INTERFACING OF AN LSI-11 MICRO PROCESSOR WITH THE SPECTRA-PHYSICS 3500B GRADIENT ELUTION LIQUID CHROMATOGRAPH

by

Gary Neal Giss

Thesis submitted to the Graduate Faculty of the
Virginia Polytechnic Institute and State University
in partial fulfillment of the requirements for the degree of

Master of Science
in
Chemistry

APPROVED

R.E. Dessy, Chairman

H.M. McNair

J.E. McGrath

August, 1978

Blacksburg, Virginia

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

PAGE

INTRODUCTION

Building and using an intelligent instrument is the domain of the chemist. He realizes what concepts are involved in real-time data acquisition and what the limitations of an instrument are. He expects the following from such an instrument:

1. An instrument that will optimize the conditions under which it runs.
2. It will control all basic functions itself. An operator does not have to perform any operations during an analysis, but can do other work.
3. It will run unattended.
4. It will record all operating parameters, saving them with the data so that a permanent record is kept.
5. Perform smoothing and filtering of the data to reduce unnecessary noise.
6. Analyze data to the point that it is easily understood.
7. Store data in a medium easily transferable to other computers for later analysis by sophisticated data techniques.
8. Possibly give determinations of the components of a sample.

A computer scientist or an electrical engineer cannot grasp the problem or communicate with the chemist sufficiently to do the job that is needed. Building equipment such as this is an iterative process,

unexpected results are found, and only by keeping the system flexible can it be improved.

The first goal of the project was to interface an LSI-11 microprocessor with a Spectra-Physics model 3500B liquid chromatograph to produce an intelligent instrument. This involves giving full control of the L.C. to the operator through a cathode ray tube (CRT). Interaction such as this is more user orientated than the knobs or switches that have been previously used. The instrument will also store all data, including the operating parameters, reducing the overhead of report writing.

A second goal was to use and test the data network system in the laboratory. A network system is controlled by a host computer consisting of a large amount of memory, a mass storage device and other necessary peripherals. It communicates with a series of satellite processors having only the minimal configuration. The satellites are dedicated to one operation or instrument. A system such as this allows the user to develop and debug programs in high level languages like Fortran and Basic on the host. The program can then be down-line loaded to the satellite where it can be run. Data can be sent from the satellite to the host and stored on the mass storage device. This allows the satellite to maintain the minimal memory for the program and yet still collect an unlimited amount of data. Sharing of the resources of the host by the satellites conserves on replication of unnecessary memory or peripherals. This saves on the amount of money spent for building a processor for a specific instrument. In our laboratory, the host is a PDP-11/03 computer using the RT-11/REMOTE operating system. This network

was designed for business applications but is being used here for real time data acquisition. The system itself is relatively new, being actively pursued as a laboratory system at Virginia Tech. Fortran compatibility with a satellite was investigated along with determinations of the best ways to transmit data from the satellite to the host while operating an instrument.

A final goal was to build the system so that another member of the research group could use the instrument.

SYSTEM DESCRIPTION

Overall System

The system is controlled by an LSI-11 KD11-F microprocessor. It is supported by sixteen kilowords of core memory, with the four kilowords of dynamic memory on the processor disabled. The peripheral units are; three DRV11 parallel line units (PLU), two DLV11 serial line units and one ADAC digital and analog converter.

One serial line unit interfaces with the teletype (figure 1), the other with the host computer. The parallel line units control the liquid chromatograph, support the timer and clock module and communicate with the NOVA system. The ADAC board receives the analog detector signal and converts it to digital format and controls the oscilloscope display.

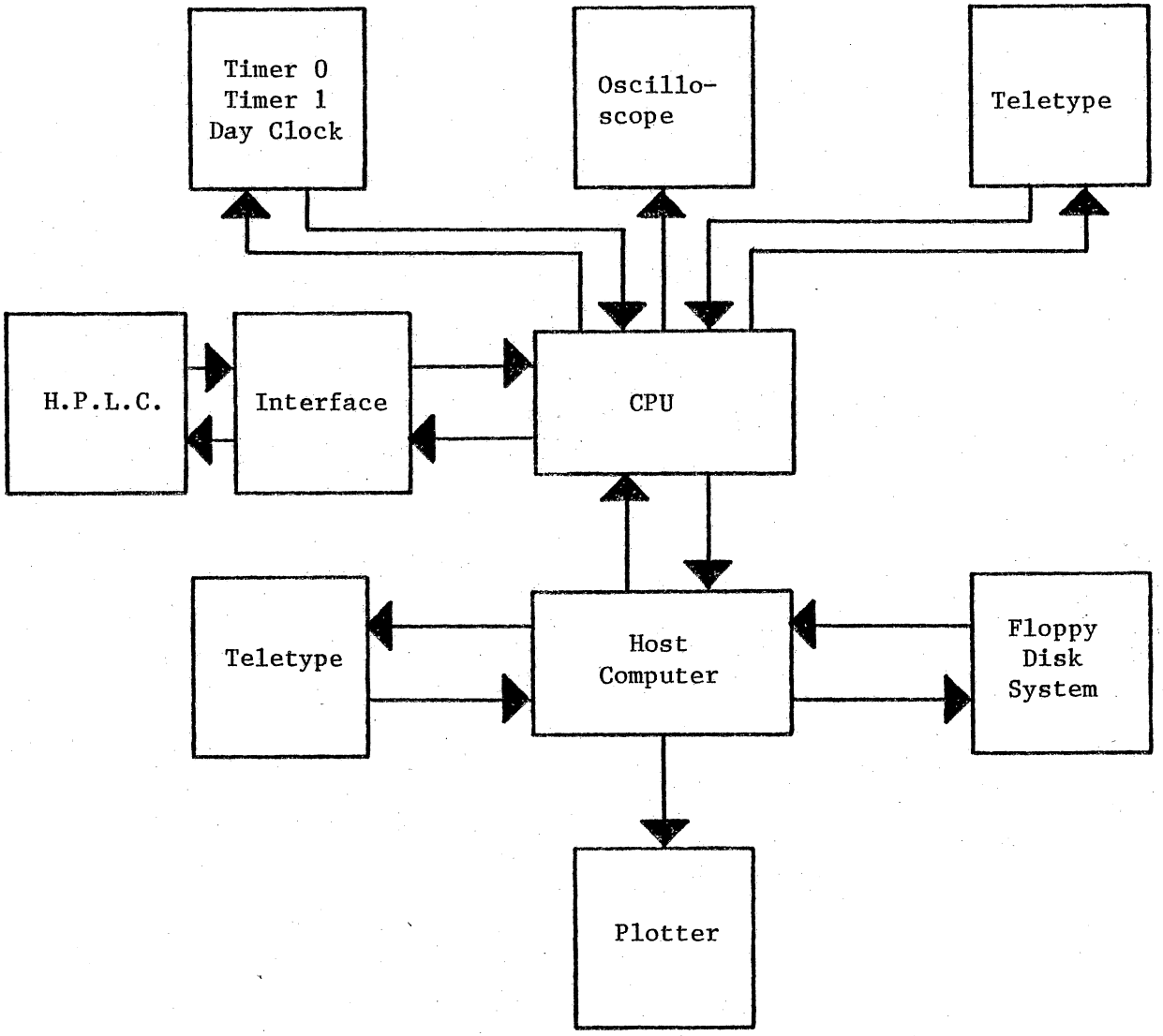


FIGURE 1 - System Layout

LIQUID CHROMATOGRAPH

The Spectra-Physics 3500B contains two Model 740 pumps, a mixing chamber and a solvent programmer for gradient elution. The detector is a fixed wavelength ultra-violet detector set at 254nm. The Model 740 pump has two speed ranges: 0 to 4 ml/min and 0 to 20 ml/min, the first for high speed analytical chromatography and the second for preparative chromatography. It is a two piston reciprocating, positive displacement pump with inlet and outlet check valves in each of the two cylinders. The pistons are driven 180° out of phase by a single cam. The cam is designed so that one piston pumps while the other fills up.

A flow feed back system is incorporated to reduce pump noise. During the end of a stroke, when the flow rate would normally decrease, due to compressability of the solvent or compliance of the piston seal, the flow feed back mechanism varies the speed of the motor so that the overall flow rate is effectively constant.

The flow feed back device has two pressure transducers. These are used to measure the pressure drop across a flow restrictor, placed in the outlet stream. A variation in the difference of the two transducers causes a signal to be sent to the motor speed control. The motor speed is then changed to compensate for the flow rate change. This technique assumes constant viscosity of the fluids and constant permeability of the restrictor.

To control both pumps for gradient elution, a Model 744 programmer module is used. This unit acts as the control center for the L.C. The three states that the L.C., can operate in are set in this module. The

INITIAL condition sets the flow rates of the two pumps to the values indicated by the gradient. The run clock is set to zero. Upon switching to the START mode, the run clock and the gradient programmer are started. When HOLD is set, the run clock and the gradient programmer are held at whatever point they were at when the HOLD was initiated. Reverting to START will resume all operations. The flow rate and the run time are set in the programmer module.

The operational modes, the run time and the flow rate are all available at the rear of the programmer module for remote control operation.

TABLE 1

REMOTE CONTROL OF MODEL 744 PROGRAMMER

PLU Bit No.	Pin No.	Programmer Condition	Signal/Function
NA	1	Remote Off	Ground = Press Initial
NA	3	" "	Ground = Press Start
NA	5	" "	Ground = Press Hold
REQ B	20	" "	Ground at End of Run
12	19	Remote On	Ground = Press Initial
13	21	" "	Ground = Press Start
14	23	" "	Ground = Press Hold
NA	11	" "	Flowrate. Full scale = +5VDC
NA	7	" "	Gradient shapes:
NA	9	" "	Jumper 7 to 25 and
NA	25	" "	9 to 27 for switch-
NA	27	" "	selected profiles
0	29	" "	+7.5VDC = 1 minute run
1	13	" "	" = 2 " "
2	14	" "	" = 4 " "
3	15	" "	" = 8 " "
4	17	" "	" = 10 " "
5	16	" "	" = 20 " "
6	18	" "	" = 40 " "
7	31	" "	" = 80 " "
8	32	" "	" = 100 " "
9	33	" "	" = 200 " "
10	34	" "	" = 400 " "
11	35	" "	" = 800 " "
GND	12	Remote On/Off	= ground
NA	36	Remote On/Off	= +7.5 VDC
NA	2	Remote On/Off	= +15 VDC
NA	4	Remote On/Off	= -15 VDC

HARDWARE

L.C. Control

The model 744 Programmer has its operational modes and run time signals in digital format and the flow rate in analog format, available for control of the L.C. The lines are not directly compatible with a micro-computer because of the difference in digital signal voltages. The L.C. has its' logic one equal to seven volts. The computer uses TTL signal levels.

A parallel line unit (PLU) is used to control the interface. The PLU address is 167760 for the control status register (CSR) and 310 for its interrupt vector. The three functions -- operational modes, run time and flow rate -- are controlled by the PLU. The run time is set by maintaining the output buffer of the PLU at the appropriate value. Bits zero through eleven are used. Each bit has a specific number of minutes associated with it (table 1). A time is set by placing a zero in the proper bit. When a bit is set to zero, it causes an open-collector NAND gate output, tied to seven volts, to go high (figure 15). The L.C. then recognizes the seven volt signal and sets the run time. For a run time of ten minutes (referring to table 1) the proper control word would be:

$$3765_8 \text{ or } 11111110101_2.$$

The operational modes, START, INITIAL and HOLD, are controled by bits 12, 13 and 14 of the PLU output buffer (table 1), which are inserted by NAND gates (figure 15 . Bit 12 is INITIAL, bit 13 is START and bit 14 is HOLD. A mode is set by placing a one in the proper bit adding run

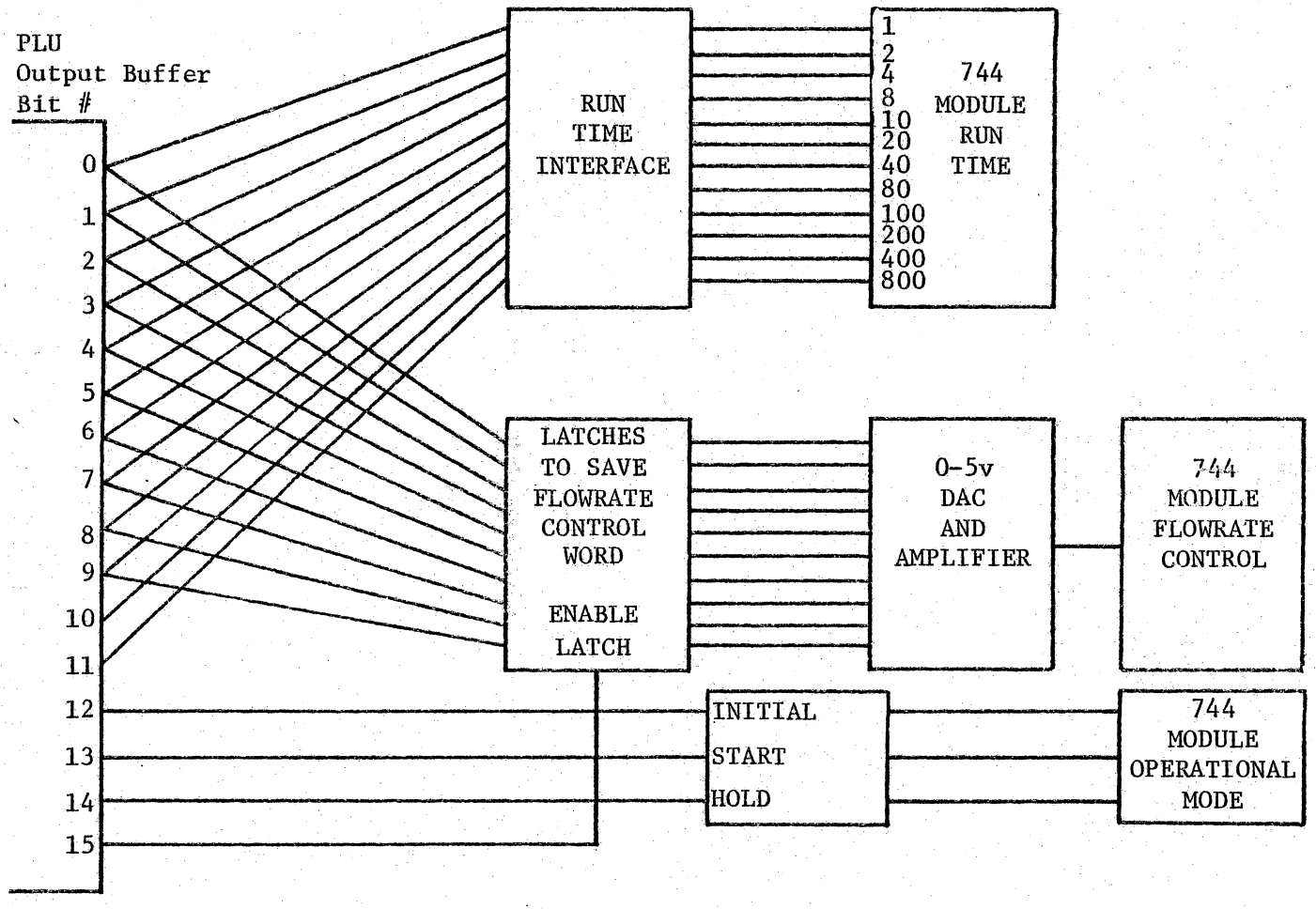


FIGURE 2 - L.C. Control Overview

time to it. To set a run time of ten minutes and to put the L.C. on INITIAL, the control word is:

$$017765_8 \text{ or } 0001111111110101_2.$$

To start the L.C. with a run time of ten minutes, the control word is:

$$027765_8 \text{ or } 0001111111110101_2.$$

The flow rate is set by bits zero through nine and fifteen. These bits are multiplexes with the run time. Setting bit 15 to one enables the latching of bits zero through nine for the flow rate. The outputs of the latches are connected to a 10 bit, zero to ten volt digital to analog converter (DAC) (figure 17). The range of the L.C. input is zero to five volts for a flow rate of zero to four milliliters per minute. The DAC is followed by an amplifier which divides the output by a factor of two. The latches are cleared by the INIT pulse when the processor is started up. This sets the flow rate to zero. To set the minimum flow rate, the control word is:

$$100000_8 \text{ or } 1000000000000000_2.$$

The maximum flow rate control word is:

$$101777_8 \text{ or } 1000001111111111_2.$$

To detect the beginning of a run, it was not feasible to have the operator indicate an injection by using the teletype. Instead, a push button was installed next to the injection port. When this is depressed, it shorts a debouncing circuit to ground (figure 18). Its' output goes

high once then returns to the ground. The pulse goes to the clock of a flip-flop. Its' output goes high and stays high until a clear signal is sent, either from INIT or from the processor by CSR Ø. The output goes to bit zero of the PLU input buffer.

So that the processor can detect when the L.C. has reached the end of its' run time, the L.C. sends an end of the run signal. This is a seven volt signal. A voltage divider circuit is used (figure 15), which then inputs to a NAND gate acting as an inverter and buffer. When the output of the gate is high, the run is over. The processor detects this through a REQ B interrupt.

The output of the U.V. detector is zero to ten millivolts. To obtain good resolution, the signal has to be amplified to ten volts since the analog to digital converter (ADC) is on a zero to ten volt scale. The signal has to be amplified by a factor of one thousand. This is done in two stages by a pair of 741 operational amplifiers (figure 16). The first stage amplifies by a factor of one hundred, the second by ten. the output is then fed straight to the input of the ADC.

Timer-Clock Circuit

A second PLU is used to set the day clock and to control the timers. The CSR address is 167750 and the vector address is 320. The timer and day clock boards are bussed together (table 4). The first time board (figure 20) divides a one megahertz crystal frequency into eight different frequencies. This is done by a series of 7490 divide by five divide by two counters. Bits 13, 14 and 15 of the PLU output buffer, control the functions of the three boards. To select a frequency of 200

kHz, the control word 020000_8 or 0010000000000000_2 must be placed in the PLU output buffer (table 2). The frequency selected is latched so that it is available at REQ A at all times. The frequency will change only when bit 13 is set, and bits 14 and 15 are zero. The frequencies and their control words are listed in table 2.

The second time board (figure 22) takes a 10 kHz signal from the first board and divides it by a series of 7490 decade counters. This timer has four frequencies made selectable by setting bits 13, 14 and 15 to zero (table 2). To select a frequency of 1 Hz, the control word is:

$$000001_8 \text{ or } 0000000000000001_2.$$

The frequencies are available at the REQ B line. The four times on this board are used to control the data rate for the L.C.

The day clock (figures 24, 25 and 26) keeps the time of day for four days before turning over. It has 25 bits for counting, divided into two parts, the high clock and the low clock. The high clock is the upper 13 bits of the clock, counting the days, hours and minutes. The low clock is the lower 12 bits, keeping track of the seconds. There are five functions enables by bits 13, 14 and 15 of the PLU: Load high, Load low, Read High, Read low and Clear clock (table 2). The control words are given in table 2. To use the Load function, the time is added to the control word and placed in the PLU output buffer. The buffer must not be changed until the load acknowledge signal has been received. For the low clock, bit 14 of the PLU input buffer goes high when the load is done. The high clock signal is on bit 13 of the input buffer. To read a time on the clock, the read control word is placed in the output buffer. The

TABLE 2

Timer 0, Timer 1 and Day Clock
Control Words and Output Select

Timer 0:	Control Word	Frequency Selected (kHz)
	020000	200
	020001	100
	020002	50
	020003	40
	020004	25
	020005	20
	020006	10
	020007	5

Timer 1:	Control Word	Frequency Selected (Hz)
	000000	0.1
	000001	1.0
	000002	10.0
	000003	100.0

Day Clock:	Control Word	Function
	140000	Clear Clock
	100000	Read Lower Clock
	120000	Read Upper Clock
	040000	Load Lower Clock
	060000	Load Upper Clock

value of each bit is given in table 3. The whole clock is cleared by placing the clear control word in the output buffer.

The day clock uses a series of sunchronous counters, driven by a 1.0 kHz signal. There are two sets of outputs from the counters, one with twelve bits and the other with thirteen bits. These lines are bussed together by a set of tri-state buffers, onto the thirteen lines of the input buffer. The read control words enable the tri-state buffers.

TABLE 3

Day Clock Times

Lower Clock		Upper Clock	
Bit Number	Time(sec)	Bit Number	Time(min)
0	0.02	0	0.914
1	0.04	1	1.6
2	0.08	2	3.2
3	0.16	3	6.4
4	0.32	4	13.7
5	0.64	5	24.0
6	1.28	6	48.0
7	2.56	7	96.0
8	5.12	8	205.7
9	12.8	9	360.0
10	25.6	10	720.0
11	25.6	11	1440.0
		12	2880.0

TABLE 4

Backplane for Timer Boards

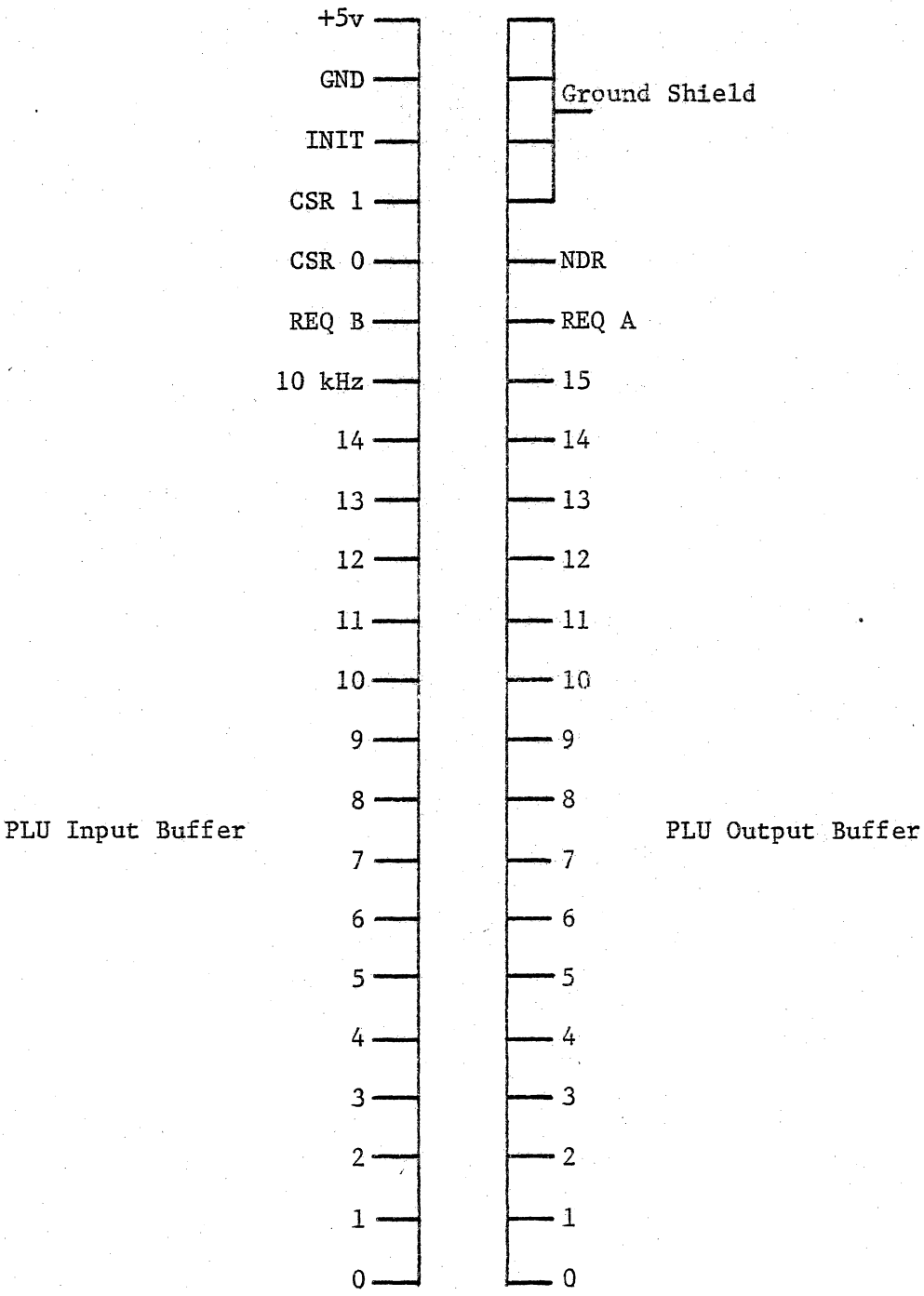


TABLE 5
Peripheral Addresses

Device	CSR/Data Buffers	Vector Area
Teletype	177560-177566	60-66
ADC	176760-176762 DACs	
	176770-176772 ADC	130-132
PLU 1	167760-167764	310-316
PLU 2	167750-167754	320-326
Satellite	177540-177546	140-146
Remote Boot	160000-162000	

SOFTWARE

The software was originally written in Fortran because of the math routines available. The language is also supplied with routines to handle basic machine operations such as inspecting or changing memory locations and setting up interrupts. It was found that these routines did not run correctly in a stand-alone system. The program had to be replaced by an assembler language program. The whole program was rewritten to assembler to assure compatability with the whole system.

The software is divided into eight modules:

A. REPORT

B. MAIN CONTROL

C. DATA INTERRUPT (D INT)

D. END OF DATA (EOD)

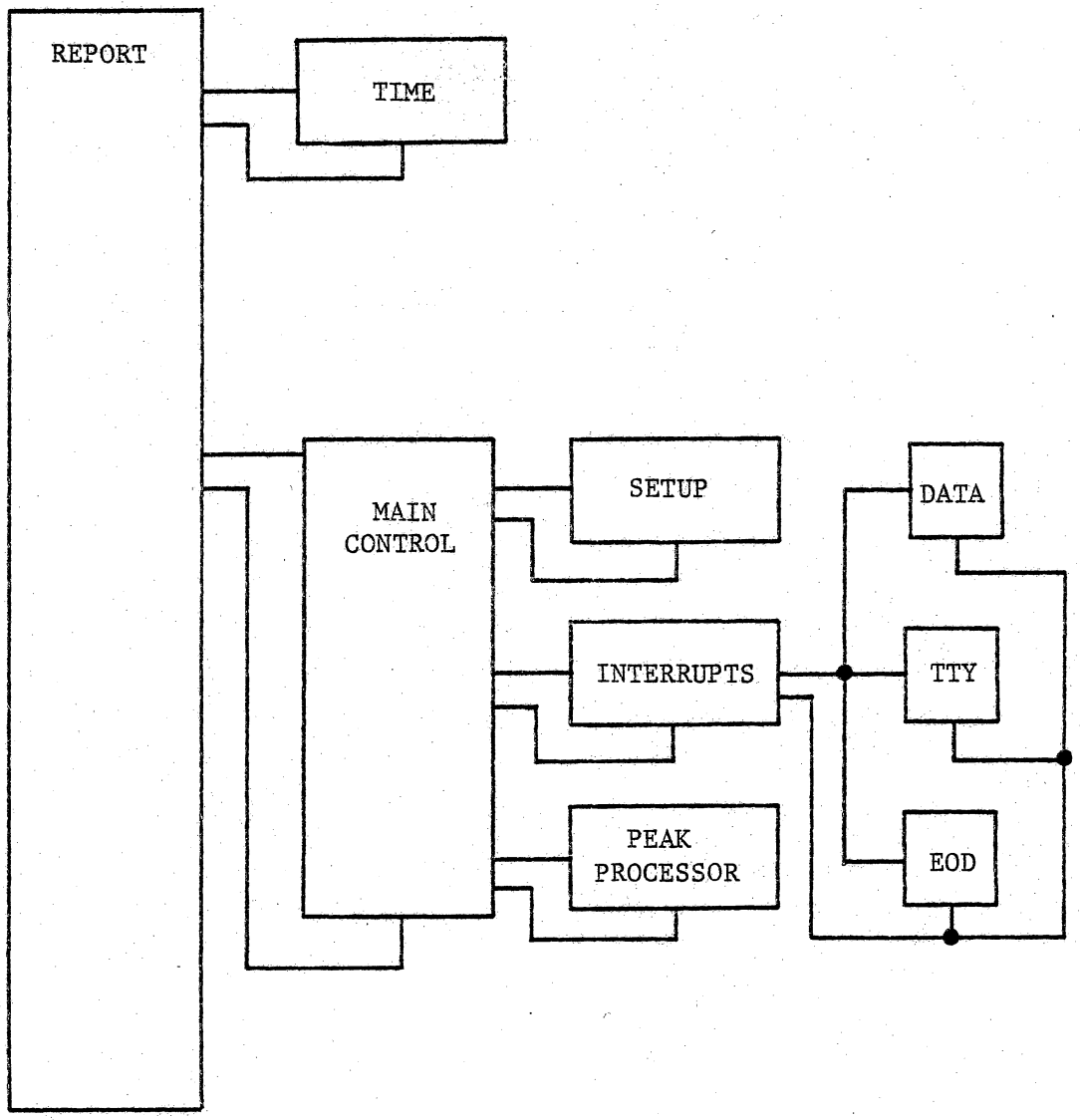
E. TELETYPE INTERRUPT (T INT)

F. TIME OF DAY

G. SETUP

H. PEAK PROCESSOR

REPORT is the main routine to begin and end a chromatographic run (figure 3). It calls TIME OF DAY and MAIN CONTROL. MAIN CONTROL calls SETUP and performs the operations necessary for controlling the run, interrupted by DATA INTERRUPT, TELETYPE INTERRUPT and END OF DATA. PEAK PROCESSOR is then called to analyze the peak data. MAIN CONTROL returns to REPORT.



Software Routine Flow

FIGURE 3

REPORT

This is a question and answer routine to obtain the parameters necessary for a complete record of a typical chromatographic run. For the user that is unfamiliar with the operation a HELP listing has been provided which explains how to set the instrument up and how to operate it. This can be seen listed in figure 4. The program is called HELP.LC. The teletype records the information permanently, the processor utilizes only the parameters necessary to operate the L.C. These parameters are: time, data file name, data rate, run time and flow rate. Figure 5 shows a typical output (also see REPORT flowchart).

The report is divided into five sections:

1. Sample
2. Column Specifications
3. Detector
4. Operational Parameters
5. Mobile Phase

The operator is requested to select which section has to be changed by the prompt: 'SECTION CHANGE' followed by the number of the section, '1?'. A 'Y' for yes or an 'N' for no, followed by a carriage return, is requested. Each section is checked. The replies are stored in a word called KEY. Only the sections which have a positive answer stored in KEY are printed. All answers must be followed by a carriage return, except where noted.

The parameters tored by the L.C. have special formats for input. The time input is described in the TIME OF DAY section.

RUNNING THE PROGRAM

THIS PROGRAM WILL TAKE DATA FROM AN L.C. AND STORE IT ON THE FLOPPY DISK. THE FIRST SECTION WILL PROMPT YOU WITH THE APPROPRIATE PARAMETERS FOR THE RUN. MOST INPUTS ARE IN THE FREE FORMAT SO THAT ANY INFORMATION CAN BE INPUT. THE ONES LISTED BELOW REQUIRE A SPECIFIC FORMAT FOR INPUT.

FILE: INPUT A 12 CHARACTER STRING CONTAINING THE FLOPPY DISK WHICH WILL BE USED FOLLOWED BY THE NAME OF THE FILE. EXAMPLE:

FD1SAMPLERUN

FLOPPY DISK ONE IS USED TO STORE THE FILE NAMED SAMPLE.RUN

RUN TIME(MIN) = : THIS REQUESTS A VALUE BETWEEN) AND 1665 MINUTES FOR THE AMOUNT OF TIME THAT THE RUN WILL TAKE.

FLOWRATE(ML/MIN X 0.04) = : REQUESTING A VALUE BETWEEN 0 AND 100 REPRESENTING THE PERCENT OF THE MAXIMUM OF THE NEEDED. EXAMPLE: IF A FLOWRATE OF 2 ML/MIN IS DESIRED, AND THE MAXIMUM FLOWRATE IS 4 ML/MIN, THEN THE VALUE INPUT WOULD BE 50.

DATA RATE = : THE RATE AT WHICH DATA WILL BE COLLECTED. VALUES OF 0, 1, 2 AND 3 ARE INPUT TO GET RATES OF 0.1 HZ, 1.0 HZ, 10.0 HZ and 100.0 HZ, RESPECTIVELY.

TO CHANGE THE L.C. WHILE DATA IS BEING COLLECTED, THE LETTERS 'H', 'S' AND 'E' ARE INPUT. 'H' PUTS THE L.C. IN THE HOLD MODE, 'S' RESTARTS IT AND 'E' TERMINATES THE RUN AND DISPLAYS THE DATA ON THE SCOPE WHILE STORING IT ON THE FLOPPY DISK. OTHERWISE THE RUN WILL TERMINATE NORMALLY AND CAUSE THE DATA TO BE SENT TO THE DISK. WHEN A RUN HAS ENDED, THE DATA IS DISPLAYED ON THE SCOPE AND THE OPERATOR CAN INCREMENT THROUGH THE FILE BY USING THE KEYS 'I' FOR THE NEXT BLOCK OR 'D' FOR THE PREVIOUS BLOCK. 'N' IS USED TO PROCEED TO THE NEXT SECTION WHICH IS THE PEAK PROCESSOR. ANALYZE PEAKS WILL BE ASKED AND A 'Y' OR AN 'N' IS INPUT FOR YES OR NO, WITH NO CARRIAGE RETURN. 'LONG FORMAT' IS THEN ASKED, ANSWER 'Y' OR 'N' WITH NO 'CR'.

IN THE PEAK PROCESSING ROUTINE, THREE PARAMETERS ARE REQUESTED.

WIDTH FACTOR = : A VALUE BETWEEN 0 AND 7 IS INPUT. THIS VALUE DETERMINES WHEN THE ROUTINE WILL BEGIN TO LOOK FOR THE END OF A PEAK OR THE BASELINE.

FIGURE 4

GATE = : VALUE BETWEEN 0 AND 7. THIS IS THE NUMBER OF UNIDIRECTIONAL CHANGES IN DATA NEEDED TO DEFINE AN INCREASING OR DECREASING TREND.

CHANGE FACTOR = : VALUE BETWEEN 0 AND 7. THIS IS THE MINIMUM INCREASE IN A DATUM FROM THE PREVIOUS ONE TO BE CONSIDERED A SIGNIFICANT INCREASE.

IT THEN ASKS 'DO ANOTHER RUN?' 'Y' FOR YES, 'N' FOR NO, WITH A CARRIAGE RETURN.

HARDWARE SETUP

TO SET THE SYSTEM UP, THE PLU AT ADDRESS 167750 MUST BE CONNECTED TO THE TIMER CLOCK INTERFACE. PLU 167760 MUST BE TO THE L.C. INTERFACE. THE SCOPE INPUT 'A' GOES TO DACO. THE TRIGGER GOES TO THE BNC OUTPUT ON THE TIMER CHASSIS. THE SCOPE IS SET TO EXTERNAL TRIGGER, 2 VOLTS/DIV, 2 MSEC TIME/DIV AND VERTICAL DISPLAY OF CHANNEL 'A' ONLY. THE INTERFACE TO THE L.C. IS POWERED BY THE PROCESSOR. IT HAS A CORD COMING FROM THE POWER SUPPLY OF THE PROCESSOR.

THE SWITCH ON THE INTERFACE CONTROLS THE ANGLE OF THE GRADIENT. IN 'A', IT IS A POSITIVE 45 DEGREE ANGLE, IN 'B', IT IS A NEGATIVE 45 DEGREE ANGLE.

A SWITCH ON THE BACK OF THE SOLVENT PROGRAMMER HAS BEEN INSTALLED NEXT TO THE INTERFACE PLUG, TO CONTROL THE FLOWRATE CONTROLLER. WHEN THE SWITCH IS UP, THE INTERFACE CONTROLS THE FLOWRATE. WHEN THE SWITCH IS DOWN, THE L.C. SWITCHES CONTROL THE FLOWRATE. THE BUTTON NEXT TO THE INJECTION PORT IS SEND A SIGNAL TO THE THAT THE START OF A RUN HAS JUST OCCURED.

SOFTWARE

THE PROGRAM IS CALLED AS 'LCCON.LDA'. THIS IS A COMBINATION OF THE PROGRAMS CALLER, TOTAL, TIME, INTRUP, P AND RISIM. THE PROGRAM IS LINKED WITH THE SWITCHES /F/L/B:1000. THE PROGRAM IS STARTED AT ADDRESS 1000.

FIGURE 4 (continued)

DATE: JULY 6,1978
OPERATOR: GARY GISS
RUN NUMBER: 1
ENTER FILE NAME FD1DATAFILE1

1 SAMPLE
NAME: BENZENE IN HEXANES
VOLUME: 20 ul
CONC: 2 mg/ml

2 COLUMN SPECIFICATIONS
MANUFACTURER: USER MADE
PACKING: SPHERISORB
LENGTH: 20 cm
INNER DIAMETER =
COLUMN PRESSURE = 26 psi
TEMP(C) = 24

3 DETECTOR
TYPE: UV
O.D. VALUE: 64
MODE: SINGLE WAVELENGTH
WAVELENGTH: 254 nm

4 RUN TIME(MIN) = 8
FLOWRATE(ml/min x 0.04) = 50
DATA RATE = 1

5 MOBILE PHASE
A: N-HEXANES
B: N.A.
GRADIENT SLOPE = 0
%A AT START: 100
%A AT END: 100

STARTE
ANALYZE PEAKS?Y
ENTER WIDTH FACTOR 4
ENTER GATE 6
ENTER CHANGE FACTOR 6

PEAK NO.	PEAK AREA	RETENTION TIME(SEC)
1	1562	150
2	1023	247

Sample Report

FIGURE 5

```
DO ANOTHER RUN?Y

SECTION CHANGE
1?Y
2?Y
3?Y
4?Y
5?Y

RUN NUMBER 2
ENTER FILE NAME: FD1DATAFILE2

STARTE
ANALYZE PEAKS? Y
LONG FORMAT? Y

ENTER WIDTH FACTOR 3
ENTER GATE 6
ENTER CHANGE FACTOR 6
```

PEAK NO.	PEAK AREA	PEAK HEIGHT (SEC)	PEAK TIME	LEADING MINIMUM HEIGHT	TIME OF LEADING MINIMUM	WIDTH	TRAILING MINIMUM HEIGHT	TIME OF TRAILING MINIMUM	PEAK TYPE
00	35853	2019	151	00	48	08	00	166	00
01	18111	1500	177	00	166	10	00	201	01
02	73750	3284	231	00	201	08	107	254	01

```
DO ANOTHER RUN?N
```

Sample Report, Continued

FIGURE 5 (continued)

The data file name requests the disk drive that will be used, FDØ, FD1, etc., and a nine character name for the data file. To use disk drive Ø, a correct name would be:

FDØDATAFILE1

After the twelve characters have been input, the teletype automatically executes a carriage return. The routine performs a RAD50 conversion of the twelve ASCII characters to put them in a form usable by the device handler.

The run time is input as a decimal number of minutes between Ø and 1665. This is changed to its' octal equivalent and stored in SWEEP. A series of subtractions are performed to set the bits in the run time control word. The values subtracted are the octal equivalents of the times available from the L.C. (listed in table 1). In the first step, 1440_8 (800_{10}) is compared to SWEEP. If SWEEP is greater, 1440_8 is subtracted from it and bit 11 of the control word is set. SWEEP is next compared to 620_8. This process continues until SWEEP is equal to zero. The complement of the control word is taken to fit it to the format of the L.C.

The data rate requests a single digit value, the value corresponding to a specific rate of data collection.

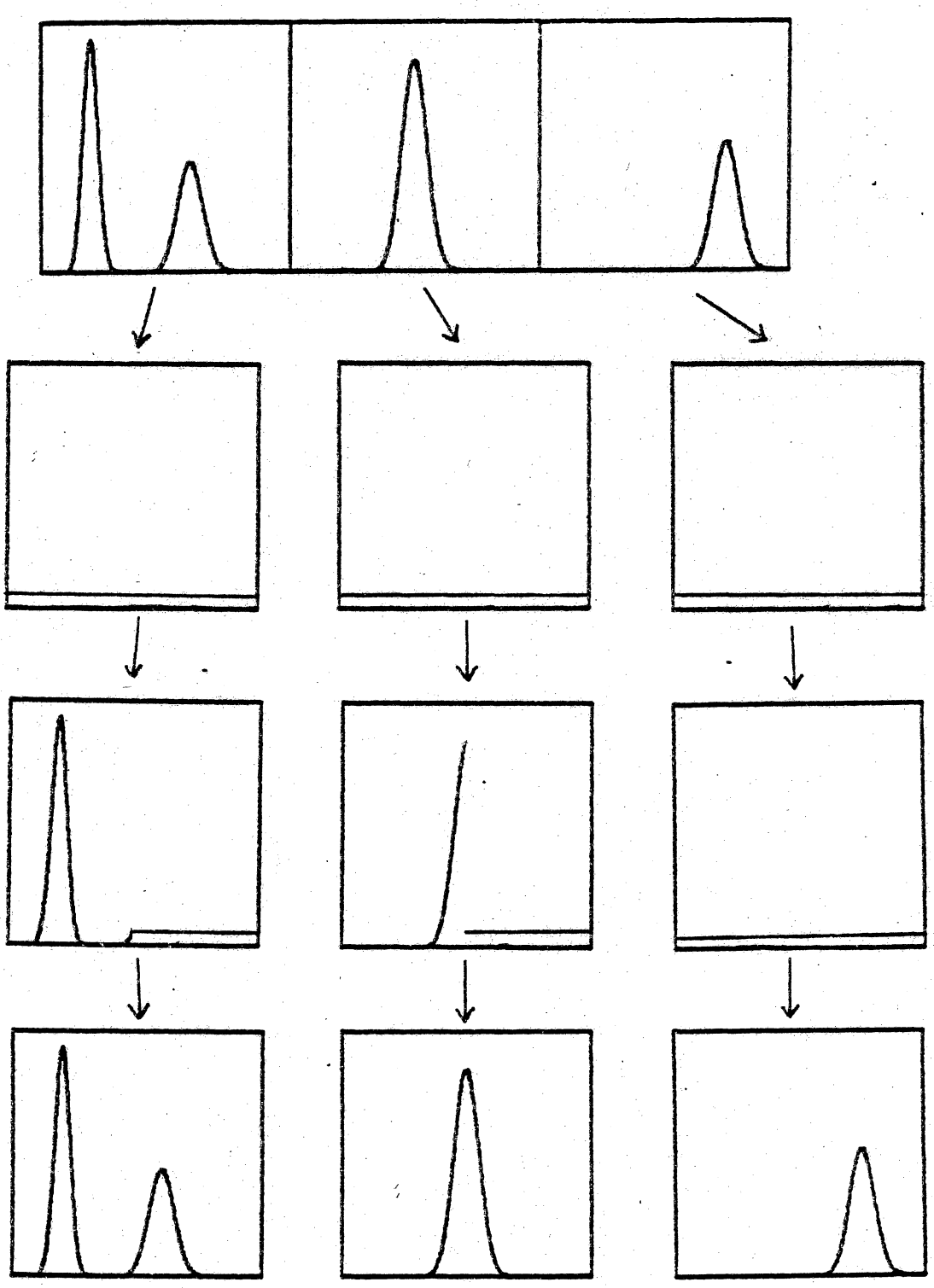
Digit Input	Frequency Selected
Ø	0.1 Hz
1	1.0 Hz
2	10.0 Hz
3	100.0 Hz

The timer 1 board is set to produce one of these rates by placing the control word in location 167752.

The routine initializes the L.C. by outputting the flow rate followed by the run time control word, set for INITIAL. The subroutine MAIN CONTROL is called. REPORT next asks whether another run should be done. Upon an answer of 'Y', the section changes are again requested and the routine begins. An answer of 'N' ends the run and returns control to the Monitor.

MAIN CONTROL

This is the main operational routine. SETUP and PEAK PROCESSOR are called by this program. EOD, D INT and T INT interrupt this routine. It begins by waiting for the injection signal (see CONTROLLER flowchart). This is bit zero of the PLU input buffer, 167764. After the bit goes to one, the injection signal is reset by making CSRØ go high and low at adress 167760. The message 'START' is output at the beginning of the run. The run time and the START signal are output to the L.C. and SETUP is called. Ten 512 point data blocks, called DATBUF, are kept in memory. As the routine collects data, it is displaying the current data block on the oscilloscope. Imagining a strip short recorder output, such as the one at the top of figure 6 , that is broken up into sections of 256 points each, one can see what is on the oscilloscope screen. As the data fills the blocks, it is seen to grow, as in the frames shown vertically in the figure. The data is collected in consecutive blocks. When all ten blocks are full, the flow rate is set to zero and the L.C. is put into the HOLD mode. The data blocks are written to disk. The



Peak Display Example

FIGURE 6

flow rate is set back to its' proper value and the L.C. is put back into the START mode. The L.C. is stopped for two seconds during this operation. The pointers are reset to the beginning of DATBUF and data is again collected. This process continues until the end of data signal is received. The last blocks are written to disk. All interrupts are disabled and the first block of data stored on the disk is displayed on the oscilloscope. To view the latter blocks, the 'I' key is depressed, the 'D' key for earlier blocks. To end the display, the 'N' key is used.

The operator is asked whether the data should be analyzed. A negative answer returns control the the REPORT routine. A positive answer sets the data buffer for the reading of twenty, 512 point blocks of data from the disk. These are analyzed by the PEAK PROCESSOR routine. If a peak overlaps the end of the buffer, the last block is reread into the first block of the next buffer to be analyzed. It is started at the minimum point before the partial peak. The blocks are analyzed until there are no more left. The program returns control to REPORT.

DATA INTERRUPT

This routine acquires the data (DATA flowchart). The processor receives an interrupt from TIMER 1. Control is vectored to DATA where the ADC conversion is initiated and waited for. The point is moved from the ADC buffer, 176772, to the data buffer, DATBUF. If it is full, the buffer pointers are reset. Control is returned to MAIN CONTROL.

END OF DATA

At the end of a run, REQ B goes high and interrupts the data acquisition (EOD flowchart). The processor vectors to the EOD routine.

The interrupts are disabled. An 'E' is printed to indicate to the operator that the run is done. The done flag is set and the interrupt routine returns control to MAIN CONTROL.

TELETYPE INTERRUPT

The teletype interrupt routine is vectored to when a key is depressed during data acquisition (TELETYPE flowchart). Keying an 'H' will put the L.C. in the HOLD mode, an 'S' puts it in the START mode and an 'E' will end the routine and return control to the MAIN CONTROL program. The routine is vectored to and moves the character from the teletype input buffer to a register. It is compared to the ASCII equivalent of each letter and branches to the proper section. The 'S' and 'H' branches add the run time to the control signal and outputs them to the L.C. through the PLU output buffer, 167762. The 'E' branch disables the interrupts, sets the done flag and pops the stack twice to point the stack to the proper return. It then branches to the section in MAIN at which the final blocks are written out to disk.

SETUP

The interrupt vectors for EOD, D INT and T INT are set in this routine (SETUP flowchart). The Monitor interrupt vectors are saved and all the interrupts are enabled.

TIME OF DAY

This routine is called by REPORT. It requests the time of day, converts it and outputs it to the clock (TIME CALCULATION flowchart). The hour of the day is input first, using a twenty-four hour format.

Two characters must be given. A colon is input, then the minutes, as two characters. Seven forty-five A.M. would be input as: 07:45. Three thirty P.M. would be: 15:30. The hours are changed to minutes and added to the number of minutes. The time set word is determined by setting bits in register two during an iterative subtraction of the octal equivalent of the clock times from the time. In the first subtraction, the number 2448_{10} is compared to the time. If it is less, it is subtracted from the time and bit 12 in register two is set. Otherwise, a lower value is compared. The values subtracted are determined from the frequencies of the different bits in the clock.

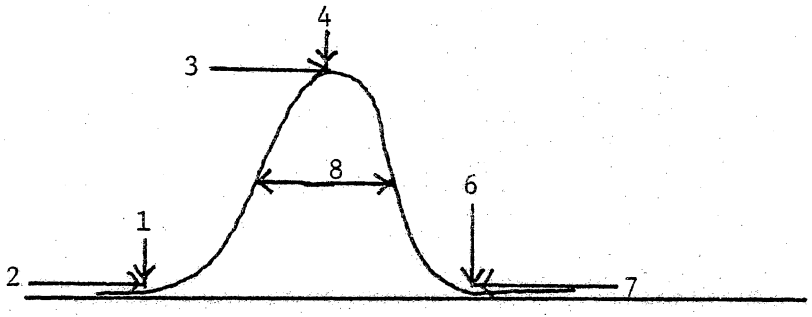
Once the subtractions are done, the clock is cleared and the high clock is loaded with the converted time. There is a wait period while the load operation takes place. Program control is returned to the REPORT routine.

PEAK PROCESSOR

The peak analysis routine takes a buffer of data and determines the peak area and the retention time in the short form and nine parameters, listed in figure 5, in the long form. The routine is a DEC program that Elaine Fiorino modified, which was later modified for use in this project (PEAK PROCESSOR flowchart). There are three parameters; baseline sensitivity, peak detection sensitivity factor and height minimum which determine the efficacy of this routine.

The baseline sensitivity (WT) is the point after the peak at which the program begins to look for the termination of the peak on the baseline. The point is determined by multipling the peak width by WT.

Full analysis of a peak.



1. Time of Leading Edge
2. Height of Leading Edge
3. Time of Maximum
4. Height of Maximum
5. Baseline Indicator (0 - ended on baseline, 1 - ended on valley)
6. Time of Trailing Edge
7. Height of Trailing Edge
8. Width of Peak
9. Area

Peak Parameters

FIGURE 7

The peak detection sensitivity factor (GT) is a gate. The gate represents the number of uni-directional changes in data needed to define an increasing or decreasing trend. The height minimum (HM) represents the minimum increase in a datum, from the previous one, to be considered a significant increase. The most efficient parameters were found to be: WT = 3, GT = 5 and HM = 6.

The flow of the program is controlled by three software switches; BS, S1 and S2 (figure 8).

BS - baseline switch

Ø - peak started on baseline

1 - looking for peak width

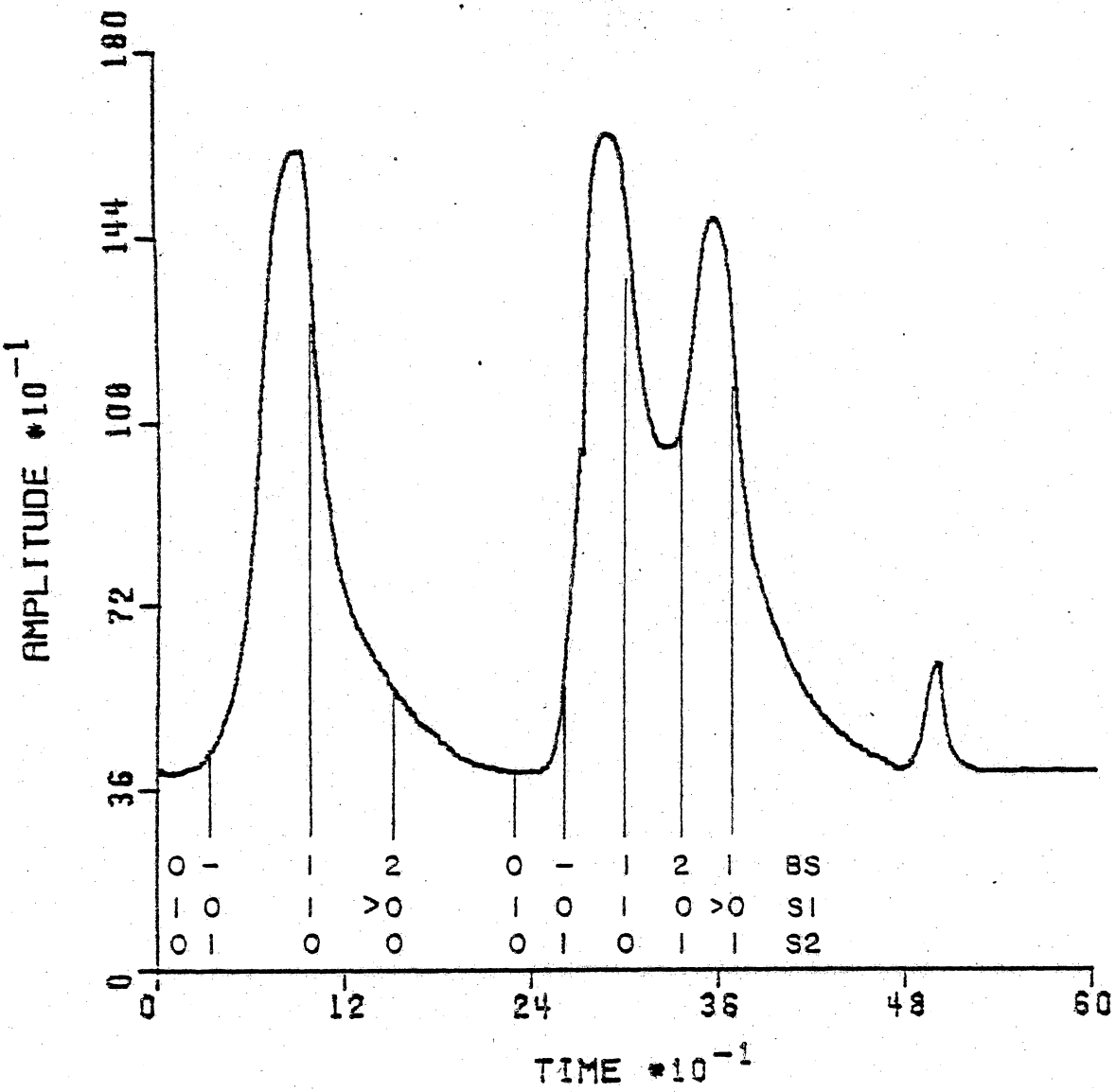
2 - looking for termination on baseline

S1 - Set to 1 to indicate data on the backside of a peak

S2 - Set to 1 to indicate an increasing trend

The program takes a point and performs tests on it to determine whether it is a baseline, a peak maximum or an increasing or decreasing trend. At the end of each peak, the peak parameters are output as decimal values. The counters are reset and the next peak is searched for. At the end of the buffer, if there is a peak only partially processed, because the peak overlaps the end of the buffer, a flag is set. The flag causes the last block to be read as the first block of the next buffer. The peak program begins at the minimum just before the last partial peak so that the whole peak is analyzed.

A post run plotter is available. The routine to run the plotter was written by David Binkley. It runs on the Host computer controlling a Bensen Lehner plotter. This can be run only after all data runs have been performed and stored on the floppy disk.



Peak Processor Software Switch Example

FIGURE 8

EXPERIMENTAL

The ability of the processor to control the L.C. under typical operating conditions was tested first by using a sample that was easily obtainable and well characterized. A mixture of benzene, naphthalene and anthracene was used. This sample is called Sample 2 - Run 1. The conditions of the run are listed in figure 9 and the plot is in figure 10. The report format gives all the pertinent information about the run. The peak areas are listed as relative values and the retention times in seconds.

In figure 11, the full peak parameters for four consecutive runs of the synthetic mixture are given. They are all analyzed with the same peak factors, giving them a consistent basis for analysis. The plot of this mixture is shown in figure 12. The values from run to run are not the same. An analysis of this difference is shown in table 6. It can be seen that most of the values have less than two percent error while many have zero percent error. The data analysis by the peak processor is very consistent over consecutive runs of the synthetic mixture.

To determine the effect of the peak factors, one run of the synthetic mixture was analyzed by the peak processor four times. The factors WIDTH, GATE and CHANGE were set to different values. The results of these changes can be seen in figure 13. In run 1 the factors are set low. This causes the peak data to be distorted and misinterpreted. Run 2 shows the factors in a good analysis. The peaks are all represented, without distortion. In run 3,

the factors are high enough to eliminate the last peak. The other peaks are still normal. Run 4 shows the effect of having the factors too high. The fourth peak is not seen and the second and third peaks have become fused. The peak factors are very important in performing an analysis. They must be sensitive enough to detect all peaks and yet not distort them.

1 SAMPLE
NAME: BENZENE, NAPHTHALENE AND ANTHRACENE IN HEXANE (SAMPLE 2)
VOLUME: 10 ul
CONC: 2 mg/ml

2 COLUMN SPECIFICATIONS
MANUFACTURER: USER MADE
PACKING: SPHERISORB
LENGTH: 20cm
INNER DIAMETER:
COLUMN PRESSURE: 45 psi
TEMP(C) = 27

3 DETECTOR
TYPE: U.V.
O.D. VALUE: 1.28
MODE: SINGLE WAVELENGTH
WAVELENGTH: 254 nm

4 RUN TIME(MIN) = 5
FLOWRATE(ml/min x 0.04) = 70
DATA RATE = 1

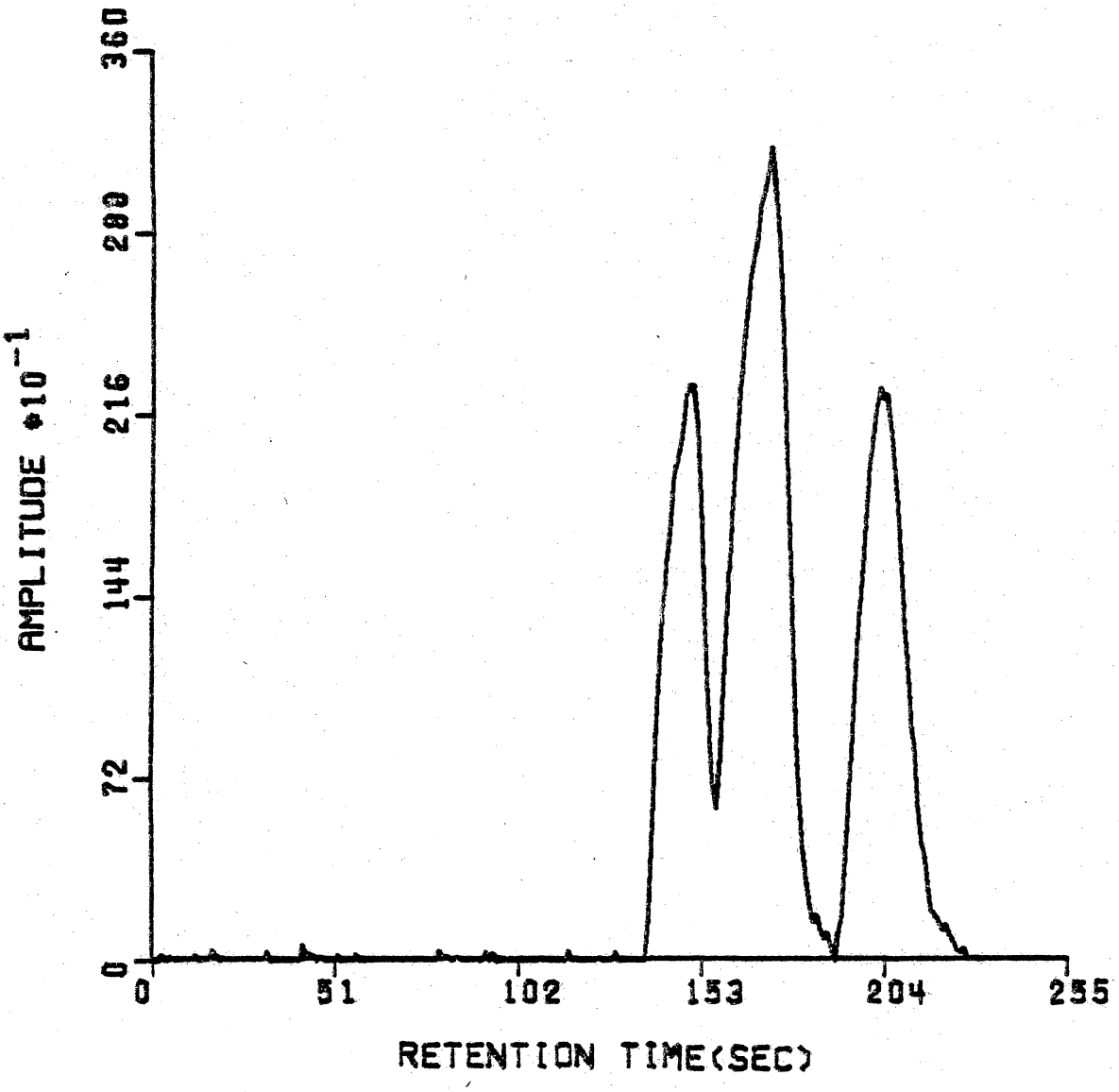
5 MOBILE PHASE
A: N-HEXANES
B: N.A.
GRADIENT SLOPE: 0
%A AT START: 100
%A AT END: 100

WIDTH FACTOR = 2
GATE = 5
CHANGE FACTOR = 6

PEAK NO.	PEAK AREA	RETENTION TIME(SEC)
1	29736	149
2	51551	172
3	33808	202

Sample 2 - Run 1

FIGURE 9



SAMPLE 2 - RUN 1

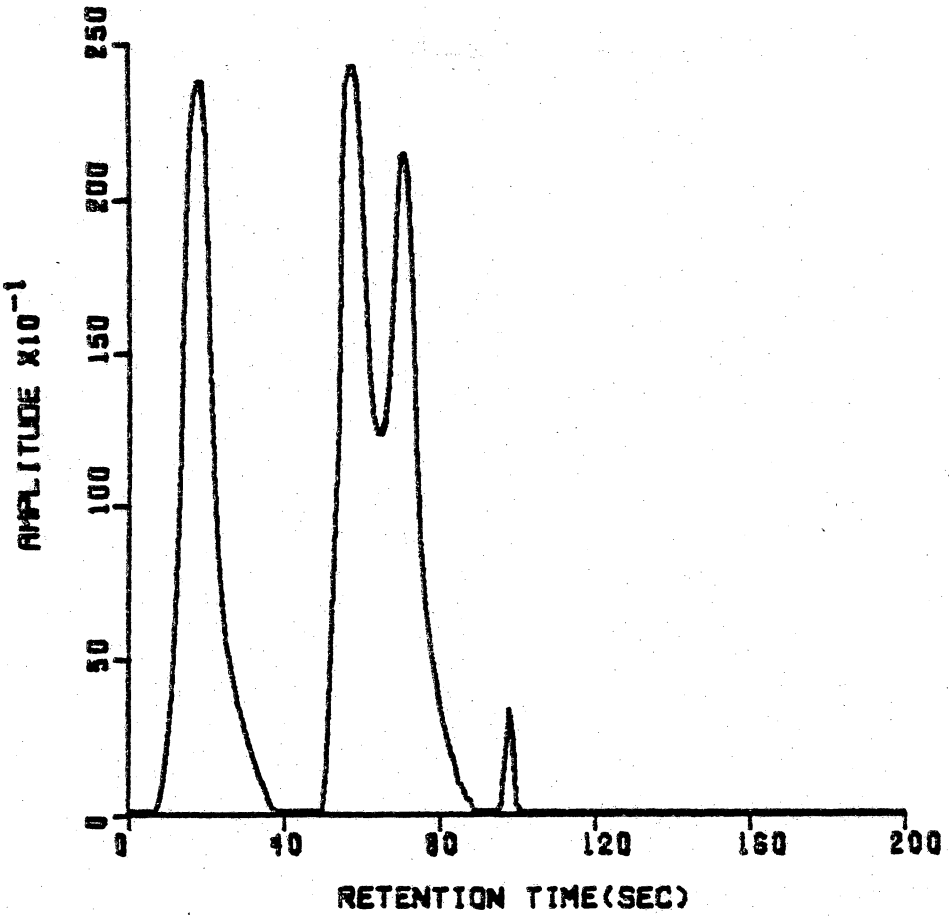
FIGURE 10

WIDTH FACTOR 3
GATE FACTOR 2
CHANGE FACTOR 2

PEAK NO.	PEAK AREA	PEAK HEIGHT	PEAK TIME (SEC)	LEADING MINIMUM HEIGHT	TIME OF LEADING MINIMUM	WIDTH	TRAILING MINIMUM HEIGHT	TIME OF TRAILING MINIMUM	PEAK TYPE
RUN 1									
00	24493	2381	17	00	01	04	00	37	01
01	22192	2425	56	00	37	03	1225	63	00
02	21863	2145	70	1225	63	03	00	89	01
03	808	337	97	00	89	03	00	100	01
RUN 2									
00	24427	2381	17	00	01	04	00	36	01
01	22997	2432	56	00	36	07	1209	63	00
02	21205	2157	69	1209	63	03	00	88	01
03	834	363	97	00	88	03	00	100	01
RUN 3									
00	24414	2387	17	00	01	04	00	36	01
01	23082	2421	56	00	36	03	1229	63	00
02	21129	2149	69	1229	63	03	00	88	01
03	832	357	97	00	88	03	00	100	01
RUN 4									
00	24518	2389	17	00	01	04	00	37	01
01	22232	2425	56	00	37	03	1227	63	00
02	21901	2153	70	1227	63	03	00	89	01
03	819	345	97	00	89	03	00	100	01

Consecutive Analysis of
Synthetic Mixture - Long Format

FIGURE 11



SYNTHETIC MIXTURE

FIGURE 12

PEAK NO.	PEAK AREA	PEAK HEIGHT	PEAK TIME (SEC)	LEADING MINIMUM HEIGHT	TIME OF LEADING MINIMUM	WIDTH	TRAILING MINIMUM HEIGHT	TIME OF TRAILING MINIMUM	PEAK TYPE
RUN 1 - Low Peak Parameters									
WIDTH FACTOR 1									
GATE FACTOR 1									
CHANGE FACTOR 1									
00	24493	2381	17	00	01	04	00	37	00
01	22192	2425	56	00	37	03	1225	63	00
02	17357	2145	70	1225	63	03	1157	73	01
03	808	337	97	00	89	02	00	100	00
RUN 2 - Good Peak Parameters									
WIDTH FACTOR 3									
GATE FACTOR 2									
CHANGE FACTOR 2									
00	24493	2381	17	00	01	04	00	37	01
01	22192	2425	56	00	37	03	1225	63	00
02	21863	2145	70	1225	63	03	00	89	01
03	808	337	97	00	89	03	00	100	01
RUN 3 - High Peak Parameters, Loss of Peak									
WIDTH FACTOR 4									
GATE FACTOR 6									
CHANGE FACTOR 7									
00	24493	2381	17	00	01	08	00	37	01
01	22192	2425	56	00	37	03	1225	63	00
02	21863	2145	70	1225	63	07	00	89	01
RUN 4 - High Peak Parameters, Fusing of Peaks									
WIDTH FACTOR 4									
GATE FACTOR 7									
CHANGE FACTOR 7									
00	24493	2381	17	00	01	09	00	37	01
01	43443	2425	56	00	37	17	00	89	01

Effect of Changing Peak Factors

FIGURE 13

TABLE 6

Peak Data Accuracy Analysis

PEAK NO.	PEAK AREA	PEAK HEIGHT	PEAK TIME (SEC)	LEADING MINIMUM HEIGHT	TIME OF LEADING MINIMUM	WIDTH	TRAILING MINIMUM HEIGHT	TIME OF TRAILING MINIMUM	PEAK TYPE
00	±0.22	± 0.19	0.0	0.0	0.0	0.0	0.0	± 1.3	0.0
01	± 2.0	± 0.26	0.0	0.0	± 0.72	± 75	± 1.1	0.0	0.0
02	± 1.8	± 0.28	± .72	± 1.1	0.0	0.0	0.0	± 0.56	0.0
03	± 1.8	± 3.8	0.0	0.0	± 0.56	0.0	0.0	0.0	0.0

All Values are Percents

CONCLUSION

The goals of this project were all attained. The instrument requests the operating parameters from the user and keeps them in a report format. They are stored as a leader to the data file. All data collected is stored on the floppy disk system. The operator does minimal interaction with the L.C. and so is left to perform other tasks in the laboratory while the instrument performs an analysis.

The data networks system worked out well. Fortran was attempted but was found to be inoperable on the satellite system due to a fault in the Fortran simulator. Assembly language was used to replace the Fortran routines. Using the Host to build the programs and edit them made the software construction much easier. The transmission of the data also worked well. The only factor causing any difficulty was the transmission of data back to the host. The problems were due to worse case response of the host (approximately two seconds) and the block transfer format demanded by DDCMP. The first was solved by halting the flow of the solvent during the data transmission process. This short delay does not result in any significant band broadening. The block transfer both to and from the processor could result in peaks being "trapped" between two adjacent frames of data. Software detection of this led to the solution of rereading the last portion of a block to collect the first portion of the trapped peak. This is then put at the beginning of the next block so that the peak is whole.

REFERENCES

1. Computer Networking - A Rational Approach to Lab Automation, Raymond E. Dessy, Analytical Chemistry, Vol. 49, No. 13, November 1977, page 1100 A.

2. Cooper, James W. The Mini-Computer in the Laboratory, Wiley-Interscience, New York, 1977.

3. Bugbooks I & II, Larsen and Rony, E & L Instruments, Inc. Southern Printing, 1974.

4. Guide to Electronic Measurements and Laboratory Practice, Stanley Wolf, Prentice Hall, New Jersey, 1973.

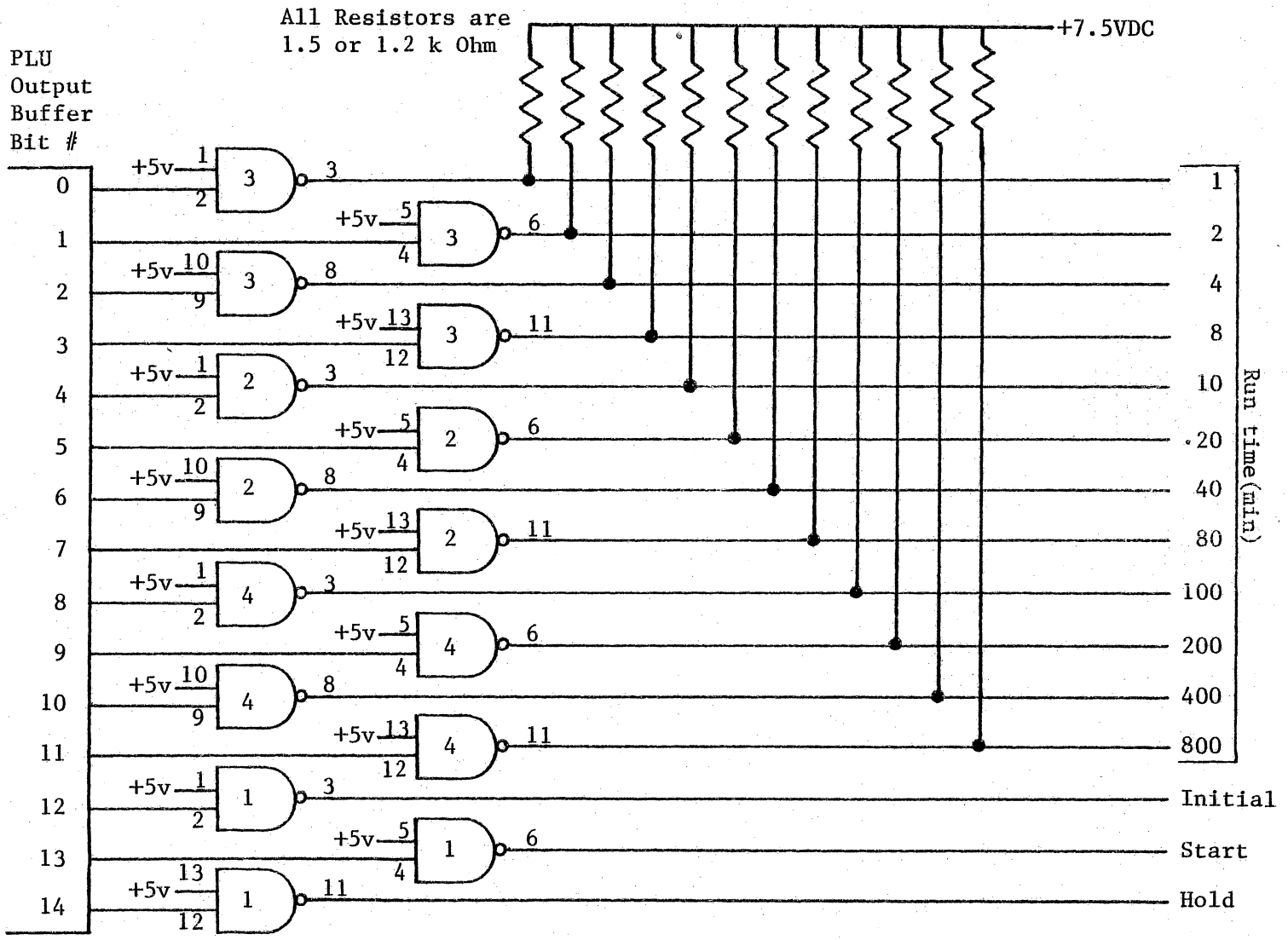
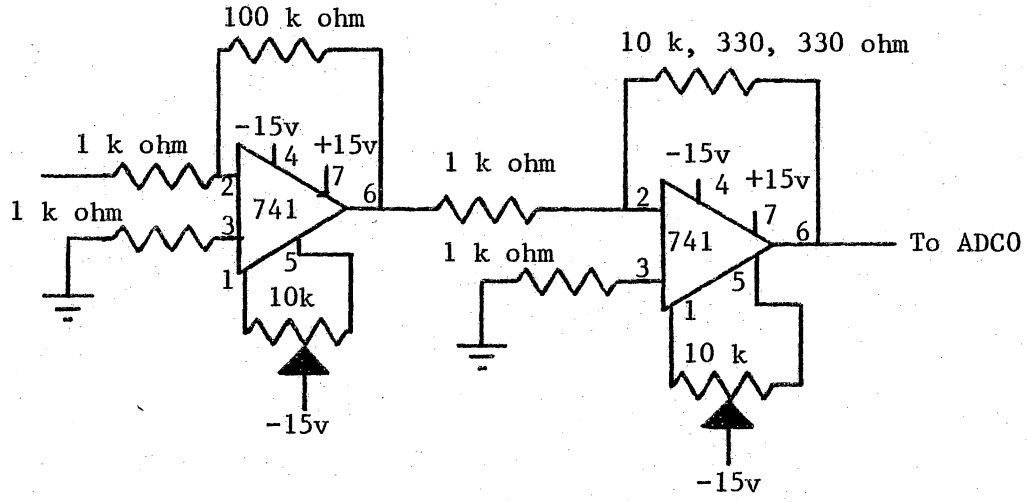
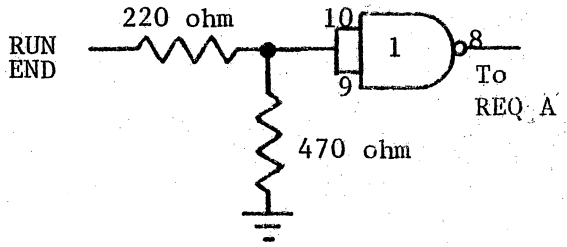


FIGURE 15 - L.C. Interface Control Schematic





Chip Layout for L.C. Interface Control

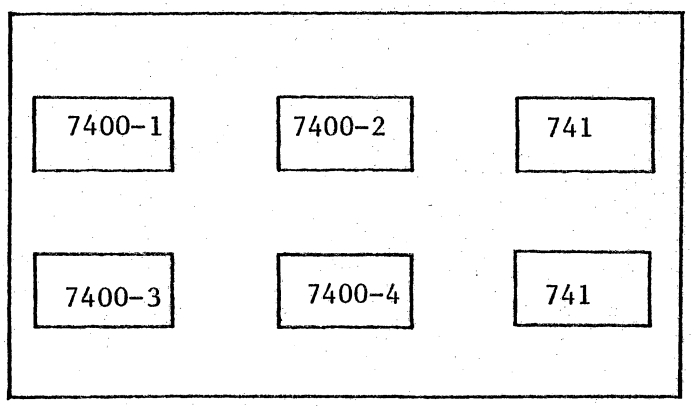


FIGURE 16 - L.C. Interface Control Schematic, Continued

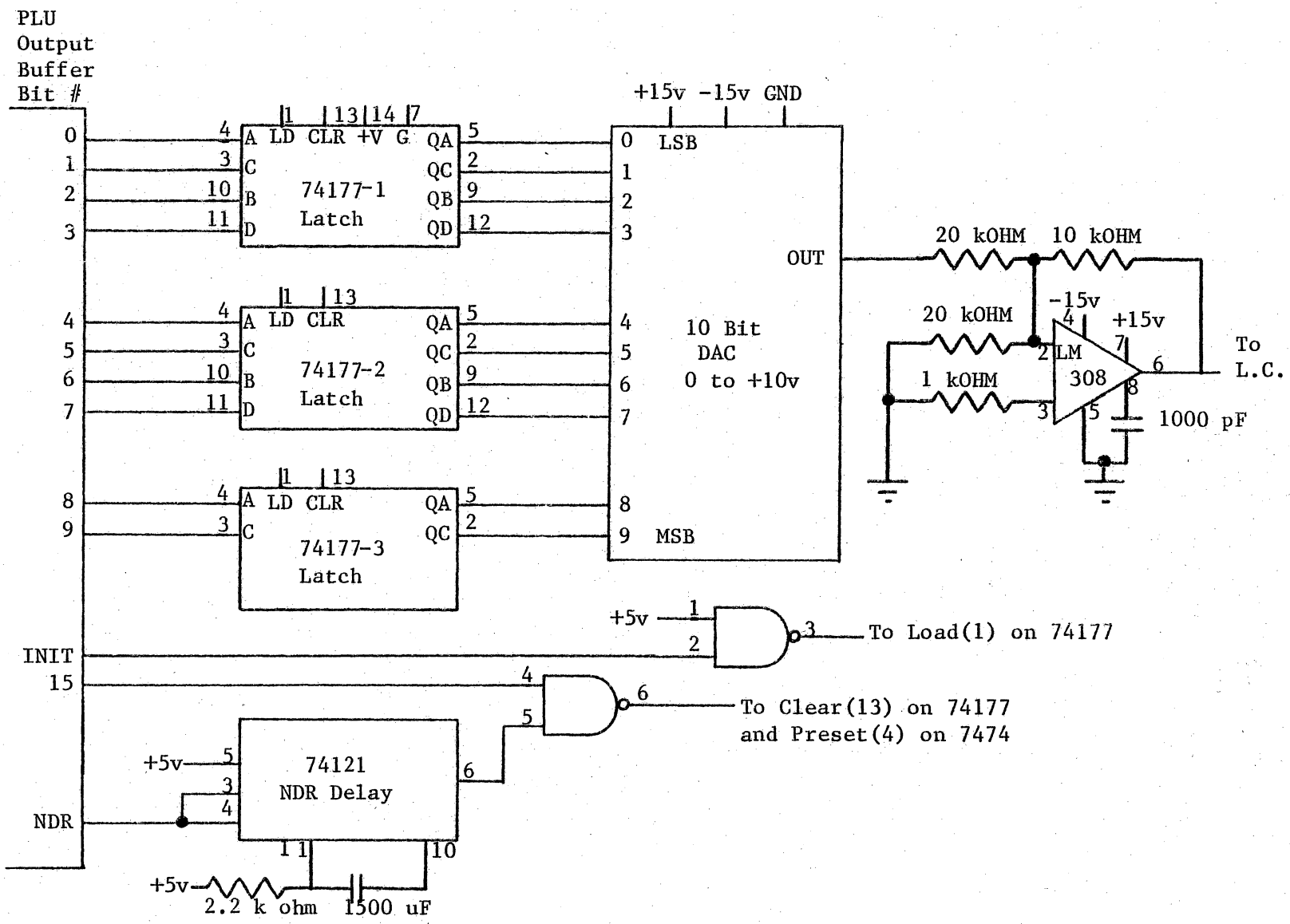


FIGURE 17 - Flow Rate Control

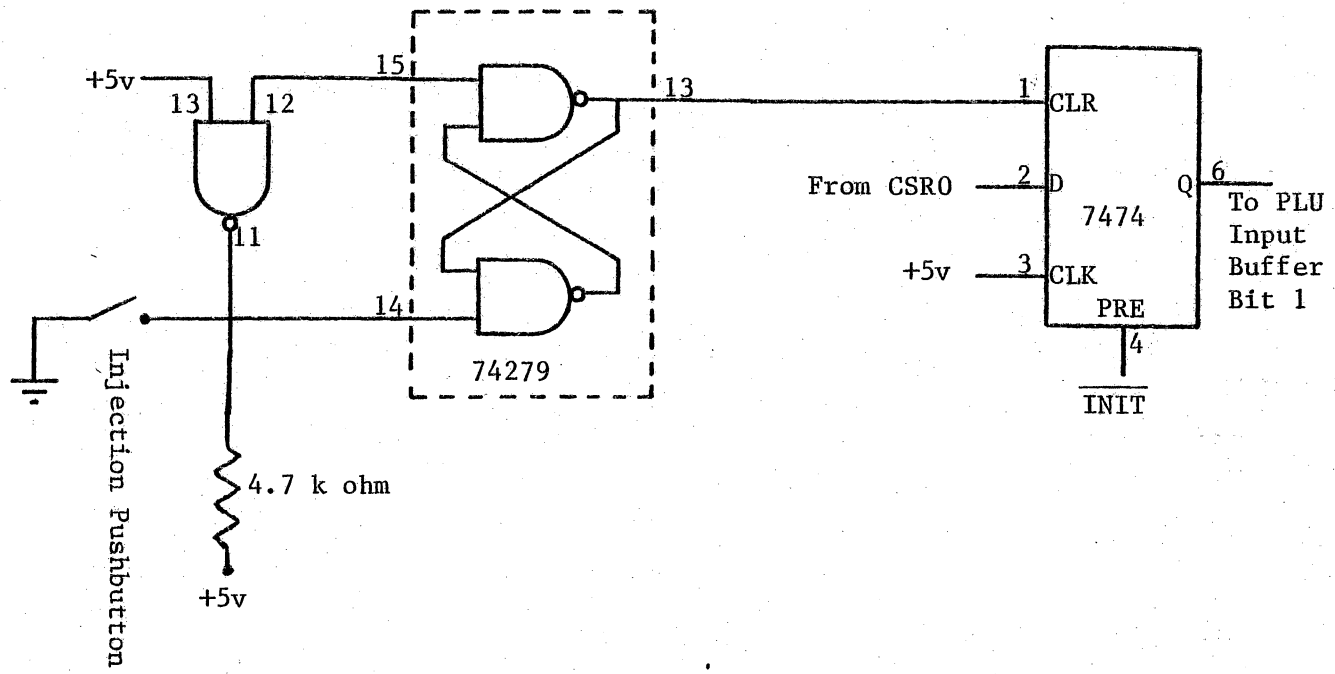


FIGURE 18 - Flow Rate Control - Continued

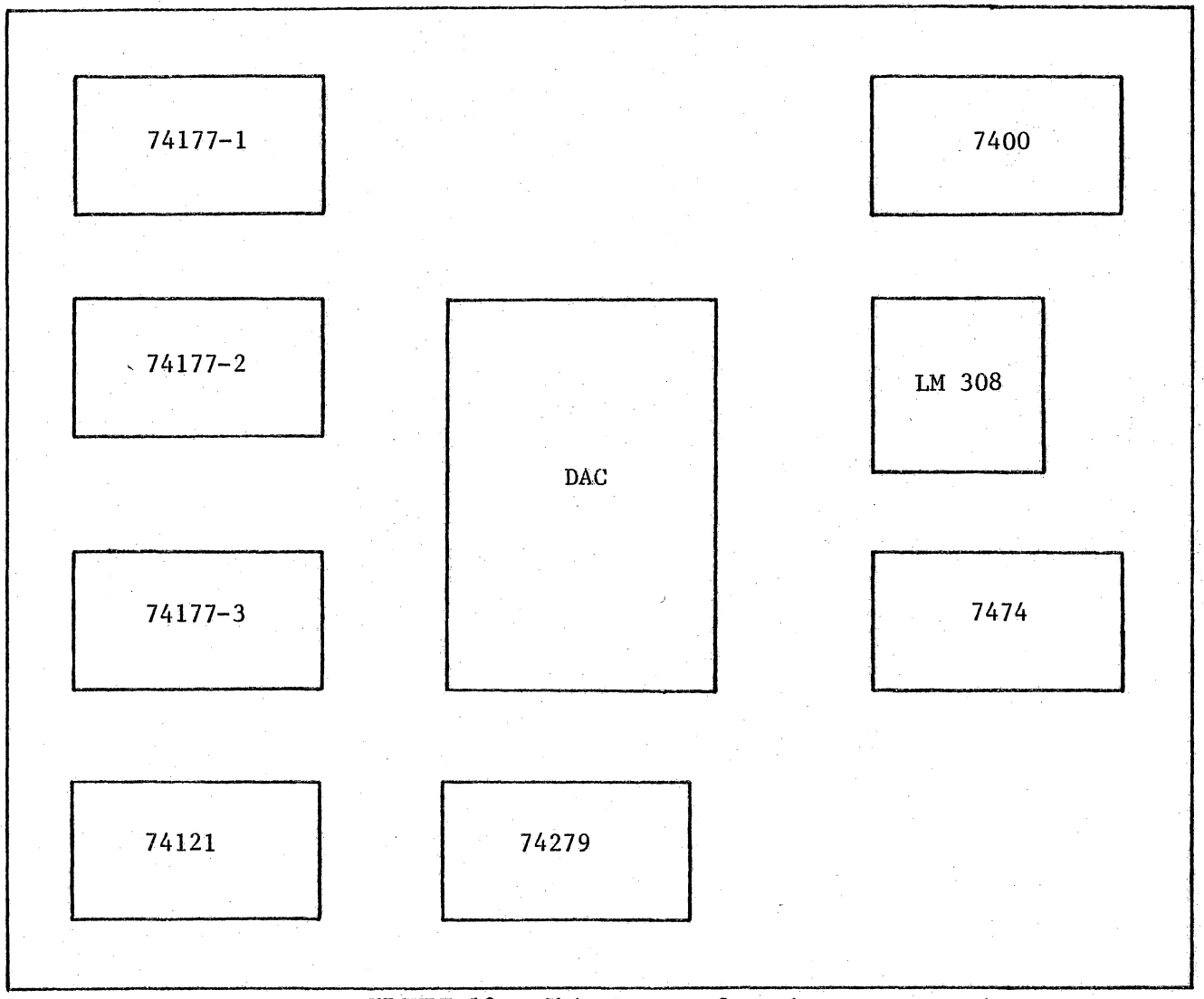


FIGURE 19 - Chip Layout for Flow Rate Board

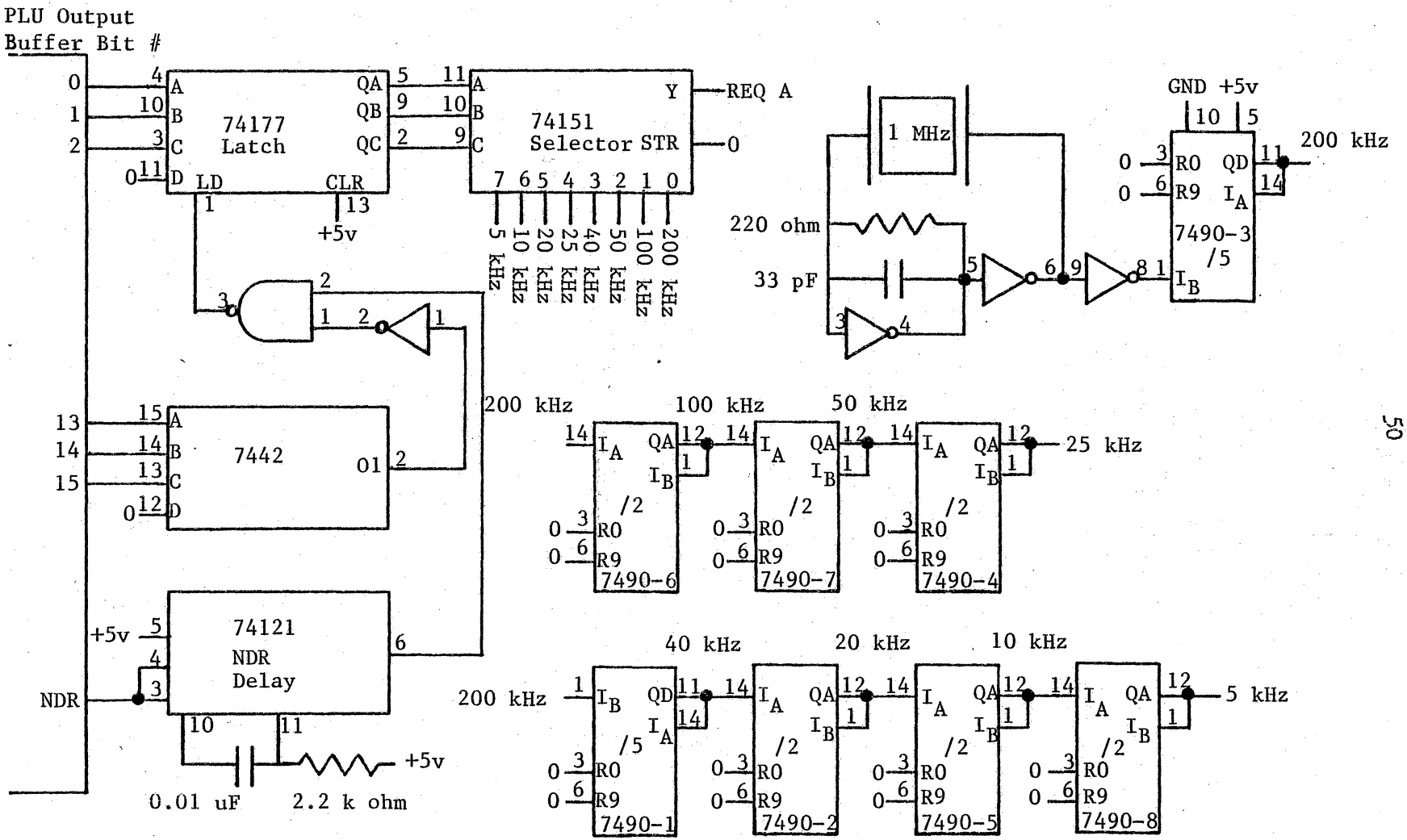


FIGURE 20 - Timer Ø Schematic

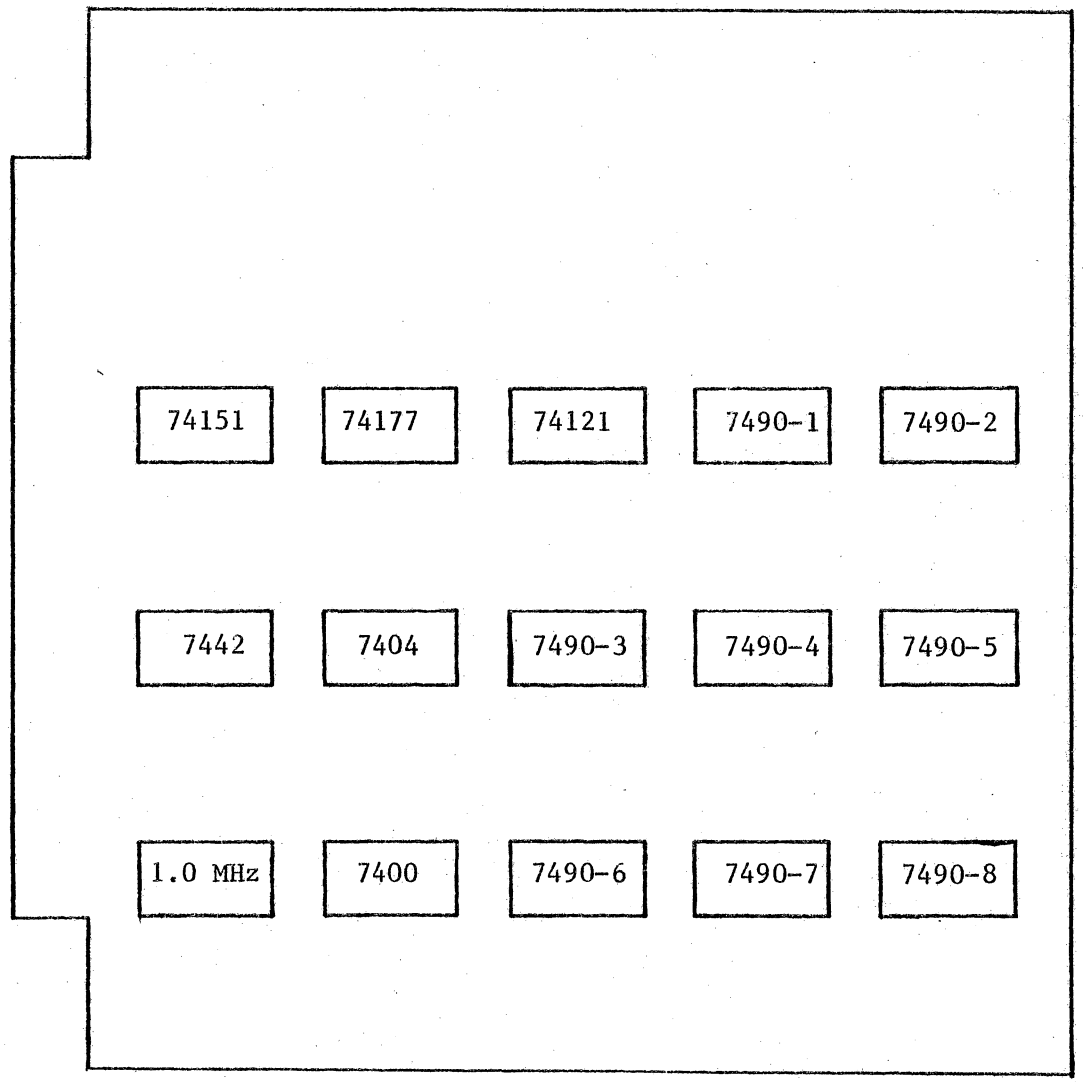


FIGURE 21 - Chip Layout for Timer Board Ø

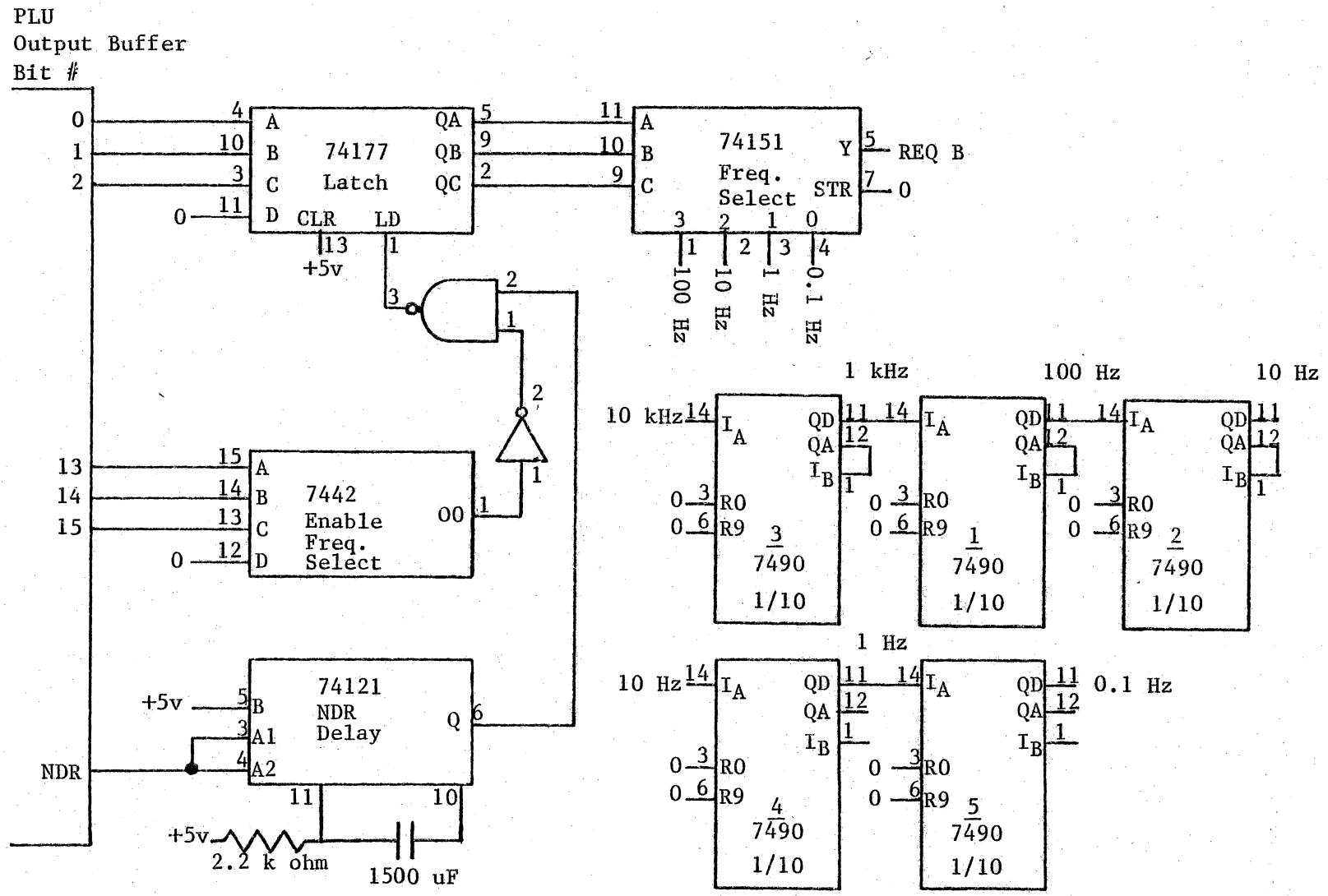


FIGURE 22 - Timer 1 Schematic

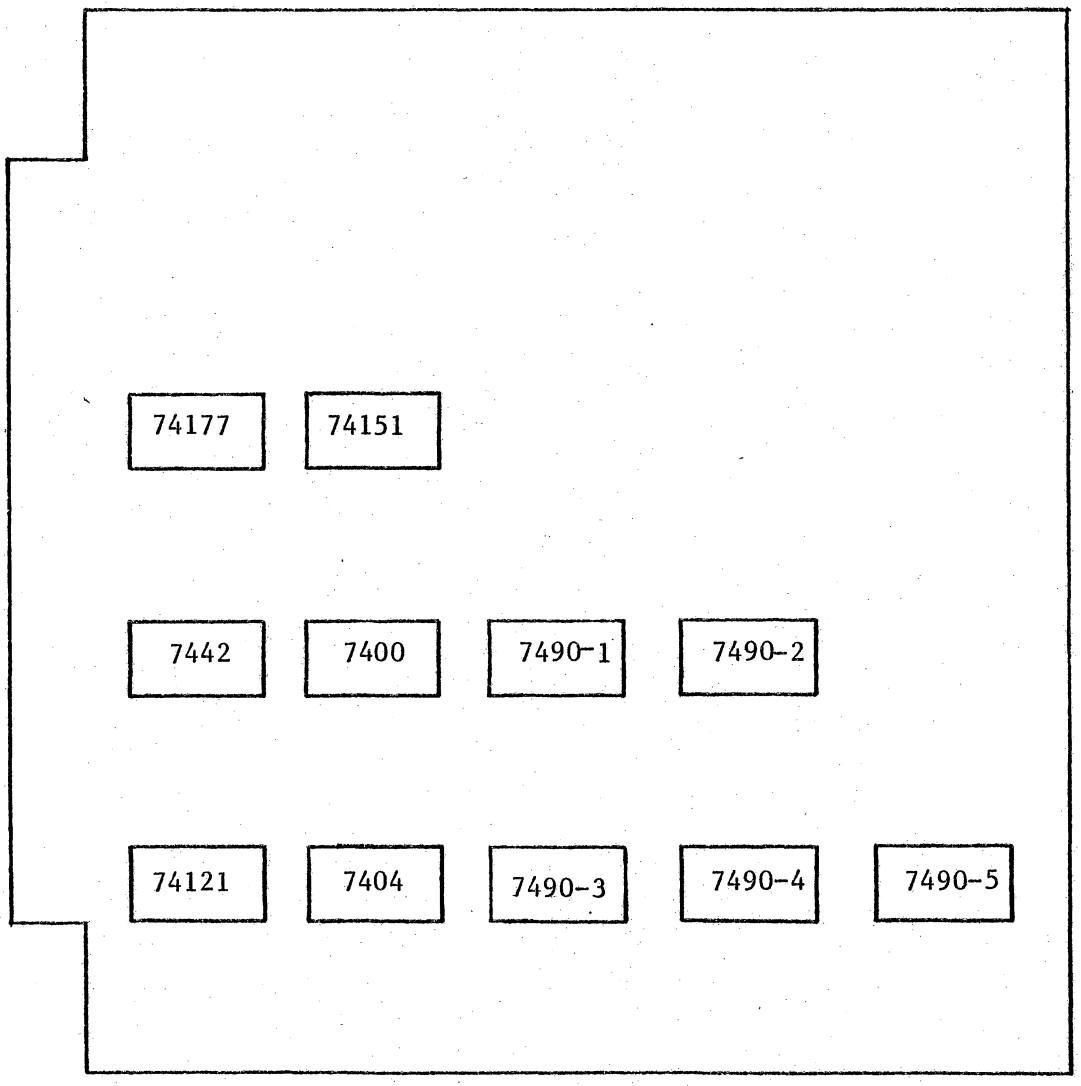


FIGURE 23 - Chip Layout of Timer Board 1

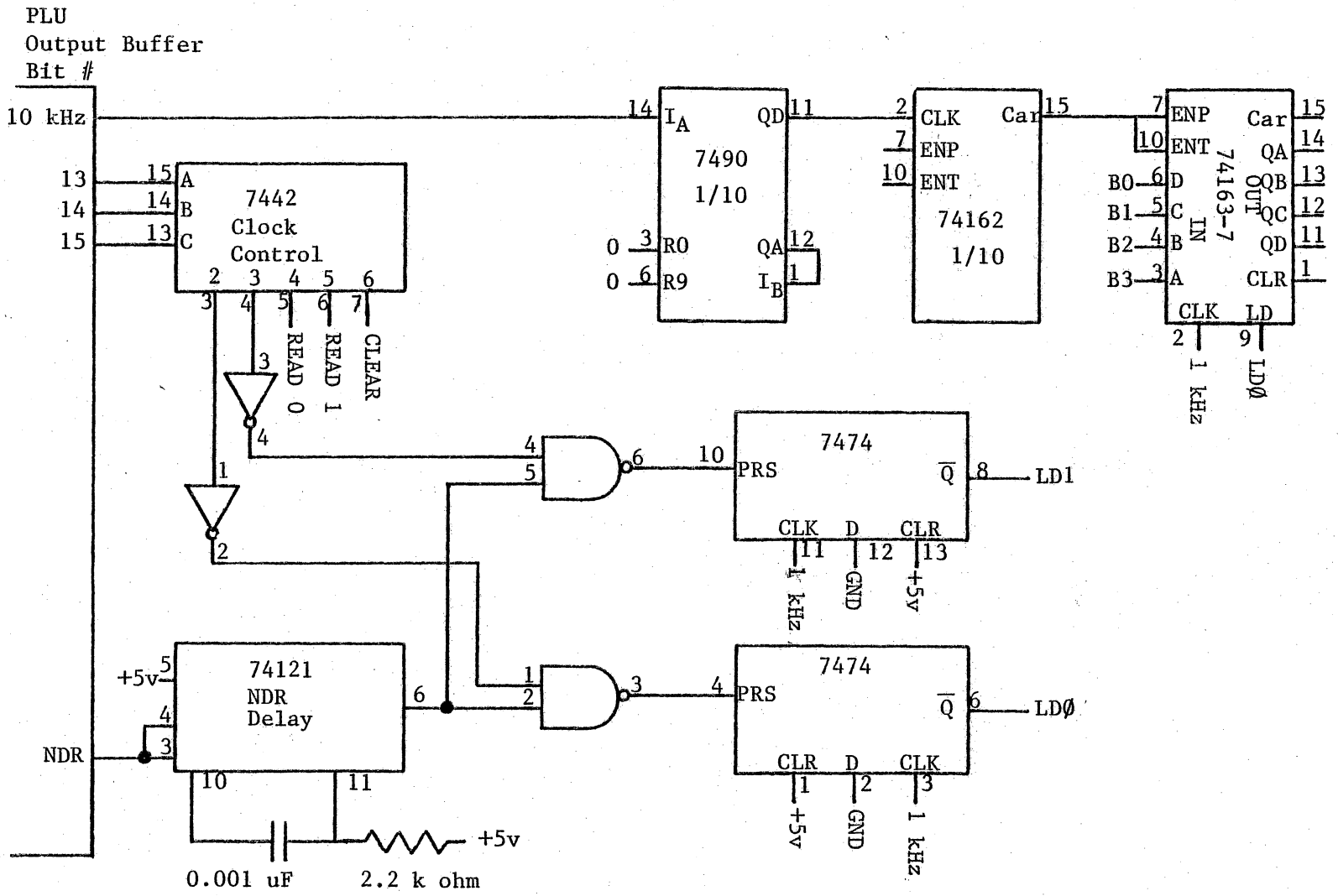


FIGURE 24 - Day Clock Control

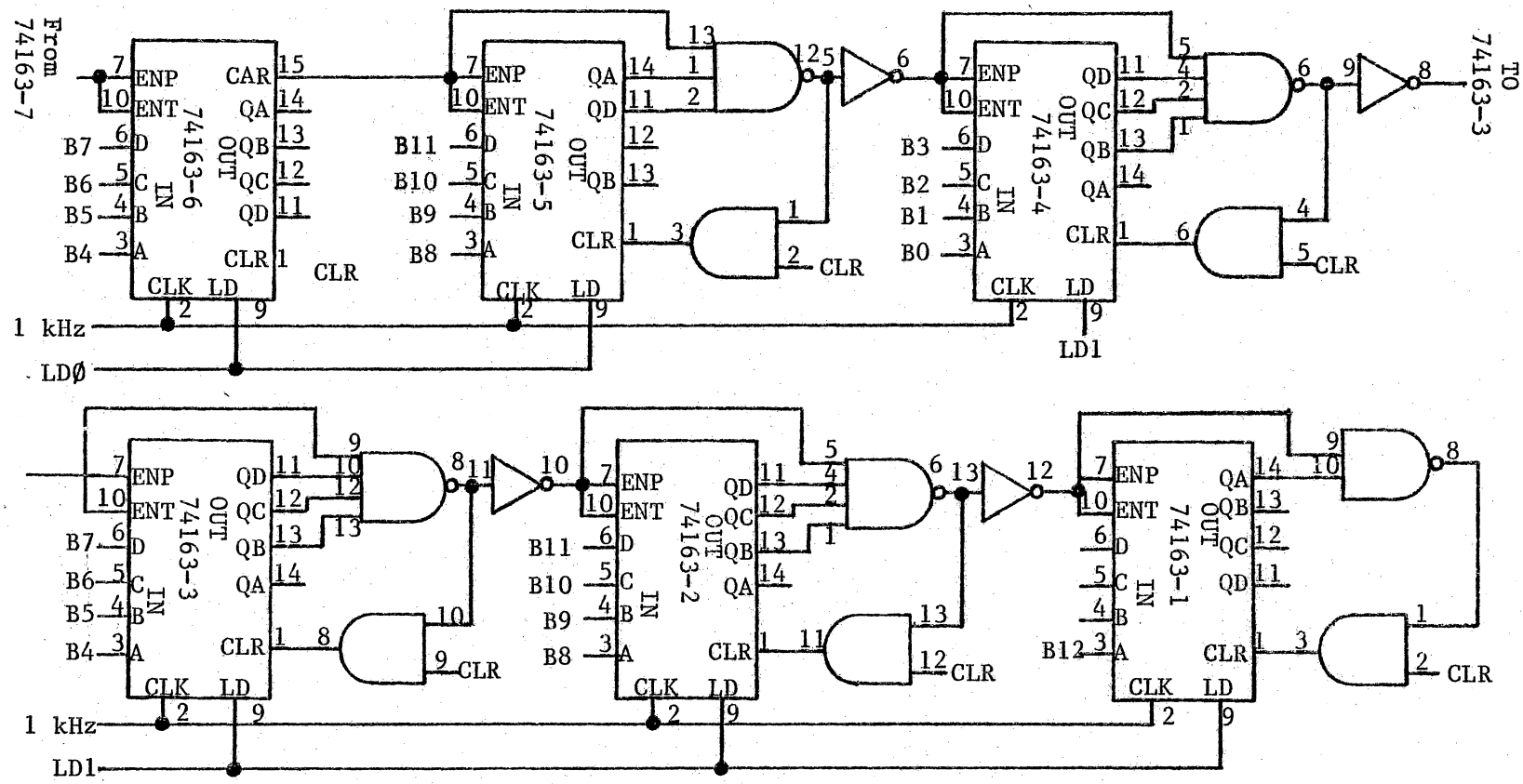
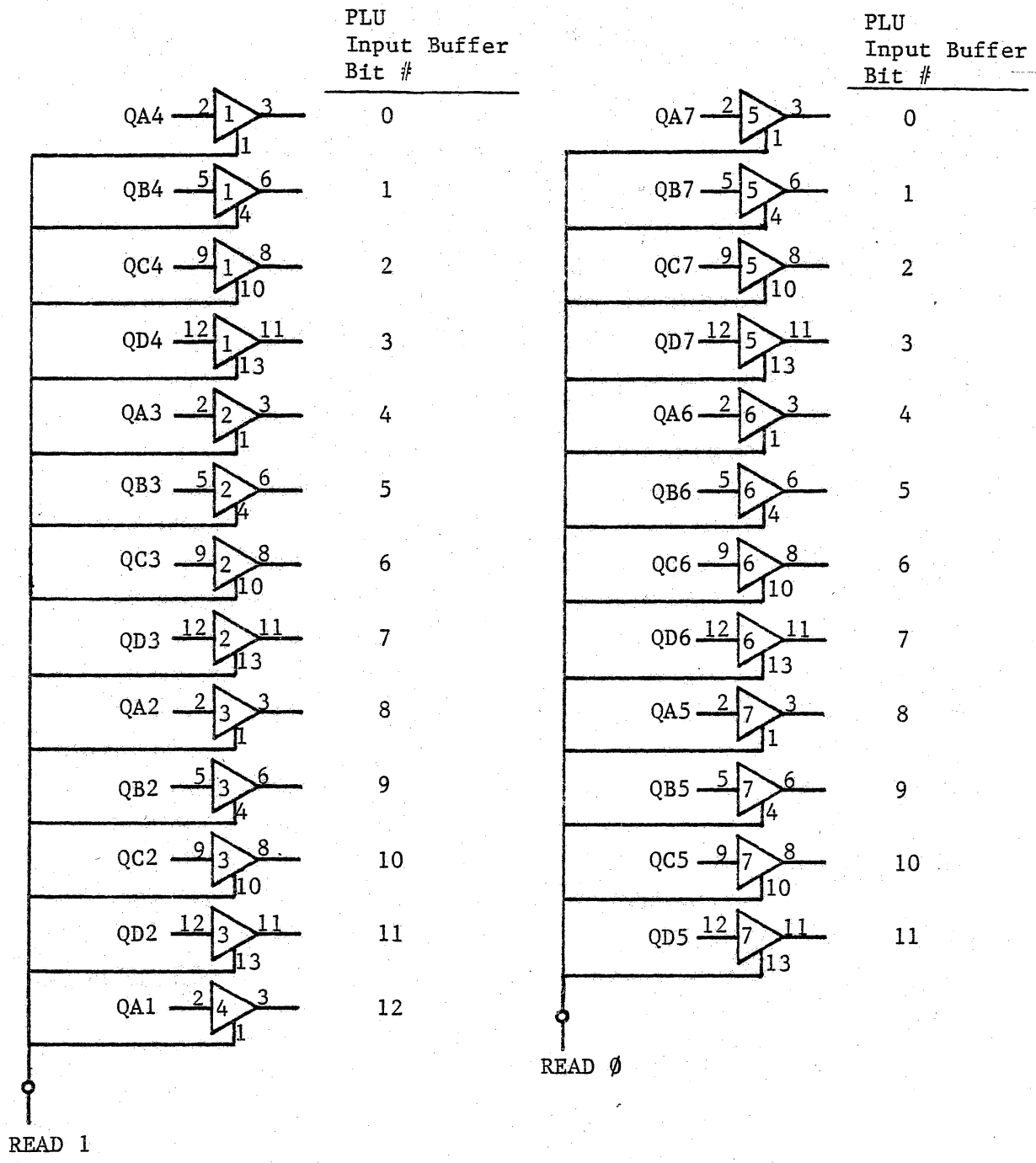


FIGURE 25 - Day Clock



Day Clock, Continued

FIGURE 26

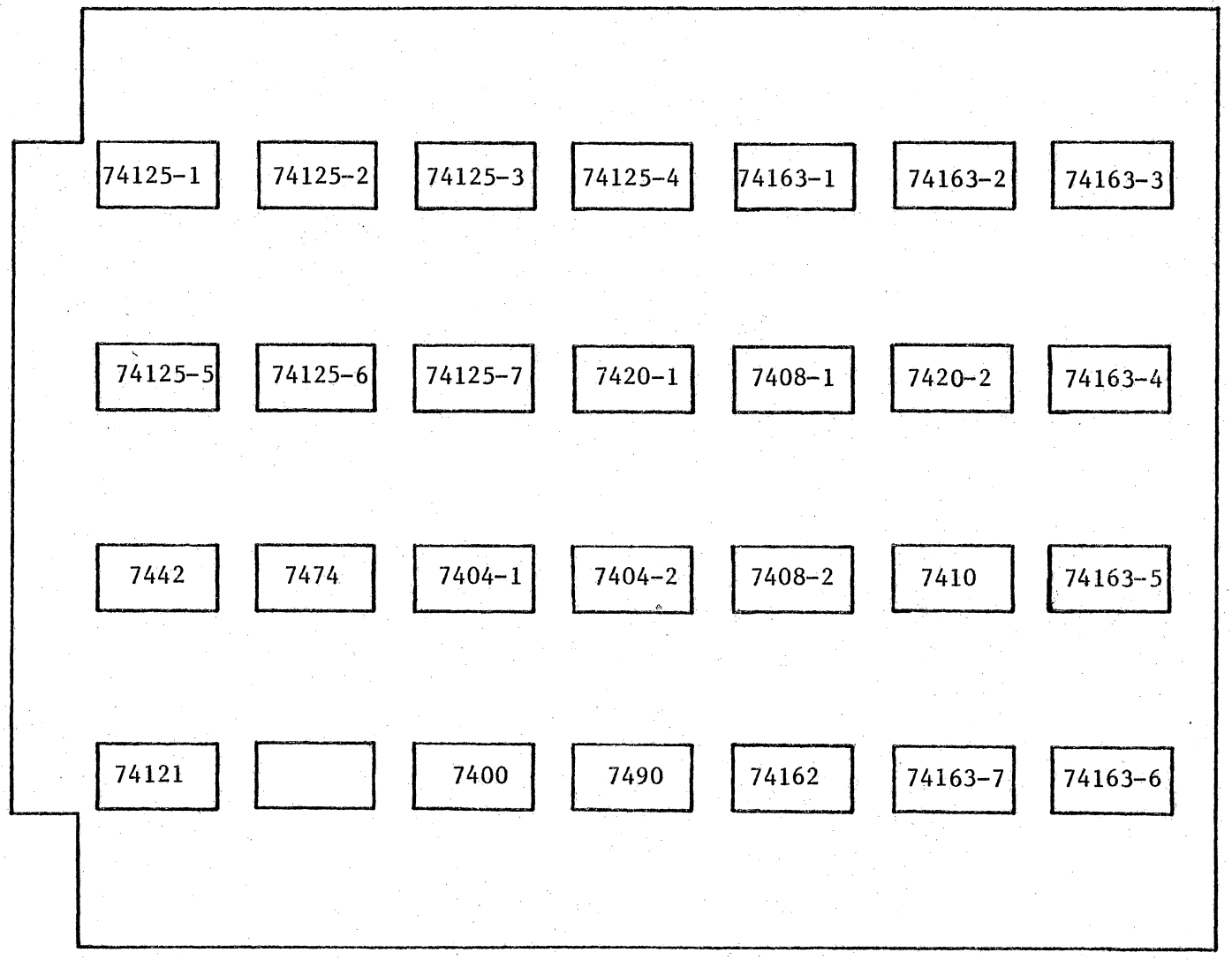


FIGURE 27 - Chip Layout for Day Clock

APPENDIX A

Report Routine Flowchart

A. Start routine

B. Initialize flowrate and run time

C. Get Date, Operator Name, and Run Number

D. Set KEY value

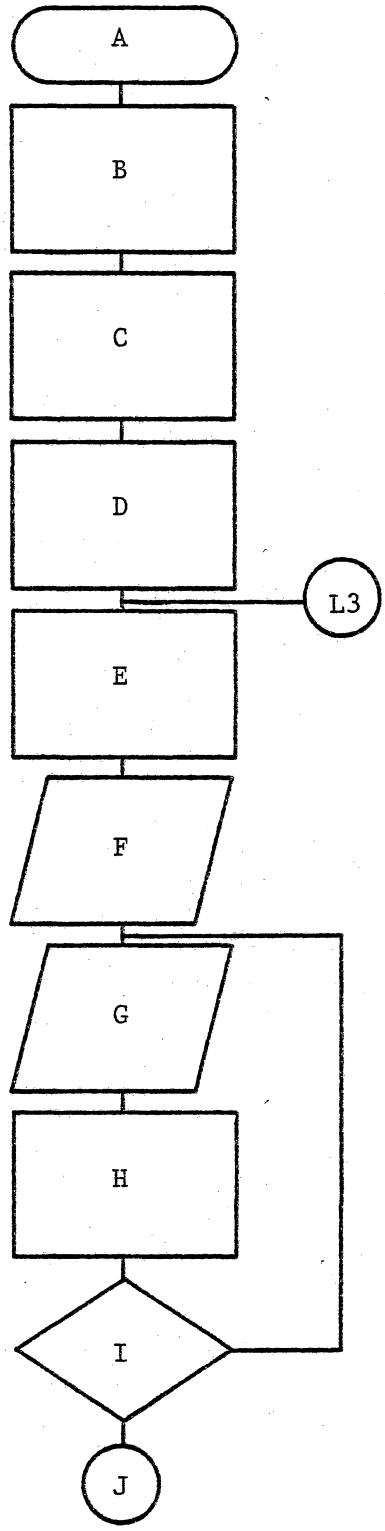
E. Set up RAD50 file name

F. Ask file name

G. Get character and echo it

H. Strip extra bits off, store it, and change counters

I. Are all 12 characters input yet?
If they are, go to J
If not, go back to G



K. Call RAD50 conversion routine

L. Shift bits in KEY

M. Do section 1?
   If yes, go to N
   If not, go to O

N. Get sample parameters

O. Shift bits in KEY

P. Do section 2?
   If yes, go to Q
   if not, go to R

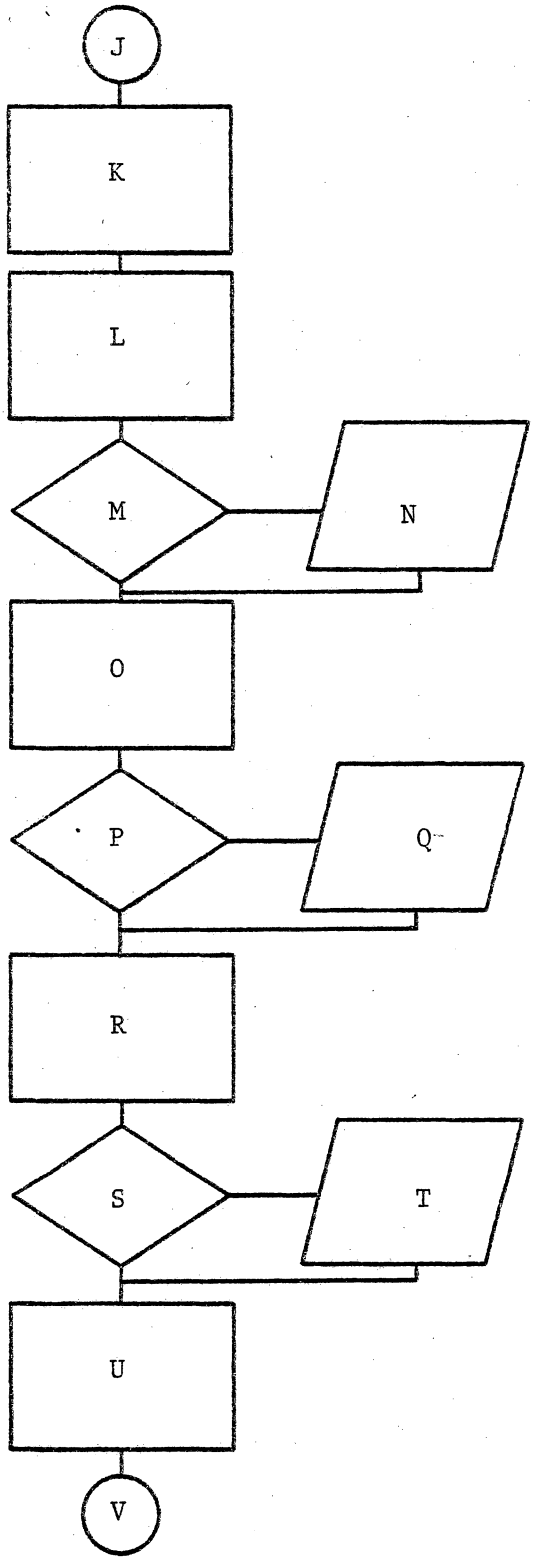
Q. Get column specifications

R. Shift bits in KEY

S. Do section 3?
   If yes, go to T
   If not go to U

T. Get detector parameters

U. Shift bits in KEY



W. Do section 4?
   If yes, go to X
   If not, go to D1

X. Get Run Time

Y. Is it within range?
   If it is, go to Z
   If not, go to X

Z. Get Flowrate

A1. Is it in range?
   If it is, go to B1
   If not, go to Z

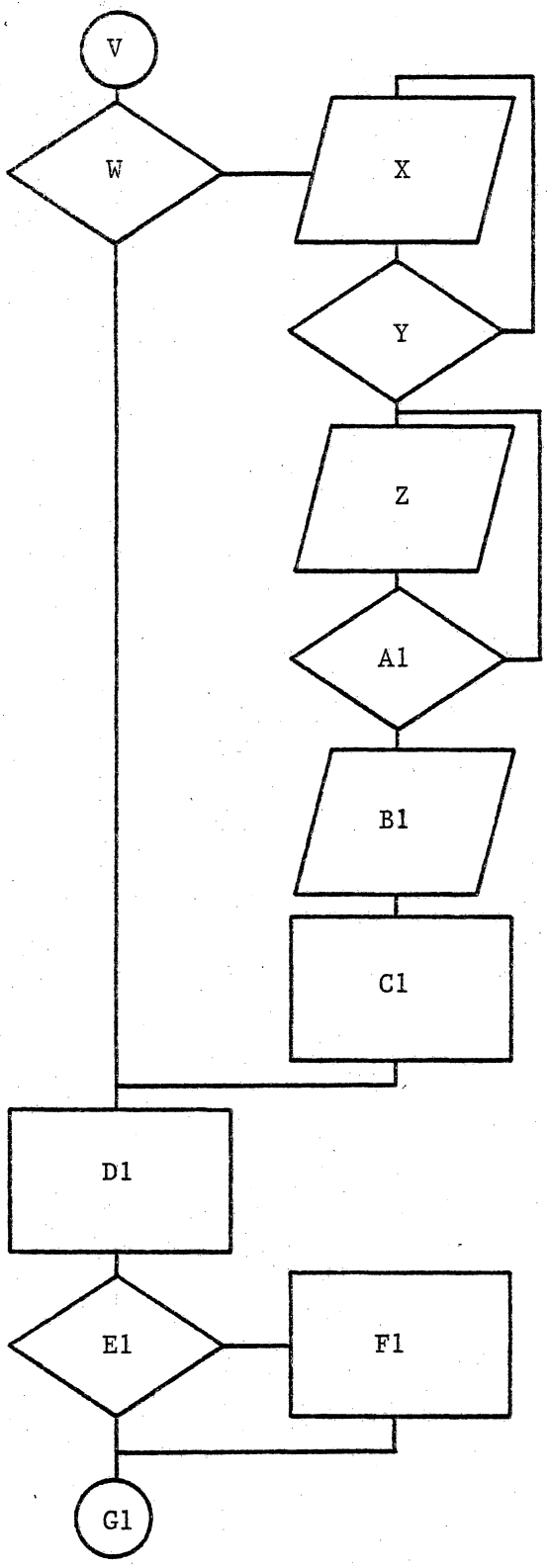
B1. Get Data Rate

C1. Move to output buffer on PLU

D1. Shift bits in KEY

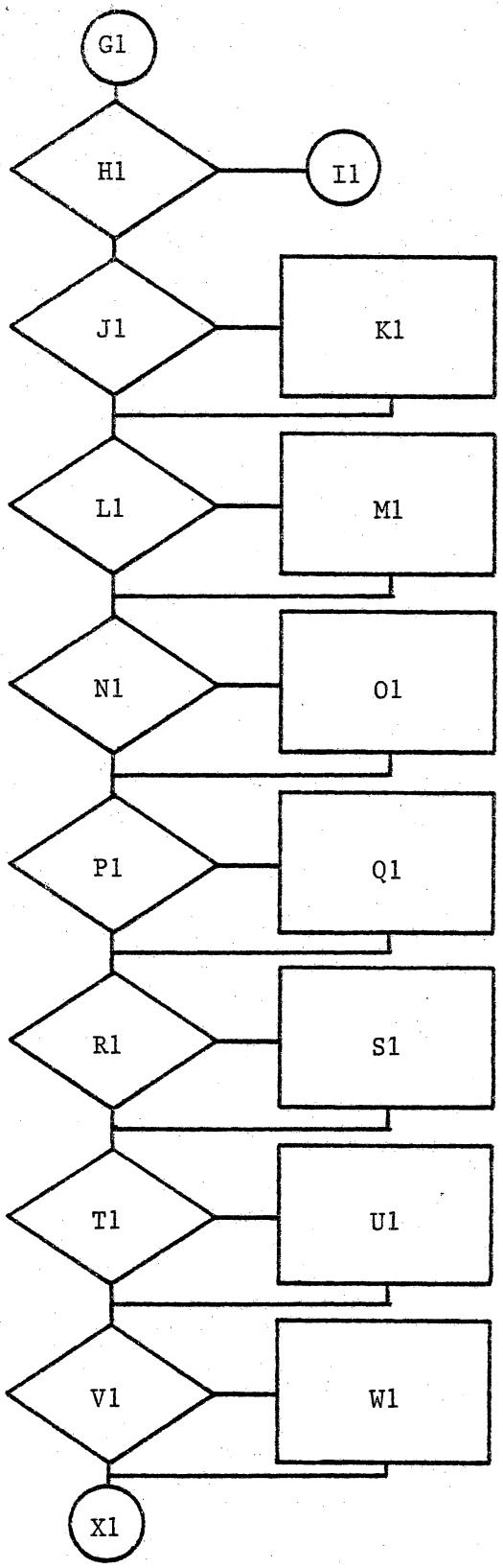
E1. Do section 5?
   If yes, go to F1
   If not, go to G1

F1. Get mobile phase parameters

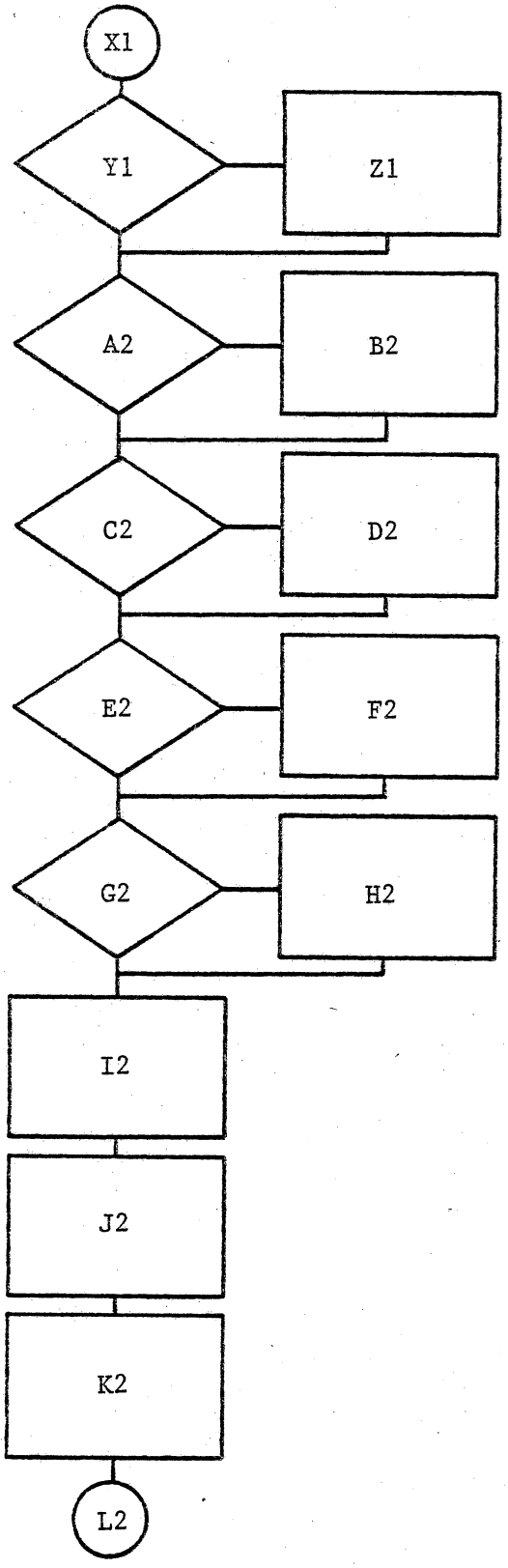


H1. Calculate flowrate and run time?
    If yes, go to J1
    If not, go to I1

J1. Time GT 1440?
    If it is, go to K1
    If not, go to L1
K1. Subtract 1440 from time R2 + 4000
L1. Time GT 620?
    If it is, go to M1
    If not, go to N1
M1. Sub 620 from time,R2 + 2000
N1. Time GT 310?
    If it is, go to O1
    If not, go to P1
O1. Sub 310 from time, R2 + 1000
P1. Time GT 144?
    If it is, go to Q1
    If not, go to R1
Q1. Sub 144 from time, R2 + 400
R1. Time GT 120?
    If it is, go to S1
    If not, go to T1
S1. Sub 120 from time, R2 + 200
T1. Time GT 50?
    If it is, go to U1
    If not, go to V1
U1. Sub 50 from time,R2 + 100
V1. Time GT 24?
    If it is, go to W1
    If not, go to X1
W1. Sub 24 from time, R2 + 40



Y1. Time GT 12?
    If it is, go to Z1
    If not, go to A2
Z1. Sub 12 from time, R2 + 20
A2. Time GT 10?
    If it is, go to B2
    If not, go to C2
B2. Sub 10 from time, R2 + 10
C2. Time GT 4?
    If it is, go to D2
    If not, go to E2
D2. Sub 4 from time, R2 + 4
E2. Time GT 2?
    If it is, go to F2
    If not, go to G2
F2. Sub 2 from time, R2 + 2
G2. Time GT 1?
    If it is, go to H2
    If not, go to I2
H2. Sub 1 from time, R2 + 1
I2. Take complement of R2 and
    move to sweep
J2. Get flowrate and multiply it
    by 82_{10}
K2. Divide flowrate by 8



M2. Set bit fifteen of flowrate

N2. Set flowrate and initialize L.C.

O2. Put flowrate and run time in
a register for control
routine

P2. Call Control routine

Q2. Output 'Do another run?'

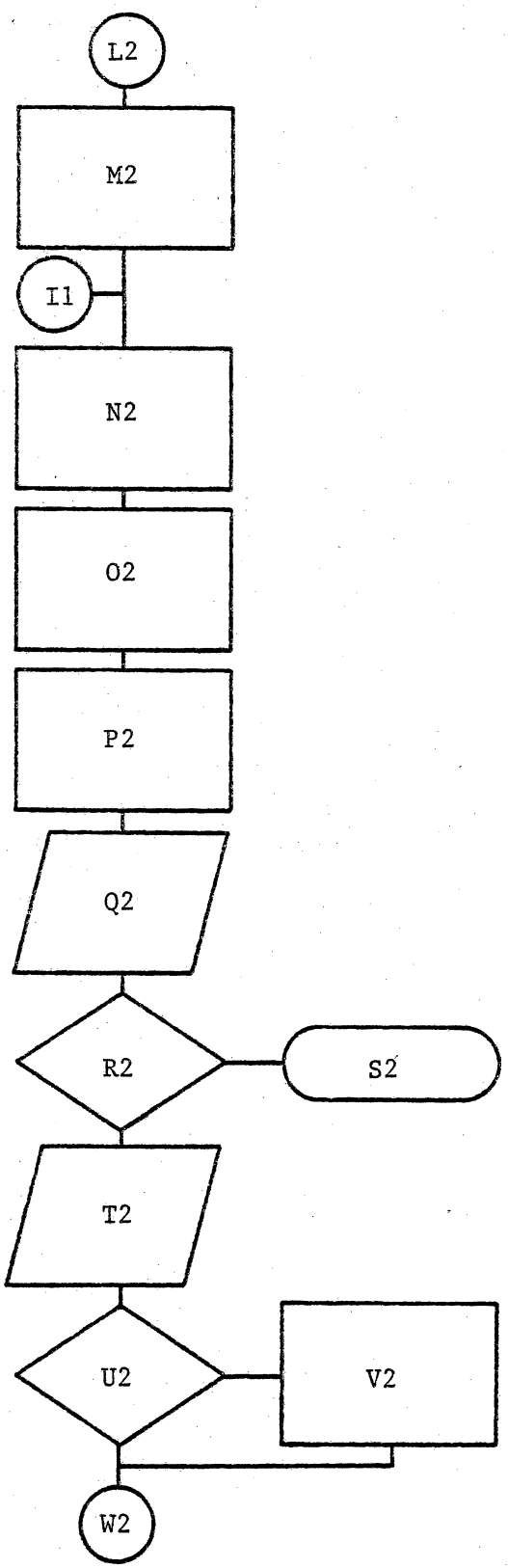
R2. Is the answer 'Y'?
If it is, go to T2
If not, go to S2

S2. Exit to monitor

T2. Output 'Section Change?'
then '1'

U2. Change section 1?
If yes, go to V2
If not, go to W2

V2. Add 1 to KEY



X2. Output '2'

Y2. Change section 2?
    If it should be, go to Z2
    If not, go to A3

Z2. Add 2 to KEY

A3. Output '3'

B3. Change section 3?
    If yes, go to C3
    If not, go to D3

C3. Add 4 to KEY

D3. Output '4'

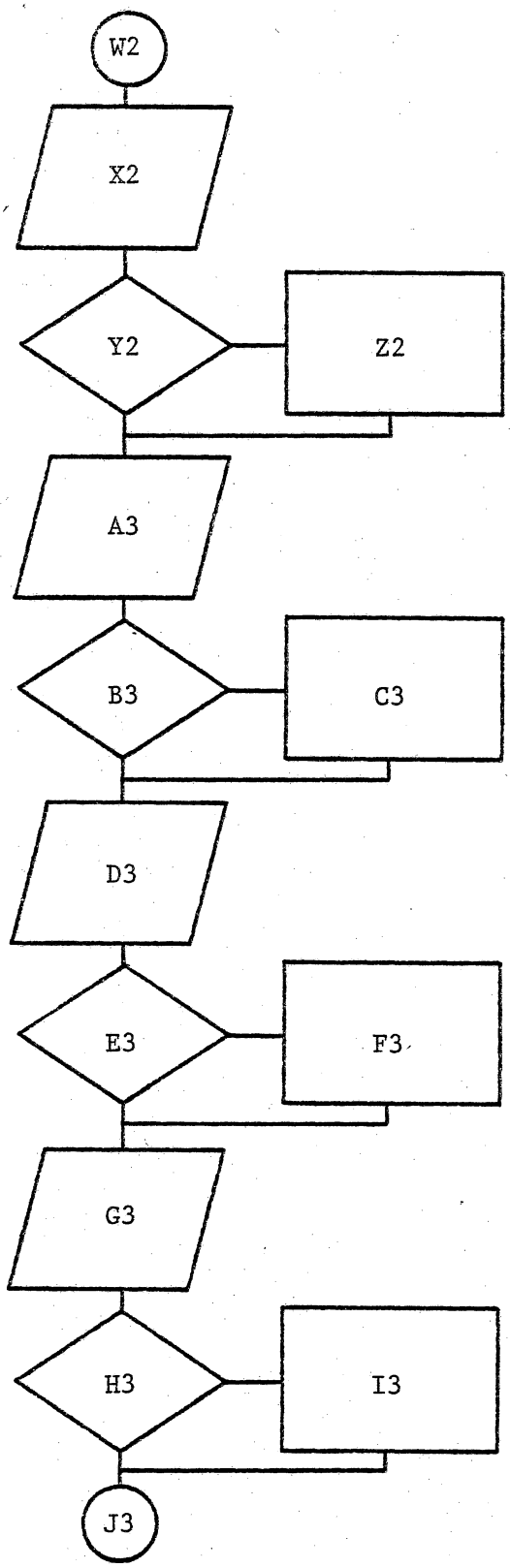
E3. Change section 4?
    If yes, go to F3
    If not, go to G3

F3. Add 10 to KEY

G3. Output '5'

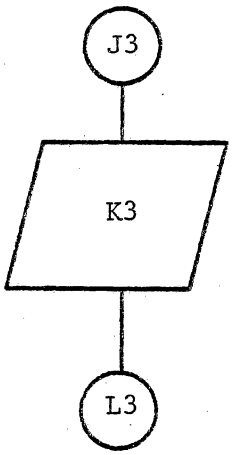
H3. Change section 5?
    If yes, go to I3
    If not, go to J3

I3. Add 20 to KEY



K3. Output run number

L3. Go back to the beginning



Controller Routine Flowchart

A. Start routine

B. Save flowrate and runtime

C. Clear buffer area

D. Set up data files

E. A correct Enter?
   If it is, go to H
   If not, go to F

F. Output bad Enter message

G. Exit to Monitor

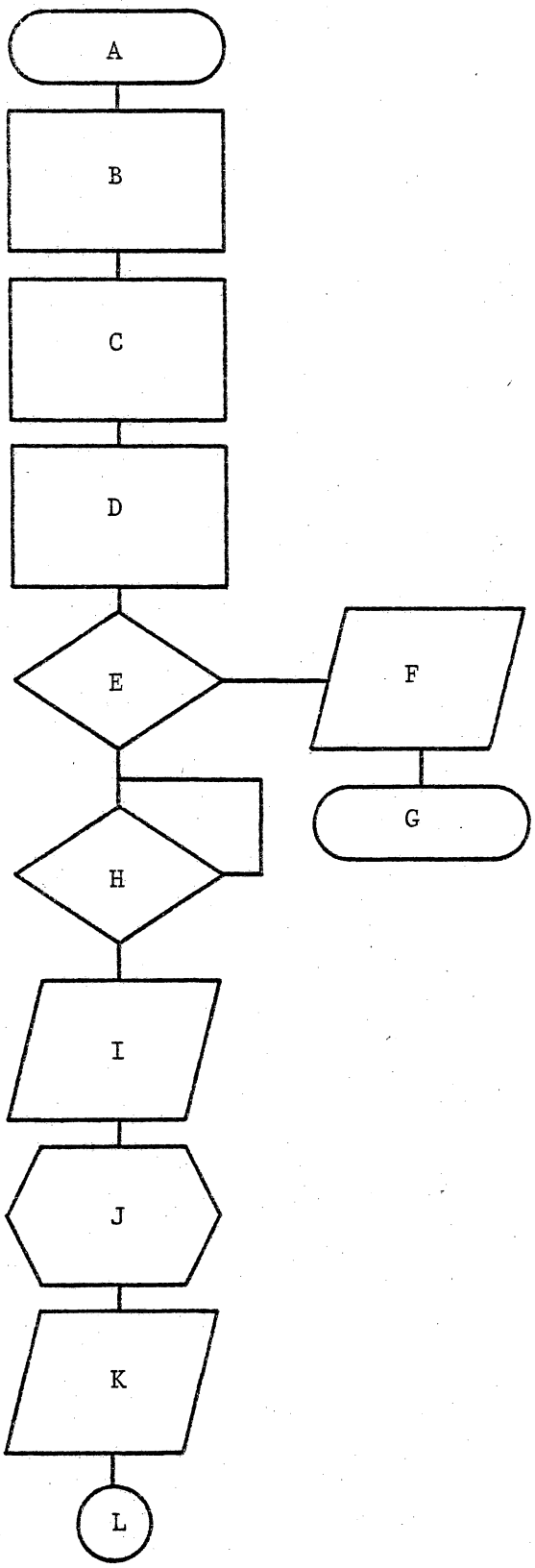
H. Injection button depressed?
   If yes, go to I
   If not, check it again

I. Output start message

J. Reset start signal

K. Output start signal to L.C.

L. Next page



L. From previous page

M. Call subroutine Setup

N. Initialize registers and
buffer addresses

O. Send trigger signal

P. Output buffer to oscilliscope

Q. Is the buffer full?
If it is, go to X
If not, go to R

R. Is it the end of data?
If it is, go to S
If not, go to O

S. Disable interrupts

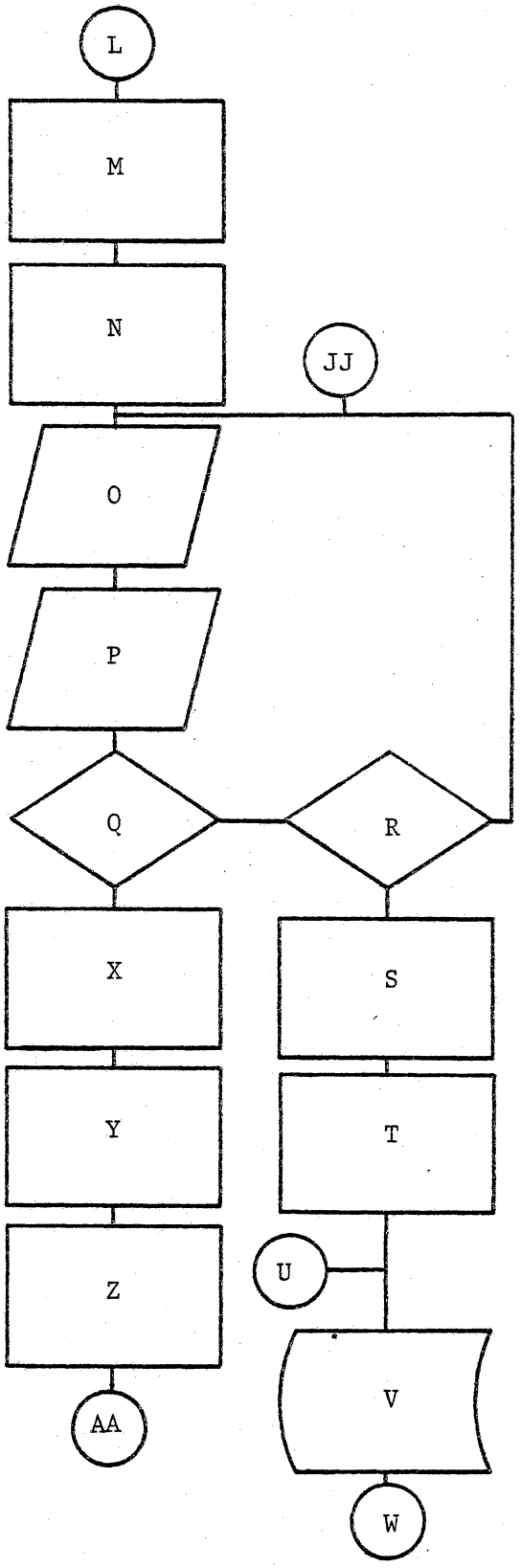
T. Adjust block count to correct value

V. Write 512 words to disk

X. Stop flow on L.C., put it
on Hold

Y. Disable interrupts

Z. Set block and buffer addresses



BB. Write 512 points to disk

CC. Good write?
If it is, go to DD
If not, go to LL

DD. All 12 blocks written?
If they are, go to EE
If not, go to BB

EE. Start flow again, put L.C.
on start

FF. Reinitialize buffer pointers

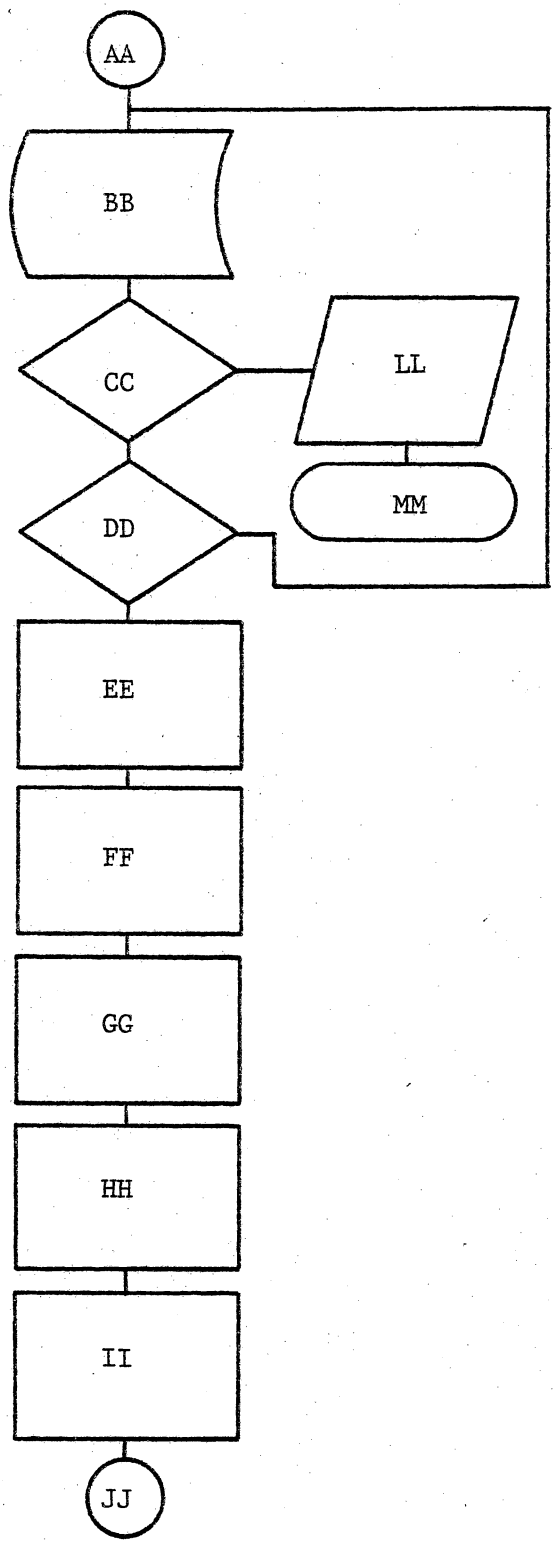
GG. Clear buffer

HH. Adjust block count

II. Enable interrupts

LL. Output bad write message

MM. Exit to Monitor



NN. Good write?
    If yes, go to QQ
    If not, go to OO

OO. Output bad write message

PP. Exit to monitor

QQ. All blocks written?
    If yes, go to SS
    If not, go to RR

SS. Close data channel

TT. Clear buffer

UU. Reinitialize L.C.

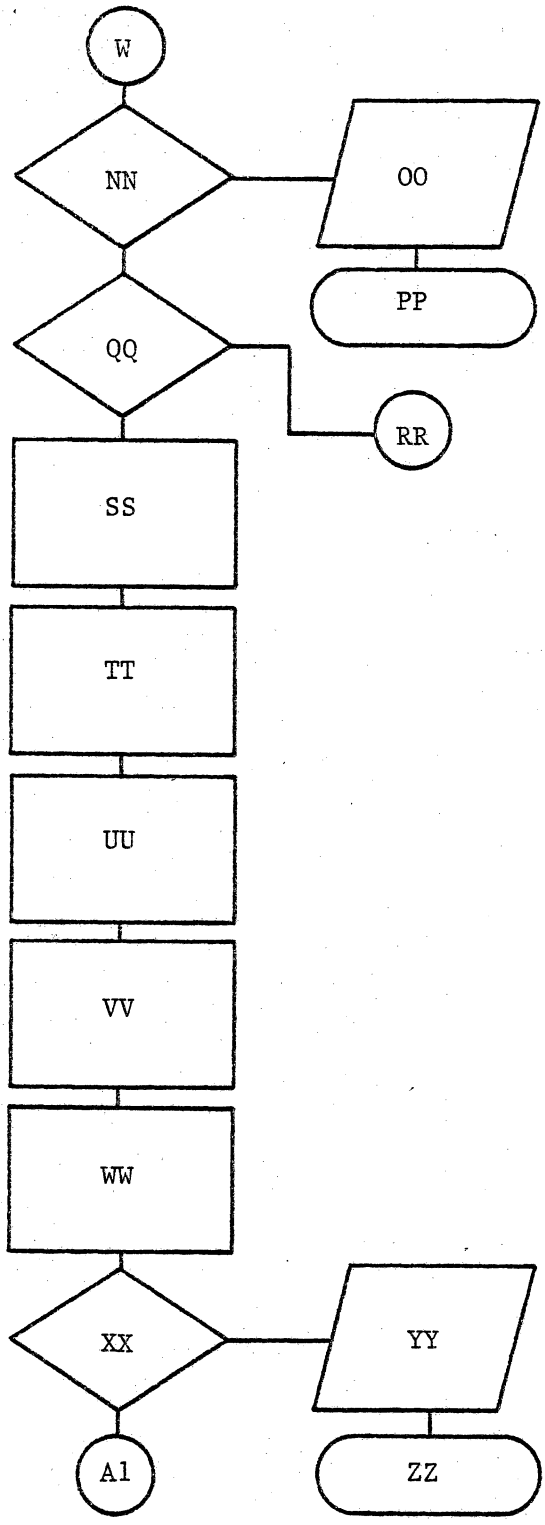
VV. Replace teletype interrupt
    vector addresses

WW. Look up data file

XX. Was it a good look up?
    If yes, go to A1
    If not, go to YY

YY. Output bad look up message

ZZ. Exit to monitor



B1. Read 512 points from the disk

C1. Was it a good read?
    If yes, go to G1
    If not, go to D1

D1. Output bad read message

E1. Exit to monitor

G1. Set buffer display

H1. Output trigger signal

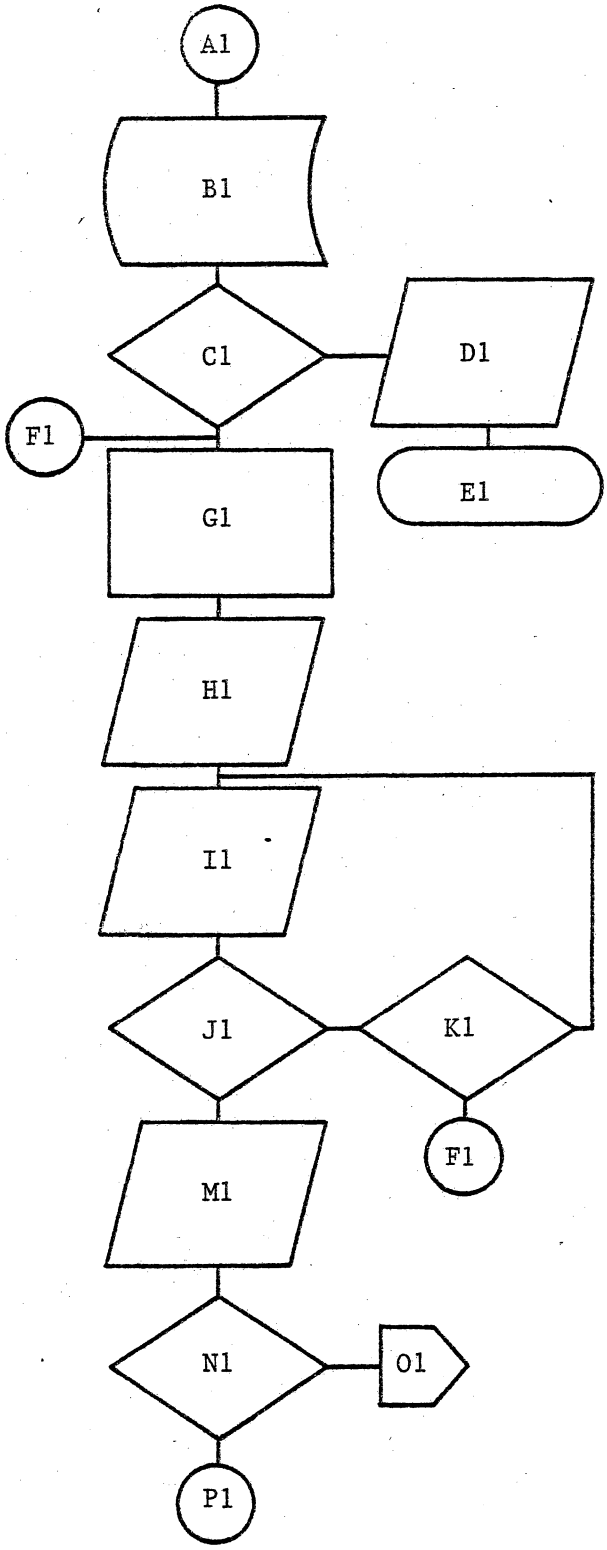
I1. Output points to oscilliscope

J1. Any teletype input?
    If yes, go to M1
    If not, go to K1

K1. Is the buffer done?
    If yes, go to F1
    If not, go to I1

M1. Get character from TTY buffer

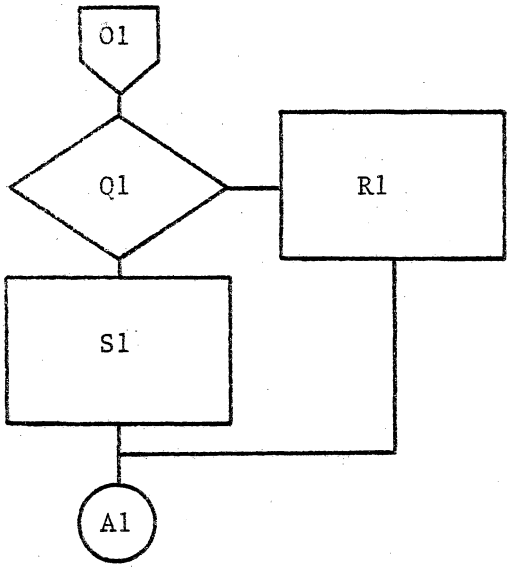
N1. Is it an 'N'?
    If it is, go to P1
    If not, go to O1



Q1. Is it an 'I'?
    If it is, go to S1
    If not, go to R1

R1. Decrement file pointer to previous data block

S1. Increment file pointer to next data block



T1. Output analysis question

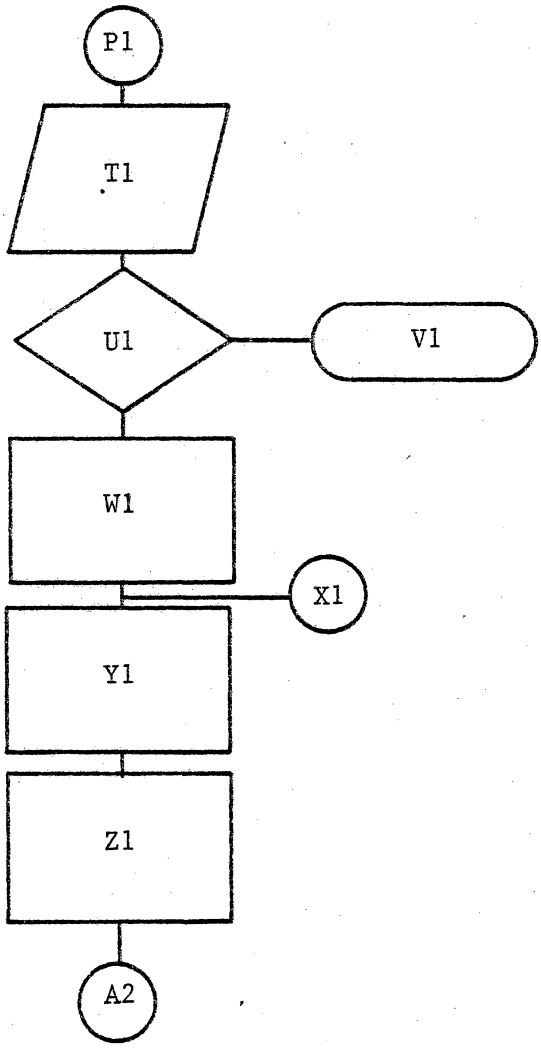
U1. Is the answer a 'Y'?
    If it is, go to W1
    If not, go to V1

V1. Return to report routine

W1. Clear counters and pointers

Y1. Set buffer area pointers

Z1. Clear buffer



B2. All blocks done?
If yes, go to D2
If not, go to G2

D2. Close channel

E2. Return to report routine

G2. Read data blocks in

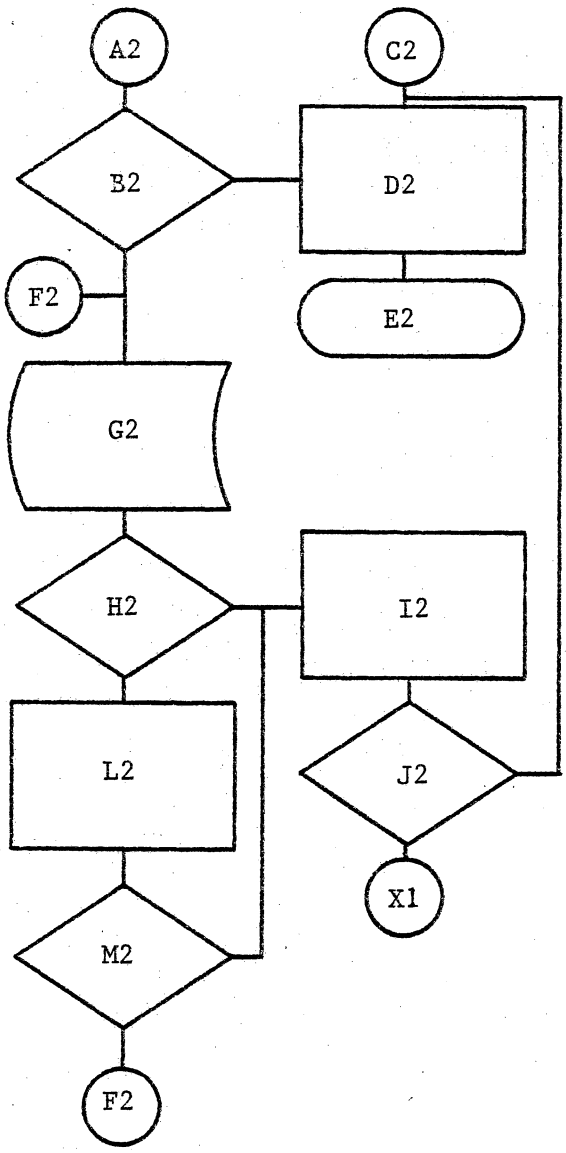
H2. Is buffer full?
If it is, go to I2
If not, go to L2

I2. Jump to peak processor subroutine

J2. All blocks done?
If yes, go to D2
If not, go to X1

L2. Increment counters over bad core

M2. Are all blocks done?
If yes, go to I2
If not, go to F2



Setup Subroutine

A. Start routine

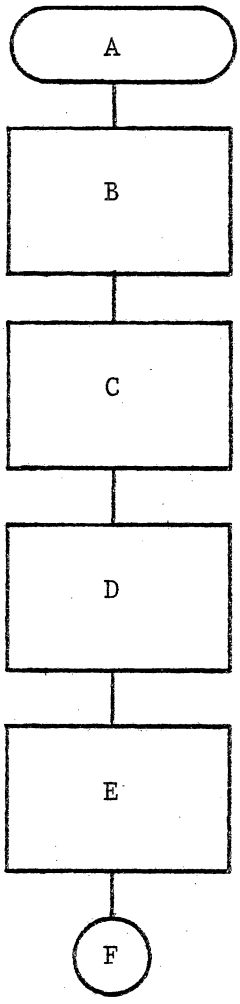
B. Set Data, EOD, and TTY interrupt vector addresses

C. Save TTY monitor vector addresses

D. Set ADC for operation

E. Enable interrupts

F. Return to control routine



Data and End of Data Interrupt Flowcharts

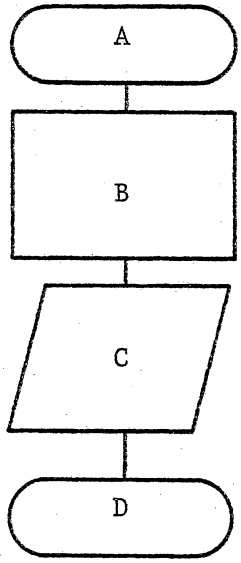
End of Data Interrupt

A. Start routine

B. Disable interrupts

C. Print 'E' for end of run

D. Return from interrupt



Data Interrupt

E. Start routine

F. Get data point

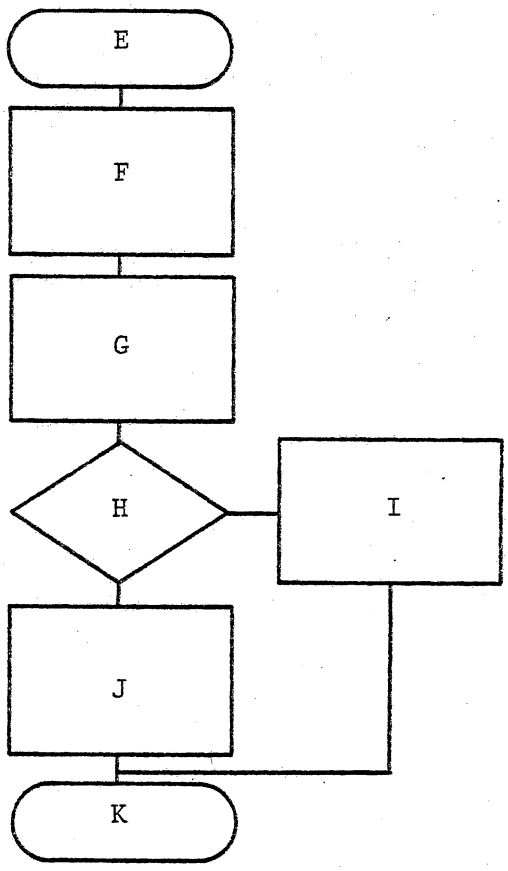
G. Store in buffer

H. End of buffer?
   If yes, go to J
   If no, go to I

I. Return from interrupt

J. Reset buffer pointers

K. Return from interrupt



Teletype Interrupt Handler

A. Start routine

B. Input character

C. Is it an 'H'?
   If it is, go to D
   If not, go to G

D. Add run time to Hold signal

E. Output it to L.C. to put it
   on hold

F. Return from interrupt

G. Is it an 'S'?
   If it is, go to H
   If not, go to K

H. Add run time to start signal

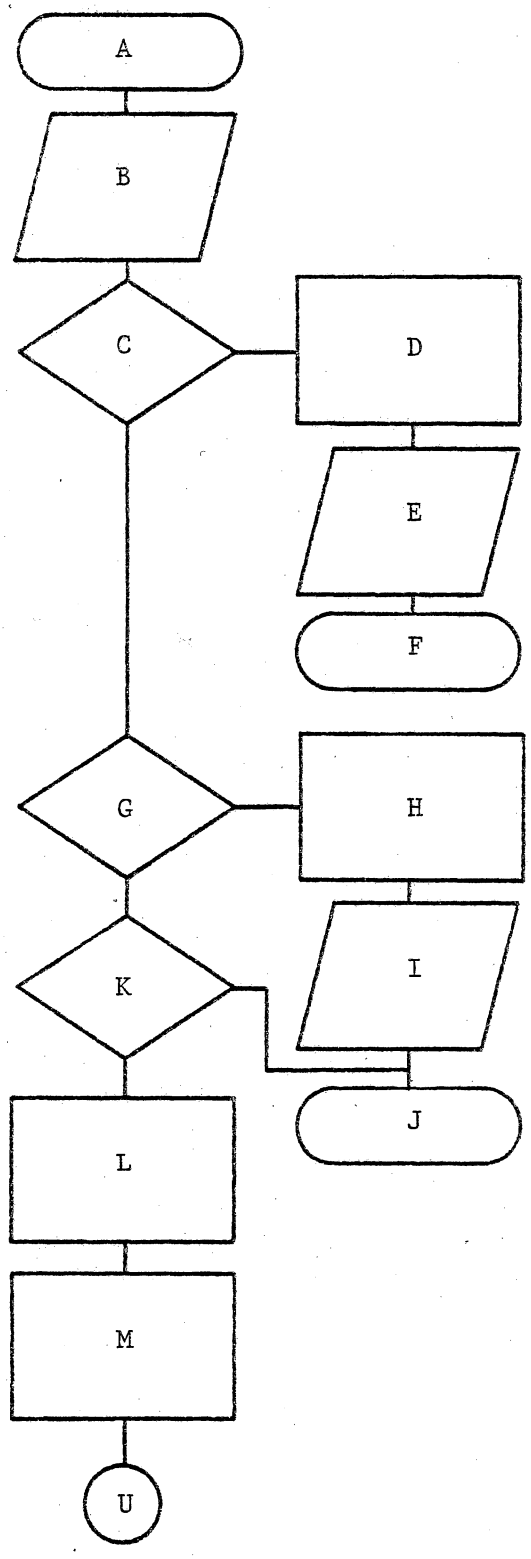
I. Output it to L.C. to start it

J. Return from interrupt

K. Is it an 'E'?
   If it is, go to L
   If not, go to J

L. Disable interrupts

M. Pop stack twice to go to controller routine at point U



Time Calculation Flowchart

A. Start routine

B. Input time of day and echo it

C. Convert hours to minutes
   add minutes to this total

D. Multiply by 10 for better
   resolution

E. Time GT 2880 min.?
   If it is, go to F
   If not, go to G

F. Subtract 2880 from time, R2 + 2000

G. Time GT 1440 min?
   If it is, go to H
   If not, go to I

H. Sub 1440 from time, R2 + 1000

I. Time GT 720 min?
   If it is, go to J
   If not, go to K

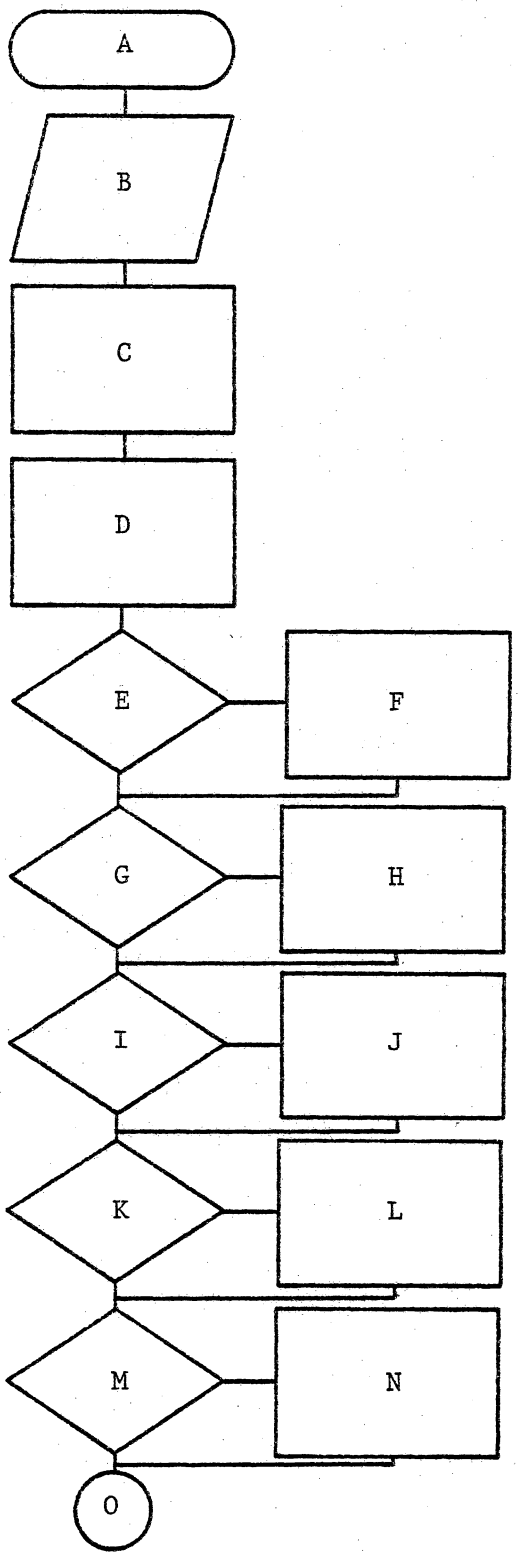
J. Sub 720 from time, R2 + 400

K. Time GT 360 min?
   If it is, go to L
   If not, go to M

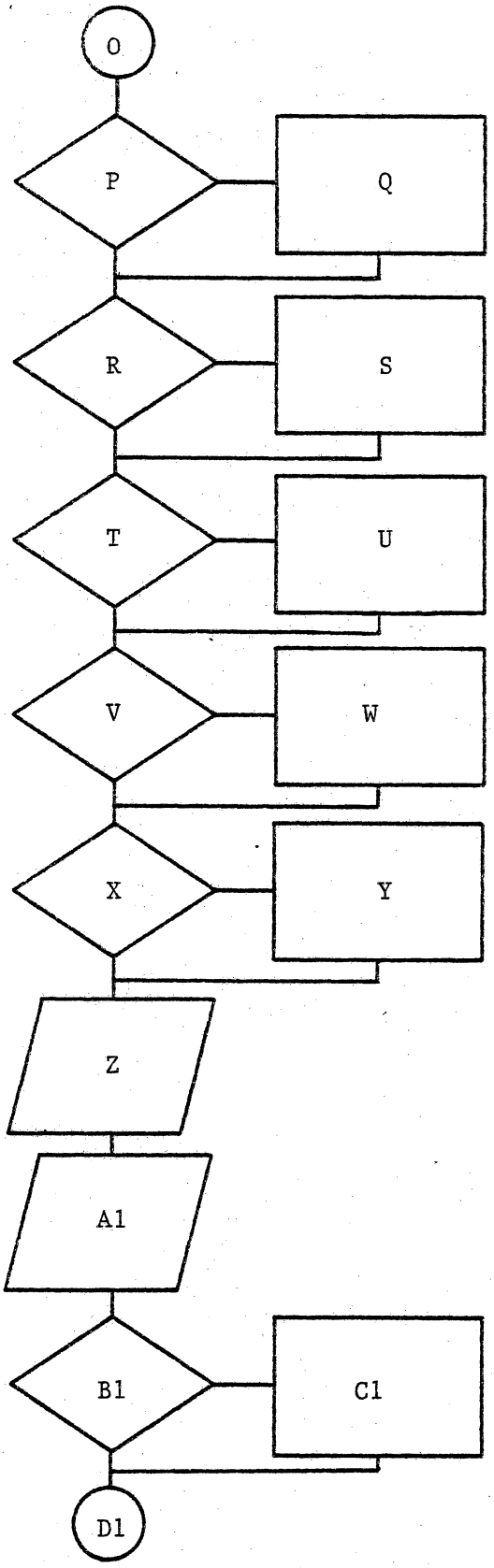
L. Sub 360 from time, R2 + 200

M. Time GT 205 min?
   If it is, go to N
   If not, go to O

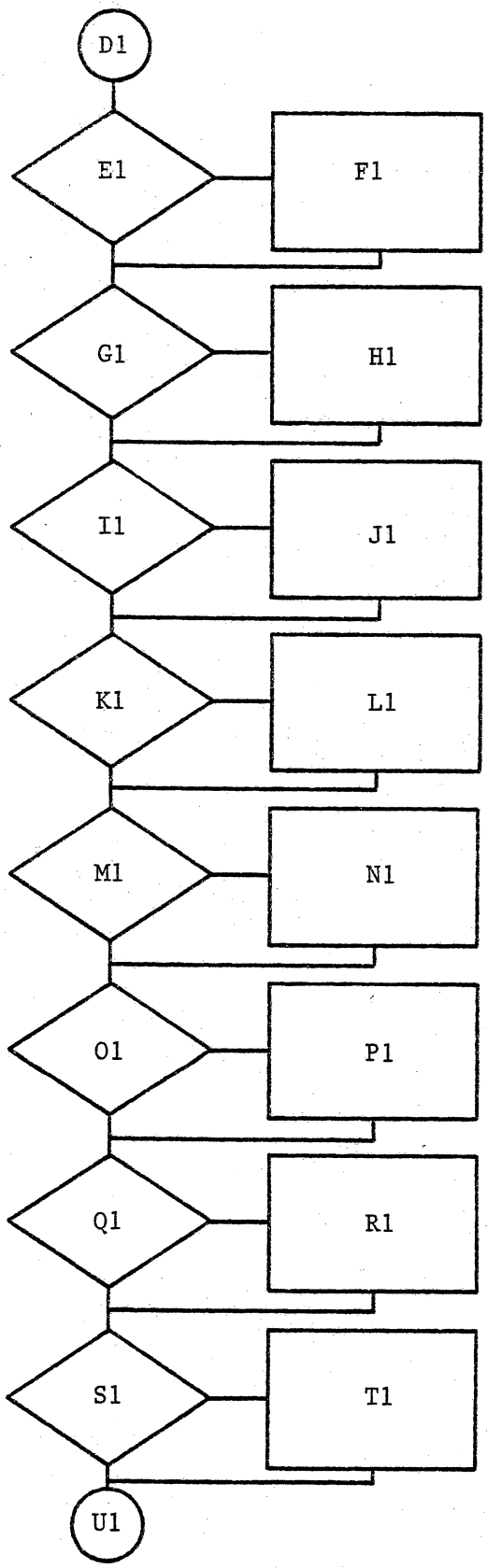
N. Sub 205 from time, R2 + 100



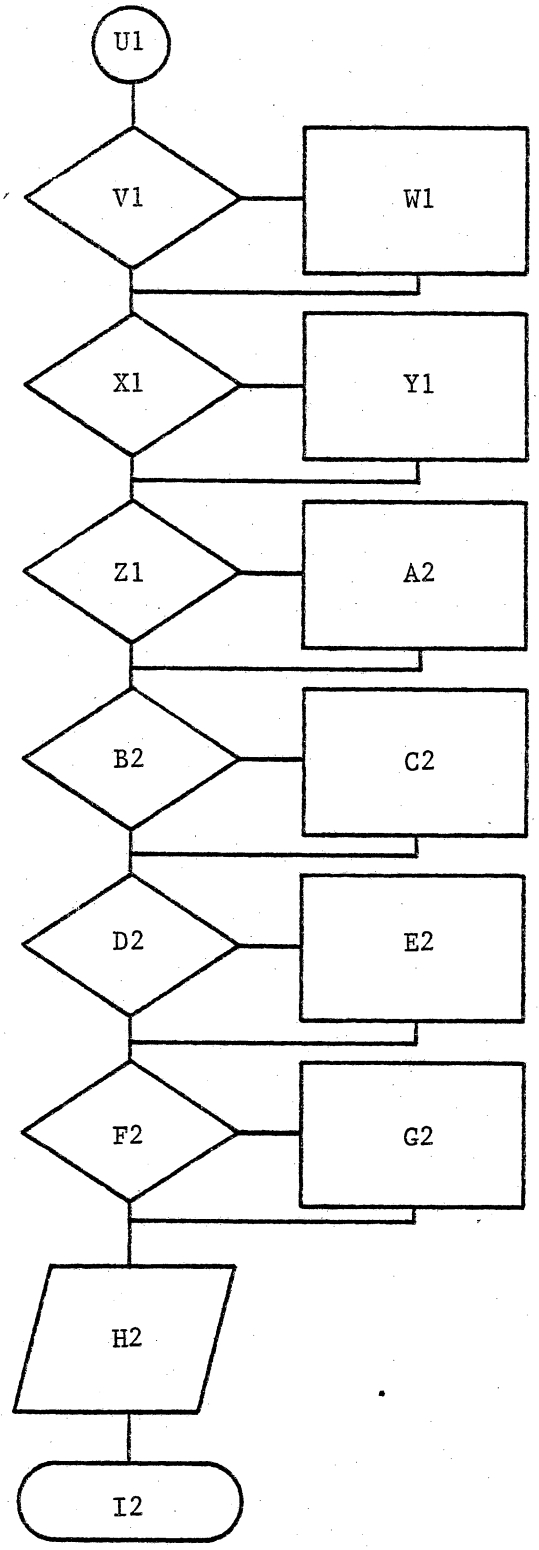
P. Time GT 96 min?
   If it is, go to Q
   If not, go to R
Q. Sub 96 from time, R2 + 40
R. Time GT 48 min?
   If it is, go to S
   If not, go to T
S. Sub 48 from time, R2 + 20
T. Time GT 24 min?
   If it is, go to U
   If not, go to V
U. Sub 24 from time, R2 + 10
V. Time GT 13.7 min?
   If it is, go to W
   If not, go to X
W. Sub 13.7 from time, R2 + 4
X. Time GT 6.4 min?
   If it is, go to Y
   If not, go to Z
Y. Sub 6.4 from time, R2 + 2
Z. Output R2 plus load signal 60000
A1. Output read control word,100000
    Input high clock value to R1
    Output read control word,120000
    Input low clock value to R2
B1. Shift bits four times
    Is the carry set?
    If it is, go to C1
    If not, go to D1
C1. Move a 2 into R3



E1. Shift bits in R2, carry set?
    If it is, go to F1
    If not, go to G1
F1. Add 1 to R3, shift R2
G1. Carry set?
    If it is, go to H1
    If not, go to I1
H1. Move 16040 into R1, shift R2
I1. Carry set?
    If it is, go to J1
    If not, go to K1
J1. Add 7020 to R1, shift R2
K1. Carry set?
    If it is, go to L1
    If not, go to M1
L1. Add 4011 to R1 shift R2
M1. Carry set?
    If it is, go to N1
    If not, go to O1
N1. Add 1700 to R1, shift R2
O1. Carry set?
    If it is, go to P1
    If not, go to Q1
P1. Add 740 to R1, shift R2
Q1. Carry set?
    If it is, go to R1
    If not, go to S1
R1. Add 360 to R1, shift R2
S1. Carry set ?
    If it is, go to T1
    If not, go to U1
T1. Add 211 to R1, shift R2



V1. Carry set?
    If it is, go to W1
    If not, go to X1
W1. Add 100 to R1, shift R2
X1. Carry set?
    If it is, go to Y1
    If not, go to Z1
Y1. Add 40 to R1, shift R2
Z1. Carry set?
    If it is, go to A2
    If not, go to B2
A2. Add 20 to R1, shift R2
B2. Carry set?
    If it is, go to C2
    If not, go to D2
C2. Add 11 to R1,shift R4 five times
D2. Carry set?
    If it is, go to E2
    If not, go to F2
E2. Add 4 to R1, shift R4 twice
F2. Carry set?
    If it is, go to G2
    If not, go to H2
G2. Add 2 to R1
H2. Output time to teletype
I2. Return to main routine



Peak Processor Flowchart

A. Start routine

B. Initialize parameters, BS = 0

D. S1 = 1, MN = 77777
S2 = 0, MX = 100000

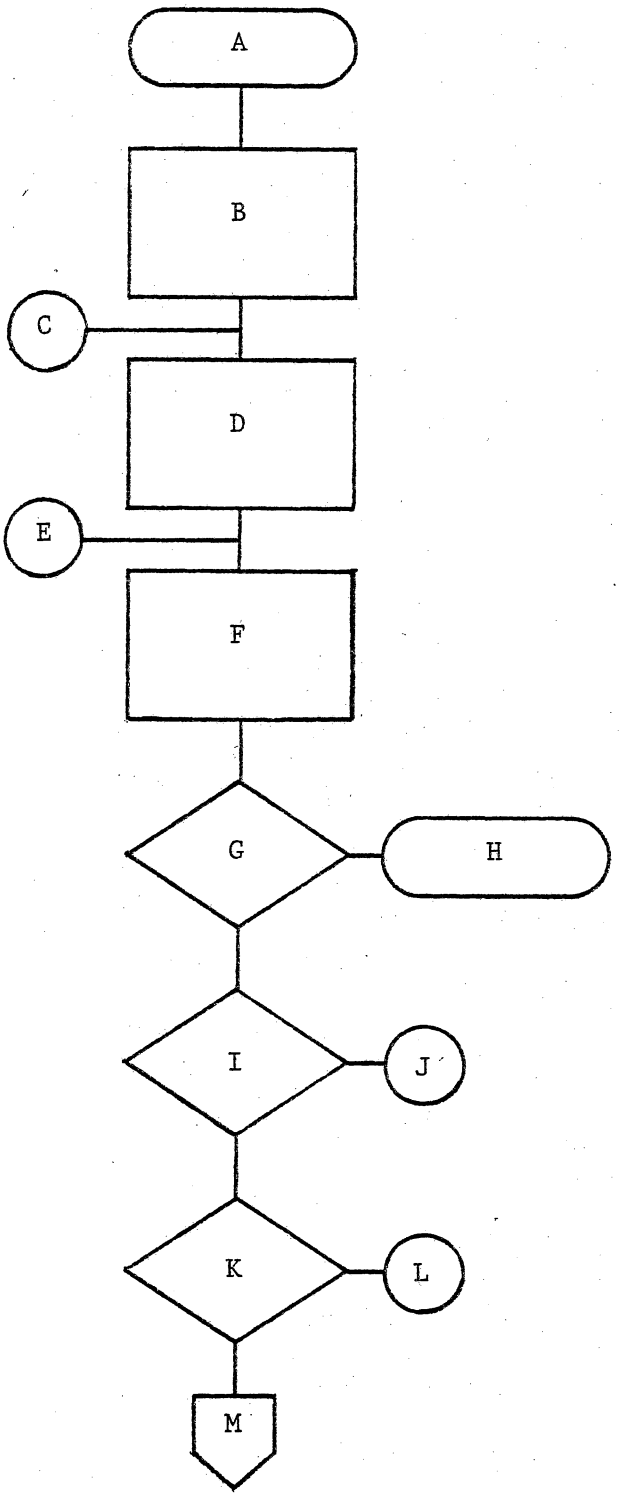
F. Decrement buffer count

G. Buffer count = 0?
If it does, go to H
If not, go to I

H. Return to Control routine

I. Point LT MN?
If it is, go to K
If not, go to J

K. BS = 1?
If it does, go to M
If not, go to L



N. Point GT 0.5(Max - Min) ?
   If it is, go to P
   If not, go to O

O. Calculate width, BS = 2

P. Update MN and time

Q. Update area if on peak

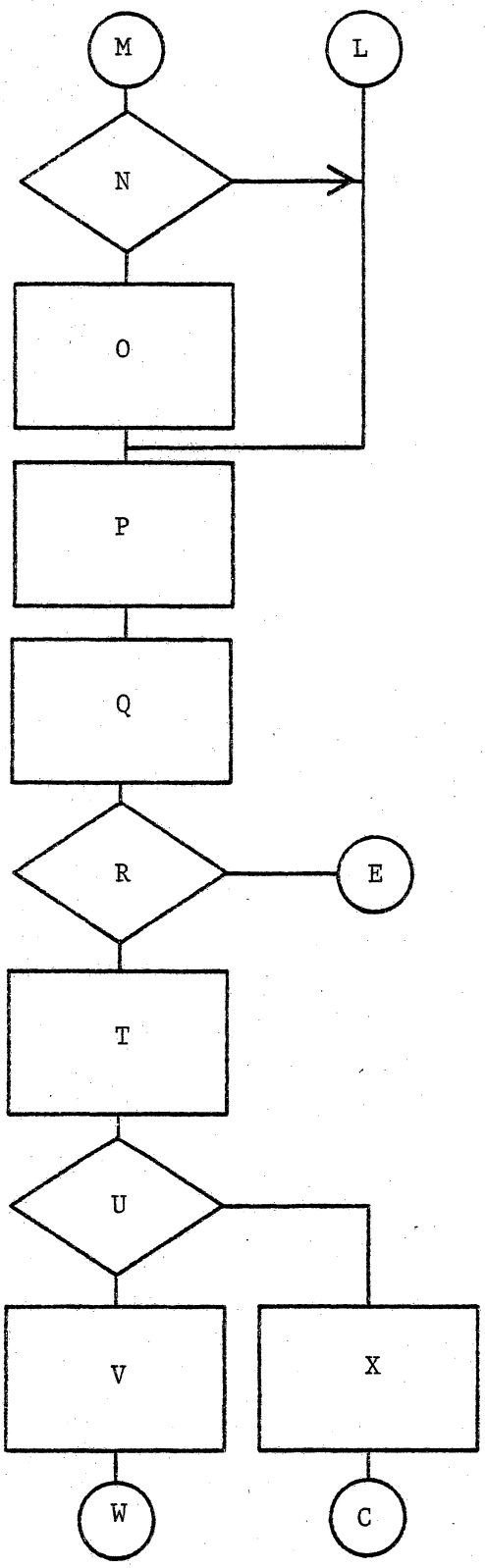
R. Decreasing trend?
   If it is, go to T
   If not, go to S

T. Increment S1

U. S2 = 0?
   If it does, go to V
   If not, go to X

V. MX = 100000

X. BS = 1



Y. Update: area, MX, and time

Z. Increasing trend?
   If it is, go to A1
   If not, go to W

A1. S1 = 0?
   If it does, go to B1
   If not, go to C1

C1. Increment S2, S1 = 0

D1. Store time and value of
   leading minimum

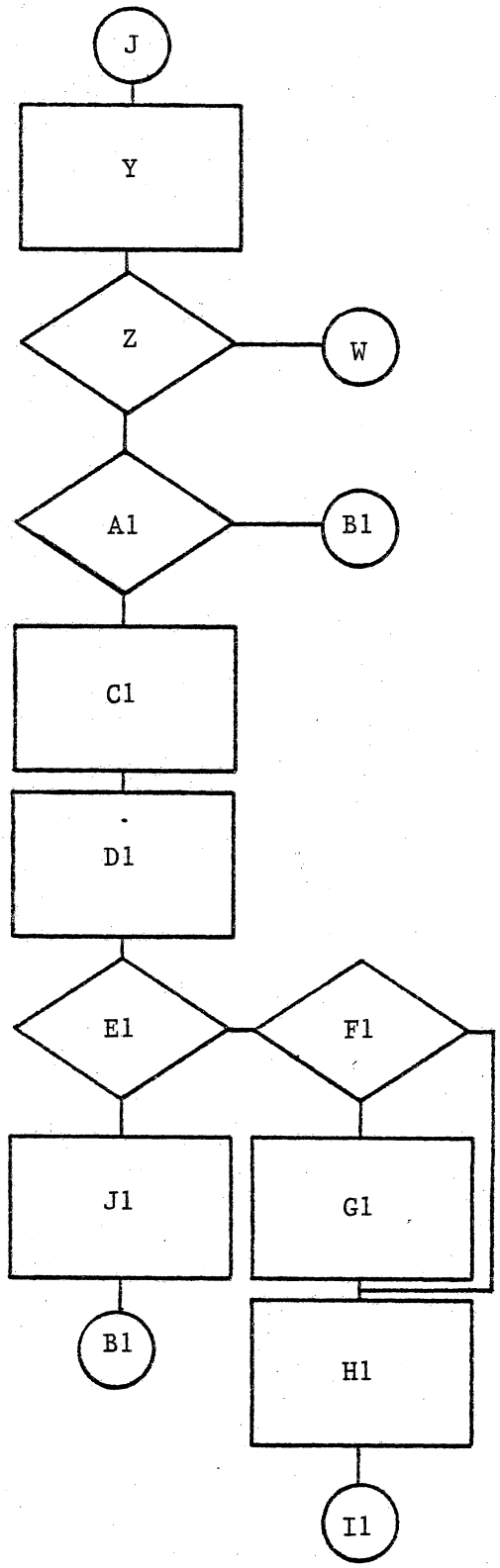
E1. BS = 0?
   If it does, go to J1
   If not, go to F1

F1. BS = 1 or 2?
   If it is 1, go to G1
   If it is 2, go to H1

G1. Calculate width

H1. Type of peak is 0

J1. Set up for baseline calculations



K1. Output peak parameters

L1. BS = 2

M1. Replace MN with MX

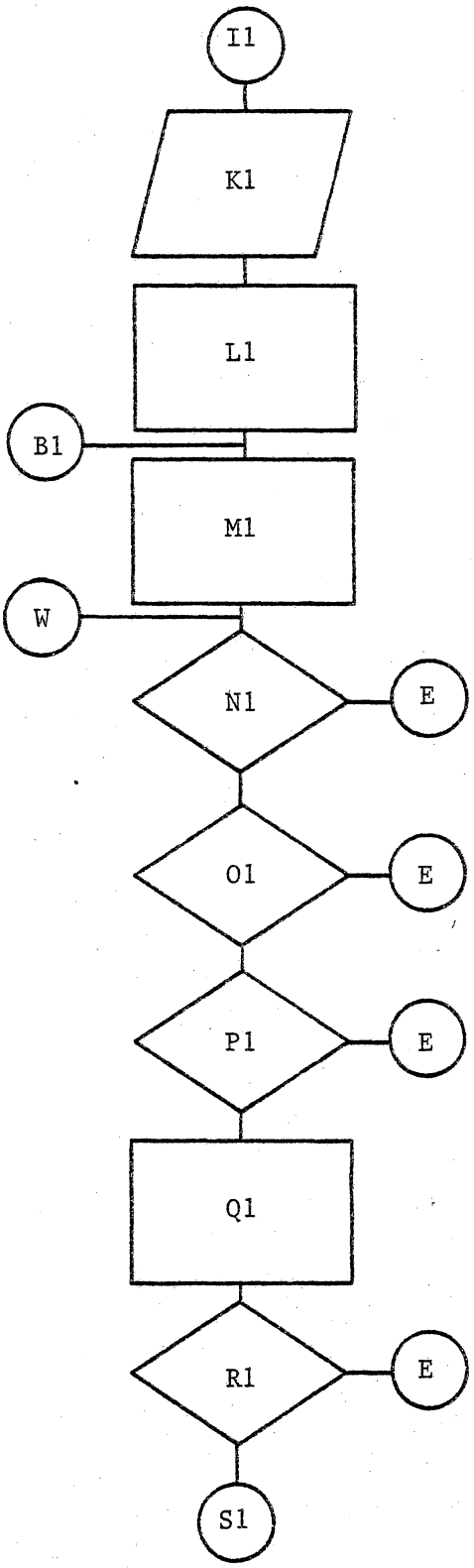
N1. BS = 2?
    If it does, go to O1
    If not, go to E

O1. S1 = 0?
    If it does, go to E
    If not, go to P1

P1. Point far enough past peak?
    If it is, go to Q1
    If not, go to E

Q1. Calculate and store slope

R1. New slope GT old slope?
    If it is, go to S1
    If not, go to E



T1. Increment slope counter

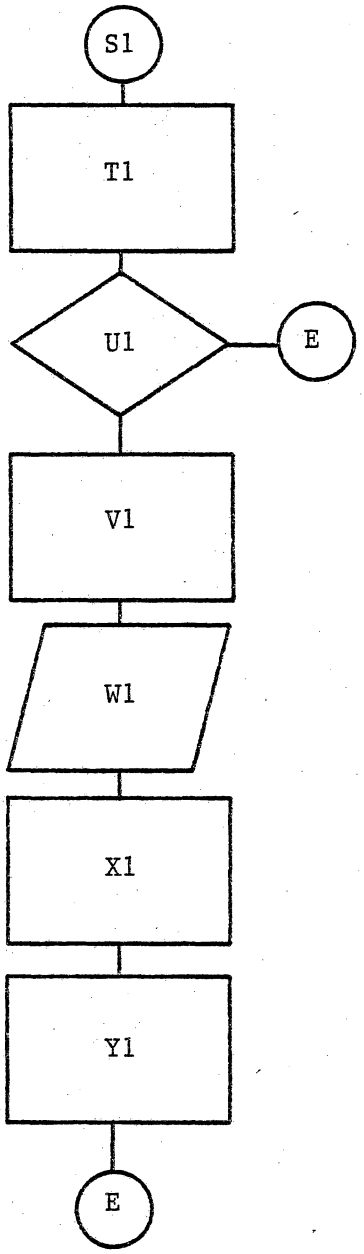
U1. Is it LT 2?
    If it is, go to E
    If not, go to V1

V1. Peak type is 1

W1. Output peak parameters

X1. BS = 0

Y1. Reinitialize parameters



APPENDIX B

```
                        .TITLE  REPORT ROUTINE
                        .GLOBL  BEGIN,FDNAM,IRAD50,RATE
                        .MCALL  ..V2..,.REGDEF,.EXIT
000000                  ..V2..
000000                  .REGDEF
                        ;
                        ;
                        ;THIS IS THE REPORT PROGRAM
                        ;THIS TAKES CARE OF MOST INTERACTIONS WITH THE
                        ;USER. ALL INFORMATION FROM THE USER IS ASKED
                        ;FOR HERE. THE PROGRAM THEN CALLS THE MAIN RUNNING
                        ;PROGRAM, CALLED CONTROL, STARTING AT BEGIN.
                        ;
                        ;
00000 012737            MOV     #100000,@#167762        ;INIATIALIZE FLOW
        100000
        167762
00006 052737            BIS     #010000,@#44            ;SET MONITOR FOR SPECIAL I/O
        010000
        000044
00014 012737            MOV     #010000,@#167762        ;INITIALIZE SWEEP
        010000
        167762
00022 012700            MOV     #DAT,R0                 ;ASK DATE
        002170'
00026 004767            JSR     PC,OUT                  ;OUTPUT DATE REQUEST
        001730
00032 004767            JSR     PC,IN                   ;GET DATE FROM USER
        001746
00036 012700            MOV     #OPER,R0                ;ASK OPERATORS NAME
        002237'
00042 004767            JSR     PC,OUT
        001714
00046 004767            JSR     PC,IN
        001732
00052 012700            MOV     #RUN,R0                 ;GET THE RUN NUMBER
        002254'
00056 004767            JSR     PC,OUT
        001700
00062 004767            JSR     PC,INP
        001744
00066 010167            MOV     R1,RUNNUM               ;SAVE THE RUN NUMBER
        002056
00072 012767            MOV     #10,KEY                 ;SET OUTPUT CONTROL SO INITIALLY ALL
        000010
        002064
                                                        ;REPORT NOTES ARE TAKEN
```

```
30 00100 012767 RDUND:  MOV     #FDNAM,RADLST+6         ;PUT FILE HANDLER ADDRESS IN RAD50 LIST
         002102'
         002014
31 00106 005067         CLR     FDNAM                   ;CLEAR THE FILE NAME AREA
         001770
32 00112 005067         CLR     FDNAM+2
         001766
33 00116 005067         CLR     FDNAM+4
         001764
34 00122 005067         CLR     FDNAM+6
         001762
35 00126 005067         CLR     FDNAM+10
         001760
36 00132 012704         MOV     #CHRSTR,R4              ;SET TTY BUFFER ADDRESS
         002126'
37 00136 012703         MOV     #14,R3                  ;SET CHAR COUNT
         000014
38 00142 012700         MOV     #FIL,R0                 ;MSG ASKING FOR FILE NAME
         002201'
39 00146 004767         JSR     PC,OUT                  ;OUTPUT MSG
         001610
40 00152 105737 INT:    TSTB    @#177560                ;GET FILE NAME
         177560
41 00156 002375         BGE     INT                     ;WAIT FOR CHAR TO SENT
42 00160 013737         MOV     @#177562,@#177566       ;ECHO THE CHAR.
         177562
         177566
43 00166 013700         MOV     @#177562,R0             ;STRIP OFF UNNECESSARY BITS AND SAVE
         177562
44 00172 042700         BIC     #177600,R0
         177600
45 00176 110014 NOMIS:  MOVB    R0,(R4)                 ;SAVE CHAR IN RAD50 LIST
46 00200 005204         INC     R4
47 00202 005303         DEC     R3                      ;RESET COUNT
48 00204 001362         BNE     INT                     ;SEE IF COUNT DONE
49 00206 012705 RADCH:  MOV     #RADLST,R5              ;SET UP JUMP TO SUB
         002114'
50 00212 004767         JSR     PC,IRAD50               ;GO CONVERT VALUES TO RAD50
         000006
51 00216 006267         ASR     KEY                     ;CHECK IF DESIRE FIRST SET
         001742
52 00222 103026         BCC     TWO                     ;OF QUESTIONS ASKED
53 00224 012700         MOV     #Q0,R0                  ;TYPE OF SAMPLE
         002273'
54 00230 004767         JSR     PC,OUT
         001526
55 00234 012700         MOV     #Q1,R0                  ;SAMPLE NAME
```

```
                002307'
56 00240 004767         JSR     PC,OUT                  ;OUTPUT QUESTION
         001516
57 00244 004767         JSR     PC,IN                   ;GET ANSWER
         001534
58 00250 012700         MOV     #Q2,R0                  ;ASK FOR VOLUME
         002317'
59 00254 004767         JSR     PC,OUT
         001502
60 00260 004767         JSR     PC,IN
         001520
61 00264 012700         MOV     #Q3,R0                  ;ASK FOR CONCENTRATION
         002331'
62 00270 004767         JSR     PC,OUT
         001466
63 00274 004767         JSR     PC,IN
         001504
64 00300 006267 TWO:    ASR     KEY                     ;CHECK IF SECTION TWO TO BE DONE
         001660
65 00304 103050         BCC     THREE
66 00306 012700         MOV     #Q4,R0                  ;OUTPUT TITLE: COLUMN SPECS
         002341'
67 00312 004767         JSR     PC,OUT
         001444
68 00316 012700         MOV     #Q5,R0                  ;MANUFACTURER
         002374'
69 00322 004767         JSR     PC,OUT
         001434
70 00326 004767         JSR     PC,IN
         001452
71 00332 012700         MOV     #A2,R0                  ;COLUMN PACKING
         002414'
72 00336 004767         JSR     PC,OUT
         001420
73 00342 004767         JSR     PC,IN
         001436
74 00346 012700         MOV     #A3,R0                  ;COLUMN LENGTH
         002427'
75 00352 004767         JSR     PC,OUT
         001404
76 00356 004767         JSR     PC,IN
         001422
77 00362 012700         MOV     #A4,R0                  ;COLUMN DIAMETER
         002441'
78 00366 004767         JSR     PC,OUT
         001370
79 00372 004767         JSR     PC,IN
```

```
                001436
80  00376 012700          MOV     #A5,R0          ;COLUMN PRESSURE
                002465'
81  00402 004767          JSR     PC,OUT
                001354
82  00406 004767          JSR     PC,IN
                001372
83  00412 012700          MOV     #A6,R0          ;TEMPERATURE
                002510'
84  00416 004767          JSR     PC,OUT
                001340
85  00422 004767          JSR     PC,IN
                001356
86  00426 006267 THREE:   ASR     KEY             ;CHECK FOR THIRD SECTION
                001532
87  00432 103034          BCC     FOUR
88  00434 012700          MOV     #B0,R0          ;TITLE: DETECTOR
                002523'
89  00440 004767          JSR     PC,OUT
                001316
90  00444 012700          MOV     #B1,R0          ;TYPE OF DETECTOR
                002541'
91  00450 004767          JSR     PC,OUT
                001306
92  00454 004767          JSR     PC,IN
                001324
93  00460 012700          MOV     #B2,R0          ;O. D. VALUE
                002551'
94  00464 004767          JSR     PC,OUT
                001272
95  00470 004767          JSR     PC,IN
                001310
96  00474 012700          MOV     #B3,R0          ;DETECTOR MODE
                002570'
97  00500 004767          JSR     PC,OUT
                001256
98  00504 004767          JSR     PC,IN
                001274
99  00510 012700          MOV     #B4,R0          ;WAVELENGTH
                002600'
100 0514  004767          JSR     PC,OUT
                001242
101 0520  004767          JSR     PC,IN
                001260
102 0524  006267 FOUR:    ASR     KEY             ;SECTION FOUR?
                001434
103 0530  103054          BCC     FIVE
```

```
104 0532 012700 REP1:   MOV     #C0,R0                  ;RUN TIME
         002516'
105 0536 004767         JSR     PC,OUT
         001220
106 0542 004767         JSR     PC,INP
         001264
107 0546 010167         MOV     R1,SWEEP
         001410
108 0552 026727         CMP     SWEEP,#3202             ;ADJUST VALUE SO THAT THE LC CAN USE IT
         001404
         003202
109 0560 100405         BMI     ALRITE                  ;CHECK IF IT IS WITHIN RANGE
110 0562 012700         MOV     #E7,R0                  ;IF NOT OUTPUT ERROR MSG
         003133'
111 0566 004767         JSR     PC,OUT
         001170
112 0572 000757         BR      REP1
113 0574 012700 ALRITE: MOV     #C1,R0                  ;GET FLOWRATE
         002644'
114 0600 004767         JSR     PC,OUT
         001156
115 0604 004767         JSR     PC,INP
         001222
116 0610 010167         MOV     R1,FLOW                 ;SAVE IT
         001352
117 0614 026727         CMP     FLOW,#145               ;SEE IF IT IS IN RANGE
         001346
         000145
118 0622 100405         BMI     DESCE
119 0624 012700         MOV     #E7,R0                  ;OUTPUT ERROR MSG IF NOT
         003133'
120 0630 004767         JSR     PC,OUT
         001126
121 0634 000757         BR      ALRITE
122 0636 012700 DESCE:  MOV     #C2,R0                  ;GET DATA RATE
         002703'
123 0642 004767         JSR     PC,OUT
         001114
124 0646 004767         JSR     PC,INP
         001160
125 0652 010137         MOV     R1,@#167752             ;OUTPUT DATA RATE TO TIMER
         167752
126 0656 010167         MOV     R1,RATE                 ;SAVE DATA RATE
         001216
127 0662 006267 FIVE:   ASR     KEY                     ;DO SECTION FIVE?
         001276
128 0666 103042         BCC     SIX
```

```
129 0670 012700          MOV     #00,R0                  ;TITLE: MOBILE PHASE
         002725'
130 0674 004767          JSR     PC,OUT
         001062
131 0700 012700          MOV     #01,R0                  ;PHASE A
         002747'
132 0704 004767          JSR     PC,OUT
         001052
133 0710 004767          JSR     PC,IN
         001070
134 0714 012700          MOV     #02,R0                  ;PHASE B
         002754'
135 0720 004767          JSR     PC,OUT
         001036
136 0724 004767          JSR     PC,IN
         001054
137 0730 012700          MOV     #03,R0                  ;GRADIENT SLOPE
         002761'
138 0734 004767          JSR     PC,OUT
         001022
139 0740 004767          JSR     PC,IN
         001040
140 0744 012700          MOV     #04,R0                  ;%A AT START
         003003'
141 0750 004767          JSR     PC,OUT
         001006
142 0754 004767          JSR     PC,IN
         001024
143 0760 012700          MOV     #05,R0                  ;%A AT END
         003022'
144 0764 004767          JSR     PC,OUT
         000772
145 0770 004767          JSR     PC,IN
         001010
146 0774 022767 SIX:     CMP     #1,SKIP                 ;IS FLOW,DATA RATE,AND RUN TIMESET?
         000001
         001154
147 1002 001574          BEQ     WAYOUT                  ;GO BR IF IT IS
148                      ;
149                      ;
150                      ;
151                      ;THIS SECTION TAKES THE RUN TIME AND CONVERTS IT
152                      ;TO A FORM USABLE BY THE LC.
153                      ;BIT0  = 1 MIN IF SET TO A ZERO
154                      ;BIT1 = 2 MIN, BIT2 = 4 MIN, BIT3 = 8 MIN
155                      ;BIT4 = 10 MIN, ETC.
156                      ;THIS IS DONE BY SUBTRACTING VALUES CORRESPONDING
```

```
157                             ;TO THE BITS TO BE SET
158                             ;
159                             ;
160                             ;
161 1004 005002                 CLR     R2
162 1006 026727                 CMP     SWEEP,#1440             ;CHECK FOR 800 MINS.
         001150
         001440
163 1014 100405                 BMI     A
164 1016 162767                 SUB     #1440,SWEEP
         001440
         001136
165 1024 062702                 ADD     #4000,R2
         004000
166 1030 026727 A:              CMP     SWEEP,#620              ;CHECK FOR 400 MINS.
         001126
         000620
167 1036 100405                 BMI     B
168 1040 162767                 SUB     #620,SWEEP
         000620
         001114
169 1046 062702                 ADD     #2000,R2
         002000
170 1052 026727 B:              CMP     SWEEP,#310              ;CHECK FOR 200 MINS.
         001104
         000310
171 1060 100405                 BMI     C
172 1062 162767                 SUB     #310,SWEEP
         000310
         001072
173 1070 062702                 ADD     #1000,R2
         001000
174 1074 026727 C:              CMP     SWEEP,#144              ;CHECK FOR 100 MINS.
         001062
         000144
175 1102 100405                 BMI     D
176 1104 162767                 SUB     #144,SWEEP
         000144
         001050
177 1112 062702                 ADD     #400,R2
         000400
178 1116 026727 D:              CMP     SWEEP,#120              ;CHECK FOR 80 MINS
         001040
         000120
179 1124 100405                 BMI     E
180 1126 162767                 SUB     #120,SWEEP
         000120
```

```
                001026
181  1134 062702        ADD     #200,R2
                000200
182  1140 026727 E:     CMP     SWEEP,#50           ;CHECK FOR 40 MINS.
                001016
                000050
183  1146 100405        BMI     F
184  1150 162767        SUB     #50,SWEEP
                000050
                001004
185  1156 062702        ADD     #100,R2
                000100
186  1162 026727 F:     CMP     SWEEP,#24           ;CHECK FOR 20 MINS.
                000774
                000024
187  1170 100405        BMI     G
188  1172 162767        SUB     #24,SWEEP
                000024
                000762
189  1200 062702        ADD     #40,R2
                000040
190  1204 026727 G:     CMP     SWEEP,#12           ;CHECK FOR 10 MINS.
                000752
                000012
191  1212 100405        BMI     H
192  1214 162767        SUB     #12,SWEEP
                000012
                000740
193  1222 062702        ADD     #20,R2
                000020
194  1226 026727 H:     CMP     SWEEP,#10           ;CHECK FOR 8 MINS.
                000730
                000010
195  1234 100405        BMI     I
196  1236 162767        SUB     #10,SWEEP
                000010
                000716
197  1244 062702        ADD     #10,R2
                000010
198  1250 026727 I:     CMP     SWEEP,#4            ;CHECK FOR 4 MINS.
                000706
                000004
199  1256 100405        BMI     J
200  1260 162767        SUB     #4,SWEEP
                000004
                000674
201  1266 062702        ADD     #4,R2
```

```
                000004
202 1272 026727 J:      CMP     SWEEP,#2                ;CHECK FOR 2 MINS.
         000664
         000002
203 1300 100405         BMI     K
204 1302 162767         SUB     #2,SWEEP
         000002
         000652
205 1310 062702         ADD     #2,R2
         000002
206 1314 026727 K:      CMP     SWEEP,#1                ;CHECK FOR 1 MIN
         000642
         000001
207 1322 100402         BMI     L
208 1324 062702         ADD     #1,R2
         000001
209 1330 005102 L:      COM     R2                      ;GET THE COMPLEMENT OF THE VALUE
210 1332 010267         MOV     R2,SWEEP                ;SAVE THE VALUE
         000624
211 1336 042767         BIC     #170000,SWEEP           ;CLEAR ALL EXTRA BITS
         170000
         000616
212                     ;
213                     ;
214                     ;
215                     ;CALCULATE THE FLOW RATE BY MULTIPLYING BY THE CONVERSION FACTOR
216                     ;
217                     ;
218 1344 016701         MOV     FLOW,R1
         000616
219 1350 070127         MUL     #122,R1
         000122
220 1354 005000         CLR     R0
221 1356 071027         DIV     #10,R0
         000010
222 1362 010067         MOV     R0,FLOW
         000600
223 1366 062767         ADD     #100000,FLOW            ;SET BIT 15 SO FLOW GOES TO DAC
         100000
         000572
224 1374 016737 WAYOUT: MOV     FLOW,@#167762           ;MOVE FLOW TO LC
         000566
         167762
225 1402 012737         MOV     #010000,@#167762        ;INITIALIZE SYSTEM AGAIN
         010000
         167762
226 1410 016700         MOV     FLOW,R0                 ;SEND FLOW AND RUN TIME TO SUBROUTINE
```

```
                000552
227 1414 016701         MOV     SWEEP,R1
         000542
228 1420 004767         JSR     PC,BEGIN                ;START THE CONTROLLER PROGRAM
         000000G
229 1424 012700         MOV     #E0,R0                  ;ANOTHER RUN?
         003037'
230 1430 005067         CLR     KEY                     ;THIS PART SETS THE KEY BY
         000530
231 1434 004767         JSR     PC,OUT                  ;SETTING THE BITS AFTER ASKING IF
         000322
232 1440 004767         JSR     PC,INP                  ;THE SECTION SHOULD BE DONE
         000366
233 1444 022701         CMP     #51,R1                  ;51 STANDS FOR YES
         000051
234 1450 001143         BNE     OVER
235 1452 012700         MOV     #E1,R0                  ;TITLE: SECTION CHANGE
         003061'
236 1456 004767         JSR     PC,OUT
         000300
237 1462 012700         MOV     #E2,R0                  ;SECTION 1?
         003102'
238 1466 004767         JSR     PC,OUT
         000270
239 1472 004767         JSR     PC,INP
         000334
240 1476 022701         CMP     #51,R1
         000051
241 1502 001003         BNE     NONE
242 1504 062767         ADD     #1,KEY
         000001
         000452
243 1512 012700 NONE:   MOV     #E3,R0                  ;SECTION 2?
         003107'
244 1516 004767         JSR     PC,OUT
         000240
245 1522 004767         JSR     PC,INP
         000304
246 1526 022701         CMP     #51,R1
         000051
247 1532 001003         BNE     NTWO
248 1534 062767         ADD     #2,KEY
         000002
         000422
249 1542 012700 NTWO:   MOV     #E4,R0                  ;SECTION 3?
         003114'
250 1546 004767         JSR     PC,OUT
```

```
                000210
251 1552 004767         JSR     PC,INP
         000254
252 1556 022701         CMP     #51,R1
         000051
253 1562 001003         BNE     NTHRE
254 1564 062767         ADD     #4,KEY
         000004
                000372
255 1572 012700 NTHRE:  MOV     #E5,R0          ;SECTION 47
         003121'
256 1576 004767         JSR     PC,OUT
         000160
257 1602 004767         JSR     PC,INP
         000224
258 1606 012767         MOV     #1,SKIP
         000001
                000342
259 1614 022701         CMP     #51,R1
         000051
260 1620 001005         BNE     NFOUR
261 1622 062767         ADD     #10,KEY
         000010
                000334
262 1630 005067         CLR     SKIP
                000322
263 1634 012700 NFOUR:  MOV     #E6,R0          ;SECTION 57
         003126'
264 1640 004767         JSR     PC,OUT
         000116
265 1644 004767         JSR     PC,INP
         000162
266 1650 022701         CMP     #51,R1
         000051
267 1654 001003         BNE     NFIVE
268 1656 062767         ADD     #20,KEY
         000020
                000300
269 1664 012700 NFIVE:  MOV     #RUN,R0         ;OUTPUT RUN NUMBER
         002254'
270 1670 004767         JSR     PC,OUT
         000066
271 1674 005267         INC     RUNNUM
         000260
272 1700 016701         MOV     RUNNUM,R1
         000254
273 1704 026727         CMP     RUNNUM,#12
```

```
                000250
                000012
274 1712 100407         BMI     NOTENS
275 1714 162767         SUB     #12,RUNNUM
                000012
                000236
276 1722 012700         MOV     #MUN,R0
                002076'
277 1726 004767         JSR     PC,OUT
                000030
278 1732 012700 NOTENS: MOV     #RUNNUM,R0
                002160'
279 1736 062767         ADD     #260,RUNNUM
                000260
                000214
280 1744 004767         JSR     PC,OUT
                000012
281 1750 010167         MOV     R1,RUNNUM
                000204
282 1754 000167         JMP     ROUND
                176120
283 1760         OVER:  .EXIT
284 1762 112037 OUT:    MOVB    (R0)+,@#177566          ;ROUTINE TO OUTPUT MESSAGES
                177566
285 1766 105737 HOLD:   TSTB    @#177564
                177564
286 1772 002375         BGE     HOLD
287 1774 122710         CMPB    #0,(R0)
                000000
288 2000 001370         BNE     OUT
289 2002 000207         RTS     PC
290 2004 105737 IN:     TSTB    @#177560                ;ROUTINE TO INPUT CHARS.
                177560
291 2010 002375         BGE     IN
292 2012 013700         MOV     @#177562,R0
                177562
293 2016 010037         MOV     R0,@#177566
                177566
294 2022 020027         CMP     R0,#215
                000215
295 2026 001366         BNE     IN
296 2030 000207         RTS     PC
297 2032 005001 INP:    CLR     R1                      ;ROUTINE TO INPUT NUMBERS
298 2034 105737 ONE:    TSTB    @#177560
                177560
299 2040 002375         BGE     ONE
300 2042 013700         MOV     @#177562,R0
```

```
                177562
301 2046 010037         MOV     R0,@#177566
                177566
302 2052 022700         CMP     #215,R0
                000215
303 2056 001406         BEQ     AWAY
304 2060 070127         MUL     #12,R1
                000012
305 2064 162700         SUB     #260,R0
                000260
306 2070 060001         ADD     R0,R1
307 2072 000760         BR      ONE
308 2074 000207 AWAY:   RTS     PC
309                     ;VARIABLES NAMES
310 2076 000261 WUN:    .WORD   000261
311 2100 000000 RATE:   .WORD   0
312 2102        FDNAM:  .BLKW   5
313 2114 000003 RADLST: .WORD   3
314 2116 002124'        .WORD   CHR
315 2120 002126'        .WORD   CHRSTR
316 2122 002102'        .WORD   FDNAM
317 2124 000014 CHR:    .WORD   12.
318 2126        CHRSTR: .BLKW   14
319 2156 000000 SKIP:   .WORD   000000
320 2160 000000 RUNNUM: .WORD   0
321 2162 000000 SWEEP:  .WORD   0
322 2164 000000 KEY:    .WORD   000000
323 2166 000000 FLOW:   .WORD   0
324                     .NLIST  BIN
325 2170        DAT:    .ASCIZ  <15><12>/DATE: /
326 2201        FIL:    .ASCIZ  <15><12>/ENTER FILE NAME   /
327 2226        TIM:    .ASCIZ  <15><12>/TIME: /
328 2237        OPER:   .ASCIZ  <15><12>/OPERATOR: /
329 2254        RUN:    .ASCIZ  <15><12>/RUN NUMBER: /
330 2273        Q0:     .ASCIZ  <15><12>/1  SAMPLE/
331 2307        Q1:     .ASCIZ  <15><12>/NAME:/
332 2317        Q2:     .ASCIZ  <15><12>/VOLUME:/
333 2331        Q3:     .ASCIZ  <15><12>/CONC:/
334 2341        Q4:     .ASCIZ  <15><12>/2  COLUMN SPECIFICATIONS/
335 2374        Q5:     .ASCIZ  <15><12>/MANUFACTURER:/
336 2414        A2:     .ASCIZ  <15><12>/PACKING:/
337 2427        A3:     .ASCIZ  <15><12>/LENGTH:/
338 2441        A4:     .ASCIZ  <15><12>/INNER DIAMETER = /
339 2465        A5:     .ASCIZ  <15><12>/COLUMN PRESSURE:/
340 2510        A6:     .ASCIZ  <15><12>/TEMP(C):/
341 2523        B0:     .ASCIZ  <15><12>/3  DETECTOR/
342 2541        B1:     .ASCIZ  <15><12>/TYPE:/
```

```
343 2551        32:     .ASCIZ  <15><12>/U. D. VALUE:/
344 2570        B3:     .ASCIZ  <15><12>/MODE:/
345 2600        B4:     .ASCIZ  <15><12>/WAVELENGTH:/
346 2616        C0:     .ASCIZ  <15><12>/4  SWEEPTIME(MIN)= /
347 2644        C1:     .ASCIZ  <15><12>$   FLOWRATE(ML/MIN X 0.04)= $
348 2703        C2:     .ASCIZ  <15><12>/   DATA RATE = /
349 2725        D0:     .ASCIZ  <15><12>/5  MOBILE PHASE/
350 2747        D1:     .ASCIZ  <15><12>/A:/
351 2754        D2:     .ASCIZ  <15><12>/B:/
352 2761        D3:     .ASCIZ  <15><12>/GRADIENT SLOPE:/
353 3003        D4:     .ASCIZ  <15><12>/%A AT START:/
354 3022        D5:     .ASCIZ  <15><12>/%A AT END:/
355 3037        E0:     .ASCIZ  <15><12>/DO ANOTHER RUN?/
356 3061        E1:     .ASCIZ  <15><12>/SECTION CHANGE/
357 3102        E2:     .ASCIZ  <15><12>/1?/
358 3107        E3:     .ASCIZ  <15><12>/2?/
359 3114        E4:     .ASCIZ  <15><12>/3?/
360 3121        E5:     .ASCIZ  <15><12>/4?/
361 3126        E6:     .ASCIZ  <15><12>/5?/
362 3133        E7:     .ASCIZ  /   ILLOGICAL VALUE/
363 3156        MAD:    .ASCIZ  <15><12>/MADE IT OUT/
364                     .EVEN
365                     .END
```

```
REPORT ROUTINE  RT-11 MACRO VM02-12   27-JUN-78 PAGE 1+
SYMBOL TABLE

A       001030R         ALRITE  000574R         AWAY    002074R
A2      002414R         A3      002427R         A4      002441R
A5      002465R         A6      002510R         B       001052R
BEGIN = ****** G        B0      002523R         B1      002541R
B2      002551R         B3      002570R         B4      002600R
C       001074R         CHR     002124R         CHRSTR  002126R
C0      002616R         C1      002644R         C2      002703R
D       001116R         DAT     002170R         DESCE   000636R
D0      002725R         D1      002747R         D2      002754R
D3      002761R         D4      003003R         D5      003022R
E       001140R         E0      003037R         E1      003061R
E2      003102R         E3      003107R         E4      003114R
E5      003121R         E6      003126R         E7      003133R
F       001162R         FDNAM   002102RG        FIL     002201R
FIVE    000662R         FLOW    002166R         FOUR    000524R
G       001204R         H       001226R         HOLD    001766R
I       001250R         IN      002004R         INP     002032R
INT     000152R         IRAD50= ****** G        J       001272R
K       001314R         KEY     002164R         L       001330R
MAD     003156R         NFIVE   001664R         NFOUR   001634R
NUMIS   000176R         NONE    001512R         NOTENS  001732R
NTHRE   001572R         NTWO    001542R         ONE     002034R
OPER    002237R         OUT     001762R         OVER    001760R
PC    =%000007          Q0      002273R         Q1      002307R
Q2      002317R         Q3      002331R         Q4      002341R
Q5      002374R         RADCH   000206R         RADLST  002114R
RATE    002100RG        REP1    000532R         ROUND   000100R
RUN     002254R         RUNNUM  002160R         R0    =%000000
R1    =%000001          R2    =%000002          R3    =%000003
R4    =%000004          R5    =%000005          SIX     000774R
SKIP    002156R         SP    =%000006          SWEEP   002162R
THREE   000426R         TIM     002226R         TWO     000300R
WAYOUT  001374R         WUN     002076R         ...V2 = 000001

. ABS.  000000      000
        003174      001
ERRORS DETECTED: 0
FREE CORE: 2510. WORDS

CALLER,LP:/L:BEX=CALLER
```

```
                        .TITLE  MAIN CONTROL ROUTINE
                        .GLOBL  SETUP,BEGIN,TTY1,TTY2,BLKNM,PEAKNM,PEAK
                        .GLOBL  ACQCNT,DN,OFFSET,FONAM,BAKCAL
                        .MCALL ..V2..,.REGDEF,.CLOSE,.PRINT,.LOOKUP,.READW
                        .MCALL  .EXIT,.ENTER,.WRITW
000000                  ..V2..
000000                  .REGDEF
                        ;
                        ;
                        ;
                        ;THIS ROUTINE HANDLES ALL THE DATA FILES, THE OUTPUT
                        ;TO THE SCOPE AND CALLS THE PEAK PROCESSING
                        ;ROUTINE.
                        ;THE DISPLAY OF DATA AS IT IS ACQUIRED IS DONE IN
                        ;THIS ROUTINE WHILE DATA IS COLLECTED IN THE DATA
                        ;ROUTINE BY INTERRUPTS.
                        ;
                        ;
                        ;
                        ;VARIABLE DEFINITIONS
00000           AREA:   .BLKW   10              ;AREA FOR EMTS
00020 000000    ERR:    .WORD   0               ;COUNT OF DATA BLOCKS
00022 000000    BAKCAL: .WORD   0               ;FLAG HOLDER FOR PEAKS OVERLAPPING BUFFERS
00024 000000    BLKCT:  .WORD   0               ;BLOCK COUNT
00026 000000    BLKNM:  .WORD   0               ;BLOCK NUMBER
00030 001000    ACQCNT: .WORD   1000            ;COUNT OF DATA POINTS ACQUIRED IN A BUFFER
00032 000001    DN:     .WORD   1               ;CHANGE IN POINTS
00034 000000    OFFSET: .WORD   0               ;OFFSET OF DATA FROM BASELINE
00036 000000    PEAKNM: .WORD   0               ;NUMBER OF PEAKS
00040 000000    BUFFAD: .WORD   000000          ;BUFFER PTR
00042 000000    FLOW:   .WORD   000000          ;FLOWRATE
00044 000000    SWEEP:  .WORD   000000          ;RUN TIME
                        .NLIST  BIN
00046           BDENT:  .ASCIZ  /BAD ENTRY/
00060           BEIN:   .ASCIZ  <15><12>/START/
00070           BDWRT1: .ASCIZ  /BAD WRITE 1/
00104           BDWRT:  .ASCIZ  /BAD WRITE 2/
00120           BDLOOK: .ASCIZ  /BAD LOOKUP/
00133           BDREAD: .ASCIZ  /BAD READ/
00144           PEAKAN: .ASCIZ  <15><12>/ANALYZE PEAKS?/
                        .EVEN
                        .LIST   BIN
00166 012737 BEGIN:     MOV     #010000,@#44            ;SET MONITOR FOR SPECIAL I/O
      010000
      000044
00174 012737            MOV     #1,@#167760             ;CLEAR INJECTION SIGNAL
      000001
```

```
                167760
45 00202 012737         MOV     #0,@#167760             ;RESET START OF DATA
         000000
         167760
46 00210 010067         MOV     R0,FLOW                 ;GET FLOW RATE
         177626
47 00214 010167         MOV     R1,SWEEP                ;GET RUN TIME
         177624
48 00220 005067         CLR     BLKCT
         177600
49 00224 012700         MOV     #23000,R0               ;ADDRESS OF BUFFER
         023000
50 00230 010001         MOV     R0,R1
51 00232 062701         ADD     #12000,R1               ;SET END OF BUFFER
         012000
52 00236 005020 CLEAR:  CLR     (R0)+                   ;CLEAR BUFFER AREA
53 00240 020001         CMP     R0,R1
54 00242 001375         BNE     CLEAR
55 00244 005000         CLR     R0
56 00246                .ENTER  #AREA,#100,#FONAM       ;SET FILE UP
57 00304 103004         BCC     START                   ;CHECK IF ANY PROBLEM
58 00306                .PRINT  #BDENT                  ;PRINT ERROR MESSAGE
59 00314                .EXIT                           ;GO BACK TO MONITOR
60 00316 005000 START:  CLR     R0                      ;WAIT FOR START SIGNAL
61 00320 013700         MOV     @#167764,R0
         167764
62 00324 042700         BIC     #177776,R0
         177776
63 00330 001772         BEQ     START
64 00332 012704         MOV     #BEIN,R4                ;OUTPUT MSG TO SHOW START
         000050'
65 00336 112437 T10:    MOVB    (R4)+,@#177566
         177566
66 00342 105737 T11:    TSTB    @#177564
         177564
67 00346 002375         BGE     T11
68 00350 121427         CMPB    (R4),#0
         000000
69 00354 001370         BNE     T10
70 00356 012737         MOV     #1,@#167760             ;RESET START SIGNAL
         000001
         167760
71 00364 012737         MOV     #0,@#167760
         000000
         167760
72 00372 016701         MOV     SWEEP,R1                ;GET SWEEPRATE
         177446
```

```
73 00376 062701          ADD     #020000,R1              ;ADD START FLAG TO SWEEP
         020000
74 00402 010137          MOV     R1,@#167762             ;OUTPUT TO LC
         167762
75 00406 012704          MOV     #23000,R4               ;GIVE SBUF ADDRESS TO DATA
         023000
76 00412 012701          MOV     #INPUT,R1               ;SET ADDRESS FOR CONSOLE INT
         001174'
77 00416 004767          JSR     PC,SETUP                ;SET UP INTERRUPTS
         000000G
78 00422 012701          MOV     #23000,R1               ;SET BUFFER START POINT
         023000
79 00426 010102          MOV     R1,R2                   ;SET END OF BUFFER PTR
80 00430 062702          ADD     #1000,R2
         001000
81 00434 005005          CLR     R5                      ;GET POINTERS READY TO DISPLAY
82 00436 005067          CLR     ERR
         177356
83 00442 005067          CLR     BUFFAD
         177372
84 00446 005000          CLR     R0
85 00450 010203 TDISP:   MOV     R2,R3                   ;GET BUFFER START POINT
86 00452 162703          SUB     #1000,R3
         001000
87 00456 012737          MOV     #43,@#167750            ;OUTPUT TRIGGER PULSE
         000043
         167750
88 00464 012737          MOV     #40,@#167750
         000040
         167750
89 00472 012337 ON:      MOV     (R3)+,@#176760          ;PUT POINT OUT
         176760
90 00476 000240          NOP
91 00500 000240          NOP
92 00502 020203          CMP     R2,R3                   ;SEE IF END OF BUFFER
93 00504 001372          BNE     ON                      ;GO BACK IF NOT
94 00506 020527          CMP     R5,#12                  ;SEE IF ALL BLOCKS FILLED
         000012
95 00512 001405          BEQ     FILSAV
96 00514 022737 DSR:     CMP     #0,@#330                ;CHECK DONE FLAG
         000000
         000330
97 00522 001752          BEQ     TDISP
98 00524 000532          BR      GOON                    ;BRANCH OUT IF SET
99 00526 012737 FILSAV:  MOV     #100000,@#167762        ;ROUTINE TO WRITE TO DISK
         100000
         167762
```

```
100 0534 016703 177304          MOV     SWEEP,R3                ;PUT THE LC ON HOLD
101 0540 062703 040000          ADD     #040000,R3
102 0544 010337 167762          MOV     R3,@#167762             ;OUTPUT HOLD TO LC
103 0550 066705 177244          ADD     ERR,R5                  ;GET TOTAL NUMBER OF BLOCKS
104 0554 106427 000200          MTPS    #200                    ;DISABLE ALL INTERRUPTS
105 0560 005037 167760          CLR     @#167760
106 0564 005037 167750          CLR     @#167750
107 0570 162705 000001          SUB     #1,R5
108 0574 016704 177220          MOV     ERR,R4                  ;SET BLOCK NUMBER
109 0600 012767 023000 177232   MOV     #23000,BUFFAD           ;SET BUFFER BEGINNING
110 0606        RIGHT:  .WRITW  #AREA,#100,BUFFAD,#1000,R4
111 0654 103004         BCC     EXC                             ;WRITE DATA TO DISK
112 0656                .PRINT  #BDWRT1                         ;PRINT ERROR MSG IF NEEDED
113 0664                .EXIT                                   ;GO BACK TO MONITOR
114 0666 020405 EXC:    CMP     R4,R5                           ;CHECK IF ALL BLOCKS DONE
115 0670 001405         BEQ     RESET                           ;BRANCH IF THEY ARE
116 0672 005204         INC     R4                              ;OTHERWISE, INCREMENT POINTERS
117 0674 062767 001000 177136   ADD     #1000,BUFFAD
118 0702 000741         BR      RIGHT
119 0704 016737 RESET:  MOV     FLOW,@#167762           ;RESET ALL POINTERS
         177132
         167762
120 0712 012700 023000          MOV     #23000,R0               ;CLEAR BUFFER AREA
121 0716 010001         MOV     R0,R1
122 0720 062701 012000          ADD     #12000,R1
123 0724 005020 CLR2:   CLR     (R0)+
124 0726 020001         CMP     R0,R1
125 0730 001375         BNE     CLR2
126 0732 016703 177106          MOV     SWEEP,R3                ;PUT RUN TIME OUT WITH START
127 0736 062703 020000          ADD     #020000,R3
```

```
128 0742 010337          MOV     R3,@#167762
         167762
129 0746 166705          SUB     ERR,R5
         177046
130 0752 060567          ADD     R5,ERR
         177042
131 0756 005267          INC     ERR                     ;GET BLOCK COUNT RIGHT
         177036
132 0762 005005          CLR     R5
133 0764 012701          MOV     #23000,R1
         023000
134 0770 012737          MOV     #100,@#167760           ;RESET INTERRUPTS
         000100
         167760
135 0776 012737          MOV     #40,@#167750
         000040
         167750
136 1004 106427          MTPS    #0
         000000
137 1010 000617          BR      TOISP                   ;GO BACK TO DISPLAY
138 1012 016704 GOON:    MOV     ERR,R4                  ;AT END OF RUN WRITE ANY STRAY BLOCKS
         177002
139 1016 106427          MTPS    #200
         000200
140 1022 066705          ADD     ERR,R5
         176772
141 1026 162704          SUB     #1,R4                   ;SET POINTERS FOR WRITE
         000001
142 1032 012767          MOV     #23000,BUFFAD           ;SET BUFFER
         023000
         177000
143 1040 005204 GOOB:    INC     R4                      ;WRITE TO DISK
144 1042                 .WRITW  #AREA,#100,BUFFAD,#1000,R4
145 1110 103004          BCC     EXCELL
146 1112                 .PRINT  #BDWRT                  ;PRINT BAD WRITE MSG
147 1120                 .EXIT                           ;GO BACK TO MONITOR
148 1122 062767 EXCELL:  ADD     #1000,BUFFAD            ;GO TO NEXT BLOCK
         001000
         176710
149 1130 020405          CMP     R4,R5                   ;ALL BLOCKS DONE?
150 1132 001342          BNE     GOOB                    ;BR BACK IF NOT
151 1134                 .CLOSE  #100                    ;CLOSE CHANNEL
152 1146 010567          MOV     R5,ERR                  ;SAVE BLOCK COUNT
         176646
153 1152 012700          MOV     #23000,R0               ;CLEAR BUFFER AREA
         023000
154 1156 010001          MOV     R0,R1
```

```
155 1160 062701          ADD     #12000,R1
         012000
156 1164 005020 CLER1:   CLR     (R0)+
157 1166 020001          CMP     R0,R1
158 1170 001375          BNE     CLER1
159 1172 000453          BR      BREAK
160                      ;
161                      ;
162                      ;
163                      ;ROUTINE TO HANDLE TTY INTERRUPTS
164                      ;S = START, H = HOLD, E = END OF RUN
165                      ;OTHERWISE, IGNORE THE CHARACTER
166                      ;
167                      ;
168                      ;
169 1174 013700 INPUT:   MOV     @#177562,R0             ;GET CHAR
         177562
170 1200 012737          MOV     #010000,@#44
         010000
         000044
171 1206 105737 T6:      TSTB    @#177564
         177564
172 1212 002375          BGE     T6
173 1214 042700          BIC     #177400,R0              ;STRIP EXTRA BITS OFF
         177400
174 1220 020027          CMP     R0,#310                 ;SEE IF AN H
         000310
175 1224 001007          BNE     NOTH                    ;BR ON IF NOT
176 1226 016701          MOV     SWEEP,R1
         176612
177 1232 062701          ADD     #040000,R1              ;ADD HOLD SIGNAL TO SWEEP
         040000
178 1236 010137          MOV     R1,@#167762             ;PUT TO LC
         167762
179 1242 000002          RTI
180 1244 020027 NOTH:    CMP     R0,#323                 ;SEE IF AN S FOR START
         000323
181 1250 001007          BNE     NOTS                    ;BR ON IF NOT
182 1252 016701          MOV     SWEEP,R1
         176566
183 1256 062701          ADD     #020000,R1              ;ADD START SIGNAL TO IT
         020000
184 1262 010137          MOV     R1,@#167762
         167762
185 1266 000002          RTI
186 1270 020027 NOTS:    CMP     R0,#305                 ;SEE IF E FOR END
         000305
```

```
187 1274 001401          BEQ     ISE                     ;BR IF IT IS
188 1276 000002          RTI
189 1300 005037 ISE:     CLR     @#177560                ;DISABLE INTERRUPTS
         177560
190 1304 005037          CLR     @#167760
         167760
191 1310 005037          CLR     @#167750
         167750
192 1314 012600          MOV     (SP)+,R0                ;POP STACK TO GET BACK
193 1316 012600          MOV     (SP)+,R0                ;TO CONTROL ROUTINE AT GOON
194 1320 000634          BR      GOON
195 1322 016701 BREAK:   MOV     SWEEP,R1                ;SET LC BACK TO INITIAL COND.
         176516
196 1326 062701          ADD     #010000,R1
         010000
197 1332 010137          MOV     R1,@#167762
         167762
198 1336 005067          CLR     BLKNM
         176464
199 1342 106427          MTPS    #200
         000200
200 1346 016737          MOV     TTY1,@#60               ;REPLACE TTY MONITOR VECTORS
         0000006
         000060
201 1354 016737          MOV     TTY2,@#62
         0000006
         000062
202 1362 012703          MOV     #23000,R3               ;SET BUFFER BEGIN
         023000
203 1366 005001          CLR     R1
204 1370                 .LOOKUP #AREA,#0,#FONAM         ;OPEN DATA FILE AGAIN
205 1420 103004          BCC     READ                    ;GOOD LOOK? GO READ
206 1422                 .PRINT  #BDLOOK                 ;PRINT BAD LOOK MSG
207 1430                 .EXIT                           ;GO BACK TO MONITOR
208 1432       READ:     .READW  #AREA,#0,R3,#1000,R1    ;GET BUFFER FILLED
209 1474 103004          BCC     GOOD
210 1476                 .PRINT  #BDREAD
211 1504                 .EXIT                           ;GO BACK TO MONITOR
212 1506 010302 GOOD:    MOV     R3,R2                   ;SET UP DISPLAY PTRS
213 1510 062702          ADD     #1000,R2
         001000
214 1514 010304          MOV     R3,R4
215 1516 012737          MOV     #3,@#167750             ;OUTPUT TRIGGER SIGNAL
         000003
         167750
216 1524 012737          MOV     #0,@#167750
         000000
```

```
                167750
217 1532 012437 LOOP:   MOV     (R4)+,@#176760          ;MOV DATA PT TO DAC
         176760
218 1536 105737         TSTB    @#177560                ;SEE IF CONSOLE INPUT
         177560
219 1542 002403         BLT     CONS
220 1544 020402         CMP     R4,R2                   ;CHECK IF BUFFER DONE
221 1546 001371         BNE     LOOP                    ;GO BACK IF NOT
222 1550 000756         BR      GOOD
223 1552 013705 CONS:   MOV     @#177562,R5             ;GET CHARACTER
         177562
224 1556 022705         CMP     #316,R5                 ;CHECK IF 'N'
         000316
225 1562 001415         BEQ     FORW                    ; LEAVE IF ITIS
226 1564 022705         CMP     #311,R5                 ;CHECK IF I
         000311
227 1570 001005         BNE     DEC
228 1572 020167         CMP     R1,ERR
         176222
229 1576 001743         BEQ     GOOD                    ;INC BLOCK TO NEXT ONE
230 1600 005201         INC     R1
231 1602 000713         BR      READ
232 1604 020127 DEC:    CMP     R1,#0                   ;DEC BLOCK TO SEE PREVIOUS ONE
         000000
233 1610 001736         BEQ     GOOD
234 1612 005301         DEC     R1
235 1614 000706         BR      READ
236 1616 012700 FORW:   MOV     #PEAKAN,R0              ;ANALYZE PEAKS?
         000144'
237 1622 112037 PUTOT:  MOVB    (R0)+,@#177566
         177566
238 1626 105737 SIT:    TSTB    @#177564
         177564
239 1632 002375         BGE     SIT
240 1634 122710         CMPB    #0,(R0)
         000000
241 1640 001370         BNE     PUTOT
242 1642 105737 OOP:    TSTB    @#177560                ;GET ANSWER
         177560
243 1646 002375         BGE     OOP
244 1650 013700         MOV     @#177562,R0
         177562
245 1654 013737         MOV     @#177562,@#177566       ;ECHO IT
         177562
         177566
246 1662 005267         INC     ERR
         176132
```

```
247 1666 005067          CLR     BLKCT
         176132
248 1672 020027          CMP     R0,#331                 ;SEE IF ANSWER IS YES
         000331
249 1676 001127          BNE     OUTFL
250 1700 012767          MOV     #0,BLKNM                ;SET BLOCK COUNTS
         000000
         176120
251 1706 005067          CLR     BAKCAL
         176110
252 1712 005004          CLR     R4
253 1714 005000          CLR     R0
254 1716 012702 BUFFUL:  MOV     #30,R2                  ;SET FOR MAX OF 24 BLOCKS PROCESSED
         000030
255 1722 012767          MOV     #1000,ACQCNT            ;SET FOR 512 POINTS PER BLK
         001000
         176100
256 1730 105767          TSTB    BAKCAL                  ;CHECK IF PEAK OVERLAPPED BLK
         176066
257 1734 001402          BEQ     NOBAK
258 1736 005367          DEC     BLKCT
         176062
259 1742 016704 NOBAK:   MOV     BLKCT,R4                ;DONT HAVE TO GET PREVIOUS BLOCK
         176056
260 1746 012700          MOV     #23000,R0               ;CLEAR BUFFER
         023000
261 1752 010001          MOV     R0,R1
262 1754 062701          ADD     #30000,R1               ;END OF BLOCK
         030000
263 1760 005020 CLR3:    CLR     (R0)+
264 1762 020001          CMP     R0,R1
265 1764 001375          BNE     CLR3
266 1766 026767          CMP     BLKCT,ERR               ;GET BLOCK COUNT
         176032
         176024
267 1774 001470          BEQ     OUTFL
268 1776 012703          MOV     #23000,R3
         023000
269                      ;
270                      ;READ A BLOCK INTO THE BUFFER
271                      ;
272 2002         RD2:    .READW  #AREA,#0,R3,#1000,R4
273 2044 103004          BCC     BUFGET
274 2046                 .PRINT  #BDREAD                 ;PRINT BAD READ MSG
275 2054                 .EXIT                           ;GO BACK TO MONITOR
276 2056 005302 BUFGET:  DEC     R2                      ;GET ALL THE BLOCKS
277 2060 001422          BEQ     AOK
```

```
278 2062 062703          ADD     #1000,R3
         001000
279 2066 020327          CMP     R3,#36000              ;CHECK IF ABOUT TO WRITE ON
         036000
280 2072 001002          BNE     NOSKIP                 ;BAD CORE  IF SO SKIP OVER
281 2074 062703          ADD     #1000,R3
         001000
282 2100 062767 NOSKIP:  ADD     #1000,ACQCNT           ;GO TO NEXT BLOCK
         001000
         175722
283 2106 005204          INC     R4
284 2110 005267          INC     BLKCT
         175710
285 2114 026767          CMP     BLKCT,ERR              ; SEE IF DONE
         175704
         175676
286 2122 001401          BEQ     AOK
287 2124 000726          BR      RD2                    ;GO READ ANOTHER BLOCK
288 2126 005001 AOK:     CLR     R1                     ;CALL THE PEAK PROCESSOR
289 2130 016701          MOV     BLKNM,R1
         175672
290 2134 004767          JSR     PC,PEAK
         0000006
291 2140 026767          CMP     BLKCT,ERR              ;CHECK IF ALL BLOCKS DONE
         175660
         175652
292 2146 001403          BEQ     OUTFL                  ;LEAVE IF TRUE
293 2150 005267          INC     BLKNM
         175652
294 2154 000660          BR      BUFFUL                 ;GET MORE IF NOT
295 2156         OUTFL:  .CLOSE  #0                     ;CLOSE CHANNEL
296 2170 000207          RTS     PC                     ;RETURN TO REPORT
297      000001'         .END
```

```
MAIN CONTROL ROUTINE    RT-11 MACRO VM02-12   27-JUN-78 PAGE 1+
SYMBOL TABLE

ACQCNT  000030RG        AUK     002126R         AREA    000000R
BAKCAL  000022RG        BUENT   000046R         BDLOOK  000120R
BUREAD  000133R         BUWRT   000104R         BUWRT1  000070R
BEGIN   000166RG        BEIN    000060R         BLKCT   000024R
BLKNM   000026RG        BREAK   001322R         BUFFAD  000040R
BUFFUL  001716R         BUFGET  002056R         CLEAR   000236R
CLER1   001164R         CLR2    000724R         CLR3    001760R
CONS    001552R         DEC     001604R         DN      000032RG
DSR     000514R         ERR     000020R         EXC     000666R
EXCELL  001122R         FDNAM = ****** G        FILSAV  000526R
FLOW    000042R         FORW    001616R         GOOB    001040R
GOOD    001506R         GOON    001012R         INPUT   001174R
ISE     001300R         LOOP    001532R         NOBAK   001742R
NOSKIP  002100R         NOTH    001244R         NOTS    001270R
OFFSET  000034RG        ON      000472R         OOP     001642R
OUTFL   002156R         PC    =%000007          PEAK  = ****** G
PEAKAN  000144R         PEAKNM  000036RG        PUTOT   001622R
RD2     002002R         READ    001432R         RESET   000704R
RIGHT   000606R         R0    =%000000          R1    =%000001
R2    =%000002          R3    =%000003          R4    =%000004
R5    =%000005          SETUP = ****** G        SIT     001626R
SP    =%000006          START   000316R         SWEEP   000044R
TDISP   000450R         TTY1  = ****** G        TTY2  = ****** G
T10     000336R         T11     000342R         T6      001206R
...V2 = 000001
. ABS.  000000      000
        002172      001
ERRORS DETECTED: 0
FREE CORE: 2075. WORDS

TOTAL,LP:/L:BEX=TOTAL
```

```
                        .TITLE  INTERRUPT SETUP, DATA INT, EOD INT
                        .GLOBL  SETUP,TTY1,TTY2
                        .MCALL ..V2..,.REGDEF
000000                  ..V2..
000000                  .REGDEF
                        ;
                        ;
                        ;THIS ROUTINE SETS UP THE INTERUPTS THAT ARE NECESSARY TO
                        ;COLLECT THE DATA, END THE DATA COLLECTION, AND
                        ;TAKE CARE OF USER TTY CHANGES
                        ;
                        ;
                        ;VARIABLE DEFINITIONS
00000 000000 TTY1:      .WORD   0
00002 000000 TTY2:      .WORD   0
00004 000000 ARRAY:     .WORD   000000
                        ;


                        ;THIS SECTION SETS UP ALL NECESSARY INTERRUPTS AND
                        ;TRANSFER ADDRESSES
                        ;
00006 012737 SETUP:     MOV     #DATA,@#324
       000146'
       000324
00014 012737            MOV     #200,@#326              ;SET DATA INTERRUPT VECTOR
       000200
       000326
00022 012737            MOV     #EOD,@#310
       000230'
       000310
00030 012737            MOV     #200,@#312              ;SET END OF DATA VECTOR
       000200
       000312
00036 013767            MOV     @#60,TTY1               ;SAVE TTY MONITOR INTRT VECTORS
       000060
       177734
00044 013767            MOV     @#62,TTY2
       000062
       177730
00052 012737            MOV     #010000,@#44            ;SET MONITOR FOR SPECIAL I/O
       010000
       000044
00060 010467            MOV     R4,ARRAY                ;PUT ADDRESS OF BUFFER IN ARRAY
       177720
00064 012737            MOV     #000002,@#176770        ;ADC SET
       000002
       176770
```

```
31 00072 010137          MOV     R1,@#60                 ;SET UP CONSOLE INT VECTOR
         000060
32 00076 012737          MOV     #200,@#62
         000200
         000062
33 00104 012737          MOV     #100,@#177560
         000100
         177560
34 00112 012737          MOV     #000000,@#330           ;CLEAR DONE FLAG
         000000
         000330
35 00120 005002          CLR     R2
36 00122 012737          MOV     #000100,@#167760        ;ENABLE INT FOR END OF DATA
         000100
         167760
37 00130 012737          MOV     #000040,@#167750        ;ENABLE INT FOR TIMER,DATA READY
         000040
         167750
38 00136 106427          MTPS    #000                    ;ENABLE ALL INT
         000000
39 00142 000207          RTS     PC
40 00144 000000          HALT
41                       ;
42                       ;
43                       ;THIS IS THE DATA INTERRUPT ROUTINE
44                       ;IT PUTS ALL POINTS INTO A BUFFER
45                       ;SET IN THE CONTROL ROUTINE
46                       ;AND KEEPS TRACK OF THE BLOCK
47                       ;IT IS IN
48                       ;
49                       ;
50 00146 005037 DATA:    CLR     @#176770                ;SET CONVERSION OF NUMBER
         176770
51 00152 105737 WINE:    TSTB    @#176770                ;WAIT TILL CONVERSION DONE
         176770
52 00156 002375          BGE     WINE
53 00160 013711          MOV     @#176772,(R1)           ;STORE IN BUFFER
         176772
54 00164 042721          BIC     #170000,(R1)+           ;REMOVE NEG SIGNS
         170000
55 00170 020201          CMP     R2,R1                   ;CHECK IF BUFFER FULL
56 00172 001401          BEQ     NEWBUF
57 00174 000002          RTI
58 00176 062702 NEWBUF:  ADD     #1000,R2                ;GO TO NEXT BLOCK
         001000
59 00202 005205          INC     R5                      ;INC BLOCK COUNT
60 00204 020527          CMP     R5,#12                  ;SEE IF END OF THE BUFFER
```

```
                000012
61 00210 001006         BNE     OUTMAN                  ;RETURN IF NOT
62 00212 016702         MOV     ARRAY,R2                ;RESET BUFFER ADDRESS
         177566
63 00216 010203         MOV     R2,R3                   ;AND END OF BUFFER
64 00220 010201         MOV     R2,R1
65 00222 062702         ADD     #1000,R2
         001000
66 00226 000002 OUTMAN: RTI                             ;GO BACK TO CONTROL ROUTINE
67                      ;
68                      ;
69                      ;THIS ROUTINE ANSWERS THE INTERRUPT
70                      ;WHEN THE END OF DATA SIGNAL GOES
71                      ;HIGH. IT SETS A FLAG AND DISABLES
72                      ;INTERRUPTS. IT ALSO OUTPUTS AN 'E'
73                      ;TO SHOW IT IS DONE
74                      ;
75                      ;
76 00230 005037 EOD:    CLR     @#167760                ;ROUTINE TO END LC RUN
         167760
77 00234 106427         MTPS    #200                    ;DISABLE ALL INTERRUPTS
         000200
78 00240 005037         CLR     @#177560
         177560
79 00244 005037         CLR     @#167750
         167750
80 00250 012737         MOV     #105,@#177566           ;PUT 'E' OUT TO SHOW ITS DONE
         000105
         177566
81 00256 105737 WAIT:   TSTB    @#177564
         177564
82 00262 002375         BGE     WAIT
83 00264 012737         MOV     #001,@#330              ;SET DONE FLAG
         000001
         000330
84 00272 000002         RTI
85 00274 000000         HALT
86       000001'        .END
```

```
INTERRUPT SETUP, DATA INT, EOD  RT-11 MACRO VM02-12   27-JUN-78 PAGE 1+
SYMBOL TABLE

ARRAY   000004R         DATA    000146R         EOD     000230R
NEWBUF  000176R         OUTMAN  000226R         PC    =%000007
R0    =%000000          R1    =%000001          R2    =%000002
R3    =%000003          R4    =%000004          R5    =%000005
SETUP   000006RG        SP    =%000006          TTY1    000000RG
TTY2    000002RG        WAIT    000256R         WINE    000152R
...V2 = 000001
. ABS.  000000      000
        000276      001
ERRORS DETECTED: 0
FREE CORE: 2967. WORDS

INTRUP,LP:<INTRUP
```

```
                        .TITLE  DAY CLOCK SET ROUTINE
                        .GLOBL  TIMIN
                        .MCALL  ..V2..,.REGDEF,.EXIT
000000                  ..V2..
000000                  .REGDEF
                        ;
                        ;
                        ;THIS ROUTINE TAKES THE TIME OF DAY
                        ;AND CONVERTS IT TO A FORMAT FOR THE
                        ;DAY CLOCK. IT THEN OUTPUTS IT TO THE
                        ;CLOCK, USING THE LOAD SIGNALS
                        ;THE FORMAT OF THE INPUT IS:
                        ;   12:34    WHEN THE 4 IS INPUT
                        ;THE ROUTINE TAKES OVER, NO CARRIAGE
                        ; RETURN IS NECESSARY.
                        ;THE ROUTINE ALSO READS THE CLOCK
                        ;AND OUTPUTS THE TIME. IT IS OUTPUT AS:
                        ; DAY;HOUR:MIN
                        ;THIS IS A 24 HOUR CLOCK SO USE 19:00
                        ;FOR 7:00 PM
                        ;
                        ;
00000 005001 TIMIN:     CLR     R1
00002 052737            BIS     #010000,@#44    ;SET SO USER INPUT AVAILABLE
      010000
      000044
00010 005000            CLR  R0
00012 004767            JSR  PC,IN              ;GET TENS DIGIT OF HOUR
      001050
00016 010005            MOV     R0,R5           ;ECHO
00020 004767            JSR     PC,OUT
      001062
00024 020027            CMP     R0,#2           ;IF TENS IS TWO THEN PUT 20 IN
      000002
00030 001003            BNE     ONEI
00032 012701            MOV     #24,R1
      000024
00036 000405            BR      NEXT
00040 020027 ONEI:      CMP     R0,#1           ;IF TENS IS ONE, THEN PUT 10 IN
      000001
00044 001002            BNE     NEXT
00046 012701            MOV  #12,R1
      000012
00052 004767 NEXT:      JSR     PC,IN           ;GET ONES DIGIT OF HOUR
      001010
00056 010005            MOV     R0,R5
00060 004767            JSR     PC,OUT
```

```
                001022
39 00064 060001         ADD  R0,R1              ;ADD UP TOTAL HOURS
40 00066 004767         JSR  PC,IN              ;GET :
                000774
41 00072 010005         MOV     R0,R5
42 00074 004767         JSR     PC,OUT
                001006
43 00100 070127         MUL     #74,R1          ;CHANGE HOURS TO MIN (MUL BY 60)
                000074
44 00104 004767         JSR     PC,IN           ;GET TENS DIGIT OF MIN
                000756
45 00110 010005         MOV     R0,R5
46 00112 004767         JSR     PC,OUT
                000770
47 00116 020027 TEN:    CMP     R0,#0           ;CHANGE DECIMAL TEN TO OCTAL
                000000
48 00122 001404         BEQ     ONE
49 00124 062701         ADD     #12,R1
                000012
50 00130 005300         DEC     R0
51 00132 000771         BR      TEN
52 00134 004767 ONE:    JSR     PC,IN           ;GET ONES DIGIT OF MIN
                000726
53 00140 010005         MOV     R0,R5
54 00142 004767         JSR     PC,OUT
                000740
55 00146 060001         ADD     R0,R1           ;ADD UP TOTAL MIN
56 00150 005002         CLR     R2
57 00152 070127         MUL     #12,R1          ;MUL BY 10 TO GET BETTER NUMBERS
                000012
58 00156 010104         MOV     R1,R4
59 00160 022701         CMP     #16040,R1       ;THIS SECTION SUBTRACTS THE VALUES
                016040
60 00164 100004         BPL     A               ;OF THE TIME INCREMENTS UNTIL THE
61 00166 162701         SUB     #16040,R1       ;NUMBER OF MIN IS TOO LOW TO
                016040
62 00172 062702         ADD     #002000,R2      ;REMOVE ANY MORE. BITS IN THE TIME
                002000
63 00176 022701 A:      CMP     #7020,R1        ;REGISTER ARE SET SO THAT THE CLOCK
                007020
64 00202 100004         BPL     B               ;CAN COUNT THE TIME.
65 00204 162701         SUB     #7020,R1
                007020
66 00210 062702         ADD     #001000,R2
                001000
67 00214 022701 B:      CMP     #4011,R1
                004011
```

```
68 00220 100004          BPL     C
69 00222 162701          SUB     #4011,R1
         004011
70 00226 062702          ADD     #000400,R2
         000400
71 00232 022701 C:       CMP     #1700,R1
         001700
72 00236 100004          BPL     D
73 00240 162701          SUB     #1700,R1
         001700
74 00244 062702          ADD     #000200,R2
         000200
75 00250 022701 D:       CMP     #740,R1
         000740
76 00254 100004          BPL     E
77 00256 162701          SUB     #740,R1
         000740
78 00262 062702          ADD     #000100,R2
         000100
79 00266 022701 E:       CMP     #360,R1
         000360
80 00272 100004          BPL     F
81 00274 162701          SUB     #360,R1
         000360
82 00300 062702          ADD     #000040,R2
         000040
83
84 00304 022701 F:       CMP     #211,R1
         000211
85 00310 100004          BPL     G
86 00312 162701          SUB     #211,R1
         000211
87 00316 062702          ADD     #000020,R2
         000020
88 00322 022701 G:       CMP     #100,R1
         000100
89 00326 100004          BPL     H
90 00330 162701          SUB     #100,R1
         000100
91 00334 062702          ADD     #000010,R2
         000010
92 00340 022701 H:       CMP     #40,R1
         000040
93 00344 100004          BPL     I
94 00346 162701          SUB     #40,R1
         000040
95 00352 062702          ADD     #000004,R2
```

```
                000004
96 00356 022701 I:      CMP     #20,R1
         000020
97 00362 100004         BPL     J
98 00364 162701         SUB     #20,R1
         000020
99 00370 062702         ADD     #000002,R2
         000002
100 0374 022701 J:      CMP     #11,R1
         000011
101 0400 100004         BPL     K
102 0402 162701         SUB     #11,R1
         000011
103 0406 062702         ADD     #000001,R2
         000001
104 0412 012737 K:      MOV     #140000,@#167752
         140000
         167752
105 0420 062702         ADD     #060000,R2      ;THIS SECTION PUTS THE VALUES OUT
         060000
106 0424 010237         MOV     R2,@#167752     ;TO THE CLOCK USING THE CODES 60000
         167752
107 0430 012702         MOV     #177700,R2      ;FOR THE HIGH CLOCK AND 40000 FOR THE
         177700
108 0434 013702 WAI:    MOV     @#167754,R2     ;LOW CLOCK. THERE ARE WAIT LOOPS TO ALLOW
         167754
109 0440 042702         BIC     #157777,R2      ;THE LOAD SIGNALS TO SETTLE.
         157777
110 0444 001773         BEQ     WAI
111 0446 012737         MOV     #040000,@#167752
         040000
         167752
112 0454 013702 WA:     MOV     @#167754,R2
         167754
113 0460 042702         BIC     #137777,R2
         137777
114 0464 001773         BEQ     WA
115 0466 012737 TIMOUT: MOV     #100000,@#167752;THIS PART GETS THE TIME FROM THE
         100000
         167752
116 0474 013701         MOV     @#167754,R1     ;CLOCK BY USING THE READ SIGNALS
         167754
117 0500 012737         MOV     #120000,@#167752;100000 FOR THE HIGH CLOCK
         120000
         167752
118 0506 013702         MOV     @#167754,R2     ;AND 120000 FOR THE LOW CLOCK. THE TIMES
         167754
```

```
119 0512 010104          MOV     R1,R4
120 0514 005001          CLR     R1              ;ARE THEN READ IN
121 0516 005003          CLR     R3
122 0520 006102          ROL     R2              ;THIS PART CHANGES THE CLOCK VALUES
123 0522 006102          ROL     R2              ;BACK TO OCTAL TIMES BY ADDING VALUES
124 0524 006102          ROL     R2              ;FOR EVERY BIT SET.BITS 12 AND
125 0526 006102          ROL     R2              ;11 TELL THE NUMBER OF DAYS.
126 0530 103002          BCC     L               ;THE LOWER ONES TELL THE NUMBER OF
127 0532 012703          MOV     #2,R3           ;MINUTES GONE BY.
             000002
128 0536 006102 L:       ROL     R2
129 0540 103002          BCC     M
130 0542 062703          ADD     #1,R3
             000001
131 0546 006102 M:       ROL     R2
132 0550 103002          BCC     N
133 0552 062701          ADD     #16040,R1
             016040
134 0556 006102 N:       ROL     R2
135 0560 103002          BCC     O
136 0562 062701          ADD     #7020,R1
             007020
137 0566 006102 O:       ROL     R2
138 0570 103002          BCC     P
139 0572 062701          ADD     #4011,R1
             004011
140 0576 006102 P:       ROL     R2
141 0600 103002          BCC     Q
142 0602 062701          ADD     #1700,R1
             001700
143 0606 006102 Q:       ROL     R2
144 0610 103002          BCC     R
145 0612 062701          ADD     #740,R1
             000740
146 0616 006102 R:       ROL     R2
147 0620 103002          BCC     S
148 0622 062701          ADD     #360,R1
             000360
149 0626 006102 S:       ROL     R2
150 0630 103002          BCC     T
151 0632 062701          ADD     #211,R1
             000211
152 0636 006102 T:       ROL     R2
153 0640 103002          BCC     U
154 0642 062701          ADD     #100,R1
             000100
155 0646 006102 U:       ROL     R2
```

```
156 0650 103002          BCC     V
157 0652 062701          ADD     #40,R1
         000040
158 0656 006102 V:       ROL     R2
159 0660 103002          BCC     W
160 0662 062701          ADD     #20,R1
         000020
161 0666 006102 W:       ROL     R2
162 0670 103002          BCC     X
163 0672 062701          ADD     #11,R1
         000011
164 0676 006104 X:       ROL     R4
165 0700 006104          ROL     R4
166 0702 006104          ROL     R4
167 0704 006104          ROL     R4
168 0706 006104          ROL     R4
169 0710 103002          BCC     Y
170 0712 062701          ADD     #4,R1
         000004
171 0716 006104 Y:       ROL     R4
172 0720 006104          ROL     R4
173 0722 103002          BCC     Z
174 0724 062701          ADD     #2,R1
         000002
175 0730 005000 Z:       CLR     R0
176 0732 071027          DIV     #12,R0
         000012
177 0736 010001          MOV     R0,R1
178 0740 005000          CLR     R0
179 0742 071027          DIV     #74,R0
         000074
180 0746 005004          CLR     R4
181 0750 022700 3A:      CMP     #12,R0
         000012
182 0754 101004          BHI     AA
183 0756 162700          SUB     #12,R0
         000012
184 0762 005204          INC     R4
185 0764 000771          BR      BA
186 0766 010305 AA:      MOV     R3,R5
187 0770 004767          JSR     PC,OUT          ;NUMBER OF DAYS
         000112
188 0774 012705          MOV     #13,R5
         000013
189 1000 004767          JSR     PC,OUT          ;PUT ; TO SEPARATE NUMBERS
         000102
190 1004 010405          MOV     R4,R5
```

```
191 1006 004767          JSR     PC,OUT          ;PUT TENS OF HOURS OUT
         000074
192 1012 010005          MOV     R0,R5
193 1014 004767          JSR     PC,OUT          ;PUT ONES OF HOURS OUT
         000066
194 1020 012705          MOV     #12,R5
         000012
195 1024 004767          JSR     PC,OUT          ;PUT : OUT
         000056
196 1030 005000          CLR     R0
197 1032 022701 AD:      CMP     #12,R1
         000012
198 1036 101004          BHI     AC
199 1040 162701          SUB     #12,R1
         000012
200 1044 005200          INC     R0
201 1046 000771          BR      AD
202 1050 010005 AC:      MOV     R0,R5
203 1052 004767          JSR     PC,OUT          ;PUT TENS OF MIN OUT
         000030
204 1056 010105          MOV     R1,R5
205 1060 004767          JSR     PC,OUT          ;PUT ONES OF MIN OUT
         000022
206 1064                 .EXIT
207 1066 105737 IN:      TSTB    @#177560
         177560
208 1072 002375          BGE     IN
209 1074 013700          MOV     @#177562,R0
         177562
210 1100 162700          SUB     #260,R0
         000260
211 1104 000207          RTS     PC
212 1106 062705 OUT:     ADD     #260,R5
         000260
213 1112 010537          MOV     R5,@#177566
         177566
214 1116 105737 T3:      TSTB    @#177564
         177564
215 1122 002375          BGE     T3
216 1124 000207          RTS     PC
217      000001'         .END
```

```
DAY CLOCK SET ROUTINE   RT-11 MACRO VM02-12   27-JUN-78 PAGE 1+
SYMBOL TABLE

A       000176R         AA      000766R         AC      001050R
AD      001032R         B       000214R         BA      000750R
C       000232R         D       000250R         E       000266R
F       000304R         G       000322R         H       000340R
I       000356R         IN      001066R         J       000374R
K       000412R         L       000536R         M       000546R
N       000556R         NEXT    000052R         O       000566R
ONE     000134R         ONEI    000040R         OUT     001106R
P       000576R         PC    =%000007          Q       000606R
R       000616R         R0    =%000000          R1    =%000001
R2    =%000002          R3    =%000003          R4    =%000004
R5    =%000005          S       000626R         SP    =%000006
T       000636R         TEN     000116R         TIMIN   000000RG
TIMOUT  000466R         T3      001116R         U       000646R
V       000656R         W       000666R         WA      000454R
WAI     000434R         X       000676R         Y       000716R
Z       000730R         ...V2 = 000001
. ABS.  000000      000
        001126      001
ERRORS DETECTED: 0
FREE CORE: 2823. WORDS

TIME,LP:<TIME
```

```
1                       .TITLE PEAK PROCESSING ROUTINE
2          000000'      .CSECT P.MAC
3                       ;MODIFIED BY M.E. FIORINO AND GARY GISS
4                       ;JAN., 1978
5                       ;PEAK INFORMATION IS ACQUIRED POST RUN
6                       ;AND PRINTED OUT IN DECIMAL.
7                       .GLOBL PEAK,ACQCNT, DN,OFFSET,PEAKNM,BLKNM,RATE,BAKCAL
8                       .MCALL .REGDEF, ..V2..
9  000000               ..V2..
10 00000                .REGDEF
11
```

```
1                       ; PARAMETER-VARIABLE TABLE OFFSETS
2
3
4               ;       THE FOLLOWING LIST OF OFFSETS INTO THE PARAMETER TABLE - SCRATCH
5               ;       AREA DEFINED BY THE ADDRESS IN REGISTER FIVE(R5) WHEN "PEAK" IS
6               ;       CALLED IS USED BY THE ROUTINE.
7
8       000000 WT=0             ;PPWIDTH, WIDTH TEST PARAMETER
9       000002 GT=2             ;PPGATE, NO. OF CONSECUTIVE CHANGES TO CHANGE DIRECTION
10      000004 MM=4             ;PPMNIN, MINIMUM CHANGE CONSIDERED AN INCREASE
11      000006 SC=MM+2
12      000007 BS=SC+1          ;BASELINE SWITCH. 0=PEAK STARTED ON BASELINE
13                              ;                 1=LOOK FOR PEAK WIDTH
14                              ;                 2=LOOK FOR END ON BASELINE
15      000010 S1=BS+1          ;SWITCH SET IF DECREASING W/OUT A PREV. INCREASE
16      000011 S2=S1+1          ;SWITCH SET IF INCREASING AFTER A DECRES OR BASELINE
17      000012 CAL=S2+1         ;LO-ORDER PART OF AREA ACCUM. DURING SIGNAL INCREASE
18      000014 CAH=CAL+2        ;HI-ORDER PART
19      000016 DSL=CAH+2        ;LO-ORDER PART OF OLD SLOPE MINIMUM
20      000020 DSH=DSL+2        ;HI-ORDER PART
21      000022 TML=DSH+2        ;LO-ORDER PART OF CURRENT PT COUNTER
22      000024 TMH=TML+2        ;HI-ORDER PART
23      000026 MC=TMH+2         ;TEMPORARY COUNTER FOR # OF MINS FOUND
24      000030 CC=MC+2          ;TEMPORARY COUNTER FOR # OF MAXS FOUND
25      000032 MX=CC+2          ;CURRENT MAXIMUM
26      000034 KM=MX+2          ;VALUE OF PEAK LEADING MINIMUM
27      000036 KH=KM+2          ;HEIGHT OF LAST PEAK CREST
28      000040 KCL=KH+2         ;LO-ORDER PART OF TIME OF LAST PEAK CREST
29      000042 KCH=KCL+2        ;HI-ORDER PART
30      000044 CTL=KCH+2        ;LO-ORDER PART OF TIME OF LATEST MAXIMUM
31      000046 CTH=CTL+2        ;HI-ORDER PART
32      000050 KTL=CTH+2        ;LO-ORDER PART OF PEAK LEADING MINIMUM TIME
33      000052 KTH=KTL+2        ;HI-ORDER PART
34      000054 BTL=KTH+2        ;LO-ORDER PART OF LEADING BASELINE MINIMUM TIME
35      000056 BTH=BTL+2        ;HI-ORDER PART
36      000060 BM=BTH+2         ;LEADING BASELINE MINIMUM
37      000062 SLL=BM+2         ;LO-ORDER PART OF NEW OR CURRENT SLOPE*1000
38      000064 SLH=SLL+2        ;HI-ORDER PART
39      000066 DAL=SLH+2        ;LO-ORDER PART OF AREA
40      000070 DAH=DAL+2        ;HI-ORDER PART
41      000072 PH=DAH+2         ;PEAK HEIGHT    (SAVED FOR OUTPUT)
42      000074 PTL=PH+2         ;LO-ORDER PART OF TIME OF PEAK (SAVED FOR OUTPUT)
43      000076 PTH=PTL+2        ;HI-ORDER PART
44      000100 LMH=PTH+2        ;LEADING MINIMUM HEIGHT(SAVED FOR OUTPUT)
45      000102 LMTL=LMH+2       ;LO-ORDER PART OF TIME OF LEADING MINIMUM
46      000104 LMTH=LMTL+2      ;HI-ORDER PART
47      000106 WDL=LMTH+2       ;LO-ORDER PART OF WIDTH(IN SAMPLE CNTS)
```

```
48        000110 MDH=MDL+2        ;HI-ORDER PART
49        000112 MN=MDH+2         ;TRAILING MINIMUM
50        000114 MTL=MN+2         ;LO-ORDER PART OF TIME OF TRAILING MINIMUM
51        000116 MTH=MTL+2        ;HI-ORDER PART
52        000120 TYPE=MTH+2       ;OUTPUT PEAK TYPE INDICATOR
53                                ;       =1, IMPLIES PEAK ENDED ON BASELINE
54                                ;       =0, IMPLIES PEAK ENDED ON VALLEY
55                ;
56                ;
57        177560 RCVCSR = 177560
58        177562 RCVBUF = 177562
59        177564 XTMCSR = 177564
60        177566 XTMBUF = 177566
```

```
1                       ; PEAK ENTRY POINT
2                       ;CODE TO ASK FOR WT,GT,AND HM
3  000000 012705 PEAK:  MOV #PTABLE, R5
          002622'
4  000004 005067        CLR     BLKNM
          000006
5  000010 012767        MOV     #0,BLKNM
          000000
          000006
6  000016 010167        MOV     R1,BLKNM
          000006
7  000022 026727        CMP     BLKNM,#0
          000006
          000000
8  000030 001043        BNE     SKIPOT
9  000032 012737        MOV     #010000,@#44
          010000
          000044
10 00040 012700         MOV #WTMSG, R0          ;OUTPUT WT MSG
          003152'
11 00044 004767         JSR PC, MSG
          002450
12 00050 004767         JSR PC, OCTIN
          002470
13 00054 010065         MOV R0, WT(R5)
          000000
14 00060 012700         MOV #GTMSG, R0          ;OUTPUT GT MSG
          003202'
15 00064 004767         JSR PC, MSG
          002430
16 00070 004767         JSR PC, OCTIN
          002450
17 00074 010065         MOV R0, GT(R5)
          000002
18 00100 012700         MOV #HMMSG, R0          ;OUTPUT HM MSG
          003222'
19 00104 004767         JSR PC, MSG
          002410
20 00110 004767         JSR PC, OCTIN
          002430
21 00114 010065         MOV R0, HM(R5)
          000004
22 00120 012700         MOV     #TILE1,R0       ;OUTPUT TITLES
          003046'
23 00124 004767         JSR     PC,MSG
          002370
24 00130 012700         MOV     #TILE2,R0       ;OUTPUT SECOND TITLE
```

```
          003106'
25 00134  004767          JSR     PC,MSG
          002360
26 00140  012704 SKIPDT:  MOV     #23000,R4
          023000
27
28
29
30 00144  105065          CLRB    SC(R5)          ;CLEAR FLAGS. BASELINE SLOPE CHANGE CNTR
          000006
31 00150  105767          TSTB    BAKCAL
          000006
32 00154  001406          BEQ     NOCAL
33 00156  166767          SUB     ACQSAV,ACQCNT
          002434
          000006
34 00164  066765          ADD     ACQSAV,TML(R5)
          002426
          000022
35 00172  066767 NOCAL:   ADD     ACQCNT,ACQSAV
          000006
          002416
36 00200  105065          CLRB    BS(R5)          ;BASLINE SWITCH
          000007
37 00204  005065          CLR     CAL(R5)         ;ACC. AREA DURING INCREASE
          000012
38 00210  005065          CLR     CAH(R5)
          000014
39 00214  012765          MOV     #77777,DSH(R5)  ;SET SLOPE TO LARGE NO.
          077777
          000020
40 00222  012765          MOV     #177777,DSL(R5) ;
          177777
          000016
41 00230  016700          MOV DN, R0              ;UPDAT TIME FOR
          000000G
42 00234  070027          MUL #0, R0              ;INITIAL 1 POINTS MISSING
          000000
43 00240  010165          MOV R1, TML(R5)         ;IN SBUF
          000022
44 00244  005065          CLR TMH(R5)             ;CLR CURRENT TIME STORAGE
          000024
45 00250  162767          SUB #0, ACQCNT          ;ADJUST COUNT ALSO
          000000
          000006
46
47                        ;START OF NEW PEAK
```

```
48
49
50 00256 112765 000001 000010 N247:  MOVB    #1,S1(R5)        ;SET DECREASING WITHOUT INCREASE
51 00264 005065 000026               CLR     MC(R5)           ;CLR MINIMUM COUNTER
52 00270 005065 000030               CLR     CC(R5)           ;CLR MAXIMUM COUNTER
53 00274 105065 000011               CLRB    S2(R5)           ;SET NOT INCREASING AFTER A DECREASE
54 00300 012765 077777 000112        MOV     #77777,MN(R5)    ;SET MINIMUM TO LARGE NO.
55 00306 012765 100000 000032        MOV     #100000,MX(R5)   ;SET MAXIMUN TO VERY LOW NO.
56
57
58                                   ;GETTING NEXT POINT
59
60 00314 005367 000000G N100:  DEC ACQCNT               ;ALL PNTS DONE?
61 00320 002012                BGE NXTPNT               ;YES-RETURN TO DISPLAY
62 00322 105765 000011         TSTB    S2(R5)
63 00326 001406                BEQ     RETURN
64 00330 012767 000001 000000G MOV     #1,BAKCAL
65 00336 166567 000050 002252  SUB     KTL(R5),ACQSAV
66 00344 000207 RETURN: RTS    PC                       ;NO - PROCESS NEXT POINT
67 00346 066765 000000G 000022 NXTPNT: ADD     DN,TML(R5)       ;UPDATE CURRENT TIME
68 00354 005565 000024         ADC     TMH(R5)
69 00360 062704 000002         ADD #2, R4               ;UPDATE DATA POINTER
70 00364 020427 036000         CMP     R4,#36000
71 00370 001002                BNE     NOSKIP
72 00372 062704 001000         ADD     #1000,R4
73 00376 042714 NOSKIP: BIC    #170000,(R4)
```

```
          170000
74
75
76
77 00402 021465 3$:     CMP     (R4),MN(R5)     ;CMPARE   VALUE TO CURRENT MIN
          000112
78 00406 103402         BLO     2$
79 00410 000167         JMP     N150            ;IF GREATER THAN OR EQUAL, B+C
          000354
```

```
1                       ;NEW POINT LOWER THAN MINIMUM
2
3 000414 122765 2$:     CMPB    #1,BS(R5)       ;ARE WE LOOKING FOR PEAK WIDTH
         000001
         000007
4 000422 001026         BNE     N117            ;IF NOT, B+C
5 000424 016501         MOV     KH(R5),R1       ;IF SO, CHECK IF LOW ENOUGH TO
         000036
6 000430 066501         ADD     KM(R5),R1       ;  CALCULATE PEAK WIDTH
         000034
7
8
9 000434 006001         ROR     R1              ;(KM+KH)/2 = TERM
10
11
12 00436 021401         CMP     (R4),R1         ;IS CENTERED PT > TERM
13 00440 101017         BHI     N117            ;IF SO, B+C
14 00442 016500 4$:     MOV     TMH(R5),R0      ;OTHERWISE, CALCULATE 1/2WID AT 1/2HT
         000024
15 00446 016501         MOV     TML(R5),R1
         000022
16 00452 166500         SUB     KCH(R5),R0
         000042
17 00456 166501         SUB     KCL(R5),R1
         000040
18 00462 005600         SBC     R0
19 00464 010065         MOV     R0,WDH(R5)      ;SET WIDTH
         000110
20 00470 010165         MOV     R1,WDL(R5)
         000106
21 00474 105265         INCB    BS(R5)          ;INDICATE LOOKING FOR END PEAK ON BASLIN
         000007
22 00500 011465 N117:   MOV     (R4),MN(R5)     ;RECORD CURRENT MINIMUM VALUE
         000112
23
24
25 00504 016565         MOV     TMH(R5),MTH(R5) ;RECORD CURRENT PT COUNTER
         000024
         000116
26 00512 016565         MOV     TML(R5),MTL(R5)
         000022
         000114
27 00520 005265         INC     MC(R5)          ;INC MINS FOUND COUNTER
         000026
28 00524 011400         MOV     (R4),R0         ;INC ACC. AREA BY
29 00526 016701         MOV     DN,R1           ;  CENTERED PT. * SAMPLE RATE
         000006
```

```
30 00532 004767          JSR PC, MULUNS
         001132
31 00536 060165          ADD     R1,DAL(R5)
         000066
32 00542 005500          ADC     R0
33 00544 060065          ADD     R0,DAH(R5)
         000070
34
35
36 00550 066565          ADD     CAL(R5),DAL(R5) ; AND INCREASE ACC. AREA BY AREA ACC.
         000012
         000066
37 00556 005565          ADC     DAH(R5)         ;   DURING INCREASE
         000070
38 00562 066565          ADD     CAH(R5),DAH(R5)
         000014
         000070
39                       .ENABL  LSB
40 00570 105765          TSTB    BS(R5)          ;IS PEAK START ON BASELINE?
         000007
41 00574 001007          BNE     1$              ;IF NOT, B+C
42 00576 105765          TSTB    S1(R5)          ;ARE WE ON BACK SIDE OF PEAK
         000010
43 00602 001404          BEQ     1$              ;IF NOT, B+C
44 00604 005065          CLR     DAH(R5)         ;IF SO, ZERO ACC. AREA
         000070
45 00610 005065          CLR     DAL(R5)
         000066
46 00614 005065 1$:      CLR     CAH(R5)         ;CLEAR ACC. AREA DURING INCREASE
         000014
47 00620 005065          CLR     CAL(R5)
         000012
48 00624 026565          CMP     MC(R5),GT(R5)   ;COMPARE #MINS TO GATE
         000026
         000002
49 00632 002002          BGE     2$              ;IF GREATER OR EQUAL, B+C
50 00634 000167          JMP     N100            ;OTHERWISE
         177454
51 00640 105265 2$:      INCB    S1(R5)          ;INDICATE NOT ON BACKSIDE
         000010
52 00644 005065          CLR     CC(R5)          ;CLEAR MAXIMUMS COUNTER
         000030
53 00650 105765          TSTB    S2(R5)          ;WAS THERE AN INCR. AFTER A DECR. OR BASL
         000011
54 00654 001005          BNE     3$              ;IF SO, B+C
55 00656 012765          MOV     #100000,MX(R5)  ;IF NOT, SET MAX LARGE NEG NUM
         100000
```

```
                000032
56
57
58 00664 000167          JMP     N200
                000434
59 00670 016565 3$:      MOV     CTL(R5),PTL(R5) ;RECORD PEAK TIME
                000044
                000074
60 00676 016565          MOV     CTH(R5),PTH(R5)
                000046
                000076
61 00704 016565          MOV     KTL(R5),LMTL(R5);RECORD LEADING MINIMUM TIME
                000050
                000102
62 00712 016565          MOV     KTH(R5),LMTH(R5)
                000052
                000104
63 00720 016565          MOV     MX(R5),PH(R5)   ;RECORD PEAK HEIGHT
                000032
                000072
64 00726 016565          MOV     KH(R5),LMH(R5)  ;RECORD LEADING MINIMUM HEIGHT
                000034
                000100
65
66
67 00734 112765          MOVB    #1,BS(R5)       ;IND. LOOKING FOR PEAK WIDTH
                000001
                000007
68 00742 016565          MOV     CTL(R5),KCL(R5) ;KEEP TIME OF LASTEST MAX
                000044
                000040
69 00750 016565          MOV     CTH(R5),KCH(R5)
                000046
                000042
70 00756 016565          MOV     MX(R5),KH(R5)   ;KEEP VALUE OF LATEST MAX
                000032
                000036
71
72
73 00764 000167          JMP     N247            ;PREPARE FOR NEXT PEAK
                177266
```

```
1                       ;NEW POINT LARGER THAN CURRENT MINIMUM
2
3  000770 011400 N150:  MOV     (R4),R0          ;UPDATE AREA ACC. DURING INCR BY
4  000772 016701        MOV     DN,R1            ;CURRENT AREA
          000000G
5  000776 004767        JSR PC, MULUNS
          000666
6  001002 060165        ADD     R1,CAL(R5)
          000012
7  001006 005500        ADC     R0
8  001010 060065        ADD     R0,CAH(R5)
          000014
9
10
11 01014 016501         MOV     MX(R5),R1        ;CHECK IF NEW PT. SIGNIFICANTLY INCREASED
          000032
12 01020 066501         ADD     HM(R5),R1        ; OVER LATEST MAX
          000004
13                      .ENABL  LSB
14
15
16 01024 021401         CMP     (R4),R1
17 01026 003002         BGT     1$
18 01030 000167 2$:     JMP     N200
          000270
19 01034 016565 1$:     MOV     TML(R5),CTL(R5) ;IF INCREASE IS SIGNIFICANT, UPDATE MAX
          000022
          000044
20 01042 016565         MOV     TMH(R5),CTH(R5) ;  INFORMATION
          000024
          000046
21 01050 005265         INC     CC(R5)           ;UPDATE TIME,MAXS FOUND COUNT,
          000030
22 01054 011465         MOV     (R4),MX(R5)      ; AND HEIGHT
          000032
23
24
25 01060 026565         CMP     CC(R5),GT(R5)    ;CMPARE #MAXS TO GATE VALUE
          000030
          000002
26 01066 002002         BGE     3$               ;IF HIGH ENOUGH, B+C
27 01070 000167         JMP     N200             ;OTHERWISE WAIT
          000230
28 01074 005065 3$:     CLR     MC(R5)           ;CLR MINS COUNTER
          000026
29 01100 105765         TSTB    S1(R5)           ;ARE WE ON BACKSIDE OF PEAK
          000010
```

```
30 01104 001002          BNE     4$              ;IF SD, B+C
31 01106 000167          JMP     N160            ;OTHERWISE
         000204
32 01112 016500 4$:      MOV     MN(R5),R0       ;UPDATE.AREA ACC. DURING INCR. BY
         000112
33 01116 016701          MOV     DN,R1           ;  1/2 LEADING MINIMUM * SAMPLE RATE
         000000G
34 01122 004767          JSR PC, MULUNS
         000542
35
36
37 01126 006200          ASR     R0
38 01130 006001          ROR     R1
39 01132 060165          ADD     R1,CAL(R5)      ;UPDATE AREA
         000012
40 01136 005500          ADC     R0
41 01140 060065          ADD     R0,CAH(R5)
         000014
42 01144 105265          INCB    S2(R5)          ;IND INCREASE AFTER DECREASE
         000011
43 01150 105065          CLRB    S1(R5)          ;IND NO DECREASE
         000010
44 01154 016565          MOV     MTL(R5),KTL(R5) ;RECORD PEAK LEADING MINIMUM TIME
         000114
         000050
45 01162 016565          MOV     MTH(R5),KTH(R5)
         000116
         000052
46 01170 016565          MOV     MN(R5),KM(R5)   ;RECORD PEAK LEADING MINIMUM
         000112
         000034
47
48
49                       .ENABL  LSB
50 01176 122765          CMPB    #1,BS(R5)       ;DETERMINE WHAT TO DO BY WHERE WE ARE
         000001
         000007
51 01204 003413          BLE     1$              ;IF NOT STARTING ON BASELINE,END OF PEAK
52 01206 016565          MOV     KTL(R5),BTL(R5) ;IF STARTING ON BASELINE, RECORD NEW
         000050
         000054
53 01214 016565          MOV     KTH(R5),BTH(R5) ;  BASELINE TIME AND
         000052
         000056
54 01222 016565          MOV     KM(R5),BM(R5)   ;  HEIGHT
         000034
         000060
```

```
55
56
57 01230 000167          JMP     N160
         000062
58 01234 001017 1$:      BNE     2$              ;IF NOT LOOKING FOR WIDTH, B+C
59 01236 016501          MOV     MTL(R5),R1      ;OTHERWISE,MUST CAL. PEAK WIDTH NOW
         000114
60 01242 016500          MOV     MTH(R5),R0      ;  SINCE PEAK HAS ENDED BEFORE
         000116
61 01246 166500          SUB     KCH(R5),R0      ;  HALF HEIGHT WAS REACHED
         000042
62 01252 166501          SUB     KCL(R5),R1
         000040
63 01256 005600          SBC     R0
64 01260 006200          ASR     R0
65 01262 006001          ROR     R1              ;WIDTH EQUALS 1/2(MIN TIM - LEAD MIN TIM)
66 01264 010065          MOV     R0,WDH(R5)      ;SET WIDTH
         000110
67 01270 010165          MOV     R1,WDL(R5)
         000106
68 01274 005065 2$:      CLR     TYPE(R5)        ;IND. NOT ON BASELINE
         000120
69 01300 004767          JSR     PC,OUTPUT       ;OUTPUT PEAK DATA
         000506
70 01304 112765          MOVB    #2,BS(R5)       ;IND. LOOKING FOR END ON BASELINE
         000002
         000007
71 01312 000167          JMP     N160
         000000
72                       ;LOOKING FOR BASELINE
73
74 01316 016565 N160:    MOV     MX(R5),MN(R5)   ;SET CURRENT MAX TO CURRENT MIN
         000032
         000112
75
76
77 01324 122765 N200:    CMPB    #2,BS(R5)       ;HAS THE WIDTH BEEN CALCULATED YET
         000002
         000007
78                       .ENABL  LSB
79 01332 001402          BEQ     1$              ;IF SO, B+C
80 01334 000167 2$:      JMP     N100            ;IF NOT, GO GET NEXT PT
         176754
81 01340 105765 1$:      TSTB    S1(R5)          ;IS PT ON BACKSIDE OF PEAK
         000010
82 01344 001773          BEQ     2$              ;IF NOT, GO GET NEXT POINT
83 01346 016500          MOV     WDH(R5),R0
```

```
                000110
84 01352 070065         MUL     WT(R5),R0       ;CAL WIDTH*WIDTH TEST
         000000
85 01356 010103         MOV     R1,R3
86 01360 016500         MOV     HDL(R5),R0      ;IF SO, TEST TO SEE IF FAR ENOUGH PAST
         000106
87 01364 016501         MOV     WT(R5),R1       ;  TO START LOOKING FOR BASELINE
         000000
88 01370 004767         JSR PC, MULUNS
         000274
89 01374 060300         ADD     R3,R0
90 01376 016503         MOV     TML(R5),R3      ;CAL TIME FROM CREST TO CURRENT POINT
         000022
91 01402 016502         MOV     TMH(R5),R2
         000024
92 01406 166502         SUB     KCH(R5),R2
         000042
93 01412 166503         SUB     KCL(R5),R3
         000040
94 01416 005602         SBC     R2
95 01420 020200         CMP     R2,R0           ;COMPARE TWO CALCULATED QUANTITIES
96 01422 003005         BGT     4$              ;IF FIRST IS GREATER, B+C
97 01424 001402         BEQ     3$              ;IF HIGH PARTS EQUAL, GO CHECK LO PARTS
98 01426 000167 5$:     JMP     N100            ;IF FIRST IS LESS, GO GET NEXT POINT
         176662
99 01432 020301 3$:     CMP     R3,R1           ;COMPARE LSH OF TWO CAL. QUANTITIES
100 1434 103774         BLO     5$              ;IF FIRST IS LESS, GO GET NEXT POINT
101 1436 011400 4$:     MOV     (R4),R0         ;OTHERWISE, CALCULATE CURRENT SLOPE
102 1440 105067         CLRB    SIGN            ; CLEAR SIGN FLAG
         001154
103 1444 166500         SUB     BM(R5),R0       ;SLOPE = (CURRENT POINT-LEADING BASELINE
         000060
104                                             ;                       MINIMUM)/
105
106 1450 002003         BGE     6$              ;IF POSITIVE, B+C
107 1452 105267         INCB    SIGN            ;OTHERWISE, INDICATE SIGN CHANGE
         001142
108 1456 005400         NEG     R0
109
110 1460 005700 6$:     TST     R0              ;CHECK FOR ZERO SLOPE
111 1462 001425         BEQ     9$              ;IF NON-ZERO , B+C
112 1464 012701 7$:     MOV     #1000,R1
         001000
113 1470 004767         JSR PC, MULUNS
         000174
114
115
```

```
116 1474 016503          MOV     TML(R5),R3      ;           (CURRENT TIME-LEADING BASELINE
         000022
117 1500 016502          MOV     TMH(R5),R2      ;                        TIME)
         000024
118 1504 166502          SUB     BTH(R5),R2
         000056
119 1510 166503          SUB     BTL(R5),R3
         000054
120 1514 005603          SBC     R3
121 1516 004767          JSR     PC,UDIVD
         000206
122 1522 105767          TSTB    SIGN
         001072
123 1526 001403          BEQ     9$
124 1530 005400          NEG     R0              ;IF SIGNS DIFFERED, SLOPE NEGATIVE
125 1532 005401          NEG     R1
126 1534 005600          SBC     R0
127 1536 010065 9$:      MOV     R0,SLH(R5)
         000064
128 1542 010165          MOV     R1,SLL(R5)
         000062
129 1546 026565          CMP     OSH(R5),SLH(R5) ;COMPARE TO OLD SLOPE
         000020
         000064
130 1554 003005          BGT     10$             ;IF OLD GREATER, GO UPDATE AND TRY AGAIN
131 1556 002416          BLT     11$             ;IF OLD IS LESS, CHECK IF BASELINE
132 1560 026565          CMP     OSL(R5),SLL(R5) ;COMPARE TO LSH TO OLD SLOPE
         000016
         000062
133 1566 101412          BLOS    11$             ;IF OLD IS LESS, CHECK IF BASELINE
134 1570 016565 10$:     MOV     SLL(R5),OSL(R5) ;SET OLD TO NEW
         000062
         000016
135 1576 016565          MOV     SLH(R5),OSH(R5)
         000064
         000020
136 1604 105065          CLRB    SC(R5)          ;CLEAR SLOPE INCR COUNTER
         000006
137 1610 000167          JMP     N100            ;GO GET NEXT POINT
         176500
138 1614 105265 11$:     INCB    SC(R5)          ;INCRE SLOPE INCR COUNTER
         000006
139 1620 122765          CMPB    #2,SC(R5)       ;HAS SLOPE INCR TWICE IN A ROW
         000002
         000006
140 1626 002002          BGE     12$             ;IF SO, B+C
141 1630 000167          JMP     N100            ;OTHERWISE, GO GET NEXT POINT
```

```
          176460
142 1634 012765 12$:    MOV     #1,TYPE(R5)     ;IND. ENDING PEAK ON BASELINE
          000001
          000120
143 1642 004767         JSR     PC,OUTPUT       ;OUTPUT PEAK DATA BLOCK
          000144
144 1646 105065         CLRB    SC(R5)          ;CLEAR SLOPE INCR. COUNTER
          000006
145 1652 012765         MOV     #77777,OSH(R5)  ;SET SLOPE TO LARGE NO.
          077777
          000020
146 1660 105065         CLRB    BS(R5)          ;SET FLAG TO INDICATE STARTING AT BASELIN
          000007
147 1664 000167         JMP     N100            ;GO GET NEXT POINT
          176424
148                     ;
149                     ;
150                     ;               SUBROUTINES
151                     ;
152                     ;
153                     ;       ROUTINETO MULTIPLY R0 BY R1 AND STORE RESULT IN R0 AND R1
154                     ;
155                     ;
156 1670                MULUNS: .ENABL  LSB
157 1670 010346         MOV     R3,-(SP)        ;SAVE R2 AND R3
158 1672 010246         MOV     R2,-(SP)
159 1674 012703         MOV     #100000,R3      ;R3 CONTAINS COMPLETION FLAG
          100000
160 1700 005002         CLR     R2              ;R2 WILL CONTAIN HIGH ORDER PRODUCT
161 1702 006001 1$:     ROR     R1              ;C = MULTIPLIER BIT
162 1704 103001         BCC     2$              ;BR IF ZERO MULTIPLIER BIT
163 1706 060002         ADD     R0,R2           ;BIT MULTIPLY
164 1710 006002 2$:     ROR R2                  ;SHIFT HIGH ORDER
165 1712 006003         ROR     R3              ;BR IF MULT IS NOT DONE
166 1714 103372         BCC     1$              ;BR IF MULT NOT DONE
167 1716 010200         MOV     R2,R0           ;LEAVE RESULT IN R0 AND R1
168 1720 010301         MOV     R3,R1
169 1722 012602         MOV     (SP)+,R2
170 1724 012603         MOV     (SP)+,R3
171 1726 000207         RTS     PC
172                     ;ROUTINE TO DIVIDE R0,R1 BY R2,R3 WITH RESULT IN R0,R1
173                     ;
174 1730 010446 DDIVD:  MOV     R4, -(SP)       ;SAVE R4
175 1732 010546         MOV     R5, -(SP)       ;SAVE R5
176 1734 012746         MOV     #32.,-(SP)      ;GET LOOP COUNT
          000040
177 1740 005004         CLR     R4
```

```
178 1742 005005          CLR     R5              ;READY REMAINDERS
179                      .ENABL LSB
180 1744 006101 1$:      ROL     R1
181 1746 006100          ROL     R0              ;GET NEW BIT OF NUMERATOR
182 1750 006104          ROL     R4
183 1752 006105          ROL     R5
184 1754 020205          CMP     R2,R5           ;DOES DENOMINATOR FIT
185 1756 101007          BHI     3$              ;NO-B+C
186 1760 002402          BLT     2$
187 1762 020304          CMP     R3,R4           ;IF HIGH EQUAL, CHECK LOW
188 1764 101004          BHI     3$              ;BRANCH IF DENOM TOO BIG
189 1766 160304 2$:      SUB     R3,R4
190 1770 005605          SBC     R5              ;SUB DENOM FROM REMAINDER
191 1772 160205          SUB     R2,R5
192 1774 000261          SEC                     ;INDICATE NEW QUOTIENT BIT
193 1776 005316 3$:      DEC     (SP)            ;CHECK LOOP COUNT
194 2000 002361          BGE     1$              ;IF MORE, B+C
195 2002 005726          TST     (SP)+           ;UP SP
196 2004 012605          MOV     (SP)+,R5        ;RESTORE R5
197 2006 012604          MOV     (SP)+,R4        ;AND R4
198 2010 000207          RTS     PC              ;RESULT IN R3,R2
199                ;ROUTINES TO OUTPUT PEAK INFO TO TTY
200                ;
201                ;
202 2012 010446 OUTPUT:MOV R4, -(SP)             ;SAV R4
203 2014 005067          CLR     SPCER           ;CLR SPACE SETTER
         000574
204 2020 004767          JSR     PC,CRLF         ;OUTPUT CR AND LF TWICE
         000426
205 2024 004767          JSR     PC,CRLF
         000422
206 2030 012702          MOV     #1,R2           ;SET FLAG FOR NO DECIMAL PT
         000001
207 2034 012704          MOV     #PEAKNM,R4      ;GET ADDRESS OF PEAK NUMBER
         000006
208 2040 012767          MOV     #7,SPCE         ;SET SPACE BETWEEN VALUES
         000007
         000776
209 2046 004767          JSR     PC,SNGL         ;OUTPUT PEAK NUMBER
         000076
210 2052 012704          MOV     #AL,R4          ;GET PEAK AREA READY
         000066
211 2056 060504          ADD     R5,R4
212 2060 012767          MOV     #27,SPCE        ;SET SPACES BETWEEN VALUES
         000027
         000756
213 2066 004767          JSR     PC,DBL          ;OUTPUT PEAK AREA
```

```
                000040
214 2072 005204         INC     R4              ;INC POINTER TO PEAK TIME
215 2074 005204         INC     R4
216 2076 012767         MOV     #1,SPCE         ;SET DUMMY SPACE
                000001
                000740
217 2104 005002         CLR     R2
218 2106 004767         JSR     PC,DBL
                000020
219 2112 005065         CLR DAL(R5)             ;CLR AREA FOR NEXT PEAK
                000066
220 2116 005267         INC     PEAKNM
                000006
221 2122 005065         CLR DAH(R5)             ;CLR HIGH PART
                000070
222 2126 012604         MOV (SP)+, R4           ;RESTORE R4
223 2130 000207         RTS PC                  ;RETURN
224 2132 012401 DBL:    MOV (R4)+, R1           ;MOV LOW PART TO R1
225 2134 012400         MOV (R4)+, R0           ;MOV HI TO R0
226 2136 012703         MOV #12, R3             ;DIVISOR FOR BINARY TO DECIMAL
                000012
227 2142 004767         JSR PC, CONVRT          ;GO CONVERT
                000020
228 2146 000207         RTS PC                  ;AND RETURN TO CALLING
229                     ;
230                     ;
231 2150 012401 SNGL:   MOV (R4)+, R1           ;SNGL PREC DATUM IN R1
232 2152 005000         CLR R0                  ;SET R0 SO CAN USE SAME
233                                             ;CONVERSION ROUTINE FOR BOTH
234                                             ;DOUBLE PREC AND SNGL
235 2154 012703         MOV #12, R3             ;DIVISOR FOR CONVERSION
                000012
236 2160 004767         JSR PC, CONVRT          ;GO CONVERT
                000002
237 2164 000207         RTS PC                  ;RTRN TO CALLING PRGRM
238                     ;
239                     ;
240                     ;ROUTINE TO CONVERT FROM BINARY TO OTHER RADIX DEPENDING ON DIVISOR
241                     ;
242                     ;
243 2166 005067 CONVRT: CLR PRNTCT
                000650
244 2172 005067 CNVRT1: CLR DIVL
                000640
245 2176 005067         CLR DIVH
                000636
246                                             ;PRINT
```

```
247 2202 005700 AGAIN:  TST R0                  ;INIT# OR RESULT .GE. DIVISOR
248 2204 003401         BLE LOWTST
249 2206 000402         BR SUBTR
250 2210 020301 LOWTST: CMP R3, R1
251 2212 003010         BGT REMAIN
252 2214 160301 SUBTR:  SUB R3,R1               ;YES-SUBTRACT AGAIN
253 2216 005600         SBC R0
254 2220 062767         ADD #1, DIVL            ;INC QUOTIENT
         000001
         000610
255 2226 005567         ADC DIVH
         000606
256 2232 000753         BR AGAIN
257 2234 020127 REMAIN: CMP     R1,#0
         000000
258 2240 002765         BLT     SUBTR
259 2242 062701         ADD     #260,R1
         000260
260 2246 010146         MOV R1, -(SP)           ;SAVE ON STACK
261 2250 005267         INC PRNTCT              ;INC PRINT
         000556
262 2254 016700         MOV DIVH, R0
         000560
263 2260 016701         MOV DIVL, R1            ;COPY RESULT OF DIV TO R0
         000552
264                                             ;-R1
265 2264 005700         TST R0                  ;SEE IF DIVISOR LE. RESULT
266 2266 003341         BGT CNVRT1
267 2270 020301         CMP R3, R1
268 2272 003737         BLE CNVRT1
269 2274 020127         CMP     R1,#0
         000000
270 2300 002734         BLT     CNVRT1
271 2302 062701         ADD #260, R1            ;YES-GO SUB AGAIN
         000260
272                                             ;NO-CONVRT TO ASCII
273 2306 010146         MOV R1, -(SP)           ;AND SAVE ON STACK
274 2310 005267         INC PRNTCT
         000526
275             ;
276             ;ROUTINE TO PRINT PK INFO ON TTY
277             ;
278 2314 105767 PRINT:  TSTB XTMCSR
         177564'
279 2320 100375         BPL PRINT
280 2322 012667         MOV (SP)+, XTMBUF
         177566'
```

```
281 2326 026727          CMP     PRNTCT,#2
         000510
         000002
282 2334 001015          BNE     NODOT
283 2336 026727          CMP     RATE,#2
         000006
         000002
284 2344 001011          BNE     NODOT
285 2346 020227          CMP     R2,#0
         000000
286 2352 001006          BNE     NODOT
287 2354 105767 CIRCL:   TSTB    XTMCSR
         177564'
288 2360 100375          BPL     CIRCL
289 2362 012767          MOV     #056,XTMBUF
         000056
         177566'
290 2370 005267 NODOT:   INC     SPCER
         000220
291 2374 005367          DEC PRNTCT
         000442
292 2400 003345          BGT PRINT
293 2402 026767          CMP     SPCER,SPCE
         000206
         000434
294 2410 002017          BGE     REG
295 2412 166767          SUB     SPCER,SPCE
         000176
         000424
296 2420 016701          MOV     SPCE,R1
         000420
297 2424 066767          ADD     SPCE,SPCER
         000414
         000162
298 2432 105767 SPACE:   TSTB XTMCSR
         177564'
299 2436 100375          BPL SPACE
300 2440 012767          MOV #240, XTMBUF
         000240
         177566'
301 2446 077107          SOB R1, SPACE
302 2450 000207 REG:     RTS PC
303              ;
304              ;
305              ;ROUTINE TO GENERATE CARRIAGE RETURN/LINE FEED
306              ;
307 2452 105767 CRLF:    TSTB XTMCSR
```

```
          177564'
308 2456 100375          BPL CRLF
309 2460 012767          MOV #215,.XTMBUF
          000215
          177566'
310 2466 105767 LF:      TSTB XTMCSR
          177564'
311 2472 100375          BPL LF
312 2474 012767          MOV #212, XTMBUF
          000212
          177566'
313 2502 105767 SRQ:     TSTB     XTMCSR
          177564'
314 2506 100375          BPL      SRQ
315 2510 012767          MOV      #240,XTMBUF
          000240
          177566'
316 2516 000207          RTS PC
317                      ;ROUTINE TO OUTPUT MESSAGES
318 2520 105767 MSG:     TSTB XTMCSR
          177564'
319 2524 100375          BPL MSG
320 2526 121027          CMPB (R0), #0
          000000
321 2532 001403          BEQ ANS
322 2534 112067          MOVB (R0)+, XTMBUF
          177566'
323 2540 000767          BR MSG
324 2542 000207 ANS:     RTS PC
325                 ;
326                 ;
327                 ;
328                      ;ROUTINE TO BRING IN WT, GT, HM
329                      ;ASSUMES LESS THAN OR EQUAL TO 7 IS LARGEST NUMBER
330 2544 105767 OCTIN:   TSTB RCVCSR
          177560'
331 2550 100375          BPL OCTIN
332 2552 105767 ECH:     TSTB XTMCSR
          177564'
333 2556 100375          BPL ECH
334 2560 116767          MOVB RCVBUF, XTMBUF
          177562'
          177566'
335 2566 126727          CMPB RCVBUF, #215
          177562'
          000215
336 2574 001406          BEQ BACK
```

```
337 2576 005000          CLR R0
338 2600 116700          MOVB RCVBUF, R0
         177562'
339 2604 042700          BIC #177770,R0
         177770
340 2610 000755          BR GETIN
341 2612 000207 BACK:    RTS PC
342 2614 000000 SPCER:   .WORD   000000
343 2616 000000 ACCSAV:  .WORD   0
344 2620 000000 SIGN:    .WORD 0
345 2622        PTABLE:  .BLKW 70.
346 3036 000000 DIVL:    .WORD 0
347 3040 000000 DIVH:    .WORD 0
348 3042 000000 PRNTCT:  .WORD 0
349 3044 000000 SPCE:    .WORD   0
350                      .NLIST  BIN
351 3046        TILE1:   .ASCIZ  <12><15>/PEAK      PEAK        RETENTION/
352 3106        TILE2:   .ASCIZ  <12><15>/  NO.      AREA        TIME(SEC)  /
353                      .EVEN
354 3152        WTMSG:   .ASCIZ  <12><15>/ENTER WIDTH FACTOR  /
355                      .EVEN
356 3202        GTMSG:   .ASCIZ<12><15>/ENTER GATE  /
357                      .EVEN
358 3222        HMMSG:   .ASCIZ<12><15>/ENTER CHANGE FACTOR  /
359                      .EVEN
360                      .END
```

SYMBOL TABLE

```
ACQCNT= ****** G        ACQSAV  002616R  002  AGAIN   002202R  002
ANS     002542R  002    BACK    002612R  002  BAKCAL= ****** G
BLKNM = ****** G        BM    = 000060        BS    = 000007
BTH   = 000056          BTL   = 000054        CAH   = 000014
CAL   = 000012          CC    = 000030        CIRCL   002354R  002
CNVRT1  002172R  002    CONVRT  002166R  002  CRLF    002452R  002
CTH   = 000046          CTL   = 000044        DBL     002132R  002
DDIVD   001730R  002    DIVH    003040R  002  DIVL    003036R  002
DN    = ****** G        ECH     002552R  002  GT    = 000002
GTMSG   003202R  002    HH    = 000004        HHMSG   003222R  002
KCH   = 000042          KCL   = 000040        KH    = 000036
KM    = 000034          KTH   = 000052        KTL   = 000050
LF      002466R  002    LMH   = 000100        LMTH  = 000104
LMTL  = 000102          LOWTST  002210R  002  MC    = 000026
MN    = 000112          MSG     002520R  002  MTH   = 000116
MTL   = 000114          MULUNS  001670R  002  MX    = 000032
NOCAL   000172R  002    NODOT   002370R  002  NOSKIP  000376R  002
NXTPNT  000346R  002    N100    000314R  002  N117    000500R  002
N150    000770R  002    N160    001316R  002  N200    001324R  002.
N247    000256R  002    OAH   = 000070        OAL   = 000066
OCTIN   002544R  002    OFFSET= ****** G      OSH   = 000020
OSL   = 000016          OUTPUT  002012R  002  PC    =%000007
PEAK    000000RG 002    PEAKNM= ****** G      PH    = 000072
PRINT   002314R  002    PRNTCT  003042R  002  PTABLE  002622R  002
PTH   = 000076          PTL   = 000074        RATE  = ****** G
RCVBUF= 177562          RCVCSR= 177560        REG     002450R  002
REMAIN  002234R  002    RETURN  000344R  002  R0    =%000000
R1    =%000001          R2    =%000002        R3    =%000003
R4    =%000004          R5    =%000005        SC    = 000006
SIGN    002620R  002    SKIPDT  000140R  002  SLH   = 000064
SLL   = 000062          SNGL    002150R  002  SP    =%000006
SPACE   002432R  002    SPCE    003044R  002  SPCER   002614R  002
SRQ     002502R  002    SUBTR   002214R  002  S1    = 000010
S2    = 000011          TILE1   003046R  002  TILE2   003106R  002
TMH   = 000024          TML   = 000022        TYPE  = 000120
WDH   = 000110          WDL   = 000106        WT    = 000000
WTMSG   003152R  002    XTMBUF= 177566        XTMCSR= 177564
...V2 = 000001
. ABS.  000000  000
        000000  001
P.MAC   003252  002
ERRORS DETECTED: 0
FREE CORE: 2345. WORDS

P,LP:<P
```

The vita has been removed from the scanned document

INTERFACING OF AN LSI-LL MICRO PROCESSOR WITH THE SPECTRA-PHYSICS 3500B GRADIENT ELUTION LIQUID CHROMATOGRAPH

by

Gary Neal Giss

(ABSTRACT)

An LSI-11 micro-processor and a Spectra-Physics model 3500B gradient elution liquid chromatograph were interfaced for the purpose of automating the chromatograph, incorporating it into the laboratory data network system and collecting data for Michael Starlings' project.

The automation involved the construction of hardware for the control of the chromatograph and for data acquisition and display and the writing of software to operate the interface. The LSI-11 collects data by clock interrupts while displaying the current data buffer on the oscilloscope. It stores all data files on a floppy disk storage device, accessed through the main host computer. An analysis of the data is performed by a peak processing routine, calculating peak area and retention time.

The network is the DEC RT-11/REMOTE system. It operates on a PDP-11-03 with a floppy disk system for mass storage. The laboratory system has three satellites operating under it. They can utilize all of the facilities of the main computer while maintaining a minimal operational configuration.

The project also had the purpose of being assimulated into Michael Starlings' data correlation system. The LSI-11 will collect data and send it to the NOVA computer system where it will be stored.

The system was tested with several mixtures and the results were found to be accurate and easily obtainable.