20 SEC., ETC
00052'000000           0         ;6 - CURRENT ELUTION VOLUME FROM START
00053'000000           0         ;7 - CLOCK 1, (RATE -- 0)
00054'000000           0         ;10- CLOCK 2, (100 -- 0)
00055'000000           0         ;11- BELL COUNTER
00056'000000           0         ;12- SECOND SAMPLE EV STOP
00057'000000           0         ;13- MONITOR STATUS (0 OR 1)
00060'000000           0         ;14- LAST MONITOR PRINT OUT LOCATION
00061'000000           0         ;15- SAMPLE 1 NUMBER
00062'000000           0         ;16- SAMPLE 2 NUMBER
00063'000000           0         ;17- PATH FOR CLOCK RECOGNITION
00064'000000           0         ;20- CLOCK 4 (600 -- 0)
00065'000000           0         ;21- HEIGHT RECORD (0-NO, 1-YES)
00066'005577  G1C:     5577      ;22- MAXIMIUM STORAGE LOCATION
00067'005447           5447      ;23- WARNING LOCATION
00070'000122'          G1A       ;24- START OF STORAGE BLOCK
00071'000000           0         ;25- CHANNEL NUMBER
00072'000000  AI:      0         ;26
00073'000074' .AT:     AT        ;27
      000003  AT:      .BLK 3    ;30,31,32 ,STORE FOR AUTO-TYPE
00077'000100' .AC:     AC        ;33
      000003  AC:      .BLK 3    ;34,35 36 ,STORE FOR AUTO-CAL
      000012  PD:      .BLK 10.  ;37-50  STORE FOR D1, D2, D3,..
      000004  PC:      .BLK 4    ;51-55 ,STORE FOR AUTO CAL PAR
00121'000122' .G1A:    G1A
00122'000000  G1A:     0

              .END
```

A:1-5 PROGRAM 1

Program 1 calculates useful information from the raw data collected by the operating system. The twelve subroutines comprising Program 1 can be divided into three main subsets as indicated in the algorithm of Program 1, Figure A-2. The discussion which follows is intended to briefly summarize the information flow and interaction of the three program subsets. The reader is directed to program listings of Program 1 in A:1-7 for specific details of individual routines.

The first set of subroutines, CAL1, PICK and RST, control the input of processing parameters and raw data. If the calculation is initiated automatically by the auto-action routine of the operating system, the processing parameters are retrieved from the operating system storage buffer. If the calculation is initiated by the operating command CALC, the user must enter the processing parameters at the teletype. Whatever the source, the processing parameters are stored in temporary locations, making it possible to reprocess old data tapes without disturbing the auto-processing of current data.

The second set of subroutinesRST, XDEL, INTD and BASE, reduce the raw fixed point heights to floating point, flow interpolated, baseline corrected heights. The interpolation routines XDEL INTP reduce the raw data, by linear interpolation, to 59 evenly spaced, flow adjusted heights. This interpolation process is necessary to reduce errors caused by changes in the flow rate over the retention volume range of the sample. The floating point data is then baseline corrected, by subtracting




PROGRAM SET I

FIGURE A-2

a linear baseline. The linear baseline is determined from the heights on either side of the chromatogram, at positions specified in the processing parameters.

The third set of subroutines NORM, SIMP, MOL, PRINT, MWCC and EXP characterize the chromatogram. Molecular weight averages are computed from the interpolated and adjusted data. The interpolated data is normalized by determining the area under the curve using Simpson's Rule. The molecular weight averages are computed from the moments of the chromatogram, also using Simpson's Rule. The differential distribution is computed from the normalized heights and the molecular weight calibration curve . Output includes important processing parameters, the area under the baseline corrected chromatogram, the mean retention volume, the molecular weight averages, and a table with the differential molecular weight distribution.

A:1-6 Program 1 - Symbols

| | |
|---|---|
| AREA | area under chromatogram |
| BB | beginning of base |
| BBH | height at beginning of base |
| BC | beginning of calculation |
| CEV | elution volume counter |
| COUNT | number of heights between elution volumes |
| DELA | elution volume interval between fixed point data |
| DELX | elution volume interval between interpolated data |
| DIN | address of data input routine |
| DP | number of data points |
| EB | end of base |
| EBH | height at end of base |
| EC | end of calculation |
| MEAN | mean elution volume |
| MN | number average molecular weight |
| MZ | z-average molecular weight |
| POINT | address of fixed point data |
| SN | sample number |
| V1 | elution volume of first interpolated data point |
| WORK | storage buffer for interpolated data |
| X | elution volume of current interpolated data point |
| X1 | elution volume of first fixed point data point |
| | address of storage buffer |

A:1-7 Program Listings - Program 1

| | |
|---|---|
| CAL1 | Calculation Control, Program Set 1 |
| RST | Input Control |
| PICK | Pick |
| XDELX | XDELX |
| INTP | Interpolate |
| BASE | Linear Baseline Correction |
| NORM | Normalize |
| SIMP | Extended Simpson's Rule |
| MM | Molecular Weights |
| PRINT | Print |
| EXP | Floating Point Exponential Routine |
| MWCC | Three Parameter Calibration Curve |
| MESSCR | Messages for Calculation Routine |

Program Calculation Control Program

Synopsis This routine controls data reduction for data either stored in memory or on paper tape. It includes a processing parameter input and storage routine, auto-calculation control routine, and a manual calculation control routine.

Method

Parameter Input for Auto-Calculation

1. Input parameters
2. Store in operating system buffer
3. Return

Auto-Calculation

1. Pick up sample processing information from buffer
2. Exit to data input and interpolation routines
3. Exit to normalize routine
4. Exit to molecular weight, molecular weight distribution and mean elution volume routine
5. Return

Manual Calculation

1. Request processing parameters
2. Exit to data input and interpolation routines
3. Exit to normalize routine
4. Exit to molecular weight, molecular weight distribution and mean elution volume routine
5. Return

```
;*********************************************

;CALCULATION  CONTROL, PROGRAM SET 1

;REVISED OCT 1, 1971

;*********************************************


;TO INPUT PARAMETERS
;CALL SEQUENCE  -          JSR@ .CPAR
;                          RETURN

;TO EXECUTE A MANUAL CALCULATION
;CALL SEQUENCE  -          JSR@ .MCAL
;                          RETURN

;TO EXECUTE AN AUTO-CALCULATION
;CALL SEQUENCE  -          JSR@ .ACAL
;                          RETURN
;INPUT  -AC1 MUST CONTAIN SAMPLE NUMBER

                 .TITL   CAL1
                 .ENT    DIST  SN BB BC EB EC .CPAR CPAR MCAL  .ACAL
                 .EXTD   FEED  TYPE DBIN MEV CRLF .BASE RST .IMEM .ITAPE
                 .EXTD   BBH .IMWCC  GETC NORM MOL DP DELX V1
                 .EXTD   CM01 CM02 CM03 CM04 CM05 CM06 CM07
                 .EXTD   PRINT  BIND D1 .DPRI .G1
                 .EXTD   .PD .PC .D1
                 .EXTN   PD

                 .ZREL
00000-000001-    .MS:    IP
00001-000000     IP:     0         ;INPUT TYPE
00002-000000     SN:     0         ;SAM NO.
00003-000000     BB:     0         ;BEG OF BASE
00004-000000     EB:     0         ;END OF BASE
00005-000000     BC:     0         ;BEG OF CAL
00006-000000     EC:     0         ;END OF CALC
00007-000000'    .CPAR:  CPAR
00010-000053'    .ACAL:  ACAL

                 .NREL

                 ;INPUT PARAMETERS

00000'054545     CPAR:   STA 3 C1
00001'030035$            LDA 2 .G1
00002'020037$            LDA 0 .PC
00003'040540             STA 0 STO           ;STORE STORAGE LOCAT
00004'024535             LDA 1 T4
00005'044537             STA 1 COUNT         ;NUMBER OF INPUTS
00006'030524             LDA 2 .XM2          ;BYTE POINTER ADDRESS
00007'004404             JSR CP              ;EXIT TO INPUT
00010'006013$            JSR@ .IMWCC        ;INPUT CAL CURVE PAR
00011'177777             PD
00012'002533             JMP@ C1             ;RETURN

00013'054533     CP:     STA 3 C2
00014'023000             LDA@ 0 0 2
```

```
00015'006002$          JSR@ TYPE         ;"MESSAG"
00016'006003$          JSR@ DBIN
00017'046524           STA 1 @STO        ;STORE INPUT
00020'010523           ISZ STO           ;INC STORAGE
00021'151400           INC 2 2           ;INC POINTER
00022'014522           DSZ COUNT         ;THROUGH
00023'000771           JMP CP+1          ;NO
00024'002522           JMP@ C2

                 ;ENTRY FOR MANUAL CALCULATION
00025'054520  MCAL:    STA 3 C1
00026'006005$          JSR@ CRLF
00027'020000-          LDA 0 .MS
00030'040513           STA 0 STO         ;STORE STORAGE LOC
00031'024511           LDA 1 T6
00032'044512           STA 1 COUNT       ;NO. OF INPUTS
00033'030476           LDA 2 .XM1        ;BYTE POINTER ADDR
00034'004757           JSR CP           ;INPUT
00035'006013$          JSR@ .IMWCC      ;INPUT CAL CURV PAR
00036'000033$          D1
00037'020510           LDA 0 T1000
00040'024002-          LDA 1 SN
00041'107000           ADD 0 1           ;AC1=STOP
00042'020001-          LDA 0 IP          ;INPUT TYPE
00043'034010$          LDA 3 .IMEM
00044'101005           MOV 0 0 SNR       ;IP=
00045'034011$          LDA 3 .ITAPE       ;0
00046'005400           JSR 0 3           ;EXIT TO APPR ROUTINE
00047'020040$          LDA 0 .D1
00050'040443           STA 0 H1
00051'040452           STA 0 H2
00052'000435           JMP CAL           ;DO CALC

                 ;ENTRY FOR AUTO CALCULATION

00053'054472  ACAL:    STA 3 C1
00054'044002-          STA 1 SN
00055'006005$          JSR@ CRLF
00056'006005$          JSR@ CRLF
00057'006001$          JSR@ FEED
00060'006005$          JSR@ CRLF
00061'020024$          LDA 0 CM03
00062'006002$          JSR@ TYPE         ;"SAMPLE NO. "
00063'024002-          LDA 1 SN
00064'006032$          JSR@ BIND
00065'006005$          JSR@ CRLF
00066'006005$          JSR@ CRLF
00067'034037$          LDA 3 .PC         ;STORE FOR PAR
00070'021400           LDA 0 0 3         ;STORE AUTO PAR IN CALC PAR
00071'040003-          STA 0 BB
00072'021401           LDA 0 1 3
00073'040004-          STA 0 EB
00074'021402           LDA 0 2 3
00075'040005-          STA 0 BC
00076'021403           LDA 0 3 3
00077'040006-          STA 0 EC
00100'020002-          LDA 0 SN
00101'024446           LDA 1 T1000
00102'107000           ADD 0 1           ;AC1=STOP
00103'006010$          JSR@ .IMEM        ;INITIALIZE MEM
```

```
00104'020036$          LDA 0 .PD
00105'040406           STA 0 H1
00106'040415           STA 0 H2

             ;INPUT HEIGHTS

00107'006007$ CAL:     JSR@ RST          ;INPUT DATA
00110'002435           JMP @C1           ;ERROR RETN
00111'006006$          JSR@ .BASE          ;FIX PAR


             ;OUTPUT PARAMETERS

00112'006034$          JSR@ .DPRI
00113'000000  H1:      0

00114'006031$          JSR@ PRINT
00115'000002           2
00116'000025$          CM04
00117'000012$          BBH

00120'006015$          JSR@ NORM         ;NORMALIZE HEIGHTS

00121'006004$          JSR@ MEV          ;CALCULATE MEAN


00122'006016$          JSR@ MOL          ;CALCULATE MOL WTS.
00123'000000  H2:      0
00124'000001  DIST:    1                 ;DISTRIBUTION ON

00125'006005$          JSR@ CRLF
00126'006005$          JSR@ CRLF
00127'006001$          JSR@ FEED
00130'002415           JMP@ C1

00131'000133' .XM1:    XM1
00132'000135' .XM2:    XM2
00133'000023$ XM1:    CM02     ;TYPE
00134'000024$          CM03     ;SAM NO
00135'000025$ XM2:    CM04     ;BB
00136'000026$          CM05     ;EB
00137'000027$          CM06     ;BC
00140'000030$          CM07     ;EC

00141'000004  T4:      4
00142'000006  T6:      6
00143'000000  STO:     0
00144'000000  COUNT:   0
00145'000000  C1:      0
00146'000000  C2:      0
00147'001750  T1000:   1000.
              .END
```

**Program** Input Control

**Synopsis** This routine is used to input raw data and then exits to the interpolation routine.

**Method**

1. Initialize
2. Exit to calculate X, X1 and DELX
3. Search for sample number
4. Search for injection code
5. Search for start of base and store
6. Search for start of calculation
7. Input 'fixed' point data for one elution volume and exit to interpolation routine, INTP
8. Repeat data input and interpolation until end of calculation is reached
9. Search for end of base and store
10. Return

**Note** - All data actually used by this routine are echoed on the teletype.

Only raw data between individual elution volume dumps are stored.

```
                ;*********************************************
                ;INPUT CONTROL
                ;REVISED OCT 1, 1971
                ;*********************************************
                ;CALL SEQUENCE  -       JSR@ RST
                ;                       ERROR RETURN
                ;                       NORMAL RETURN

                .TITL   RST
                .ENT    COUNT POINT  WORK  BBH EBH CEV  .WORK RST
                .ENT    .WW1 EV1  HE1 HE2 .TAB HE11 HE22
                .EXTD   INTP .X1DELA .XDELX  SN BB BC EC EB DIN ERROT
                .EXTD   SPACE BIND CRLF

                .ZREL
00000-000000    COUNT:  0               ;NUMBER OF POINTS
00001-000000    POINT:  0               ;ADDRESS OF NEXT STORAGE
00002-000000    HE1:    0               ;FIRST HEIGHT
00003-000000    HE11:   0
00004-000000    HE2:    0               ;SECOND HEIGHT
00005-000000    HE22:   0
00006-000000    EV1:    0.0             ;EV COUNTER (FP)
00007-000000
00010-000275'   .WORK:  WORK            ;ADDRESS OF WORKING AREA
00011-000000    BBH:    0.0             ;HEIGHT AT BEGINNING OF BASE
00012-000000
00013-000000    EBH:    0.0             ;HEIGHT AT END OF BASE
00014-000000
00015-000000'   RST:    XRST
00016-000000    .WW1:   0
00017-000000    CEV:    0               ;ELUTION VOLUME COUNTER
00020-000233'   .TAB:   TAB

                .NREL
00000'054511    XRST:   STA 3 SAVE
00001'102400            SUB 0 0
00002'040017-           STA 0 CEV
00003'040006-           STA 0 EV1
00004'040000-           STA 0 COUNT
00005'020474            LDA 0 .POINT
00006'040001-           STA 0 POINT
00007'020006$           LDA 0 BC
00010'040007-           STA 0 EV1+1
00011'006003$           JSR@ .XDELX     ;CALCULATE X, AND DELX
00012'006004            FETR
00013'060006-           FFLO EV1
00014'100000            FEXT
00015'024010-           LDA 1 .WORK
00016'044016-           STA 1 .WW1      ;.WW1=.W1 , STORAGE POINTER
00017'020470            LDA 0 T10
00020'040470            STA 0 TCO       ;SET UP TABLE OUTPUT

                ;SEARCH FOR SAMPLE NO.

00021'006011$ P1:       JSR@ DIN
```

```
00022'002467          JMP@ SAVE
00023'030004$         LDA 2 SN
00024'146414          SUB# 2 1 SZR      ;SEARCH FOR SAMPLE NUMBER
00025'000774          JMP P1
00026'006020-         JSR@ .TAB         ;OUTPUT NUMBER

                ;SEARCH FOR INJ
00027'006011$ P2:     JSR@ DIN
00030'002461          JMP@ SAVE
00031'030453          LDA 2 T4000
00032'146433          SUBZ# 2 1 SNC     ;AC1>4000
00033'000774          JMP P2            ;NO
00034'030451          LDA 2 T5000
00035'132433          SUBZ# 1 2 SNC     ;AC1<=5000
00036'000771          JMP P2            ;NO
00037'006020-         JSR@ .TAB         ;OUTPUT INJECTION

                ;SEARCH FOR BEGINING OF BASE

00040'006011$ P3:     JSR@ DIN
00041'002450          JMP@ SAVE
00042'030444          LDA 2 T7000
00043'146433          SUBZ# 2 1 SNC     ;ELUTION VOLUME
00044'000774          JMP P3            ;NO
00045'006020-         JSR@ .TAB
00046'010017- P3A:    ISZ CEV
00047'030005$         LDA 2 BB
00050'034017-         LDA 3 CEV
00051'156414          SUB# 2 3 SZR          ;BEGINING OF BASE
00052'000766          JMP P3            ;NO
00053'030433          LDA 2 T7000
00054'146400          SUB 2 1 ;AC1-7000.
00055'044425          STA 1 CL1         ;CLOCK READING

                ;SEARCH FOR FIRST HEIGHT AFTER BEG OF BASE

00056'006011$ P4:     JSR@ DIN
00057'002432          JMP@ SAVE
00060'030423          LDA 2 T1000
00061'132433          SUBZ# 1 2 SNC     ;HEIGHT
00062'000774          JMP P4            ;NO
00063'044012-         STA 1 BBH+1
00064'006020-         JSR@ .TAB         ;OUTPUT FIRST HEIGHT AFTER BEGIB
00065'152400          SUB 2 2
00066'050011-         STA 2 BBH
00067'006004          FETR
00070'060011-         FFLO BBH
00071'020011-         FLDA 0 BBH
00072'040002-         FSTA 0 HE1
00073'100000          FEXT
00074'030005$         LDA 2 BB
00075'034006$         LDA 3 BC
00076'156414          SUB# 2 3 SZR      ;BC-BB=0
00077'000413          JMP P5  ;NO
00100'000441          JMP P6A
00101'000255' .POINT: STORE
00102'000000  CL1:    0
00103'001777  T1000:  1023.
00104'007640  T4000:  4000.
00105'011610  T5000:  5000.
```

```
00106'015530  T7000:  7000.

00107'000010  T10:    10
00110'000000  TCO:    0
00111'000000  SAVE:   0
              ;SEARCH FOR BEGINING OF CALCULATION

00112'006011$ P5:     JSR@ DIN
00113'002776          JMP@ SAVE
00114'030772          LDA 2 T7000
00115'146433          SUBZ# 2 1 SNC     ;EV
00116'000774          JMP P5            ;NO
00117'004514          JSR TAB           ;OUTPUT EV
00120'010017-         ISZ CEV
00121'030017-         LDA 2 CEV
00122'034006$         LDA 3 BC
00123'156414          SUB# 2 3 SZR      ;EV=BEG OF CAL
00124'000766          JMP P5            ;NO
00125'030761          LDA 2 T7000
00126'146400          SUB 2 1
00127'044753          STA 1 CL1         ;CLOCK READING

              ;STORE  DATA AND COUNT EV

00130'006011$ P6:     JSR@ DIN
00131'002760          JMP@ SAVE
00132'030754          LDA 2 T7000
00133'132033          ADCZ# 1 2 SNC     ;EV
00134'000421          JMP P7            ;YES
00135'030746          LDA 2 T1000
00136'132433          SUBZ# 1 2 SNC     ;HEIGHT
00137'000771          JMP P6            ;NO
00140'004473          JSR TAB
00141'030001- P6A:    LDA 2 POINT
00142'045000          STA 1 0 2         ;STORE HEIGHT
00143'010000-         ISZ COUNT         ;COUNT HEIGHTS
00144'151400          INC 2 2
00145'050001-         STA 2 POINT
00146'020506          LDA 0 .FILL       ;FILLED BUFFER
00147'112433          SUBZ# 0 2 SNC     ;AC2>=.FILL
00150'000760          JMP P6            ;NO
00151'020403          LDA 0 FILL
00152'006012$         JSR@ ERROT
00153'002736          JMP@ SAVE         ;ERROR RETURN
00154'000062  FILL:   50.

00155'010017- P7:     ISZ CEV            ;INC EV COUNTER
00156'004455          JSR TAB
00157'020727          LDA 0 T7000
00160'106400          SUB 0 1           ;CLOCK
00161'020720          LDA 0 .POINT
00162'040001-         STA 0 POINT
00163'020717          LDA 0 CL1         ;LAST CLOCK
00164'044716          STA 1 CL1         ;SET CL1= NEW CLOCK
00165'030000-         LDA 2 COUNT
00166'006002$         JSR@ .X1DELA      ;CAL X1 AND DELA
00167'006001$         JSR@ INTP         ;INTERPOLATE
00170'020711          LDA 0 .POINT
00171'040001-         STA 0 POINT       ;STORAGE TO START
00172'030017-         LDA 2 CEV
```

```
00173'034007$          LDA 3 EC
00174'156414           SUB# 2 3 SZR       ;END OF CAL
00175'000733           JMP P6             ;NO
00176'030010$          LDA 2 EB
00177'156414           SUB# 2 3 SZR       ;EC=EB
00200'000406           JMP P8  ;NO
00201'006004           FETR
00202'020002-          FLDA 0 HE1
00203'040013-          FSTA 0 EBH
00204'100000           FEXT
00205'000424           JMP OUT

               ;SEARCH FOR END OF BASE

00206'006011$ P8:      JSR@ DIN
00207'002702           JMP@ SAVE
00210'030676           LDA 2 T7000
00211'146433           SUBZ# 2 1 SNC
00212'044014-          STA 1 EBH+1
00213'030673           LDA 2 T7000
00214'146433           SUBZ# 2 1 SNC      ;EV
00215'000771           JMP P8
00216'004415           JSR TAB
00217'010017-          ISZ CEV
00220'030017-          LDA 2 CEV
00221'034010$          LDA 3 EB
00222'156414           SUB# 2 3 SZR       ;END OF BASE
00223'000763           JMP P8             ;NO
00224'102400           SUB 0 0
00225'040013-          STA 0 EBH
00226'006004           FETR
00227'060013-          FFLO EBH           ;END OF BASE HEIGHT
00230'100000           FEXT
00231'034660  OUT:     LDA 3 SAVE
00232'001401           JMP 1 3            ;NORMAL RETN

00233'054417  TAB:     STA 3 TSAVE
00234'044415           STA 1 HOLD
00235'006014$          JSR@ BIND
00236'014652           DSZ TCO
00237'000406           JMP TA1
00240'006015$          JSR@ CRLF
00241'020646           LDA 0 T10
00242'040646           STA 0 TCO
00243'024406           LDA 1 HOLD
00244'002406           JMP@ TSAVE
00245'024406  TA1:     LDA 1 T4
00246'006013$          JSR@ SPACE
00247'024402           LDA 1 HOLD
00250'002402           JMP@ TSAVE
00251'000000  HOLD:    0
00252'000000  TSAVE:   0
00253'000004  T4:      4

00254'000276' .FILL:   STORE+17.
      000020  STORE:   .BLK 16.
      000174  WORK:    .BLK 124.          ;ADJUSTED DATA STORAGE


              .END
```

<u>Program</u> Pick

<u>Synopsis</u> This routine controls data acquisition from either paper tape or memory

<u>Method</u>

1. Initialize either paper tape input or memory search
2. Input data through this routine which checks for errors

<u>Notes</u> - <u>Error 200</u>, if data is not found in two complete memory searches

<u>Error 201</u>, if data is not complete before stop is encountered

```
                ;********************************************
                ;PICK
                ;REVISED NOV 1, 1971
                ;******************************************

                ;INITALIZE MEMORY
                ;CALL SEQUENCE  -        JSR@ .IMEM
                ;                        RETURN
                ;INPUT  - AC0 CONTAINS STOP CHECK


                ;INITIALZE TAPE
                ;CALL SEQUENCE  -        JSR@ .ITAPE
                ;                        RETURN


                ;INPUT DATA
                ;CALL SEQUENCE  -        JSR@ DIN
                ;                        ERROR RETURN
                ;                        NORMAL RETURN


                .TITL   PICK
                .ENT    DIN .IMEM .ITAPE
                .EXTD   CEV DBIN ERROT .G1 GETC

                .ZREL
00000-000000    DIN:    0
00001-000000'   .IMEM:  IMEM
00002-000034'   .ITAPE: ITAPE

                .NREL

                ;SOURCE OF DATA IS MEMEORY

00000'044463    IMEM:   STA 1 CHECK
00001'030004S           LDA 2 .G1
00002'025024            LDA 1 24 2      ;MIN MEMORY
00003'044475            STA 1 MIN
00004'044473            STA 1 CURR      ;SET CURRENT TO MINIMIUM
00005'025022            LDA 1 22 2      ;MAXIMIUM MEMORY LOC
00006'044473            STA 1 MAX
00007'024405            LDA 1 .XM
00010'044000-           STA 1 DIN
00011'020472            LDA 0 T2
00012'040470            STA 0 C1        ;SET MAXIMIUM MEMORY RECYCLE TO0
00013'001400            JMP 0 3         ;RETURN

00014'000015'   .XM:    MEM
00015'054461    MEM:    STA 3 SAVE
00016'030461            LDA 2 CURR
00017'025000            LDA 1 0 2
00020'151400            INC 2 2         ;INC POINTER
00021'050456            STA 2 CURR
00022'020457            LDA 0 MAX
00023'112033            ADCZ# 0 2 SNC   ;CURR > MAX
00024'000426            JMP .CHECK
```

```
00025'030453  RECYC:  LDA 2 MIN
00026'050451          STA 2 CURR          ;SET CURRENT TO MINIMIUM
00027'014453          DSZ C1              ;CHECK RECYCLES
00030'000422          JMP .CHECK
00031'020454          LDA 0 E200
00032'006003$         JSR@ ERROT
00033'002443          JMP@ SAVE

              ;SOURCE OF DATA IS TAPE

00034'044427  ITAPE:  STA 1 CHECK         ;SAVE STOP
00035'020403          LDA 0 .TAPE
00036'040000-         STA 0 DIN           ;SET DIN TO TAPE
00037'001400          JMP 0 3

00040'000041' .TAPE:  TAPE
00041'054435  TAPE:   STA 3 SAVE
00042'020422          LDA 0 .IN
00043'040040          STA 0 40            ;CHANGE INPUT ROUTINE TO AVOID B
00044'006002$         JSR@ DBIN           ;INPUT NUMBER
00045'020005$         LDA 0 GETC
00046'040040          STA 0 40            ;RESTORE NORMAL INPUT ROUTINE


00047'125005          MOV 1 1 SNR         ;CHECH FOR ZERO INPUT
00050'000772          JMP TAPE+1
00051'034425          LDA 3 SAVE
00052'020411  .CHECK: LDA 0 CHECK
00053'122414          SUB# 1 0 SZR
00054'001401          JMP 1 3             ;RETN
00055'020001$         LDA 0 CEV
00056'101005          MOV 0 0 SNR
00057'001401          JMP 1 3             ;RETN IF NO HEIGHTS YET
00060'020426          LDA 0 E201
00061'006003$         JSR@ ERROT
00062'002414          JMP@ SAVE           ;ERROR RETN
00063'000000  CHECK:  0

00064'000065' .IN:    IN
00065'060110  IN:     NIOS TTI
00066'063610          SKPDN TTI
00067'000777          JMP .-1
00070'060610          DIAC 0 TTI
00071'024413          LDA 1 MSK
00072'123400          AND 1 0             ;7 BIT MASK
00073'122414          SUB# 1 0 SZR
00074'001400          JMP 0 3             ;RETURN
00075'002401          JMP@ SAVE           ;RETURN BY LAST TAPE RETURN
                                          ;WHEN A RUBOUT IS INPUT

00076'000000  SAVE:   0
00077'000000  CURR:   0
00100'000000  MIN:    0
00101'000000  MAX:    0
00102'000000  C1:     0
00103'000002  T2:     2
00104'000177  MSK:    177
00105'000310  E200:   200.
00106'000311  E201:   201.
              .END
```

Program XDELX

Synopsis This routine calculates X, V1, DELX and X1, DELA using the clock readings added to elution volume code.

Method

XDELX

1. Determine elution volume range of calculation
2. Calculate elution volume increment for interpolated points (DELX) and store
3. Calculate elution volume of first height (X, V1) and store
4. Return

X1DELA

1. Calculate elution volume interval of fixed point data (DELA)
2. Calculate elution volume of first point (X1)
3. Return

```
                  ;*************************************************

                  ;XDELX

                  ;REVISED OCT 1, 1971

                  ;*************************************************

                  ;TO CALCULATE X, DELX
                  ;CALL SEQUENCE  -          JSR@ .XDELX
                  ;                          RETN


                  ;TO CALCULATE X1, DELA
                  ;CALL SEQUENCE  -          JSR@ .X1DELA
                  ;                          RETN
                  ;INPUT   -AC0 MUST CONTAIN CL1
                  ;          AC1 MUST CONTAIN CL2
                  ;          AC2 MUST CONTAIN  NUMBER OF DATA POINTS


                  .TITL    XDEL
                  .ENT     .XDELX .X1DELA X DELX X1 DELA V1
                  .EXTD    BC EC EV1

                  .ZREL
00000-000000'     .XDELX:  XDELX
00001-000021'     .X1DEL:  X1DELA
00002-000000      X:       0.0
00003-000000
00004-000000      DELX:    0.0
00005-000000
00006-000000      X1:      0.0
00007-000000
00010-000000      DELA:    0.0
00011-000000
00012-000000      V1:      0.0                 ;ELUTION VOLUME OF FIRST POINT
00013-000000

                  .NREL

                  ;CALCULATE X, AND DELX

00000'054464      XDELX:   STA 3 SAVE
00001'020001$              LDA 0 BC
00002'040465               STA 0 C1+1
00003'020002$              LDA 0 EC
00004'040465               STA 0 C2+1
00005'004443               JSR CF

00006'020460               FLDA 0 C1
00007'024461               FLDA 1 C2
00010'030464               FLDA 2 T60
00011'106400               FSUB 0 1            ;EC-BC
00012'144200               FDIV 2 1            ;(EC-BC)/DP
00013'044004-              FSTA 1 DELX
00014'123000               FADD 1 0            ;DELX+BC
00015'040002-              FSTA 0 X
00016'040012-              FSTA 0 V1           ;EV OF FIRST POINT
00017'100000               FEXT
```

```
00020'002444          JMP@ SAVE

               ;CALCULATE X1, AND DELA

00021'054443  X1DEL:  STA 3 SAVE
00022'040445          STA 0 C1+1
00023'044446          STA 1 C2+1
00024'050447          STA 2 C3+1
00025'004423          JSR CF

00026'020440          FLDA 0 C1
00027'024441          FLDA 1 C2
00030'030432          FLDA 2 F100
00031'034441          FLDA 3 C3
00032'170100          FMPY 3 2        ;100.*DP
00033'112400          FSUB 0 2        ;-CLOCK AT BEG
00034'133000          FADD 1 2        ;+CLOCK AT END=TOTAL UNITS
00035'024425          FLDA 1 F100
00036'106400          FSUB 0 1
00037'144200          FDIV 2 1
00040'034003$         FLDA 3 EV1      ;EV
00041'137000          FADD 1 3
00042'054006-         FSTA 3 X1       ;EV OF FIRST POINT
00043'024417          FLDA 1 F100
00044'144200          FDIV 2 1
00045'044010-         FSTA 1 DELA     ;100*UNITS PER EV
00046'100000          FEXT
00047'002415          JMP@ SAVE       ;RETN

00050'054415  CF:     STA 3 S2
00051'102400          SUB 0 0
00052'040414          STA 0 C1
00053'040415          STA 0 C2
00054'040416          STA 0 C3
00055'006004          FETR
00056'060410          FFLO C1
00057'060411          FFLO C2
00060'060412          FFLO C3
00061'002404          FJMP @S2        ;RETN

00062'041144  F100:   100.0
00063'000000
00064'000000  SAVE:   0
00065'000000  S2:     0
00066'000000  C1:     0.0
00067'000000
00070'000000  C2:     0.0
00071'000000
00072'000000  C3:     0.0
00073'000000
00074'041074  T60:    60.0
00075'000000

               .END
```

<u>Program</u> Interpolate

<u>Synopsis</u> This routine produces 59 floating point, flow adjusted heights from fixed point raw data.

<u>Method</u>

1. Find fixed point heights on either side of desired point and float them
2. Use linear interpolation to produce new heights
3. Store interpolated height in storage buffer WORK
4. Increment X1 by DELA and repeat interpolation process if possible, otherwise return

```
                  ;*********************************************
                  ;INTERPRET
                  ;WRITTEN OCT 1,1971
                  ;*********************************************
                  ;CALL SEQUENCE  -        JSR@ INTP
                  ;                        RETURN

                  .TITL   INTP
                  .ENT    DP INTP
                  .EXTD   CRLF .WW1  HE1 HE2 HE11 HE22
                  .EXTD    EV1 DELA DELX X X1 POINT COUNT

                  .ZREL
00000-000000' INTP:   XINTP             ;POINTER TO THIS ROUTINE
00001-000073  DP:     59.               ;NUMBER OF POINTS PRODUCED

                  .NREL
00000'054450  XINTP:  STA 3 SAVE
00001'020015$         LDA 0 COUNT       ;NUMBER OF POINTS
00002'101005          MOV 0 0 SNR       ;=ZERO
00003'001400          JMP 0 3           ;YES RETURN

00004'022014$ AGAIN:  LDA 0 @POINT      ;ADDRESS OF DATA
00005'010014$         ISZ POINT
00006'040006$         STA 0 HE22
00007'102400          SUB 0 0
00010'040004$         STA 0 HE2
00011'006004          FETR
00012'060004$         FFLO HE2          ;PREPARE SECOND HEIGHT
00013'020013$ START:  FLDA 0 X1
00014'024012$         FLDA 1 X          ;CURRENT DESIRED X FOR F(X) DETN
00015'122415          FSUB# 1 0 FSGE    ;SKP IF X1>=X
00016'000412          FJMP NEXT         ;GO ON TO NEXT POINT
00017'004434  INT:    FJSR LINTP        ;INTERPOLATE
00020'042002$         FSTA 0 @.WW1
00021'010002$         FISZ .WW1
00022'010002$         FISZ .WW1
00023'020011$         FLDA 0 DELX
00024'024012$         FLDA 1 X
00025'107000          FADD 0 1
00026'044012$         FSTA 1 X          ;X=X+DELX
00027'000764          FJMP START        ;TRY AGAIN

00030'024010$ NEXT:   FLDA 1 DELA
00031'123000          FADD 1 0          ;X1+DELA
00032'040013$         FSTA 0 X1         ;X1=X1+DELA
00033'024004$         FLDA 1 HE2
00034'044003$         FSTA 1 HE1        ;HE1+HE2
00035'014015$         FDSZ COUNT
00036'000402          FJMP .+2          ;NOT ZERO DO AGAIN
00037'000403          FJMP OUT          ;GO ON TO NEXT EV
00040'100000          FEXT
00041'000743          JMP AGAIN

00042'020007$ OUT:    FLDA 0 EV1
```

```
00043'024406            FLDA 1 F1
00044'123000            FADD 1 0
00045'040007$           FSTA 0 EV1       ;EV1=EV1+1.0
00046'100000            FEXT
00047'002401            JMP@ SAVE

00050'000000  SAVE:     0
00051'040420  F1:       1.0
00052'000000

00053'070415  LINTP:    FST3 S1
00054'020010$           FLDA 0 DELA
00055'024013$           FLDA 1 X1
00056'030012$           FLDA 2 X
00057'146400            FSUB 2 1
00060'104200            FDIV 0 1         ;(X1-X)/DELA
00061'020003$           FLDA 0 HE1
00062'030004$           FLDA 2 HE2
00063'112400            FSUB 0 2         ;HE2-HE1
00064'144100            FMPY 2 1
00065'020004$           FLDA 0 HE2
00066'122400            FSUB 1 0
00067'002401            FJMP@ S1
00070'000000  S1:       0

              .END
```

Program Linear Baseline Correction

Synopsis This routine replaces raw chromatogram heights with baseline corrected data.

Method

1. Approximate baseline with linear relationship
2. Subtract baseline and replace original data
3. Output in graphic form the corrected chromatogram
4. Return

```
                ;*******************************************

                ;LINEAR BASELINE CORRECTION WITH GRAPH OUTPUT

                ;REVISED NOV 1 1971

                ;*******************************************

                ;CALL SEQUENCE  -       JSR@ BASE
                ;                       RETURN

                .TITL   BASE
                .ENT    .BASE
                .EXTD   CRLF DELX V1 .WORK DP BBH EBH BB EB
                .EXTD   SPACE  TYPE


                .ZREL
00000-000000'   .BASE:  BASE            ;POINTER TO BASE ROUTINE

                .NREL
00000'054502    BASE:   STA 3 SAVE
00001'006001$           JSR@ CRLF
00002'006001$           JSR@ CRLF
00003'102400            SUB 0 0
00004'040505            STA 0 EVBB
00005'040506            STA 0 DELB
00006'020010$           LDA 0 BB
00007'040503            STA 0 EVBB+1
00010'024011$           LDA 1 EB
00011'106400            SUB 0 1
00012'044502            STA 1 DELB+1
00013'030004$           LDA 2 .WORK     ;ADD OF DATA
00014'024005$           LDA 1 DP
00015'044471            STA 1 NUM       ;NUM OF DATTA POINTS
00016'006004            FETR
00017'060474            FFLO DELB
00020'060471            FFLO EVBB

                ;CALCULATE SLOPE

00021'020006$   B1:     FLDA 0 BBH      ;HEIGHT AT BEG OF BASE
00022'024007$           FLDA 1 EBH      ;  "   " END  "  "
00023'030470            FLDA 2 DELB     ;DEL EV BASE
00024'106400            FSUB 0 1        ;EBH-BBH
00025'144200            FDIV 2 1
00026'034003$           FLDA 3 V1
00027'054460            FSTA 3 V        ;V=EV OF FIRST POINT
00030'000406            FJMP B3

00031'034456    B2:     FLDA 3 V
00032'030002$           FLDA 2 DELX
00033'157000            FADD 2 3
00034'054453            FSTA 3 V        ;V+DELX
00035'104000            FIC2            ;INC POINTER

                ;SUBTRACT BASE LINE
                ;;OUTPUT RESULTS IN GRAPH FORM

00036'154000    B3:     FFDC 3          ;OUTPUT EV
```

```
00037'100000            FEXT
00040'126520            SUBZL  1 1          ;AC1=1
00041'006012$           JSR@ SPACE
00042'006004            FETR
00043'030446            FLDA 2 EVBB
00044'156400            FSUB 2 3            ;EV-EVBB
00045'134100            FMPY 1 3            ;SLOPE*V
00046'117000            FADD 0 3            ;BBH+V*SLOPE
00047'031000            FLDA 2 0 2          ;PICK UP HEIGHT
00050'172400            FSUB 3 2            ;ADJ
00051'034427            FLDA 3 LEAST
00052'172415            FSUB# 3 2 FSGE      ;>= LEAST
00053'030442            FLDA 2 F0
00054'051000            FSTA 2 0 2          ;REPLACE
00055'050427            FSTA 2 HOLD
00056'074426            FFIX HOLD           ;CONVERT TO FIXED POINT
00057'100000            FEXT
00060'024425            LDA 1 HOLD+1
00061'125220            MOVZR 1 1
00062'125220            MOVZR 1 1
00063'125220            MOVZR 1 1
00064'125220            MOVZR 1 1
00065'006012$           JSR@ SPACE
00066'020415            LDA 0 T52
00067'006041            JSR@ 41             ;OUTPUT STAR
00070'006001$           JSR@ CRLF
00071'006004            FETR
00072'014414            FDSZ NUM
00073'000736            FJMP B2

00074'100000  B4:       FEXT
00075'006001$           JSR@ CRLF
00076'006001$           JSR@ CRLF
00077'002403            JMP@ SAVE           ;RETN

00100'040420  LEAST:  1.0                   ;SMALLEST POSSIBLE HEIGHT
00101'000000
00102'000000  SAVE:     0
00103'000052  T52:      52
00104'000000  HOLD:     0.0
00105'000000
00106'000000  NUM:      0
00107'000000  V:        0.0
00110'000000
00111'000000  EVBB:     0.0
00112'000000
00113'000000  DELB:     0.0
00114'000000
00115'000000  F0:       0.0
00116'000000

              .END
```

Program Normalize

Synopsis This routine normalizes an odd number of points

Method

1. Call Simpson's Rule to calculate the area under the curve
2. Set:

$$F(X) = \frac{F(X)}{\text{area}}$$

3. Return

Note - Outputs area under curve

```
                ;**********************************************

                ;NORMALIZE DATA
                ;OUTPUT AREA UNDER THE ORIGNAL CURVE

                ;REVISED OCT 2, 1971

                ;**********************************************

                ;CALL SEQUENCE  -       JSR@ NORM
                ;                       RETN



                .TITL   NORM
                .ENT    NORM XNORM AREA
                .EXTD   CM25 DP .WORK PRINT SIMP

                .ZREL
00000-000000'   NORM:   XNORM
00001-000000    AREA:   0.0             ;AREA UNDER ORIGNAL CURVE
00002-000000

                .NREL

                ;CALCULATE AREA UNDER CURVE

00000'054427    XNORM:  STA 3 SAVE
00001'006004            FETR
00002'006005$           FJSR@ SIMP
00003'000026'           N1

00004'040001-           FSTA 0 AREA     ;STORE AREA
00005'100000            FEXT

                ;DIVIDE POINTS BY AREA UNDER CURVE

00006'024002$           LDA 1 DP        ;NUM OF POINTS
00007'044421            STA 1 DATA
00010'030003$           LDA 2 .WORK
00011'006004            FETR
00012'025000    LOOP:   FLDA 1 0 2
00013'104200            FDIV 0 1
00014'045000            FSTA 1 0 2
00015'104000            FIC2
00016'014412            FDSZ DATA
00017'000773            FJMP LOOP
00020'100000            FEXT
00021'006004$           JSR@ PRINT      ;PRINT AREA
00022'000001            1
00023'000001$           CM25
00024'000001-           AREA

00025'002402            JMP@ SAVE       ;RETN
00026'001400    N1:     FJMP 0 3        ;NO OPERATION

00027'000000    SAVE:    0
00030'000000    DATA:    0
```

Program — Extended Simpson's Rule

Synopsis — This routine calculates the area under the curve described by F(X).

Method — 1. For odd number of points $x_n$,

$$AREA = \left(F(X_1) + 4.0 \sum_{k=1}^{(n-1)/2} F(X_{2k}) + 2.0 \sum_{k=2}^{(n-1)/2} F(X_{2k-1}) + F(x_n)\right) DELX/3.0$$

Note - ERROR HALT - even number of points

```
;*********************************************

;EXTENDED SIMPSON'S RULE

;REVISED SEPT 20, 1971

;*********************************************

; CALL SEQUENCE -         FJSR@ SIMP
;                         ADDRESS OF F(X) SUBROUTINE
;                         RETURN
;INPUT  - ODD NUMBER OF POINTS
;OUTPUT - AREA IN FAC0

;  NOTE - F(X) SUBROUTINE MUST ACCEPT X IN FAC0
;         AND RETURN F(X) IN FAC0
;       - VALID ONLY FOR AN ODD NUMBER OF POINTS

               .TITL   SIMP
               .ENT    SIMP
               .EXTD   DP DELX .WORK

               .ZREL
00000-000000'  SIMP:   XSIMP

               .NREL
00000'100000   XSIMP:  FEXT
00001'025400           LDA 1 0 3
00002'044464           STA 1 S1          ;ADDR OF F(X) SUBROUTINE
00003'175400           INC 3 3
00004'054477           STA 3 SAVE        ;RETN
00005'030003$          LDA 2 .WORK
00006'024001$          LDA 1 DP
00007'044456           STA 1 DATA        ;NUM OF POINTS
00010'006004           FETR
00011'176400           FSUB 3 3
00012'054457           FSTA 3 ODD
00013'054454           FSTA 3 EVEN
00014'021000           FLDA 0 0 2        ;FIRST POINT
00015'006451           FJSR@ S1          ;OPERATE ON
00016'040455           FSTA 0 FIRST      ;STORE
00017'104000           FIC2
00020'014445           FDSZ DATA         ;DATA-1
00021'021000   LOOP:   FLDA  0 0 2
00022'006444           FJSR@ S1          ;OPERATE
00023'024444           FLDA 1 EVEN
00024'107000           FADD 0 1
00025'044442           FSTA 1 EVEN
00026'014437           FDSZ DATA
00027'000402           FJMP .+2
00030'114000           FHLT              ;ERROR, EVEN NUMBER OF POINTS
00031'104000           FIC2
00032'014433           FDSZ DATA
00033'000402           FJMP .+2
00034'000410           FJMP OUT
00035'021000           FLDA  0 0 2
00036'006430           FJSR@ S1          ;OPERATE
00037'024432           FLDA 1 ODD
00040'123000           FADD 1 0
```

```
00041'040430            FSTA 0 ODD
00042'104000            FIC2
00043'000756            FJMP LOOP
00044'021000   OUT:     FLDA  0 0 2       ;LAST
00045'006421            FJSR@ S1
00046'030421            FLDA 2 EVEN
00047'034422            FLDA 3 ODD
00050'024431            FLDA 1 T4
00051'144100            FMPY 2 1
00052'123000            FADD 1 0
00053'024422            FLDA 1 T2
00054'134100            FMPY 1 3
00055'163000            FADD 3 0
00056'024415            FLDA 1 FIRST
00057'123000            FADD 1 0
00060'024002$           FLDA 1 DELX
00061'120100            FMPY 1 0
00062'024415            FLDA 1 T3
00063'120200            FDIV 1 0          ;(F+4*EVEN+2*ODD+L)*DELX/3
00064'002417            FJMP@ SAVE
00065'000000   DATA:    0
00066'000000   S1:      0
00067'000000   EVEN:    0.0
00070'000000
00071'000000   ODD:     0.0
00072'000000
00073'000000   FIRST:   0.0
00074'000000
00075'040440   T2:      2.0
00076'000000
00077'040460   T3:      3.0
00100'000000
00101'040500   T4:      4.0
00102'000000
00103'000000   SAVE:    0

               .END
```

Program — Molecular Weights, Molecular Weight Distribution and Mean Elution Volume

Synopsis — This routine uses Simpson's Rule to calculate the molecular weight averages and mean elution volume. It also calculates and outputs the differential distribution if requested.

Method

1. Calculate zero, second and third moments
2. Compute and output

   $\bar{M}_n$ = 1 /zero moment

   $\bar{M}_w$ = second moment

   $\bar{M}_z$ = third moment/second moment
3. Output distribution
4. Return

```
;*******************************************

;MOLECULAR WEIGHTS,
;MOLECULAR WEIGHT DISTRIBUTION,
;MEAN ELUTION VOLUME

;REVISED NOV 1, 1971

;*******************************************

;TO CALCULATE MOLECULAR WEIGHTS
;CALL SEQUENCE  -        JSR@ MOL
;                        ADDRS OF D1,D2, ...
;                        DISTRIBUTION OUTPUT (1=YES)
;                        RETURN

;TO CALCULATE MEAN ELUTION VOLUME
;CALL SEQUENCE  -        JSR@ MEV
;                        RETURN

               .TITL    MM
               .ENT     MEV MOL MEAN MN MW MZ
               .EXTD    CM01 .MWCC SPACE TYPE CRLF
               .EXTD    DELX SIMP PRINT
               .EXTD    CM20 CM21 CM22 V1

               .ZREL
00000-000000' MOL:     XMOL
00001-000156' MEV:     XMEV
00002-000000  MEAN:    0.0              ;MEAN ELUTION VOLUME
00003-000000
00004-000000  MN:      0.0              ;NUMBER AVERAGE
00005-000000
00006-000000  MW:      0.0              ;WEIGHT AVERAGE
00007-000000
00010-000000  MZ:      0.0              ;Z AVERAGE
00011-000000
00012-000000  POLY:    0.0              ;POLYDISPERSITY
00013-000000

               .NREL

              ;ENTRY FOR MOL

00000'054456  XMOL:    STA 3 SAVE
00001'021400           LDA 0 0 3
00002'040463           STA 0 H1
00003'040524           STA 0 H2
00004'040470           STA 0 H3
00005'040476           STA 0 H4

00006'021401           LDA 0 1 3
00007'040450           STA 0 DT

00010'006004           FETR
00011'004504           FJSR LOAD        ;INITIALIZE FOR M=0
00012'006007$          FJSR@ SIMP
00013'000062'          OP1
```

```
00014'024444            FLDA 1 F1
00015'104200            FDIV 0 1          ;1/AREA
00016'044004-           FSTA 1 MN         ;NUMBER AVERAGE

00017'004476            FJSR LOAD         ;INIT FOR M=2
00020'006007$           FJSR@ SIMP
00021'000071'           OP2
00022'040006-           FSTA 0 MW         ;WEIGHT AVERAGE

00023'004472            FJSR LOAD         ;INIT FOR M=3
00024'006007$           FJSR@ SIMP
00025'000100'           OP3
00026'024006-           FLDA 1 MW
00027'120200            FDIV 1 0
00030'040010-           FSTA 0 MZ         ;Z AVERAGE

00031'020004-           FLDA 0 MN
00032'104200            FDIV 0 1
00033'044012-           FSTA 1 POLY       ;POLYDISPERSITY
00034'100000            FEXT

                ;PRINT AVERAGES

00035'006005$           JSR@ CRLF
00036'006005$           JSR@ CRLF
00037'006010$           JSR@ PRINT
00040'000004            4
00041'000011$           CM20
00042'000004-           MN
00043'006005$           JSR@ CRLF

                ;OUTPUT DISTRIBUTION IF SET

00044'020413    DIST:   LDA 0 DT
00045'101005            MOV 0 0 SNR
00046'000406            JMP OUT
00047'006004            FETR
00050'004445            FJSR LOAD
00051'006007$           FJSR@ SIMP
00052'000122'           OP1A
00053'100000            FEXT
00054'034402    OUT:    LDA 3 SAVE
00055'001402            JMP 2 3           ;RETN

00056'000000    SAVE:   0
00057'000000    DT:     0
00060'040420    F1:     1.0
00061'000000

                ;OPERATIONAL SUBROUTINES MUST NOT DESTROY FAC0

                ;ZERO MOMENT
00062'070471    OP1:    FST3 S1
00063'024435            FLDA 1 XV
00064'006002$           FJSR@ .MWCC
00065'000000    H1:     0
00066'120200            FDIV 1 0          ;HEIGHT/MW
```

```
00067'004421              FJSR DINCR
00070'002463              FJMP@ S1

                 ;SECOND MOMENT
00071'070462    OP2:      FST3 S1
00072'024426              FLDA 1 XV
00073'006002$             FJSR@ .MWCC
00074'000000    H3:       0
00075'120100              FMPY 1 0
00076'004412              FJSR DINCR
00077'002454              FJMP@ S1

                 ;THIRD MOMENT
00100'070453    OP3:      FST3 S1
00101'024417              FLDA 1 XV
00102'006002$             FJSR@ .MWCC
00103'000000    H4:       0
00104'124100              FMPY 1 1
00105'120100              FMPY 1 0
00106'004402              FJSR DINCR
00107'002444              FJMP@ S1

                 ;INCREMENT XV BY DELX

00110'024410    DINCR:    FLDA 1 XV
00111'030006$             FLDA 2 DELX
00112'147000              FADD 2 1
00113'044405              FSTA 1 XV
00114'001400              FJMP 0 3

                 ;XV = V1

00115'024014$ LOAD:       FLDA 1 V1
00116'044402              FSTA 1 XV
00117'001400              FJMP 0 3
00120'000000    XV:       0.0
00121'000000

                 ;OUTPUT DISTRIBUTION
                 ;TABLE FORM
                 ;EV - NORMALIZED HT. - MW - DIFF DISTRIBUTION
00122'070431    OP1A:     FST3 S1
00123'024775              FLDA 1 XV
00124'144000              FFDC 1
00125'004420              FJSR SP
00126'006002$             FJSR@ .MWCC
00127'000000    H2:       0
00130'140000              FFDC 0
00131'004414              FJSR SP
00132'144000              FFDC 1
00133'004412              FJSR SP
00134'115000              FMOV 0 3
00135'154200              FDIV 2 3
00136'154000              FFDC 3
00137'120200              FDIV 1 0
00140'004750              FJSR DINCR
00141'100000              FEXT
00142'006005$             JSR@ CRLF
00143'006004              FETR
00144'002407              FJMP@ S1
```

```
00145'100000  SP:     FEXT
00146'054406          STA 3 S2
00147'024406          LDA 1 SP2
00150'006003$         JSR@ SPACE
00151'006004          FETR
00152'002402          FJMP@ S2
00153'000000  S1:     0
00154'000000  S2:     0
00155'000003  SP2:    3


              ;ENTRY FOR MEAN ELUTION VOLUME

00156'054700  XMEV:   STA 3 SAVE

              ;CALCULATE AREA UNDER EV*H CURVE

00157'006004          FETR
00160'004735          FJSR LOAD
00161'006007$         FJSR@ SIMP
00162'000173'         OP4
00163'040002-         FSTA 0 MEAN      ;MEAN ELUTION VOLUME
00164'100000          FEXT

              ;PRINT MEAN ELUTION VOLUME

00165'006010$         JSR@ PRINT
00166'000001          1
00167'000001$         CM01
00170'000002-         MEAN
00171'006005$         JSR@ CRLF
00172'002664          JMP@ SAVE

              ;EV(I) * H(I)

00173'070760  OP4:    FST3 S1
00174'024724          FLDA 1 XV
00175'120100          FMPY 1 0
00176'004712          FJSR DINCR
00177'002754          FJMP@ S1
              .END
```

**Program** Print

**Synopsis** This routine prints n lables and n floating point numbers

**Method**

1. Assume byte pointers and floating point numbers are in order
2. Output message, floating point number and carriage return line feed
3. Repeat n times

```
;***********************************************

;PRINT

;REVISED AUG 15, 1971

;***********************************************

;CALL SEQUENCE  -        JSR@ PRINT
;                        NUMBER OF OUTPUTS
;                        ADDRS OF BYTE POINTERS
;                        ADDRS OF FLOATING POINT DATA
;                        RETN


                .TITL   PRINT
                .ENT    PRINT
                .EXTD   CRLF TYPE

                .ZREL
00000-000000'   PRINT:  XPRINT

                .NREL
00000'054425    XPRINT: STA 3 S1        ;RETN
00001'025400            LDA 1 Ø 3
00002'044425            STA 1 COUNT     ;NUM OF OUTPUTS
00003'021401            LDA Ø 1 3
00004'040424            STA Ø MESS      ;ADDR OF MESS
00005'021402            LDA Ø 2 3
00006'040420            STA Ø NUM       ;ADDRS OF FLOATING POINT DATA

00007'022421    LOOP:   LDA@ Ø MESS     ;BYTE
00010'006002$           JSR@ TYPE       ;LABEL

00011'006004            FETR
00012'022414            FLDA@ Ø NUM     ;NUMBER
00013'140000            FFDC Ø          ;OUTPUT
00014'100000            FEXT

00015'006001$           JSR@ CRLF
00016'010412            ISZ MESS
00017'010407            ISZ NUM
00020'010406            ISZ NUM
00021'014406            DSZ COUNT       ;THROUGH
00022'000765            JMP LOOP        ;NO
00023'034402            LDA 3 S1
00024'001403            JMP 3 3         ;RETN

00025'000000    S1:     Ø
00026'000000    NUM:    Ø
00027'000000    COUNT:  Ø
00030'000000    MESS:   Ø

                .END
```

**Program** Floating Point Exponential Routine

**Synopsis** This routine computes the exponential of X.

**Method**

1. Determine sign of X and flag
2. Set z = ABS(X)
3. Compute

$$y = \frac{z}{2^m}$$

   where $y \leq 2.0$

4. Sum the exponential series

$$\exp(y) = 1 + (y) \frac{(y)^2}{2!} + \frac{(y)^3}{3!} + \frac{(y)^n}{n!}$$

   until

$$\frac{(y)^n}{n!} \leq 1.0 \times 10^{-5}$$

5. Compute exp z, where

$$\exp(z) = [\exp(y)]^{2^m}$$

6. Invert exp(z) if X was negative
7. Return

```
                        ;*********************************************
                        ;FLOATING POINT EXPONENTIAL ROUTINE
                        ;REVISED NOV 1, 1971
                        ;*********************************************
                        ;CALL SEQUENCE  -        JSR@ EXP
                        ;                        RETURN
                        ;INPUT  - FAC1 CONTAINS X
                        ;OUTPUT - FAC1 CONTAINS EXP(X)

                        .TITL   EXP1
                        .ENT    EXP

                        .ZREL
00000-000000'  EXP:     XEXP

                        .NREL
00000'100000   XEXP:    FEXT
00001'054465            STA 3 SAVE      ;RETN
00002'102520            SUBZL 0 0       ;AC0=1
00003'040464            STA 0 N         ;STORE
00004'040461            STA 0 SIGN      ;SIGN=1
00005'006004            FETR
00006'125005            FMOV 1 1 FSGE   ;X>=0.0
00007'014456            FDSZ SIGN       ;NO, SIGN=0
00010'000401            FJMP .+1
00011'125400            FPOS 1 1        ;ABS(X)
00012'034460            FLDA 3 F2

00013'136412   TEST:    FSUB# 1 3 FSLT  ;FAC1<2
00014'000410            FJMP EXP1       ;FAC2<2

00015'164200   XDIV:    FDIV 3 1        ;FAC1/2.0
00016'100000            FEXT
00017'020450            LDA 0 N
00020'101120            MOVZL 0 0
00021'040446            STA 0 N         ;N*2
00022'006004            FETR
00023'000770            FJMP TEST       ;TRY AGAIN

00024'121000   EXP1:    FMOV 1 0
00025'024443            FLDA 1 F1
00026'107000            FADD 0 1        ;1+X
00027'034443            FLDA 3 F2       ;2
00030'054444            FSTA 3 NUM
00031'040445            FSTA 0 TERM     ;SAVE X/1

00032'030444   LOOP:    FLDA 2 TERM
00033'110100            FMPY 0 2        ;X*TERM
00034'170200            FDIV 3 2        ;TERM*X/N
00035'147000            FADD 2 1        ;SUM+NEW TERM
00036'034442            FLDA 3 TOL
00037'050437            FSTA 2 TERM     ;SAVE TERM
00040'151400            FPOS 2 2        ;ABS TERM
00041'156416            FSUB# 2 3 FSLE  ;ABS(TERM)<= TOL
00042'000406            FJMP SHIFT
```

```
00043'034431            FLDA 3 NUM      ;NO
00044'030424            FLDA 2 F1
00045'157000            FADD 2 3        ;NUM + 1.0
00046'054426            FSTA 3 NUM
00047'000763            FJMP LOOP

00050'121000  SHIFT:    FMOV 1 0        ;FAC1=FAC0
00051'014416  LOOP2:    FDSZ N          ;N-1
00052'000402            FJMP .+2
00053'000403            FJMP OUT
00054'104100            FMPY 0 1
00055'000774            FJMP LOOP2

00056'014407  OUT:      FDSZ SIGN       ;SKIP IF X WAS POSITIVE
00057'000402            FJMP NEG1
00060'002406            FJMP@ SAVE
00061'020407  NEG1:     FLDA 0 F1
00062'120200            FDIV 1 0
00063'105000            FMOV 0 1        ;FAC1=1.0/FAC1
00064'002402            FJMP@ SAVE

00065'000000  SIGN:     0
00066'000000  SAVE:     0
00067'000000  N:        0

00070'040420  F1:       1.0
00071'000000
00072'040440  F2:       2.0
00073'000000
00074'000000  NUM:      0.0
00075'000000
00076'000000  TERM:     0.0
00077'000000
00100'036247  TOL:      0.00001
00101'142654

              .END
```

**Program** Three Parameter Calibration Curve

**Synopsis** This routine computes the molecular weight and the slope of a three parameter molecular calibration curve at a given retention volume. It also includes a routine to input and output the calibration curve parameters $D_1$, $D_2$, $D_3$.

**Method**

1. Compute

$$M = D_1 \exp(-D_2 v - D_3 v^2)$$

2. Compute

$$\frac{dM}{dv} = D_1 \exp(-D_2 v - D_3 v^3)(-D_2 - 2D_3 v)$$

3. Return

```
;*******************************************

;THREE PARAMETER CALIBRATION CURVE

;REVISED NOV 1, 1971

;*********************************************

;FORM    -         MW = D1*EXP(-(D2*V+D3*V**2))


;CALL SEQUENCE TO INPUT PARAMETERS
;                  JSR@ .IMWCC
;                  ADDRESS FOR STORAGE
;                  RETN

;CALL SEQUENCE FOR OUTPUT OF PARAMETERS
;                  JSR@ .DPRI
;                  ADDRESS OF PARAMETERS
;                  ;RETN

;CALL SEQUENCE FOR MOLECULATR WEIGHT AND SLOPE
;                  JSR@ .MWCC
;                  ADDRESS PF PARAMETERS
;                  RETN


                  .TITL   MWCC
                  .ENT    .D1 D2 D1 .IMWCC .MWCC .DPRI
                  .EXTD PRINT TYPE EXP

                  .ZREL
00000-000012-     .IMWCC: IMWCC
00001-000051-     .MWCC:  MWCC
00002-000040-     .DPRI:  DPRI
00003-000004-     .D1:    D1
00004-000000      D1:     0.0
00005-000000
00006-000000      D2:     0.0
00007-000000
00010-000000      D3:     0.0
00011-000000

                  ;INPUT PARAMETERS

00012-054100-     IMWCC:  STA 3 SAVE
00013-025400              LDA 1 0 3          ;STORAGE LOC
00014-044101-             STA 1 SLOPE
00015-020103-             LDA 0 MA01
00016-004025-             JSR IN
00017-020104-             LDA 0 MA02
00020-004025-             JSR IN
00021-020105-             LDA 0 MA03
00022-004025-             JSR IN
00023-034100-             LDA 3 SAVE
00024-001401              JMP 1 3

00025-054036-     IN:     STA 3 S2
00026-006002$             JSR@ TYPE          ;"MESS "
00027-006004              FETR
```

```
00030-120000            FDFC 0
00031-042101-           FSTA 0 @SLOPE     ;STORE
00032-100000            FEXT
00033-010101-           ISZ SLOPE
00034-010101-           ISZ SLOPE
00035-002036-           JMP@ S2

00036-000000  S2:       0.0
00037-000000

              ;OUTPUT PARAMETERS

00040-054100- DPRI:     STA 3 SAVE
00041-035400            LDA 3 0 3
00042-054046-           STA 3 TEMP
00043-006001$           JSR@ PRINT
00044-000003            3
00045-000103-           MA01
00046-000000  TEMP:     0

00047-034100-           LDA 3 SAVE
00050-001401            JMP 1 3

              ;CALCULATE MW (FAC1) AND SLOPE (FAC2) FOR V (FAC1)

00051-070100- MWCC:     FST3 SAVE
00052-065400            FLD3 0 3          ;ADDR OF PAR
00053-040076-           FSTA 0 FAC0
00054-070036-           FST3 S2
00055-031404            FLDA 2 4 3
00056-130100            FMPY 1 2
00057-035402            FLDA 3 2 3
00060-157000            FADD 2 3
00061-173000            FADD 3 2
00062-050101-           FSTA 2 SLOPE
00063-164100            FMPY 3 1
00064-006003$           FJSR@ EXP
00065-064036-           FLD3 S2
00066-021400            FLDA 0 0 3        ;D1
00067-120200            FDIV 1 0
00070-105000            FMOV 0 1
00071-030101-           FLDA 2 SLOPE
00072-130100            FMPY 1 2
00073-020076-           FLDA 0 FAC0
00074-064100-           FLD3 SAVE
00075-001401            FJMP 1 3          ;RETN

00076-000000  FAC0:     0.0
00077-000000
00100-000000  SAVE:     0
00101-000000  SLOPE:    0.0
00102-000000

00103-000214= MA01:     A01*2
00104-000222= MA02:     A02*2
00105-000230= MA03:     A03*2

              A01:      .TXT "D1= "
```

```
00106-030504
00107-020075
00110-000000
                A02:     .TXT "D2= "
00111-031104
00112-020075
00113-000000
                A03:     .TXT "D3= "
00114-031504
00115-020075
00116-000000
                .END
```

Program — Messages for Calculation Routines

Synopsis — This routine contains all the messages for the calculation routines

Method — Byte pointers stored on page-zero

```
;***************************************************
;MESSAGES FOR CALCULATIONAL ROUTINES
;REVISED SEPT 1, 1971
;*************************************************

.TITL   MESSCR
.ENT    CMØ1 CMØ2 CMØ3 CMØ4 CMØ5 CMØ6 CMØ7 CM25
.ENT    CM2Ø CM21 CM22 CM24

.ZREL
CMØ1:   MØ1*2
CMØ2:   MØ2*2
CMØ3:   MØ3*2
CMØ4:   MØ4*2
CMØ5:   MØ5*2
CMØ6:   MØ6*2
CMØ7:   MØ7*2
CM2Ø:   M2Ø*2
CM21:   M21*2
CM22:   M22*2
CM24:   M24*2
CM25:   M25*2

.NREL

MØ1:    .TXT "MEAN =  "
MØ2:    .TXT " Ø-TAPE, 1-MEM "
MØ3:    .TXT "SAMPLE NO. "
MØ4:    .TXT "BEGIN BASE "
MØ5:    .TXT "END BASE "
MØ6:    .TXT "BEGIN CALC "
MØ7:    .TXT "END CALC "
M2Ø:    .TXT "MN = "
M21:    .TXT "MW = "
M22:    .TXT "MZ = "
M24:    .TXT "PD = "
M25:    .TXT "AREA = "

.END
```

## A:2 Details of Program 2, Calibration Curve Search

Program 2 uses a golden section, single variable search to compute an effective linear molecular weight calibration curve. This program was written to operate off-line in a Nova 1200 with 4K words of memory.

### Application

Program 2 can be used to determine a molecular weight calibration curve when calibrating with broad standards, or, to produce a corrected differential molecular weight distribution using corrected molecular weight averages and uncorrected chromatogram heights.

### Details

Program 2 subroutines input processing parameters and raw data, interpolate to produce adjusted data, search for an effective calibration curve, and output the results.

The program begins in the parameter input and computation control routine CCS. The processing parameters required to produce interpolated data are requested and stored. The subroutines and methods used to input the chromatogram heights are identical to the methods used in Program 1 (see A:1-4). In addition, the interpolated heights are produced from the raw data by a method identical to that described for Program 1 (A:1-4)

The adjusted heights are used to compute a polydispersity with an effective value of $D_2'$ in the molecular weight calibration curve:

$$M = D_1' \exp(-D_2 v) \qquad \text{A:2:1}$$

A golden section subroutine, GOLD, searchs for a $D_2'$ which minimizes the function OB, where

$$OB = (P(t) - (P(D_2') \qquad \text{A:2:2}$$

The effective $D_2'$ is then used to compute $D_1'$ by requiring:

$$\bar{M}_w(t) = \bar{M}_w\ (D_1', D_2')$$

The final step is to use the effective calibration curve to characterize the chromatogram, once again using the methods and subroutines of Program 1 (A:1-4) and the effective calibration curve. The molecular weight averages are computed and outputted. In addition, Program 2 will compute and output the differential molecular weight distribution if requested.

The reader is referred to A:4 for the complete operating instructions for this program.

## Symbols

The following symbols are defined in addition to the symbols defined for Program 1, in A:1-5:

| | |
|---|---|
| MIND | minimum difference in objective function |
| MNT | true number average molecular weight |
| MWT | true weight average molecular weight |
| NL | maximum number of loops |
| PDT | true polydispersity |
| START | starting address of Program 2 |

## Program Listings

Program 2 requires the RST, PICK, XDEL, IOSER, INTP, BASE, NORM, SIMP, PRINT and MESSCR subroutines. In addition, the following subroutines are required.

| | |
|---|---|
| | Calibration Control, Program 2 |
| CCS | Calibration Curve Search Program |
| GOLD | Golden Section Search |
| IPCS | Input Parameter for Calibration Curve Search |
| MANDP | Molecular Weight |
| MWCL | Linear Calibration Curve |
| CSM | Messages for Calibration Curve Search |

Extended Floating Point Interpreter

<u>Program</u> Calculation Control - Program 2

Synopsis This routine controls data input and data reduction for the calibration curve search program, Program 2.

<u>Method</u>

1. Input processing parameters
2. Exit to data input and interpolation routines
3. Exit to golden section search routines, return with $D_2'$
4. Compute $D_1'$
5. Exit to characterization routines
6. Halt

```
                    ;*************************************************

                    ;PROGRAM 2, CALCULATION CONTROL

                    ;WRITTEN NOV 2, 1971

                    ;*************************************************

                    ;STARTING ADDRESS        -       START

                    .TITL   CCS
                    .ENT    PDT START NL MIND .THR RETN CMASK
                    .EXTD   D1 D2 DBIN .DPRI PDC CRLF TYPE .GOLD MOL
                    .EXTD   POLY SM01 SM02 SM03 SN SM04 SM05 CM03
                    .EXTD   SM06 SM07 SM08 SM09 SM10
                    .EXTN   MCAL

                    .ZREL
00000-000000        PDT:    0.0                 ;POLYDISPERSITY (TRUE)
00001-000000
00002-000000        NL:     0                   ;NUMBER OF LOOPS
00003-000000        MNT:    0.0
00004-000000
00005-000000        MWT:    0.0
00006-000000
00007-000000        XLOW:   0.0                 ;LOW GUESS
00010-000000
00011-000000        XHIGH:  0.0                 ;HIGH GUESS
00012-000000
00013-000000        MIND:   0.0                 ;MINIMUM DIFF
00014-000000
00015-000065'       .THR:   THR
00016-000000'       RETN:   START
00017-000000        CMASK:  0

                    .NREL
00000'063077        START:  HALT
00001'020535                LDA 0 .WA
00002'040007                STA 0 7
00003'006005                JSR@ 5              ;INITIALIZE FLOATING POINT
00004'006006$               JSR@ CRLF
00005'006006$               JSR@ CRLF
00006'006006$               JSR@ CRLF
00007'020523                LDA 0 POINT
00010'040521                STA 0 .POINT        ;POINTER TO STORAGE
00011'020013$               LDA 0 SM01
00012'006007$               JSR@ TYPE
00013'006006$               JSR@ CRLF
00014'006006$               JSR@ CRLF
00015'006006$               JSR@ CRLF

00016'020021$               LDA 0 CM03
00017'006007$               JSR@ TYPE           ;"SAM NO."
00020'006003$               JSR@ DBIN
00021'044016$               STA 1 SN

00022'020014$               LDA 0 SM02          ;"MN(T)"
00023'004474                JSR IN
00024'020015$               LDA 0 SM03
00025'004472                JSR IN              ;"MW(T)"
```

```
00026'020017$          LDA 0 SM04
00027'004470           JSR IN           ;"XLOW"
00030'020020$          LDA 0 SM05
00031'004466           JSR IN           ;XHIGH"
00032'020022$          LDA 0 SM06
00033'004464           JSR IN           ;MIN DIFF"
00034'020023$          LDA 0 SM07
00035'006007$          JSR @ TYPE
00036'006003$          JSR@ DBIN        ;NUMBER OF LOOPS
00037'044002-          STA 1 NL         ;STORE

00040'006004           FETR
00041'020003-          FLDA 0 MNT
00042'024005-          FLDA 1 MWT
00043'104200           FDIV 0 1
00044'044000-          FSTA 1 PDT       ;PD=MWT/MNT
00045'024467           FLDA 1 F1
00046'044001$          FSTA 1 D1
00047'100000           FEXT

00050'006463           JSR@ DATIN       ;INPUT DATA

                 ;READY FOR SEARCH
00051'020005$          LDA 0 PDC
00052'040406           STA 0 FU

00053'020024$          LDA 0 SM08
00054'006007$          JSR@ TYPE        ;"LOOP   DIFF"
00055'006006$          JSR @CRLF
00056'006006$          JSR@ CRLF

00057'006010$          JSR@ .GOLD
00060'000000     FU:   0
00061'000002-          NL
00062'000007-          XLOW
00063'000011-          XHIGH
00064'000000     TAB:  0

00065'006006$    THR:  JSR@ CRLF
00066'006006$          JSR@ CRLF
00067'030775           LDA 2 TAB        ;ADDRS OF RESULTS

                 ;CALCULATE D1

00070'006004           FETR
00071'020005-          FLDA 0 MWT
00072'024012$          FLDA 1 POLY      ;TEMP SECOND MOM STORAGE
00073'120200           FDIV 1 0         ;MWT/MW
00074'040001$          FSTA 0 D1
00075'100000           FEXT

                 ;OUTPUT RESULTS

00076'020025$          LDA 0 SM09
00077'006007$          JSR@ TYPE        ;"MW=D1*EXP(-D2*V)
00100'006006$          JSR@ CRLF
00101'006006$          JSR@ CRLF
00102'006004$          JSR@ .DPRI       ;OUTPUT D1,D2
00103'000001$          D1
```

```
                      ;CALCULATE MOLECULAR WEIGHTS

00104'006006$          JSR@ CRLF
00105'020026$          LDA 0 SM10
00106'006007$          JSR@ TYPE          ;"MW COM. W EFF. MW CURV"
00107'006006$          JSR@ CRLF
00110'006006$          JSR@ CRLF
00111'006011$          JSR@ MOL
00112'000001$          D1
00113'000000  DST:     0

00114'006006$          JSR@ CRLF
00115'006006$          JSR@ CRLF
00116'000662           JMP START          ;REPEAT

00117'054411  IN:      STA 3 S1
00120'006007$          JSR@ TYPE
00121'006004           FETR
00122'120000           FDFC 0
00123'042406           FSTA 0 @.POINT
00124'100000           FEXT
00125'010404           ISZ .POINT
00126'010403           ISZ .POINT
00127'002401           JMP@ S1

00130'000000  S1:      0
00131'000000  .POINT:  0
00132'000003- POINT:   MNT

00133'177777  DATIN:   MCAL
00134'040420  F1:      1.0
00135'000000
00136'000137' .WA:     W
      000170  W:       .BLK 120.
              .END
```

Program — Input

Synopsis — This routine is used to input data processing parameters, input raw data and control data interpolation.

Method

1. Input processing parameters
2. Exit to data input and interpolation routines
3. Print important processing parameters
4. Return.

```
                ;*************************************************

                ;INPUT

                ;REVISED NOV 2, 1971

                ;*************************************************

                ;CALL SEQUENCE  -        JSR .MCAL
                ;                        RETURN


                .TITL   IPCS
                .ENT    DT .MCAL SN BB BC EB EC MCAL
                .EXTD    FEED  TYPE DBIN  CRLF .BASE RST .IMEM .ITAPE
                .EXTD    BBH GETC NORM MOL DP DELX V1
                .EXTD    CM03 CM04 CM05 CM06 CM07 SM11
                .EXTD   PRINT BIND D1 .DPRI

                .ZREL
00000-000000    DT:     0
00001-000012'   .MCAL:  MCAL
00002-000004-   .MS:    BB
00003-000000    SN:     0                 ;SAM NO.
00004-000000    BB:     0                 ;BEG OF BASE
00005-000000    EB:     0                 ;END OF BASE
00006-000000    BC:     0                 ;BEG OF CAL
00007-000000    EC:     0                 ;END OF CALC
00010-000000    DIS:    0

                .NREL
                ;INPUT AND STORE SUBROUTINE

00000'054460    CP:     STA 3 C2
00001'023000            LDA@ 0 0 2
00002'006002$           JSR@ TYPE         ;"MESSAG"
00003'006003$           JSR@ DBIN
00004'046451            STA 1 @STO        ;STORE INPUT
00005'010450            ISZ STO           ;INC STORAGE
00006'151400            INC 2 2           ;INC POINTER
00007'014447            DSZ COUNT         ;THROUGH
00010'000771            JMP CP+1          ;NO
00011'002447            JMP@ C2


                ;ENTRY TO INPUT PARAMETERS

00012'054445    MCAL:   STA 3 C1
00013'006004$           JSR@ CRLF
00014'020002-           LDA 0 .MS
00015'040440            STA 0 STO         ;STORE STORAGE LOC
00016'024436            LDA 1 T5
00017'044437            STA 1 COUNT       ;NO. OF INPUTS
00020'030426            LDA 2 .XM1        ;BYTE POINTER ADDR
00021'004757            JSR CP          ;INPUT
00022'020010-           LDA 0 DIS
00023'040000-           STA 0 DT

00024'020435            LDA 0 T1000
00025'024003-           LDA 1 SN
```

```
00026'107000          ADD 0 1         ;AC1=STOP
00027'034010$         LDA 3 .ITAPE    ;0
00030'005400          JSR 0 3         ;EXIT TO APPR ROUTINE

00031'006006$ CAL:    JSR@ RST        ;INPUT DATA
00032'002425          JMP @C1         ;ERROR RETN
00033'006005$         JSR@ .BASE        ;FIX PAR
00034'006004$         JSR@ CRLF
00035'006004$         JSR@ CRLF

00036'006026$         JSR@ PRINT
00037'000002          2
00040'000021$         CM04
00041'000011$         BBH

00042'006013$         JSR@ NORM

00043'006004$         JSR@ CRLF
00044'006004$         JSR@ CRLF
00045'002412          JMP@ C1

00046'000047' .XM1:   XM1
00047'000021$ XM1:    CM04     ;BB
00050'000022$         CM05     ;EB
00051'000023$         CM06     ;BC
00052'000024$         CM07     ;EC
00053'000025$         SM11     ;DIST

00054'000005  T5:     5
00055'000000  STO:    0
00056'000000  COUNT:  0
00057'000000  C1:     0
00060'000000  C2:     0
00061'001750  T1000:  1000.
              .END
```

**Program** Polydispersity

**Synopsis** This routine is used to compute molecular weights. In addition it includes a routine to compute the absolute difference in the true polydispersity and the polydispersity calculated with $D_2'$.

**Method**

Molecular Weights

See subroutine Molecular Weights, Program 1

Polydispersity

1. Assume $D_1' = 1$
2. Compute zero moment
3. Compute second moment
4. Compute $P(D_2')$
5. Set FACØ equal to $abs(P(t) - P(D_2'))$
6. Return

```
;***********************************************

;POLYDISPERSITY

;REVISED NOV 1, 1971

;*******************************************

;TO CALCULATE MOLECULAR WEIGHTS
;CALL SEQUENCE  -          JSR@ MOL
;                          ADDRS OF D1,D2, ...
;                          DISTRIBUTION OUTPUT (1=YES)
;                          RETN


;TO CALCULATE ABS(PD(T)-PD)
;CALL SEQUENCE  -          FJSR@ PDC
;                          RETN


                 .TITL   MANDD
                 .ENT    POLY PDC MOL MN MW MZ
                 .EXTD   .D1 D1 D2 CM01 .MWCC SPACE
                 .EXTD   TYPE CRLF DELX SIMP PRINT
                 .EXTD   MIND .THR CNT BIND PDT CM20 CM21
                 .EXTD   SM12 CM22 V1
                 .EXTD   NL

                 .ZREL
00000-000000' MOL:      XMOL
00001-000176' PDC:      XPDC
00002-000000  MN:       0.0
00003-000000
00004-000000  MW:       0.0
00005-000000
00006-000000  MZ:       0.0
00007-000000
00010-000000  POLY:     0.0
00011-000000
00012-000000  MZ1:      0.0
00013-000000

                 .NREL
00000'054465  XMOL:     STA 3 SAVE
00001'021401            LDA 0 1 3
00002'040464            STA 0 DT

00003'006004            FETR


              ;ZERO MOMENT

00004'004531            FJSR LOAD       ;INITIALIZE FOR M=0
00005'006012$           FJSR@ SIMP
00006'000071'           OP1

00007'024460            FLDA 1 F1
00010'104200            FDIV 0 1        ;1/AREA
00011'044002-           FSTA 1 MN
```

```
;SECOND MOMENT

00012'004523          FJSR LOAD        ;INIT FOR M=2
00013'006012S         FJSR@ SIMP
00014'000100'         OP2
00015'040004-         FSTA 0 MW

;THIRD MOMENT

00016'004517          FJSR LOAD        ;INIT FOR M=3
00017'006012S         FJSR@ SIMP
00020'000107'         OP3
00021'024004-         FLDA 1 MW
00022'040012-         FSTA 0 ME1

00023'120200          FDIV 1 0
00024'040006-         FSTA 0 ME

;FOURTH MOMENT

00025'004510          FJSR LOAD
00026'006012S         FJSR@ SIMP
00027'000117'         OP4
00030'024012-         FLDA 1 ME1
00031'120200          FDIV 1 0         ;ME+1
00032'040012-         FSTA 0 ME1

00033'020002-         FLDA 0 MN
00034'024004-         FLDA 1 MW
00035'104200          FDIV 0 1
00036'044010-         FSTA 1 POLY

00037'100000          FEXT
00040'006010S         JSR@ CRLF
00041'006013S         JSR@ PRINT
00042'000004          4
00043'000021S         CM20
00044'000002-         MN
00045'006013S         JSR@ PRINT
00046'000001          1
00047'000023S         SM12
00050'000012-         ME1

00051'006010S         JSR@ CRLF
00052'006010S         JSR@ CRLF

;OUTPUT DISTRIBUTION IF SET

00053'020413  DIST:   LDA 0 DT
00054'101005          MOV 0 0 SNR
00055'000406          JMP OUT
00056'006004          FETR
00057'004456          FJSR LOAD
00060'006012S         FJSR@ SIMP
00061'000142'         OP1A
00062'100000          FEXT
00063'034402  OUT:    LDA 3 SAVE
```

```
00064'001402            JMP 2 3         ;RETN

00065'000000    SAVE:   0
00066'000000    DT:     0
00067'040420    F1:     1.0
00070'000000

                ;OPERATIONAL SUBROUTINES MUST NOT DESTROY FAC0

00071'070502    OP1:    FST3 S1
00072'024446            FLDA 1 XV
00073'006005$           FJSR@ .MWCC
00074'000002$           D1
00075'120200            FDIV 1 0        ;HEIGHT/MW
00076'004432            FJSR DINCR
00077'002474            FJMP@ S1

00100'070473    OP2:    FST3 S1
00101'024437            FLDA 1 XV
00102'006005$           FJSR@ .MWCC
00103'000002$           D1
00104'120100            FMPY 1 0
00105'004423            FJSR DINCR
00106'002465            FJMP@ S1

00107'070464    OP3:    FST3 S1
00110'024430            FLDA 1 XV
00111'006005$           FJSR@ .MWCC
00112'000002$           D1
00113'124100            FMPY 1 1
00114'120100            FMPY 1 0
00115'004413            FJSR DINCR
00116'002455            FJMP@ S1

00117'070454    OP4:    FST3 S1
00120'024420            FLDA 1 XV
00121'006005$           FJSR@ .MWCC
00122'000002$           D1
00123'120100            FMPY 1 0
00124'120100            FMPY 1 0
00125'120100            FMPY 1 0
00126'004402            FJSR DINCR
00127'002444            FJMP @S1

00130'024410    DINCR:  FLDA 1 XV
00131'030011$           FLDA 2 DELX
00132'147000            FADD 2 1
00133'044405            FSTA 1 XV
00134'001400            FJMP 0 3

00135'024025$   LOAD:   FLDA 1 V1
00136'044402            FSTA 1 XV
00137'001400            FJMP 0 3
00140'000000    XV:     0.0
00141'000000

00142'070431    OP1A:   FST3 S1
00143'024775            FLDA 1 XV
00144'144000            FFDC 1
```

```
00145'004420            FJSR SP
00146'006005$           FJSR@ .MWCC
00147'000002$           D1
00150'140000            FFDC 0
00151'004414            FJSR SP
00152'144000            FFDC 1
00153'004412            FJSR SP
00154'115000            FMOV 0 3
00155'154200            FDIV 2 3
00156'154000            FFDC 3
00157'120200            FDIV 1 0
00160'004750            FJSR DINCR
00161'100000            FEXT
00162'006010$           JSR@ CRLF
00163'006004            FETR
00164'002407            FJMP@ S1
00165'100000  SP:       FEXT
00166'054406            STA 3 S2
00167'024406            LDA 1 SP2
00170'006006$           JSR@ SPACE
00171'006004            FETR
00172'002402            FJMP@ S2
00173'000000  S1:       0
00174'000000  S2:       0
00175'000003  SP2:      3

              ;ENTRY TO COMPUTE ABS(PDT-PD)

00176'070667  XPDC:     FST3 SAVE
00177'040003$           FSTA 0 D2
00200'004735            FJSR LOAD
00201'006012$           FJSR @SIMP
00202'000071'           OP1
00203'040442            FSTA 0 HOLD
00204'004731            FJSR LOAD
00205'006012$           FJSR@ SIMP
00206'000100'           OP2
00207'040010-           FSTA 0 POLY       ;SAVE MWTEMP IN POLY
00210'030435            FLDA 2 HOLD
00211'140100            FMPY 2 0
00212'030020$           FLDA 2 PDT
00213'142400            FSUB 2 0
00214'101400            FPOS 0 0
00215'100000            FEXT
00216'020016$           LDA 0 CNT
00217'024026$           LDA 1 NL
00220'106400            SUB 0 1
00221'006017$           JSR@ BIND
00222'024422            LDA 1 T4
00223'006006$           JSR@ SPACE
00224'006004            FETR
00225'140000            FFDC 0
00226'100000            FEXT
00227'024415            LDA 1 T4
00230'006006$           JSR@ SPACE
00231'006004            FETR
00232'034003$           FLDA 3 D2
00233'154000            FFDC 3
00234'100000            FEXT
```

```
00235'006010$          JSR@ CRLF
00236'006004           FETR
00237'024014$          FLDA 1 MIND
00240'106411           FSUB# 0 1 FSGT
00241'002624           FJMP @SAVE
00242'100000           FEXT
00243'002015$          JMP@ .THR

00244'000004  T4:      4
00245'000000  HOLD:    0.0
00246'000000

              .END
```

```
;*************************************************

;GOLDEN SECTION SEARCH

;*************************************************

;FLOATING POINT SUBROUTINE TO MINIMIZE A SINGLE
;VARIABLE FUNCTION BY THE GOLDEN SECTION METHOD.
;THE SUBROUTINE REQUIRES A USER SUBROUTINE TO
;ACCEPT  X  IN FACØ AND TO RETURN  F(X) IN FACØ.
;CALLING SEQUENCE
;
;        JSR@ .GOLD
;        ADDRESS OF F(X) ROUTINE
;        ADDRESS OF NUMBER OF ITERATIONS (INTEGER)
;        ADDRESS OF XLOW
;        ADDRESS OF XHIGH
;        ADDRESS OF RESULT TABLE
;  THE SUBROUTINE RETURNS HERE.
;
;WRITTEN BY J.D.WRIGHT .  JUNE 1970
;REVISED BY G. WALTHER.   SEPT 1971
;

                .TITL   GOLDSEC
                .ENT    .GOLD CNT


                .ZREL
00000-000000'   .GOLD:  GOLD
00001-000000    CNT:    Ø


                .NREL
00000'054473    GOLD:   STA 3 TEMP      ;RETN
00001'021400            LDA Ø    Ø 3    ;ADDR OF F(X)
00002'040507            STA Ø FUNCX
00003'023401            LDA@ Ø 1 3      ;NUMBER OF LOOPS
00004'040001-           STA Ø CNT
00005'020467            LDA Ø .XLOW
00006'041404            STA Ø 4 3       ;TABLE OF RESULTS
00007'006004            FETR
00010'064463            FLD3 TEMP
00011'033402            FLDA    2,@2,3  ;XLOW
00012'027403            FLDA    1,@3,3  ;XHIGH
00013'050462            FSTA    2,XLOW
00014'044463            FSTA    1,XHIGH
00015'146400            FSUB    2,1     ;DELX
00016'124400            FNEG    1,1     ;-DELX
00017'030473            FLDA    2,GFAC
00020'144100            FMPY    2,1     ;REDUCE DELX
00021'044466            FSTA    1,DELX
00022'020455            FLDA    Ø,XHIGH
00023'024464            FLDA    1,DELX
00024'123000            FADD    1,Ø
00025'040454            FSTA    Ø,XF
00026'006463            FJSR    @FUNCX
00027'040454            FSTA    Ø,FXW
00030'024457    CYCLE:  FLDA    1,DELX
00031'125002            FMOV    1,1,FSLT
```

```
00032'000403            FJMP    .+3       ;POS
00033'020442            FLDA    0,XLOW    ;NEG
00034'000402            FJMP    .+2
00035'020442            FLDA    0,XHIGH ;POS
00036'122400            FSUB    1,0       ;NEW X
00037'040446            FSTA    0,NEWX
00040'006451            FJSR    @FUNCX    ;RETURN WITH NEW F(X) IN FAC0
00041'024442            FLDA    1,FXW     ;OLD F(X)
00042'030445            FLDA    2,DELX
00043'122412            FSUB#   1,0,FSLT;SKIP IF NEW F<OLD F
00044'000405            FJMP    .+5       ;OLD<NEW
00045'040436            FSTA    0,FXW     ;NEW<OLD
00046'020437            FLDA    0,NEWX
00047'040432            FSTA    0,XF
00050'150400            FNEG    2,2
00051'151002            FMOV    2,2,FSLT ;SIGN OF DELX?
00052'000405            FJMP    .+5
00053'024422            FLDA    1,XLOW    ;NEG
00054'146400            FSUB    2,1
00055'044422            FSTA    1,XHIGH
00056'000404            FJMP    .+4
00057'024420            FLDA    1,XHIGH ;POS
00060'146400            FSUB    2,1
00061'044414            FSTA    1,XLOW
00062'150400            FNEG    2,2
00063'024427            FLDA    1,GFAC
00064'144100            FMPY    2,1       ;REDUCE DELX
00065'044422            FSTA    1,DELX
00066'014001-           FDSZ    CNT       ;COUNT CYCLES
00067'000741            FJMP    CYCLE
00070'100000            FEXT
00071'034402            LDA 3 TEMP
00072'001405            JMP 5 3           ;RETN
00073'000000  TEMP:     0

              ;TABLE OF RESULTS
00074'000075' .XLOW:    XLOW      ;ADDR OF TABLE
00075'000000  XLOW:     0         ;LOW VALUE OF X
00076'000000            0
00077'000000  XHIGH:    0         ;HIGH VALUE
00100'000000            0
00101'000000  XF:       0         ;MIDDEL VALUE
00102'000000            0
00103'000000  FXW:      0         ;F(XF)

00104'000000            0
00105'000000  NEWX:     0
00106'000000            0
00107'000000  DELX:     0
00110'000000            0
00111'000000  FUNCX:    0
00112'040236  GFAC:     0.618035
00113'033612
                        .END
```

**Program** Linear Calibration Curve

**Synopsis** This routine computes M and the slope at a particular retention volume for a given linear calibration curve.

**Method**

1. Compute

$$M = D_1 \exp(-D_2 v)$$

2. Return

**Note** - This routine includes a calibration curve constant output routine

```
;*********************************************

;LINEAR CALIBRATION CURVE

;REVISED NOV 1, 1971

;*********************************************


;FORM    - MW = D1*EXP(-D2*V)

;TO OUTPUT CALIBRATION CURVE PARAMETERS
;CALL SEQUENCE  -         JSR@ .DPRI
;                         ADDRESS OF PARAMETERS
;                         RETURN

;TO CALCULATE MOLECULAR WEIGHT AND SLOPE
;CALL SEQUENCE  -         JSR@ .MWCC
;                         ADDRESS OF CALIBRATION CURVE PA
;                         RETURN
;INPUT   - FAC1 CONTAINS V
;OUTPUT  - FAC1 CONTAINS MOLECULAR WEIGHT
;        - FAC2 CONTAINS SLOPE OF CALIBRATION CURVE

;NOTE    - REQUIRES EXTENDED FLOATING POINT

        .TITL   MWCC
        .ENT    .D1 D2 D1  .MWCC .DPRI
        .EXTD   PRINT TYPE EXP

        .ZREL
00000-000020- .MWCC:   MWCC
00001-000007- .DPRI:   DPRI
00002-000003- .D1:     D1
00003-000000  D1:      0.0
00004-000000
00005-000000  D2:      0.0
00006-000000



              ;OUTPUT PARAMETERS

00007-054040- DPRI:    STA 3 SAVE
00010-035400           LDA 3 0 3
00011-054015-          STA 3 TEMP
00012-006001$          JSR@ PRINT
00013-000002           2
00014-000042-          MA01
00015-000000  TEMP:    0

00016-034040-          LDA 3 SAVE
00017-001401           JMP 1 3

              ;CALCULATE MW (FAC1) AND SLOPE (FAC2) FOR V (FAC1)

00020-070040- MWCC:    FST3 SAVE
00021-065400           FLD3 0 3       ;ADDR OF PAR
00022-040036-          FSTA 0 FAC0
```

```
00023-021402            FLDA 0 2 3      ;D2
00024-104100            FMPY 0 1        ;D2*V
00025-124220            FEXP 1 1
00026-021400            FLDA 0 0 3      ;D1
00027-120200            FDIV 1 0
00030-105000            FMOV 0 1
00031-031402            FLDA 2 2 3      ;D2
00032-130100            FMPY 1 2
00033-020036-           FLDA 0 FAC0
00034-064040-           FLD3 SAVE
00035-001401            FJMP 1 3        ;RETN

00036-000000  FAC0:     0.0
00037-000000
00040-000000  SAVE:     0
00041-000000  S2:       0

00042-000110= MA01:     A01*2
00043-000116= MA02:     A02*2

              A01:      .TXT "D1= "
00044-030504
00045-020075
00046-000000
              A02:      .TXT "D2= "
00047-031104
00050-020075
00051-000000
              .END
```

<u>Program</u> Special Messages for Calibration Curve Search Program

<u>Synopsis</u> This routine includes special messages required for the calibration curve search program

<u>Method</u> 1. Byte routines are stored on page zero

```
;*********************************************

;SPECIAL MESSAGES FOR CALIBRATION CURVE SEARCH
;REVISED NOV 1, 1971

;*********************************************

.TITL   CSM
.ENT    SM01 SM02 SM03 SM04 SM05 SM06 SM07
.ENT    SM08 SM09 SM10 SM11 SM12


.ZREL
SM01:   M01*2
SM02:   M02*2
SM03:   M03*2
SM04:   M04*2
SM05:   M05*2
SM06:   M06*2
SM07:   M07*2
SM08:   M08*2
SM09:   M09*2
SM10:   M10*2
SM11:   M11*2
SM12:   M12*2

.NREL
M01:    .TXT "CALIBRATION CURVE SEARCH "
M02:    .TXT "MN(T) = "
M03:    .TXT "MW(T) = "
M04:    .TXT "XLOW = "
M05:    .TXT "XHIGH = "
M06:    .TXT "MIN DIFF ="
M07:    .TXT "MAX NO. OF LOOPS "
M08:    .TXT "LOOP          PDT-PD              D2"
M09:    .TXT "EFFECTIVE LINEAR CALIBRATION CURVE CONSTANTS "
M10:    .TXT "MOLECULR WEIGHTS COMPUTED WITH D1, D2"
M11:    .TXT "OUTPUT DISTRIBUTION "
M12:    .TXT "MZ+1="

.END
```

## A:3 Details of Program 3, Axial Dispersion Calibration Program for Standards

Program 3 is used to characterize the GPC chromatogram of standards by computing the spreading parameters h and Sk. h and Sk are computed from the true molecular weight averages $\bar{M}_n(t)$ and $\bar{M}_w(t)$ and the infinite resolution values $\bar{M}_n(\infty)$, and $\bar{M}_w(\infty)$, where:

$$h = \frac{D_2^2}{2}\left[\frac{P(\infty)}{P(\infty) - P(t)}\right] \qquad \text{A:3:1}$$

$$Sk = \frac{M_w(t)}{M_w(\infty)} - \frac{M_n(t)}{M_n(\infty)} - \left[\exp\left(\frac{D_2^2}{4h}\right) + \exp\left(\frac{-D_2^2}{4h}\right)\right] \qquad \text{A:3:2}$$

### Details

Program 3 uses the identical subroutines and methods outlined in Program 1, to input raw data, interpolate to produce the 59 adjusted heights, and characterize the chromatogram. In addition a subroutine CALS, is used to input $\bar{M}_n(t)$, $\bar{M}_w(t)$, and a subroutine SPREAD, is used to compute the spreading parameters h and Sk directly from A:3:1 and A:3:2.

### Program Listings

Program 3 includes the RST, PICK, XPEL, IOSER, INTP, BASE, NORM, SIMP, PRINT, MESSCR, MOL, MWCC of A:1-7. In addition, the following subroutines are required:

| | |
|---|---|
| CALS | Calculation Control - Program 3 |
| MESSP3 | Special Messages for Program 3 |
| SPREAD | Spreading Parameters |

Extended Floating Point Interpreter

Program Calculation Control - Program 3

Synopsis This routine is used to control Program 3, Axial Dispersion Calibration for Standards.

Methods

1. Initialize floating point package
2. Input processing parameters
3. Exit to data input and interpolation routines
4. Exit to chromatogram characterization routines
5. Exit to spreading parameter calculation routine
6. Halt

```
              ;**********************************************
              ;CALCULATION CONTROL - PROGRAM 3
              ;REVISED NOV 2, 1971
              ;*******************************************
              ;STARTING ADDRESS        -        START

              .TITL   CALS
              .ENT    START MNT MWT .MCAL SN BB BC EB EC MCAL
              .EXTD   CM35 CM36  TYPE DBIN MEV
              .EXTD   CRLF .BASE RST .IMEM .ITAPE
              .EXTD   BBH .IMWCC  GETC NORM MOL
              .EXTD   CM30 CM03 CM04 CM05 CM06 CM07 CM44
              .EXTD   PRINT  BIND D1 .DPRI SPREAD CM40

              .ZREL
00000-000000  MNT:    0.0
00001-000000
00002-000000  MWT:    0.0
00003-000000
00004-000222' .MCAL:  MCAL
00005-000006- .MS:    SN
00006-000000  SN:     0         ;SAM NO.
00007-000000  BB:     0         ;BEG OF BASE
00010-000000  EB:     0         ;END OF BASE
00011-000000  BC:     0         ;BEG OF CAL
00012-000000  EC:     0         ;END OF CALC
00013-000000  DIS:    0

              .NREL

              ;INITIALIZE CALCULATION

00000'063077  START:  HALT
00001'006006$         JSR@ CRLF
00002'006006$         JSR@ CRLF
00003'020015$         LDA 0 GETC
00004'040040          STA 0 40
00005'020034$         LDA 0 CM40
00006'006003$         JSR@ TYPE         ;"PROGRAM 3"
00007'006006$         JSR@ CRLF
00010'006006$         JSR@ CRLF

00011'020406          LDA 0 .WA
00012'040007          STA 0 7           ;STORAGE FOR FLOATING POINT
00013'006005          JSR@ 5            ;INITIALIZE
00014'006004-         JSR@ .MCAL        ;DO CALCULATION
00015'063077          HALT
00016'000763          JMP START+1


00017'000020' .WA:    WA
      000170  WA:     .BLK 120.         ;STORAGE FOR FLOATING POINT INTP


              ;INPUT AND STORE SUBROUTINE
```

```
00210'054515  CP:     STA 3 C2
00211'023000          LDA@ 0 0 2
00212'006003$         JSR@ TYPE       ;"MESSAG"
00213'006004$         JSR@ DBIN
00214'046506          STA 1 @STO      ;STORE INPUT
00215'010505          ISZ STO         ;INC STORAGE
00216'151400          INC 2 2         ;INC POINTER
00217'014504          DSZ COUNT       ;THROUGH
00220'000771          JMP CP+1        ;NO
00221'002504          JMP@ C2


00222'054502  MCAL:   STA 3 C1
00223'006006$         JSR@ CRLF

00224'020005-         LDA 0 .MS
00225'040475          STA 0 STO       ;STORE STORAGE LOC
00226'024473          LDA 1 T6
00227'044474          STA 1 COUNT     ;NO. OF INPUTS
00230'030461          LDA 2 .XM1      ;BYTE POINTER ADDR
00231'004757          JSR CP          ;INPUT
00232'020013-         LDA 0 DIS
00233'040447          STA 0 DT

00234'006014$         JSR@ .IMWCC    ;INPUT CAL CURV PAR
00235'000031$         D1

00236'020001$         LDA 0 CM35
00237'006003$         JSR@ TYPE       ;"MNT"
00240'006004          FETR
00241'120000          FDFC 0
00242'040000-         FSTA 0 MNT
00243'100000          FEXT
00244'020002$         LDA 0 CM36
00245'006003$         JSR@ TYPE       ;"MWT"
00246'006004          FETR
00247'120000          FDFC 0
00250'040002-         FSTA 0 MWT
00251'100000          FEXT
00252'020454          LDA 0 T1000
00253'024006-         LDA 1 SN
00254'107000          ADD 0 1         ;AC1=STOP
00255'034012$         LDA 3 .ITAPE
00256'005400          JSR 0 3         ;EXIT TO APPR ROUTINE


00257'006010$ CAL:    JSR@ RST        ;INPUT DATA
00260'002444          JMP @C1         ;ERROR RETN
00261'006006$         JSR@ CRLF
00262'006006$         JSR@ CRLF
00263'006007$         JSR@ .BASE      ;SUB BASE LINE
00264'006006$         JSR@ CRLF
00265'006006$         JSR@ CRLF

00266'006032$         JSR@ .DPRI
00267'000031$         D1

00270'006027$         JSR@ PRINT
00271'000002          2
```

```
---
00272'000022$         CM04
00273'000013$         BBH

00274'006016$         JSR@ NORM

00275'006005$         JSR@ MEV          ;EXIT TO ROUTINE


00276'006006$         JSR@ CRLF
00277'006006$         JSR@ CRLF
00300'006017$         JSR@ MOL          ;CAL MOL WTS
00301'000031$         D1
00302'000000  DT:     0
00303'006006$         JSR@ CRLF
00304'006033$         JSR@ SPREAD
00305'006006$         JSR@ CRLF
00306'006006$         JSR@ CRLF
00307'006006$         JSR@ CRLF
00310'002414          JMP@ C1

00311'000313' .XM1:   XM1
00312'000314' .XM2:   XM2
00313'000021$ XM1:    CM03     ;SAM NO
00314'000022$ XM2:    CM04     ;BB
00315'000023$         CM05     ;EB
00316'000024$         CM06     ;BC
00317'000025$         CM07     ;EC
00320'000026$         CM44     ;DIST

00321'000006  T6:     6
00322'000000  STO:    0
00323'000000  COUNT:  0
00324'000000  C1:     0
00325'000000  C2:     0
00326'001750  T1000:  1000.
      000000' .END START
```

Program H and Sk

Synopsis This routine computes the spreading parameter h and Sk.

Method

1.

$$h = \frac{D_2^2}{2}\left[\frac{P(\infty)}{P(\infty) - P(t)}\right]$$

$$Sk = \frac{\bar{M}_w(t)}{\bar{M}_w(\infty)} - \frac{\bar{M}_n(t)}{\bar{M}_n(\infty)} - \left[\exp\left(\frac{D_2^2}{4h}\right) + \exp\left(\frac{-D_2^2}{4h}\right)\right]$$

2. Output h, Sk

3. Return

```
                  ;*************************************************
                  ; H AND SK
                  ;REVISED NOV 1, 1971
                  ;*************************************************
                  ;CALL SEQUENCE  -        JSR@ SPREAD
                  ;                        RETURN
                  ;NOTE    - REQUIRES EXTENDED FLOATING POINT PACKAGE
                  .TITL    SPREAD
                  .ENT     SPREAD H SK
                  .EXTD    .MWCC MEAN CM40 CM42 PRINT MN
                  .EXTD     MNT MW MWT CRLF D1

                  .ZREL
00000-000000'     SPREAD:  XSPRE
00001-000000      H:       0.0
00002-000000
00003-000000      SK:      0.0
00004-000000

                  .NREL
00000'054457      XSPRE:   STA 3 SAVE
00001'006004               FETR
00002'024002$              FLDA 1 MEAN
00003'006001$              FJSR@ .MWCC       ;COMPUTE SLOPE AT MEAN
00004'000013$              D1
00005'130200               FDIV 1 2          ;SLOPE/MW
00006'050460               FSTA 2 D2LOC
00007'150100               FMPY 2 2          ;D2LOC*D2LOC
00010'020452               FLDA 0 F4
00011'110200               FDIV 0 2
00012'050452               FSTA 2 A          ;D2**2/4.0

                  ;COMPUTE H

00013'020006$              FLDA 0 MN
00014'024010$              FLDA 1 MW
00015'030007$              FLDA 2 MNT
00016'034011$              FLDA 3 MWT
00017'104200               FDIV 0 1
00020'154200               FDIV 2 3
00021'121000               FMOV 1 0
00022'162400               FSUB 3 0          ;PD-PDT
00023'104200               FDIV 0 1          ;PD/(PD-PDT)
00024'020440               FLDA 0 A
00025'104100               FMPY 0 1          ;A*PD/(PD-PDT)
00026'030432               FLDA 2 F2
00027'144100               FMPY 2 1          ;H
00030'044001-              FSTA 1 H

                  ;COMPUTE SK
```

```
---
00031'120200            FDIV 1 0          ;A/H
00032'104220            FEXP 0 1
00033'100400            FNEG 0 0          ;-A/H
00034'100220            FEXP 0 0
00035'107000            FADD 0 1          ;EXP(A/H) + EXP(-A/H)

00036'020011$           FLDA 0 MWT
00037'030010$           FLDA 2 MW
00040'140200            FDIV 2 0          ;MWT/MW
00041'122400            FSUB 1 0
00042'030007$           FLDA 2 MNT
00043'034006$           FLDA 3 MN
00044'170200            FDIV 3 2
00045'143000            FADD 2 0          ;MWT/MW +MNT/MN -(E(A/H) + E(-AX

00046'040003-           FSTA 0 SK

00047'100000            FEXT

00050'006005$           JSR@ PRINT
00051'000002            2
00052'000004$           CM42
00053'000001-           H

00054'006012$           JSR@ CRLF
00055'006012$           JSR@ CRLF
00056'002401            JMP@ SAVE         ;RETURN


00057'000000   SAVE:    0
00060'040440   F2:      2.0
00061'000000
00062'040500   F4:      4.0
00063'000000
00064'000000   A:       0.0
00065'000000
00066'000000   D2LOC:   0.0       ;LOCAL D2
00067'000000

               .END
```

```
;**********************************************
;SPECIAL MESSAGES FOR PROGRAM 3
;REVISED NOV 2, 1971
;*********************************************
.TITL   MESSP3
.ENT    CM35 CM36 CM40 CM42 CM43 CM44


.ZREL
CM35:   M35*2
CM36:   M36*2
CM40:   M40*2
CM42:   M42*2
CM43:   M43*2
CM44:   M44*2

.NREL
M35:    .TXT "MN(TRUE) = "
M36:    .TXT "MW(TRUE) = "
M40:    .TXT "PROGRAM 3 "
M42:    .TXT "H = "
M43:    .TXT "SK = "
M44:    .TXT "OUTPUT DIST "

.END
R
```

A:5 The Contact Sense Device

Each of the eight contact sense lines will cause an interrupt when there is a change in state (i.e., open to closed or closed to open) in an external circuit. The eight contact sense inputs are wired as device 40. When an interrupt occurs, the change in state is determined with an INTA Ø, instruction. Bit's 1-15 of accumulator zero will contain the octal device number (4) plus the octal number of the contact sense line (0-7) which caused the interrupt. In addition bit Ø will be 1 if the change in state was from closed to open. For example, if the contents of accumulator zero were $10042_8$, contact sense line two caused the interrupt, because the external circuit to which it is connected, has closed.

The status of device 40 may also be monitored by testing the done flag. In this case the sense line which caused the done flag to be set is determined with a DIA Ø 40 instruction.

# OPERATING MANUAL FOR

# THE GPC MINICOMPUTER SYSTEM

The instructions contained in this manual are to be used with:

1. The Data Acquisition and Reduction Program

   - GPC.SV-ϕI

2. The Effective Calibration Curve Search Program

   - CCS.SV-ϕI

3. The Axial Dispersion Calibration Program for Standards

   - ADCS.SV-ϕI

# CONTENTS


| | | PAGE |
|---|---|---|
| A:4 | General | 1 |
| A:4-1 | Interfacing the Computer to the GPC | 2 |
| A:4-2 | Loading a Binary Program | 4 |
| A:4-3 | Data Coding | 5 |
| A:4-4 | Data Acquisition and Reduction Program GPC.SV | |
| | a) Operating Procedure | 6 |
| | b) Operating Restriction | 15 |
| | c) Input Errors | 16 |
| | d) Error Flags | 17 |
| | e) Processing Examples | 18 |
| A:4-5 | Effective Calibration Curve Search Program CCS.SV | |
| | a) Operating Procedure | 22 |
| | b) Processing Examples | 24 |
| A:4-6 | Axial Dispersion Calibration Program for Standards | |
| | a) Operating Procedure | 27 |
| | b) Processing Examples | 29 |

## A:4 General

The instructions contained in this manual outline how the programs GPC.SV, CCS.SV and ADCS.SV are used for simultaneous data acquisition and reduction from one gel permeation chromatograph. These programs will operate with the basic Data Acquisition and Reduction Minicomputer System, which includes a Nova 1200 CPU, 4k words of memory, an interface, and an ASR.33 teletype.

The Data Acquisition and Reduction Program (GPC.SV) is used to sample, store, output, and compute information from one GPC.

The Effective Calibration Curve Search Program (CCS.SV) and the Axial Dispersion Calibration Program for standards (ADCS.SV) are used for special purpose computations, and may only be used when the GPC minicomputer is not being used for on-line data acquisition and reduction.

The user must select one of the three programs and then load that program into the computer, following the directions in A:4-2. If the Data Acquisition and Reduction Program, GPC.SV is to be used for the first time, the GPC must first be interfaced to the computer as outlined in A:4-1.

A:4-1 Interfacing the Computer to a GPC

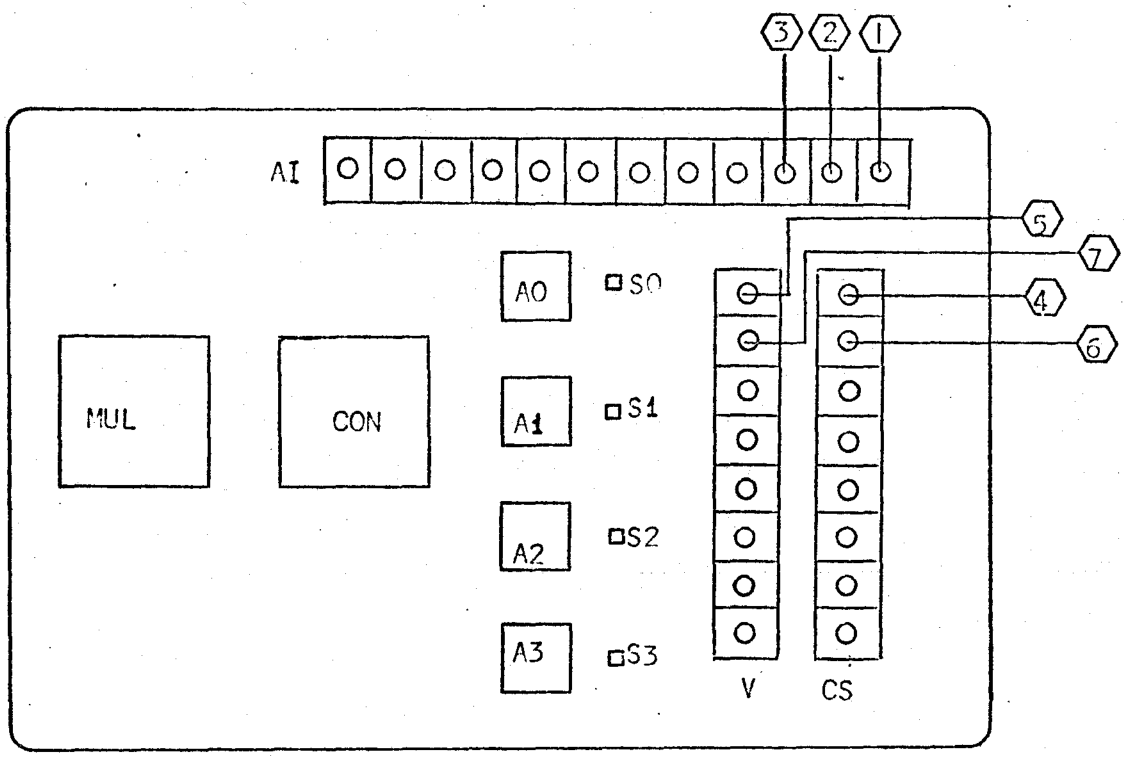
Refer to Figure 1.

1. Determine the maximum GPC detector output voltage and set analog input switch S0 in the interface to 10 millivolts or 100 millivolts accordingly.
(Warning, do not interface the minicomputer system to analog signals greater than 200 millivolts).

2. Connect analog input leads 1, 2, and 3 from the computer interface to the positive, negative, and ground inputs on the GPC's recorder respectively.

3. [For GPC's with retention volume syphon only]
Connect contact sense leads 4 and 5 from the interface to the retention volume dump relay in the GPC syphon circuit. In a Waters Model 200 GPC the syphon relay leads are located behind the main panel, terminals 17 and 18 on the main terminal strip. In a Waters liquid volume indicator,the relay terminal points are inside the cover, marked C1 and NO1.

4. [Optional, automatic sample recognition]
Connect leads 6 and 7 from the interface to the sample injection relay in the GPC. The sample injection relay is located behind the main panel on the GPC, and should be clearly marked. If leads 6 and 7 are not connected to the GPC sample injection relay, a sample injection is simulated at the teletype. (See A:4-4)




TOP VIEW

| GPC TIE POINTS | | CODE | |
|---|---|---|---|
| ①②③ | TO CHROMATOGRAM SIGNAL | A0, A1, A2, A3 | AMPLIFIERS |
| | | AI | ANALOG INPUTS |
| | | CON | A/D CONVERTOR |
| ④⑤ | TO RETENTION VOLUME RELAY | CS | CONTACT SENSE INPUT |
| | | MUL | MULTIPLEXER |
| ⑥⑦ | TO INJECTION RELAY | S0, S1, S2, S3 | ANALOG INPUT SWITCHES |
| | | V | + 5 VOLT |

FIGURE 1

## A:4-2 Loading a Binary Program

1. Plug in computer, teletype and interface.

2. Turn key on computer to "ON" position. Some of the lights on the computer console should go on.

3. Turn interface power supply switch on back of computer to "ON" position.

4. Turn switch on teletype to "LINE".

5. Enter starting address of binary loader by moving the data switches (switches on the computer console numbered 0-15) 4-15 inclusive to the up position.

6. Load paper tape program in the teletype reader. The tape should be loaded so that the direction of motion is towards the user. Check arrows on tape to ensure it is loaded properly.

7. Turn switch on paper tape reader to "START" position.

8. Press the RESET and START switches on the computer. Tape should read in.

Tape Does Not Read - Check to see if binary loader is in core. See Data General publication 091-00004-03 for details.

- Check teletype reader operation by turning teletype power switch to "local". Tape should advance through the reader.

Tape Stops Before Completion - Reload tape by starting at step 5. If tape fails a a second time, the program tape should be replaced.

## A:4-3 Data Coding

The data coding method used in the GPC minicomputer system is:

| Number | Meaning |
|---|---|
| 0 - 1023 | Chromatogram height |
| 1100 - 2000 | Sample numbers |
| 2000 - 3000 | Sample stop (1000 + sample no.) |
| 4000 | Injection marker |
| 7000 - 7100 | Retention volume dump (7000) plus clock reading (0-99) |

## A:4-4 Data Acquisition and Reduction Program - GPC.SV

The Data Acquisition and Reduction System includes an Operating System and several calculation routines (1). This system is used to collect, store, and output data on command. In addition, it will calculate and output:

1. Graph of the baseline corrected chromatogram
2. Area under chromatogram
3. Mean retention volume
4. Molecular weight averages, $\overline{M}_n$, $\overline{M}_w$, $\overline{M}_z$
5. Differential Molecular Weight Distribution.

### a) OPERATING PROCEDURE

The user controls data acquisition and reduction by typing in a series of operating commands. There are three basic types of commands, test commands, automatic data acquisition and reduction commands, and manual data acquisition and reduction commands. The purpose of each command is summarized in Table I.

The test commands are LOOK, ATD, and MON. These commands are used to test the system operation and monitor data storage.

The automatic data processing command DATA, implements automatic data acquisition and/or reduction. It is normally used when a series of similar polymers are to be characterized. When used to initiate automatic data reduction,the user must estimate,before the sample is injected, between which retention volumes the baseline is to be drawn and what are the limits for the calculation.

(1) W.G. Walther, M.Eng. Thesis, "New Aspects of Data Acquisition and Reduction in Gel Permeation Chromatography", McMaster University, (1972).

TABLE I

OPERATING COMMANDS

| COMMAND | PURPOSE |
| --- | --- |
| LOOK | Output current operating status |
| MONITOR | Output all information as it is collected |
| ATD | Test analog to digital channels. |
| BEGIN | Enter data collection parameters. |
| INJ | Note manual sample injection |
| STOP | Stop storing data |
| TYPE | Output data for a particular sample |
| CALC | Calculate and output Molecular Weight Averages and Molecular Weight Distribution. |
| DATA | Enter automatic collection and processing parameters |

Data may also be collected, outputted and reduced manually with the BEGIN, TYPE, INJ, CALC and STOP commands. These commands are used for acquisition and reduction when processing parameters cannot be estimated before a sample is injected. When using the manual commands, the GPC's strip chart may be used as a guide in determining processing parameters.

The instructions which follow, outline a procedure for loading, testing and operating the data acquisition and reduction system using these nine commands. When operating the system for the first time, it might be of some assistance to refer to the examples beginning on page 16.

In the instructions which follow all computer teletype output will be enclosed in quotation marks, and all user input will be underlined.

Loading the Data Acquisition and Reduction Program

1. Load the paper tape program GPC.SV-ϕ1 following the instructions in A:4-2.

Starting the Program

1. Enter the starting address (S.A.) of the GPC program in the data switches.
   - S.A. 2 (only data switch 14 up) Start with storage buffer clear.
   - S.A. 3 (switch 14 and 15 up) Start without storage clear.
2. Press the reset and start switches on computer. System should respond with "*". If not, reload GPC.SV tape as described in A:3-2.

## Issuing A Command

The system should now respond to the operating commands. The user issues a command by typing the command followed by an escape character. The escape key is located on the left hand side of the keyboard and is labeled ESC. The ESC key is used to signal the computer that a message is to be decoded. If a mistake is made when typing a command, issue a carriage return (RETURN) and repeat the command. If the system does not respond to any command, reload the program.

## Test Commands

System Operation: LOOK Command

1. Type LOOK (ESC). System should respond with:
   "GPC OFF AT EV 0 MONITOR OFF"

Analog to Digital Converter Test: ATD Command

1 Type ATD (ESC). System should respond with carriage return, line feed. Type any other key, but an escape. System will convert analog signal and type out decimal result. Repeat as many times as necessary. Return control to the system by typing an escape character.

Monitor: MON Command

1. Type MON (ESC). System should respond with:
   "MONITOR".

2. Type ϕ (zero) for off, 1 (one) for on, followed by a carriage return. If the monitor is turned on, any data stored in memory

will also be echoed on the teletype. The monitor command is used to follow the system performance. It does not have to be turned on for normal operation.

## Automatic Operation

1. Type DATA (ESC). Answer the following questions followed by carriage return. (Answer ϕ for NO, 1 for YES, where appropriate).

| Question | Explanation |
|---|---|
| "Auto Inj" | Should the computer expect a series of samples? |
| "Auto Type" | Following completion of a sample run should the data be typed out automatically? |
| "Auto Calc" | Following completion of a sample run should the computer compute molecular weight averages? |
| "Sample No." | First sample number? The sample number will be automatically incremented for the second sample. |
| "RV Sense" | Should the computer expect retention volume dumps? If set to NO (Zero), an artifical dump will be created by the computer every 60 seconds for reference purposes. |
| "Rate" | Chromatogram sampling rate (See operating restrictions)<br>1 = one sample every ten seconds<br>2 = one sample every twenty seconds<br>3 = one sample every thirty seconds |
| "On at RV" | Start storing heights after which retention volume? |
| "Off at RV" | Stop sampling at which retention volume? |

If Auto Calc is set to 1, answers to the following questions will be requested:

| Question | Explanation |
|---|---|
| "Begin Base" | Start baseline at which retention volume? |
| "End Base" | End base at which retention volume? |

| | |
|---|---|
| "Begin Calc" | Begin calculation at which retention volume? |
| "End Calc" | End calculation at which retention volume? |
| "$D_1$=" "$D_2$=" "$D_3$=" | Molecular weight calibration curve constants for three parameter calibration curve $M = D_1 \exp(-(D_2 V + D_3 V^2))$ Enter floating point numbers in E Format e.g. 1.065E-10 ($1.065 \times 10^{-10}$) -7.321E-10 ($-7.321 \times 10^{-10}$) |

2. The computer is now ready to sample and process data. Turn data switch ϕ on the computer to up position. The bell will ring every time a retention volume dump occurs. Inject a sample into the GPC immediately after the bell rings. If injection recognition lines are connected (See A:4-1) the system should respond with:
"INJECTION NOTED FOR SAMPLE XXXXX"
after a sample has been injected. If the injection lines are not connected to the GPC, the user must type INJ(ESC) to indicate to the computer that a sample has been injected.

3. The status of the system may be checked at any time by typing LOOK (ESC).

4. If the monitor is turned on at any time (MON (ESC)) all the subsequent data which is stored in memory will be echoed on the teletype.

5. When sampling is complete the system will respond with:
"STOP NOTED FOR SAMPLE XXXXX"
The auto-action routine will then initiate the action specified in the DATA command.

5. The raw data will reside in memory for approximately 2 hours of continuous operation. A second calculation can be performed with that data by typing the command CALC (see CALC command).

## Overlapping Samples

A second sample may be injected into the GPC at any time. A second DATA command is unneccessary, unless the user wishes to change one of the data collection or reduction parameters. The sample number will be automatically incremented by one for the second sample.

## Manual Operation

To Collect Data - The BEGIN Command

1. Type BEGIN (ESC). Answer the following questions followed by a carriage return. Answer ϕ for NO, 1 for YES where appropriate.

| Question | Explanation |
|---|---|
| "Sample No." | Sample number? |
| "RV Sense" | Should the computer recognize retention volume dumps? If set to zero, create a retention column dump artifically every minute. |
| "Rate" | Sampling rate (See operating restriction).<br>1 = one sample every 10 seconds<br>2 = one sample every 20 seconds<br>3 = one sample every 30 seconds |
| "On at RV" | Start storing samples of which retention volume? |
| "Off at RV" | Stop sampling at which retention volume? |

2. Turn data switch ϕ to the up position. Inject a sample into the GPC after the bell rings. If the injection recognition lines are connected the system will respond with

"INJECTION NOTED FOR SAMPLE XXXXX"

| | |
|---|---|
| "Begin Base" | Start baseline at which retention volume? |
| "End Base" | End baseline at which retention volume? |
| "Start CALC" | Start calculation at which retention volume? |
| "End CALC" | End calculation at which retention volume? |
| "$D_1$=" "$D_2$=" "$D_3$=" | Enter molecular weight calibration curve constants. The form of the equation is: $M = D_1 \exp(-(D_2 V + D_3 V^2))$ Constants are floating point numbers with E format e.g. 1.064 E-10 -7.251 E-11 |

2. If the data are stored on paper tape, load the tape at this point and set the reader switch to "START".

3. When the calculation is complete, control will return to the operating system.

To Manually Stop Data Collection - The STOP Command

1. Type STOP (ESC). This command will stop the sampling process for the first sample only. The stop routine is identical to the automatic stop of the DATA and BEGIN commands.

To Signify an Injection - INJ Command

If the injection recognition lines are not connected to the GPC relay, an injection must be relayed to the computer by typing INJ(ESC). The system should respond with:

"INJECTION NOTED FOR SAMPLE XXXX"

If the injection lines are not connected, the user must issue INJ(ESC) to signify to the computer that a sample has been injected.

To Type Data - The TYPE Command

1. When the sampling is complete the system will respond with:

"STOP NOTED FOR SAMPLE XXXXX"

Enter TYPE (ESC). The system will respond with:

"TURN ON TAPE"

Turn on the paper-tape punch and hit any key. The system will respond with:

"SAMPLE NO."

Enter sample number and a return. Raw data should be typed out. If the data is not found the system will respond with:

"DATA NOT FOUND"

"STORAGE DUMP"

Enter ϕ for NO storage dump, 1 for storage dump. If data is only partially lost, often a storage dump will provide significant data.

To Compute Molecular Weight Averages - The CALC Command

1. Type CALC (ESC). The user must provide processing information. Answer the following questions followed by a carriage return:

| Question | Explanation |
|---|---|
| "Sample No." | Sample Number? |
| "ϕ-tape, 1 mem" | Source of Data, paper tape or memory? |

## b) OPERATING RESTRICTIONS

1. Do not interrupt an auto-calculation to perform a second calculation until the auto-calculation is complete. All other user interrupts (teletype input) are permissable at any time.

2. Data collection is limited to 400 data points per sample, or 200 data points per sample when overlapping samples. The sampling rate ("RATE" in Begin and Data commands) must be set accordingly.

If data collection must exceed the above limits, the following procedure is recommended:

a) Turn monitor ON

b) Turn paper tape punch ON

c) Use BEGIN command

d) Inject sample

e) Use monitor output (on paper tape) as data for further processing

## c) INPUT ERRORS

Digit Error

1. If an error is made in a single digit and it is noticed before the number is complete, type rub out and repeat the entire number. The rub out re-initializes input for both floating point and fixed point inputs.

Number Error

2. If an error is made when entering a digit and it is not noticed before the number is completed, the entire command must be repeated. Type Control A (CTRL key and A key simultaneously). This returns control to the operating system.

Paper Tape Data Error

3. If paper tape is used as input, and the data is not found, the user must type rub out to return to the operating system.

## d) ERROR FLAGS

| Error | Cause | Action |
|---|---|---|
| HALT | Unknown Interrupt | Check Interface operation |
| 10 | Illegal Command | Type carriage return and repeat command. |
| 11 | Illegal Stop | Stop not proceeded by an injection. |
| 200 | Data not found in memory | Use TYPE Command to determine if data is there. |
| 201 | Data not complete | Check input for unrealistic processing parameters. |

## e) PROCESSING EXAMPLES

### TEST COMMANDS

TYPE THE COMMAND FOLLOWED BY AN ESCAPE

ATD

```
00000
00000
00000
00000
00000
00000
00000
```

LOOK

```
GPC   OFF     AT RV 00000   MONITOR OFF
```

MON

```
MONITOR      1
```

AUTO-PROCESSING

TYPICAL EXAMPLE OF AUTO-COLLECTION AND PROCESSING

DATA
AUTO INJ 1
AUTO TYPE 1
AUTO CALC 1

SAMPLE NO. 1300
RV SENSE 1
RATE 2
ON AT RV 20
OFF AT RV 40

BEGIN BASE 20
END BASE 40
BEGIN CALC 22
END CALC 38
D1 = 3.4567E9
D2 = 6.78E-1
D3 = 0.0

MANUAL PROCESSING

BEGIN

SAMPLE NO. 1300
RV SENSE 0
RATE 2
ON AT RV 20
OFF AT RV 40

INJ

INJECTION NOTED FOR 01300

STOP

STOP NOTED FOR 01300

TYPE
TURN ON TAPE
SAMPLE NO. 1300
DATA NOT FOUND
STORAGE DUMP 0

CALC

Ø-TAPE, 1-MEM Ø
SAMPLE NO. 13ØØ
BEGIN BASE 2Ø
END BASE 4Ø
BEGIN CALC 22
END CALC 38
D1= 3.5478E1Ø
D2= .78
D3= Ø.Ø

## A:4-5 Calibration Curve Search Program, CCS

The calibration curve search program CCS uses a golden section optimization algorithm to search for an effective linear calibration curve. It is used to calibrate with one or more broad standards or when a corrected differential distribution is required. Output includes:

1. Graph of Chromatogram
2. Area under baseline corrected chromatogram
3. $D_1'$, $D_2'$
4. $\overline{M}_n$, $\overline{M}_w$, $\overline{M}_z$, $\overline{M}_{z+1}$
5. Differential distribution.

### a) OPERATING PROCEDURE

1. Load paper tape program CCS.SV-ϕ1 as outlined in A:4-2.
2. Start the program in Location 2.
   (Move only data switch 14 to the UP position, press reset and start).
3. The system should respond with:
   "CALIBRATION CURVE SEARCH"
4. Answer the following questions followed by a carriage return

| Question | Explanation |
| --- | --- |
| "Sample No." | Sample Number? |
| "$M_n$(t)" | True number average molecular weight? |
| "$M_w$(t)" | True weight average molecular weight? |

| | |
|---|---|
| "XLOW" | Initial guess of minimum value for $D_2'$? |
| "XHIGH" | Initial guess of maximum value for $D_2'$? |
| "Min Diff" | Search should stop when the difference in the absolute values of the true polydispersity and the polydispersity computed with $D_2'$ is less than what number? |
| "Max.No.of Loops" | Maximum number of sub-sections of initial interval. |
| "Begin Base" | Start base line at which retention volume? |
| "End Base" | End base line at which retention volume? |
| "Start CALC" | Start calculation at which retention volume? |
| "End CALC" | End calculation at which retention volume? |
| "Output Distribution" | Output distribution, $\phi$ - NO, 1 - YES? |

5. Load the paper tape containing the raw data in the reader and move reader switch to "ON"

6. Data should read in and the calculation proceeds.

7. Program should halt after calculation is complete. Press continue to repeat this program.

b) PROCESSING EXAMPLES

CALIBRATION CURVE SEARCH

SAMPLE NO. 2006
MN(T) = 3600
MW(T) = 4000
XLOW = .3
XHIGH = .8
MIN DIFF =1.0E-4
MAX NO. OF LOOPS 40

BEGIN BASE 30
END BASE 37
BEGIN CALC 30
END CALC 35
OUTPUT DISTRIBUTION 0

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 02006 | 04000 | 07063 | 07057 | 07046 | 07032 | 07024 | 07020 |
| 07099 | 07089 | 07064 | 07052 | 07042 | 07018 | 07007 | 07084 |
| 07072 | 07048 | 07039 | 07025 | 07013 | 07092 | 07077 | 07058 |
| 07044 | 07029 | 07006 | 07089 | 07063 | 07051 | 07037 | 07021 |
| 00097 | 00097 | 00097 | 00097 | 00097 | 00097 | 00097 | 00097 |
| 00097 | 00097 | 00097 | 00097 | 00097 | 07008 | 00097 | 00097 |
| 00097 | 00097 | 00098 | 00098 | 00100 | 00101 | 00104 | 00108 |
| 00113 | 00122 | 07090 | 00131 | 00148 | 00168 | 00193 | 00224 |
| 00261 | 00304 | 00352 | 00405 | 00461 | 00517 | 00571 | 00622 |
| 07073 | 00665 | 00703 | 00729 | 00746 | 00750 | 00743 | 00726 |
| 00699 | 00665 | 00625 | 00581 | 00535 | 00488 | 07045 | 00442 |
| 00398 | 00357 | 00319 | 00285 | 00255 | 00229 | 00206 | 00187 |
| 00171 | 00157 | 00146 | 00136 | 07029 | 07009 | 07092 | |

```
+.3008332E+02 *
+.3016666E+02 *
+.3024999E+02 *
+.3033332E+02 *
+.3041664E+02 *
+.3049997E+02 *
+.3058330E+02 *
+.3066663E+02 *
+.3074996E+02 *
+.3083329E+02 *
+.3091661E+02 *
+.3099994E+02 *
+.3108327E+02 *
+.3116660E+02 *
+.3124993E+02 *
+.3133326E+02 *
+.3141658E+02 *
+.3149991E+02 *
+.3158324E+02 *
+.3166657E+02 *
+.3174990E+02 *
+.3183323E+02 *
+.3191655E+02  *
+.3199988E+02   *
```

```
+.3208321E+02     *
+.3216654E+02      *
+.3224986E+02        *
+.3233320E+02          *
+.3241652E+02            *
+.3249985E+02               *
+.3258318E+02                   *
+.3266651E+02                      *
+.3274983E+02                          *
+.3283317E+02                              *
+.3291649E+02                                 *
+.3299982E+02                                    *
+.3308315E+02                                       *
+.3316648E+02                                         *
+.3324980E+02                                          *
+.3333314E+02                                          *
+.3341646E+02                                          *
+.3349979E+02                                         *
+.3358312E+02                                       *
+.3366644E+02                                    *
+.3374977E+02                                  *
+.3383311E+02                               *
+.3391643E+02                            *
+.3399976E+02                         *
+.3408309E+02                      *
+.3416641E+02                   *
+.3424974E+02                *
+.3433307E+02              *
+.3441640E+02            *
+.3449973E+02          *
+.3458306E+02        *
+.3466638E+02       *
+.3474971E+02      *
+.3483304E+02     *
+.3491637E+02    *
```

BEGIN BASE +.9700000E+02
END BASE +.9800000E+02
AREA = +.9799509E+03

| LOOP | PDT-PD | D2 |
|---|---|---|
| 00000 | +.2413280E-01 | +.4909825E+00 |
| 00000 | +.2564719E-01 | +.6090174E+00 |
| 00001 | +.4875019E-01 | +.4180338E+00 |
| 00002 | +.6625258E-02 | +.5360683E+00 |
| 00003 | +.5116403E-02 | +.5639324E+00 |
| 00004 | +.1273054E-01 | +.5811533E+00 |
| 00005 | +.5467795E-03 | +.5532889E+00 |
| 00006 | +.2226110E-02 | +.5467112E+00 |
| 00007 | +.2280514E-02 | +.5573543E+00 |
| 00008 | +.5168989E-03 | +.5507764E+00 |
| 00009 | +.1171932E-02 | +.5492236E+00 |
| 00010 | +.1121796E-03 | +.5517361E+00 |
| 00011 | +.1394898E-03 | +.5523293E+00 |
| 00012 | +.2671704E-03 | +.5513695E+00 |
| 00013 | +.1563877E-04 | +.5519626E+00 |

EFFECTIVE LINEAR CALIBRATION CURVE CONSTANTS

D1= +.3834851E+12
D2= +.5519626E+00

MOLECULR WEIGHTS COMPUTED WITH D1, D2

MN = +.3600048E+04
MW = +.4000003E+04
MZ = +.4427371E+04
PD = +.4919238E+04
MZ+1=+.4873433E+04

## A:4-6 Axial Dispersion Calibration Program for Standards

The Axial Dispersion Calibration Program for Standards is used to compute values for the axial dispersion spreading parameters h and SK from the chromatograms of polymer standards. Output includes:

1. Graph of chromatogram
2. Mean retention volume
3. Area under baseline corrected chromatogram
4. $\overline{M}_n(\infty)$, $\overline{M}_w(\infty)$, $\overline{M}_z(\infty)$
5. h, SK

### a) OPERATING PROCEDURE

1. Load paper tape program ADCS.SV-ϕ1 as outlined in A:4-2.

2. Start the program in location 2.
(Move only data switch 14 to up position, then press RESET, START.

3. The system should respond with:
"PROGRAM 3"

4. Answer the following questions followed by a carriage return:

| Question | Explanation |
|---|---|
| "Sample No." | Sample number? |
| "$M_n(t)$" | True number average molecular weight? |
| "$M_w(t)$" | True weight average molecular weight? |
| "Begin Base" | Start baseline at which retention volume? |

"Start Calc" Start calculation at which retention volume?

"End Calc" End calculation at which retention volume?

5. Load the paper tape containing the raw data in the reader and move the reader switch to "START"

6. Data tape should read in and the calculation procede.

7. When the calculation is complete the CPU will HALT. Press continue to repeat the program.

PROGRAM 3

b) PROCESSING EXAMPLES

SAMPLE NO. 2008
BEGIN BASE 29
END BASE 35
BEGIN CALC 29
END CALC 35
OUTPUT DIST 1
D1 = .1566E6
D2 = -.638
D3 = .2253E-1
MN(TRUE) = 9700
MW(TRUE) = 10300

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 02008 | 04000 | 07031 | 07092 | 07052 | 07005 | 07080 | 07042 |
| 07008 | 07065 | 07017 | 07077 | 07033 | 07093 | 07044 | 07097 |
| 07060 | 07015 | 07077 | 07035 | 07095 | 07050 | 07008 | 07065 |
| 07022 | 07081 | 07029 | 07081 | 07039 | 07096 | 07052 | 00104 |
| 00104 | 00105 | 00105 | 00105 | 00105 | 00105 | 00105 | 00105 |
| 00105 | 00105 | 00105 | 00105 | 07009 | 00106 | 00107 | 00108 |
| 00111 | 00114 | 00119 | 00126 | 00136 | 00150 | 00169 | 00193 |
| 00224 | 07060 | 00263 | 00308 | 00361 | 00421 | 00486 | 00556 |
| 00626 | 00696 | 00762 | 00822 | 00874 | 00915 | 00945 | 07006 |
| 00962 | 00965 | 00956 | 00934 | 00902 | 00859 | 00810 | 00754 |
| 00696 | 00636 | 00577 | 00519 | 07051 | 00464 | 00414 | 00368 |
| 00328 | 00292 | 00261 | 00235 | 00213 | 00195 | 00179 | 00166 |
| 00156 | 07093 | 00145 | 00141 | 00135 | 00130 | 00127 | 00125 |
| 00122 | 00120 | 00118 | 00117 | 00116 | 00116 | 00115 | 07031 |

```
+.2909999E+02 *
+.2919999E+02 *
+.2929998E+02 *
+.2939997E+02 *
+.2949996E+02 *
+.2959995E+02 *
+.2969994E+02 *
+.2979993E+02 *
+.2989993E+02 *
+.2999991E+02 *
+.3009991E+02 *
+.3019990E+02 *
+.3029989E+02 *
+.3039988E+02 *
+.3049987E+02 *
+.3059986E+02  *
+.3069985E+02   *
+.3079984E+02     *
+.3089984E+02       *
+.3099982E+02         *
+.3109982E+02             *
+.3119981E+02                 *
+.3129979E+02                     *
+.3139979E+02                           *
+.3149978E+02                                *
+.3159977E+02                                      *
+.3169976E+02                                           *
+.3179975E+02                                               *
+.3189974E+02                                                  *
+.3199973E+02                                                     *
+.3209972E+02                                                      *
```

```
+.3219971E+02                                                       *
+.3229970E+02                                                     *           30.
+.3239970E+02                                                   *
+.3249968E+02                                               *
+.3259968E+02                                           *
+.3269967E+02                                       *
+.3279966E+02                                  *
+.3289965E+02                             *
+.3299964E+02                        *
+.3309963E+02                    *
+.3319962E+02                 *
+.3329961E+02             *
+.3339961E+02           *
+.3349959E+02         *
+.3359959E+02       *
+.3369958E+02     *
+.3379957E+02    *
+.3389956E+02   *
+.3399955E+02   *
+.3409954E+02  *
+.3419953E+02  *
+.3429952E+02 *
+.3439951E+02 *
+.3449950E+02 *
+.3459950E+02 *
+.3469948E+02 *
+.3479948E+02 *
+.3489947E+02 *
```

D1= +.1566000E+06
D2= -.6380000E+00
D3= +.2253000E-01
BEGIN BASE +.1040000E+03
END BASE +.1150000E+03
AREA = +.1390861E+04
MEAN = +.3222171E+02

MN = +.7798066E+04
MW = +.1043758E+05
MZ = +.1325281E+05
PD = +.1338483E+01

| | | | |
|---|---|---|---|
| +.2909999E+02 | +.0000000E+00 | +.9377263E+05 | +.0000000E+00 |
| +.2919998E+02 | +.0000000E+00 | +.8764805E+05 | +.0000000E+00 |
| +.2929997E+02 | +.0000000E+00 | +.8188662E+05 | +.0000000E+00 |
| +.2939996E+02 | +.0000000E+00 | +.7646939E+05 | +.0000000E+00 |
| +.2949995E+02 | +.0000000E+00 | +.7137840E+05 | +.0000000E+00 |
| +.2959994E+02 | +.0000000E+00 | +.6659635E+05 | +.0000000E+00 |
| +.2969993E+02 | +.0000000E+00 | +.6210667E+05 | +.0000000E+00 |
| +.2979993E+02 | +.0000000E+00 | +.5789359E+05 | +.0000000E+00 |
| +.2989991E+02 | +.0000000E+00 | +.5394201E+05 | +.0000000E+00 |
| +.2999991E+02 | +.0000000E+00 | +.5023751E+05 | +.0000000E+00 |
| +.3009990E+02 | +.0000000E+00 | +.4676632E+05 | +.0000000E+00 |
| +.3019989E+02 | +.1000180E-02 | +.4351539E+05 | +.3179895E-07 |
| +.3029988E+02 | +.2978515E-02 | +.4047220E+05 | +.1011863E-06 |
| +.3039987E+02 | +.5679216E-02 | +.3762489E+05 | +.2062576E-06 |
| +.3049986E+02 | +.1053951E-01 | +.3496211E+05 | +.4094059E-06 |
| +.3059985E+02 | +.1798754E-01 | +.3247317E+05 | +.7477027E-06 |

| | | | |
|---|---|---|---|
| +.3079983E+02 | +.4604002E-01 | +.2797639E+05 | +.2194699E-05 |
| +.3089982E+02 | +.6924819E-01 | +.2594966E+05 | +.3537576E-05 |
| +.3099982E+02 | +.1004062E+00 | +.2405892E+05 | +.5499557E-05 |
| +.3109980E+02 | +.1388536E+00 | +.2229589E+05 | +.8158397E-05 |
| +.3119980E+02 | +.1857630E+00 | +.2065274E+05 | +.1171380E-04 |
| +.3129979E+02 | +.2405913E+00 | +.1912209E+05 | +.1628996E-04 |
| +.3139978E+02 | +.3007831E+00 | +.1769689E+05 | +.2187791E-04 |
| +.3149977E+02 | +.3633617E+00 | +.1637054E+05 | +.2840625E-04 |
| +.3159976E+02 | +.4257335E+00 | +.1513677E+05 | +.3578866E-04 |
| +.3169975E+02 | +.4833469E+00 | +.1398968E+05 | +.4371286E-04 |
| +.3179974E+02 | +.5337093E+00 | +.1292371E+05 | +.5195258E-04 |
| +.3189973E+02 | +.5737378E+00 | +.1193357E+05 | +.6014198E-04 |
| +.3199973E+02 | +.6013988E+00 | +.1101434E+05 | +.6792008E-04 |
| +.3209971E+02 | +.6134486E+00 | +.1016133E+05 | +.7467830E-04 |
| +.3219971E+02 | +.6112853E+00 | +.9370163E+04 | +.8025091E-04 |
| +.3229970E+02 | +.5956963E+00 | +.8636697E+04 | +.8437812E-04 |
| +.3239969E+02 | +.5681188E+00 | +.7957066E+04 | +.8686637E-04 |
| +.3249968E+02 | +.5282776E+00 | +.7327613E+04 | +.8723503E-04 |
| +.3259967E+02 | +.4816291E+00 | +.6744911E+04 | +.8593428E-04 |
| +.3269966E+02 | +.4302571E+00 | +.6205751E+04 | +.8298796E-04 |
| +.3279965E+02 | +.3767474E+00 | +.5707122E+04 | +.7859206E-04 |
| +.3289964E+02 | +.3239890E+00 | +.5246189E+04 | +.7313215E-04 |
| +.3299963E+02 | +.2731124E+00 | +.4820310E+04 | +.6673872E-04 |
| +.3309962E+02 | +.2265542E+00 | +.4427011E+04 | +.5996168E-04 |
| +.3319962E+02 | +.1846193E+00 | +.4063969E+04 | +.5294833E-04 |
| +.3329960E+02 | +.1493864E+00 | +.3729019E+04 | +.4644803E-04 |
| +.3339959E+02 | +.1188104E+00 | +.3420135E+04 | +.4006813E-04 |
| +.3349959E+02 | +.9356921E-01 | +.3135423E+04 | +.3424313E-04 |
| +.3359957E+02 | +.7297432E-01 | +.2873118E+04 | +.2899436E-04 |
| +.3369957E+02 | +.5693317E-01 | +.2631571E+04 | +.2457082E-04 |
| +.3379956E+02 | +.4346681E-01 | +.2409245E+04 | +.2038589E-04 |
| +.3389955E+02 | +.3320140E-01 | +.2204709E+04 | +.1692983E-04 |
| +.3399954E+02 | +.2348568E-01 | +.2016628E+04 | +.1302658E-04 |
| +.3409953E+02 | +.1917897E-01 | +.1843762E+04 | +.1157684E-04 |
| +.3419952E+02 | +.1399453E-01 | +.1684956E+04 | +.9197455E-05 |
| +.3429951E+02 | +.1033051E-01 | +.1539131E+04 | +.7395750E-05 |
| +.3439950E+02 | +.8158683E-02 | +.1405295E+04 | +.6365579E-05 |
| +.3449950E+02 | +.5527489E-02 | +.1282520E+04 | +.4702285E-05 |
| +.3459948E+02 | +.3615466E-02 | +.1169943E+04 | +.3355175E-05 |
| +.3469948E+02 | +.2128252E-02 | +.1066768E+04 | +.2155506E-05 |
| +.3479947E+02 | +.1106340E-02 | +.9722531E+03 | +.1223479E-05 |
| +.3489946E+02 | +.8033078E-03 | +.8857129E+03 | +.9704595E-06 |

H = +.1602657E+01
SK = +.2200295E+00