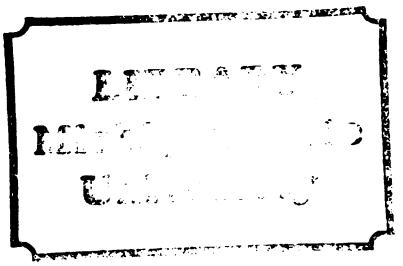
THESIS



This is to certify that the

thesis entitled

A MICROCOMPUTER-CONTROLLED BOXCAR INTEGRATOR
FOR LASER SPECTROSCOPY EXPERIMENTS

presented by

John David Stanley

has been accepted towards fulfillment
of the requirements for

M.S. degree in Chemistry

G. A. Crosby

Major professor

Date May 20, 1982

O-7639

A MICROCOMPUTER-CONTROLLED BOXCAR INTEGRATOR FOR LASER SPECTROSCOPY EXPERIMENTS

By

John David Stanley

A THESIS

Submitted to
Michigan State University
in partial fulfillment of the requirements
for the degree of

MASTER OF SCIENCE

College of Natural Science

1982

ABSTRACT

A MICROCOMPUTER-CONTROLLED BOXCAR INTEGRATOR FOR LASER SPECTROSCOPY EXPERIMENTS

By

John David Stanley

The large amount of data presented to the modern experimenter, if manipulated by hand, requires a great amount of time to process. Time to perform more experiments is lost while past results are analyzed. The limited control functions on some data collection equipment also prevent the efficient use of an experimental system.

One such system is a laser enhanced ionization (LEI) or dual laser ionization (DLI) spectroscopic system that uses a boxcar integrator for data collection. The construction of a microcomputer interface to the boxcar integrator, and the software written to control the boxcar integrator is presented. A study was performed to determine the noise reduction capability of the computer controlled boxcar integrator, and to verify the proper operation of the LEI/DLI data collection electronics.

TABLE OF CONTENTS

Chapter Page

Chapter Page

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

In the past few years, a new analytical technique called Dual Laser Ionization (DLI) Spectroscopy has been developed in this laboratory. The DLI technique involves the measurement of both the relative amplitude and the time decay characteristics of laser-induced ionization signals in a flame. These signals are present at low repetition rates (~20Hz) and for short periods of time (10 ns to 100 μs), and are superimposed on a noisy DC background. The ionization signal also has some radio frequency noise produced by the nitrogen laser present at the beginning of each peak.

The types of signals encountered in the DLI technique preclude the use of simple DC integrators and amplifiers for measuring the desired quantities. Previous workers [1] have lessened these measurement problems by the use of a boxcar integrator, which provides many of the functions necessary to measure brief, noisy signals. The boxcar integrator integrates the input signal for a brief period of time after the application of a trigger signal. The delay between the

trigger signal and the beginning of the integration period can be varied. The time period integrated is called the aperture time, and the time between the trigger and the beginning of the aperture is called the aperture delay. If a noisy, repetitive signal is observed with a boxcar integrator, and the aperture delay is held constant, the boxcar integrator will serve to average the signal from many repetitions into one value, thereby reducing the amount of noise measured. If the aperture delay is increased at a rate that is slower than the repetition rate of the observed signal, the boxcar integrator will sample progressively later and later portions of the input signal, and plotting the output of the boxcar integrator with respect to time will produce a time resolved picture of the input signal. The boxcar integrator, however, generates a data reduction problem. The output recording device from a boxcar integrator is usually a stripchart recorder. The amount of time consumed in averaging multiple passes of time resolved data, or noisy non-time resolved data limits the amount of time available to the experimenter, and is a psychological hurdle to long experiments.

A second problem, or rather, challenge, is presented by the fact that for this application, more fuctional variables need to be controlled than are available from the boxcar. For example, using the front panel controls, the boxcar

aperture can be scanned linearly across the aperture delay range. The exponential nature of the decay curves means that the early part of the signal should be sampled with higher resolution than the later regions of lower slope. Fortunately, the boxcar being used has most of the control lines brought out to the back panel, so that the user can generate control signals appropriate to the experimental requirements.

The work presented in this thesis represents the author's attempts to solve these two problems by interfacing a microprocessor to the boxcar integrator. This microprocessor controls the operation of the boxcar integrator, and collects the data generated. The data are then sent to an LSI 11/23 minicomputer for manipulation and storage.

This work consists of five major parts: the historical aspects, a discussion of the instrument, the software developed, an evaluation of the system with well-behaved test signals, and a presentation of data from the DLI spectroscopic system.

CHAPTER 2

HISTORICAL

Introduction

With the advent of LSI integrated circuits, the ease and practicality of interfacing a computer to analog data collection circuitry has greatly increased. Prior to that time the cost of minicomputers often prohibited their dedication to a single instrument for the length of time required for data collection and the control of experimental parameters. Early data collection schemes generally required the entry of data to the computer by hand; however, systems have been devised to circumvent this. For example, one early system[2] converted the binary coded decimal output of a Keithly digital multimeter to the ASCII codes required to run a Teletype paper tape punch. This paper tape could then be loaded into any computer that had a Teletype terminal. As late as 1980, a system that used a Tektronix 4051 programmable calculator and an X-Y strip chart recorder with an image sensor in place of the pen was

developed to convert an analog chart recording into the form required by a digital computer automatically[3].

Laboratory Microcomputers

The Digital Equipment Corporation (DEC) PDP8 series of minicomputers has been a popular choice for laboratory computer applications. This series has a large base of software support (OS/8 operating system, FORTRAN, BASIC, etc.), but because of the cost of memory (both core and mass storage) that is needed to run these higher level operating systems and languages, the PDP8 is still a larger system than desired for dedication to a single, small laboratory instrument.

The Intersil IM6100 integrated circuit provided the answer to the cost versus software problem for PDP8 users. The IM6100 is a one chip implementation of the PDP8 processor; it made dedicated control of experimental equipment practical. This chip was easily adapted for use in a master/slave processor system. The IM6100 processor, control electronics, and just enough memory (either read-only or random access) for the desired application are connected to the experiment. A PDP8 which can run OS/8, FORTRAN, etc. could function as the master minicomputer, and could be used to compile programs to run on the 6100

system, which would recieve these programs over a serial communication line. The same serial line would then be used to send data back to the PDP8 for storage. This technique has been successfully used in this laboratory[4,5].

The PDP8 and IM6100 suffer from two problems: 1) a weak instruction set, and 2) complicated interfacing requirements. Advances in LSI technology and microprocessor architecture have made available processors with the speed and power of larger machines at a cost that permits system dedication. The interfacing electronics have also been simplified by the production of integrated circuits designed to interface to specific microprocessor bus systems.

The Intel 8080, an 8 bit, one-chip microprocessor, is such a processor, and it has been used in such a dedicated system[6,7] with a cross assembler and a downloader running on a PDP8. Further developments at both Intel and DEC have led to more powerful microcomputers and minicomputers for less cost. The Intel 8085 is the offspring of the 8080, and is the microprocessor used in this work. The 8085 is also an 8-bit processor, but is faster than the 8080, and requires only a +5 volt power supply, while the 8080 requires +5, +12, and -5 volt supplies. The 8085 also has four interrupt and serial input and output lines not present on the 8080.

Other workers [8,9] at this university have developed a relatively inexpensive, twin bus system based on this microprocessor and the support chips designed by Intel. The master minicomputer is a DEC LSI-11/23 with the RSX-11M multi-user, multi-tasking operating system. An 8085 cross assembler which uses the extremely powerful Macro-11 assembler available on the LSI-11 has been developed by Steve Johnson and Hugh Gregg. This sytem will be described in more detail in the next chapter.

Boxcar Integrators

Boxcar averagers and boxcar integrators have become a familiar sight in many laboratories in which noisy repetitive signals must be measured[10]. The cost of these devices has decreased, as should be expected, with the increase in the large scale integration (LSI) of the modules needed in a boxcar integrator. High speed and high resolution devices are still not inexpensive, and may not be compatible with modern microcomputers.

The interfacing problems can be avoided by designing one's own boxcar integrator. One such device has been developed for use in a transient detector, which has nanosecond resolution[11]. This design used emitter coupled logic for the high speed section of the digital logic, which

for lower resolution (~μs) boxcar integrators is not necessary.

With the increasing use of microcomputers in the laboratory, boxcar integrator manufacturers have begun to provide user access to what used to be internal signals. The boxcar integrator used in this work has most of the control voltages available on a rear panel connector. Microcomputers themselves have begun to take over the functions of the control logic in the boxcar integrator. The AIM-65, a low cost microcomputer based on the 6502 processor, is able to function as a boxcar integrator, with the addition of a sample-and-hold and an ADC[12].

CHAPTER 3

Instrumentation

Introduction

The electronics in this instrumentation project consists of two parts: the boxcar integrator and the microcomputer. The time scale of the measurements prevents the use of a microcomputer and an ADC alone as a software boxcar integrator. The flexibility desired requires use of more than a boxcar integrator alone. Together, the two allow measurements of fast signals with a large degree of flexibility. The boxcar integrator is discussed first, followed by a brief description of the microcomputer. A block diagram of the instrumentation is in Figure 3-1.

Boxcar Integrator

The boxcar integrator used is a Princeton Applied Research Model 162 boxcar averager mainframe with a Model 163 sampling integrator, and a Model 164 gated integrator

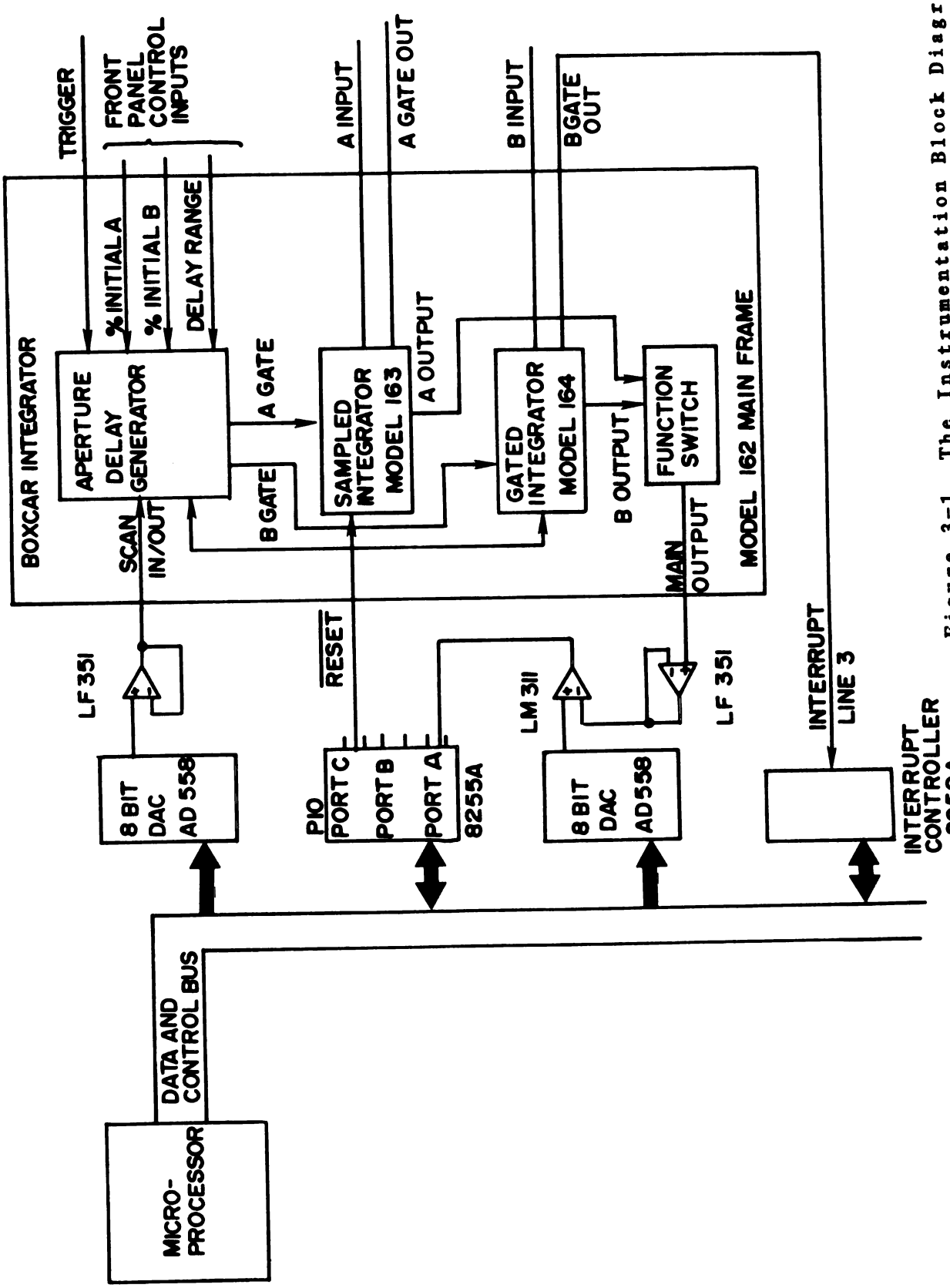


Figure 3-1. The Instrumentation Block Diagram

module[13]. The Model 164 allows greater flexibility in the aperture duration selection than the Model 163, and is the module with which the data are acquired.

Model 164

The Model 164 gated integrator module is designed so that upon each trigger signal generated by the mainframe, an integrator inside the 164 integrates the input signal for a time determined by the setting of the aperture duration switch on the mainframe front panel. The aperture duration can be set from 5 ns to 500 μs. Two different modes of operation may be selected for the integration. In the summation mode, the input signal samples are integrated linearly, so that the output signal is the simple sum of all the input samples.

In the exponential mode, the difference between the input signal and the output signal is integrated. In this mode, the output exponentially approaches the value of the input, and after 5 time constants of the integrator, will be essentially the same as the input. The value for the time constant is switch-selectable from the front panel of the Model 164. This is the mode normally used, as the output value is independent of the number of samples taken, within

the time constant constraint.

The output from the Model 164 goes to the function switch of the mainframe (at lower right in Figure 3-1). The function switch selects the output of either channel A, channel B, the difference between the A and B outputs, or one of several options as the Model 162 mainframe output. In this system, the Model 164 is installed in channel B of the boxcar integrator.

Mainframe Model 162

The mainframe of the boxcar integrator provides two functions. It determines the time delay for the aperture, and provides signal conditioning for the output of the Model 164.

The aperture delay is the time between the trigger signal to the boxcar integrator, and the opening of the aperture in the gated integrator. This delay time is determined by 3 factors. The aperture delay range determines the largest delay available, and is set by a Model 162 front panel switch. The choices for delay range vary from 0.1 μs to 50 ms. The % Initial A and % Initial B controls are 10 turn precision potentiometers located on the

front panel of the Model 162, and set the lowest value the aperture delay can have, in terms of the aperture delay range.

The final parameter in the determination of aperture delay is the scan signal. When using the internal scan function of the boxcar integrator, a voltage ramp is generated, which slowly sweeps the aperture delay from the limit set by the % Initial control, to the limit set by the aperture delay range control. However, when placed in the external scan mode, the ramp signal can be externally supplied via back panel connectors. It is this input signal that allows the computer to control the aperture delay time. The method by which the microcomputer does this is discussed later.

The three inputs work in the following way. The aperture delay range switch determines the amount of time from the application of the trigger signal it will take for an internal ramp signal to go from 0 to full scale. The time for this to occur is equal to the aperture delay range. This signal is sent to three comparators. The first comparator determines when the delay ramp has reached full scale, and resets the ramp generation circuitry.

The other two comparators generate the gate signals for

the Models 163 and 164. The second input to these comparators receives the sum of the % Initial control for the appropriate channel and the scan voltage. In the internal scan mode, this signal is a slow ramp. When the delay ramp passes the summed scan ramp, the gate signal for the channel is generated.

Mathematically, the aperture delay is a function of three variables: the aperture delay range, R, the % Initial A or B represented by %I, and the voltage from the internal scan circuit or from the rear panel V. The equation relating aperture delay D to these three variables is:

$$D = R(\%I + 10V)/100 \qquad (3\text{-}1)$$

where V is the magnitude of the input scan voltage, and D has the same units as R. The contribution, S, to the aperture delay from the scan signal is:

$$S = RV/10 \qquad (3\text{-}2)$$

The second function of the mainframe is to provide various combinations of the channel A and B outputs. As only channel B is used in this work, the function selector for the mainframe output is set to channel B out. The mainframe also contains a filter on the output to help reduce the noise components of the signal. The time constant for this filter is switch selectable from the front panel, and can vary from 0.1 ms to 10 s.

The Microcomputer

The microcomputer used was designed by Bruce Newcome, and is described in detail elsewhere[8]. The processor used is the Intel 8085A, an 8-bit data, 16-bit address device. This processor runs at 3 MHz, and has a maximum address space of 65,534 bytes.

The 8085 is mounted on the CPU board, which contains the electronics to buffer the address, data and control lines as they go off the CPU board. The CPU and all other boards are mounted on a motherboard, which has the sole function of carrying the microprocessor bus from module to module. The main motherboard is connected to other motherboards in the system by a backplane, which carries the processor bus between motherboards.

The entire system, called the Multi-Micro System, is modular in design. Instead of having each peripheral device decode it's own address, one module performs this function. This module, the Chip Select (CS) board, generates 8 non-qualified selects, and 8 qualified selects. The qualified selects are only active during a processor read or write operation, while the non-qualified selects can be activated by random states of the address bus.

Terminal I/O, and I/O to the minicomputer is controlled by a USART module which contains 2 Intel 8251A Universal Synchronous Asynchronous Receiver Transmitters. The system interrupts are controlled by an Intel 8259 Programmable Interrupt Controller (PIC).

Memory is provided by two modules, a 16k RAM/ROM board, and an 8k RAM board. The RAM/ROM board in the system used in this work, contains 4k of ROM which stores the operating system SLOPS, and 10k of RAM, which contains the software to control data collection. The operating system and software are discussed in detail later.

One final standard module is used, the Intel 8255 Programmable Peripheral Interface (PPI), also referred to as a Parallel I/O (PIO) module. This module allows for 3 parallel 8-bit input or output ports. The configuration of

this device is discussed later.

Interface

The interface between the boxcar integrator and the microcomputer consists mostly of buffering to protect the microcomputer from being damaged. One additional module for the microcomputer system has been designed, and one external circuit board contains the buffers. This board also contains the comparator used in the analog-to-digital converter (ADC).

Dual Digital-to-Analog Converter

The only function missing in the basic Multi-Micro system was a means of converting the analog output of the boxcar integrator to the digital signals required by the microcomputer, and vice versa. This function was provided by a module which contains two Analog Devices AD558 digital-to-analog converters (DACs). The AD558 is a 16 pin IC which has an internal voltage reference, and can provide either 10 or 2.56 volts full scale output. In order to set an analog output voltage, all that is necessary is to write the digital value to the address of one of the CS selects.

This signal is connected to a chip select on the AD558 which latches the data into the converter.

The output of one of the two DACs is used as an input to the external scan input on the boxcar integrator. The output voltage on the 10 volt scale from the DAC is 10B/256 where B is the 8-bit data value. It follows from equation 3-2 that the aperture delay due to the external scan input is:

$$S = RB/256 \qquad (3-3)$$

where R and S have the same meaning as in equation 3-2.

The other DAC is used with an LM311 comparator to function as a successive approximation ADC, under software control. This software ADC was chosen for use because a general purpose Analog-to-Digital Converter (ADC) module had not been designed for the multi-microsystem. However, a 12-bit ADC module was designed recently[14]. Future plans include the use of this module for data acquisition.

Digital Interface

The remaining interfacing is accomplished using digital signals. The boxcar integrator provides a copy of the aperture gate signal fed to the Model 164, and a reset function is available from a 50 conductor edge connector on the rear panel. The rear edge connector also makes the analog signals available, and the one external circuit board connects to the back of the boxcar integrator, with power supplied by the boxcar integrator.

The aperture gate signal from the Model 164 is used to trigger an interrupt in the microcomputer. The gate out is connected to the clock input of a J/K flip-flop on the interrupt controller board, which latches the trigger pulse. The output of the flip-flop is sent into the interrupt 3 line of the 8259 PIC chip. One of the qualified CS selects is used to reset the flip-flop at the end of the interrupt.

The boxcar integrator can be reset by taking an external reset line LO. This function is provided by bit 7 of port C on the PIO module. Port A, bit 0 is used to examine the output of the LM311 comparator.

LSI-11/23 Minicomputer

The computer used for storage and data manipulation is a Digital Equipment Corporation (DEC) LSI-11/23 minicomputer, with 128K words of RAM, and 2 DEC DLV-11J boards, each of which provides 4 serial data lines. Thirty seven million bytes of hard disk storage are available, as are dual density floppy disk drives. Printed listings and graphics are generated on an Integral Data Systems Paper Tiger 560-G.

The operating system used on the minicomputer is RSX-11M, supplied by DEC. This system has Fortran and Macro programming languages available for data manipulation software.

The hardware connection between the mini and microcomputers consists of a 4 wire cable connected to a serial line on the LSI-11. RSX-11 treats this line as if it were a terminal, and programs have been written which accept data coming in this line for storage on the hard disk. The other end of this cable is connected to the second USART on the microcomputer's USART board. A routine in the microcomputer connects the user's terminal to the link to the LSI-11 through software. This transparent mode allows one terminal to be used for both the LSI-11 and the

microcomputer.

CHAPTER 4

Software

Introduction

Because much of the instrumentation was purchased commercially, a major effort was in the development of software. There are three major sections to the software. The first is the section written to run on the microcomputer, and the second is the part written for the LSI-11 minicomputer. The final section is that written for communications between the two computers. The latter portion of code was developed primarilly by other workers[15], and will not be described extensively here.

The development of the software was guided by one major philosophy: the microcomputer should be used for instrument control, and any mathematical operations on data should be done on the larger machine. This leads to lower equipment cost, since only one mass storage device is needed within the laboratory, and overall faster data turnaround. The

availability of higher level languages and plotting packages is also greater on the LSI-11 than on the microcomputer.

Microcomputer software

The microcomputer has 5 functions to perform:

1. Allow the operator to adjust experimental parameters
2. Control the boxcar integrator
3. Collect data from the boxcar integrator
4. Store the data temporarily
5. Send the data to the LSI-11

Prior workers[15] have developed the operating system used on the microcomputer, which is called SLOPS. SLOPS is a very simple system, stored in ROM, which allows an operator to enter commands from the terminal keyboard. SLOPS then searches a linked list of names to find the command. When a match is found the machine code immediately following the matched command is executed. The code to accomplish the desired function is written in assembly language, which is cross assembled on the LSI-11, and downloaded into the microcomputer memory. The operator interface (no. 1 above) is the largest section of code and

will be discussed first.

PARAM

The portion of the software that allows the user to control the parameters for data collection is called PARAM; it is executed by typing PARAM from the terminal keyboard. Once started, PARAM displays the current operating parameters, and waits for user action. This action consists of one of several commands, most of which are the names of the parameters that can be modified. Those parameters are:

IPDPT - the number of laser flashes, and therefore interrupts, between data point collection

RESADP - should the boxcar integrator be reset between data points

DBUFF - the beginning of the data buffer

BUFFE - the end of the buffer

Those parameters dealing with time resolution are:

STSCAN - start of scan

LSCAN - length of scan

SCANINC- the increment of change during the scan

PPSCNC - the number of data points to be taken between scan steps

The use of these parameters is discussed later along with the programs in which they are employed. The remaining commands do not deal with parameters. They are: DISPLAY, EXIT, and PB.

The commands do not need to be entered in their entirety. Once enough of the command has been entered so that the program can unambiguously determine what is desired, the remainder of the command is displayed on the terminal. For example, to change IPDPT, only the letter I need be typed. A carriage return (CR) is then entered, and the command is echoed back on the screen to ascertain that the proper command was entered. The operator may then enter a number, which will replace the old entry. If no number is entered, the old entry is not changed.

Since values in PARAM are used to control the collection of data, it is important that once data are being collected the parameters not be changed accidentally. Included in the table in which all the parameters are stored is a software lock, which is set by the data collection routine as soon as one data point is taken. This lock prevents the operator from changing any parameters. Commands to change parameters are rejected until the operator unlocks the table with the UNLOCK command.

A secondary function of the PARAM program is the recording of the operating parameters of the boxcar integrator. This function is activated by using the PB command. The entries in this table are a description of the settings of the boxcar integrator front panel. There are locations for the entry of time constants, %A INITIAL, and other settings. There are also two lines for the entry of textual comments. The entries in this table are accessed by entering the line number of the desired line. Entering an 11 returns the user to the main PARAM program.

The entire parameter table is stored immediately in front of the data buffer, and is sent with the data to the LSI-11 for storage. This provides a permanent record of the experimental parameters with the data.

After the operator has set the parameters to the desired values, PARAM is exited with the EXIT command, or the values are listed with the DISPLAY command.

TAKE

This is the second major portion of the microcomputer software. TAKE is responsible for actually taking the data, and controlling the boxcar integrator. When the operator types TAKE in response to the SLOPS prompt, the following events occur:

1. BUFFE (the pointer to the end of the data buffer)
   is set equal to DBUFF.
2. The interrupt counter is reset to 0.
3. The parallel input/output (PIO) port is initialized.
4. Interrupt line 3 is activated on the interrupt controller IC.

A flowchart of the operation of TAKE is shown in Figure 4-1. Control of the processor is then returned to SLOPS. The remainder of the data collection operation is under interrupt control. A flowchart of the interrupt handler

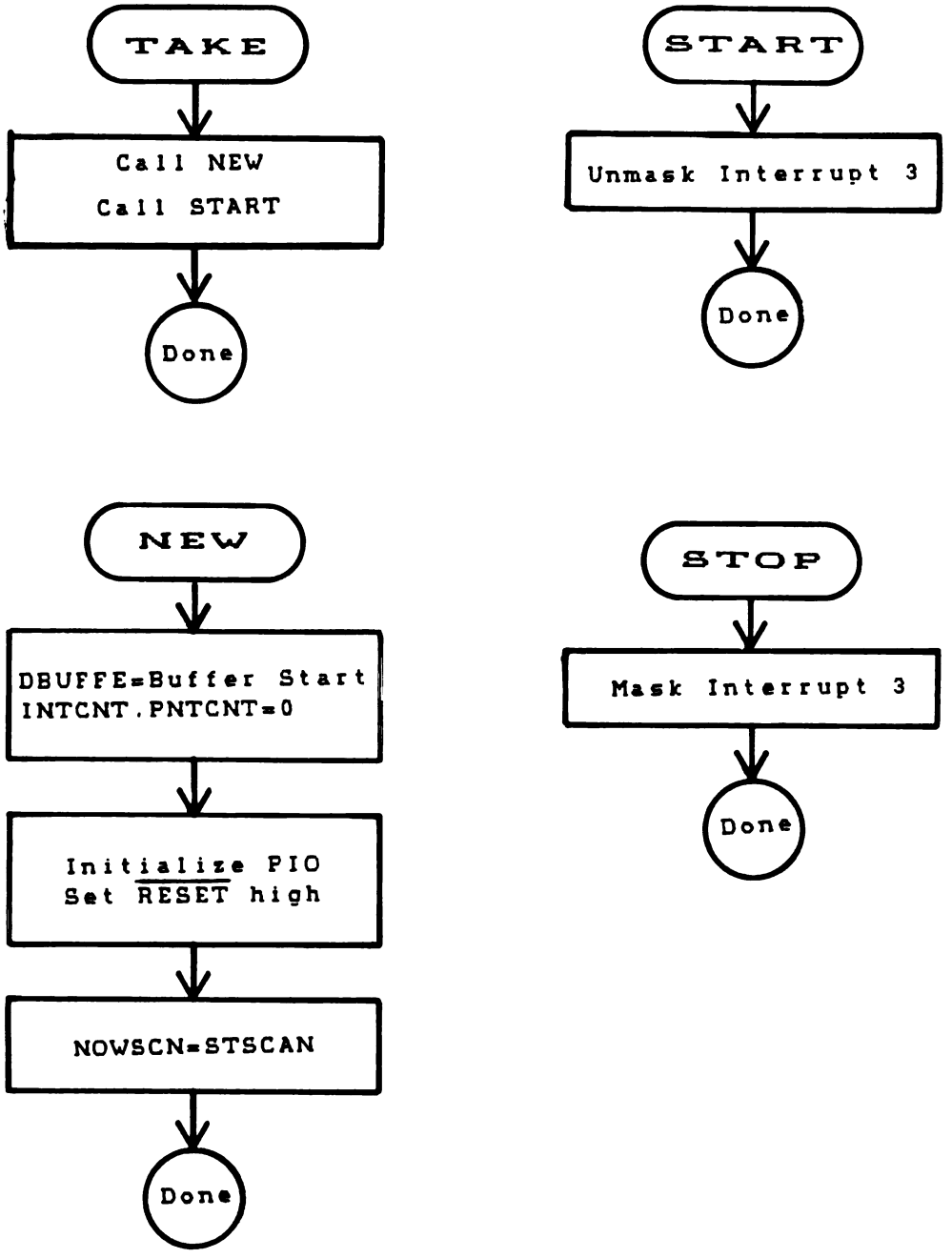


Figure 4-1. TAKE, STOP, NEW, and START

operation is in Figure 4-2.

As the laser fires, it sends a trigger pulse to the boxcar integrator, which in turn, generates a gate signal for the sampling head. This gate signal is sent to the microcomputer to act as an interrupt. This interrupt causes control of the processor to pass to the data collection routine through a CALL to an entry in the interrupt table, which contains a JUMP to the beginning of the interrupt handling routine. This routine is the one that uses the parameters entered during PARAM to control the operation of the boxcar integrator.

The interrupt handler first saves all the processor's registers so that at the end of the interrupt, the processor can return to what it was doing earlier. The status of the reset line (C7STAT) is examined first, to determine the state of the reset line. If the C7STAT contains the number 16, the boxcar integrator is being reset by the user, and the interrupt handler will not attempt to collect data. If C7STAT if any other non-zero value, the boxcar integrator is being reset by virtue of RESADP after collecting a data point. The value of C7STAT is then decremented by one, and if it becomes zero, the reset signal is removed from the boxcar integrator, and the handler exits. This use of a count in C7STAT is necessary to provide the electronics

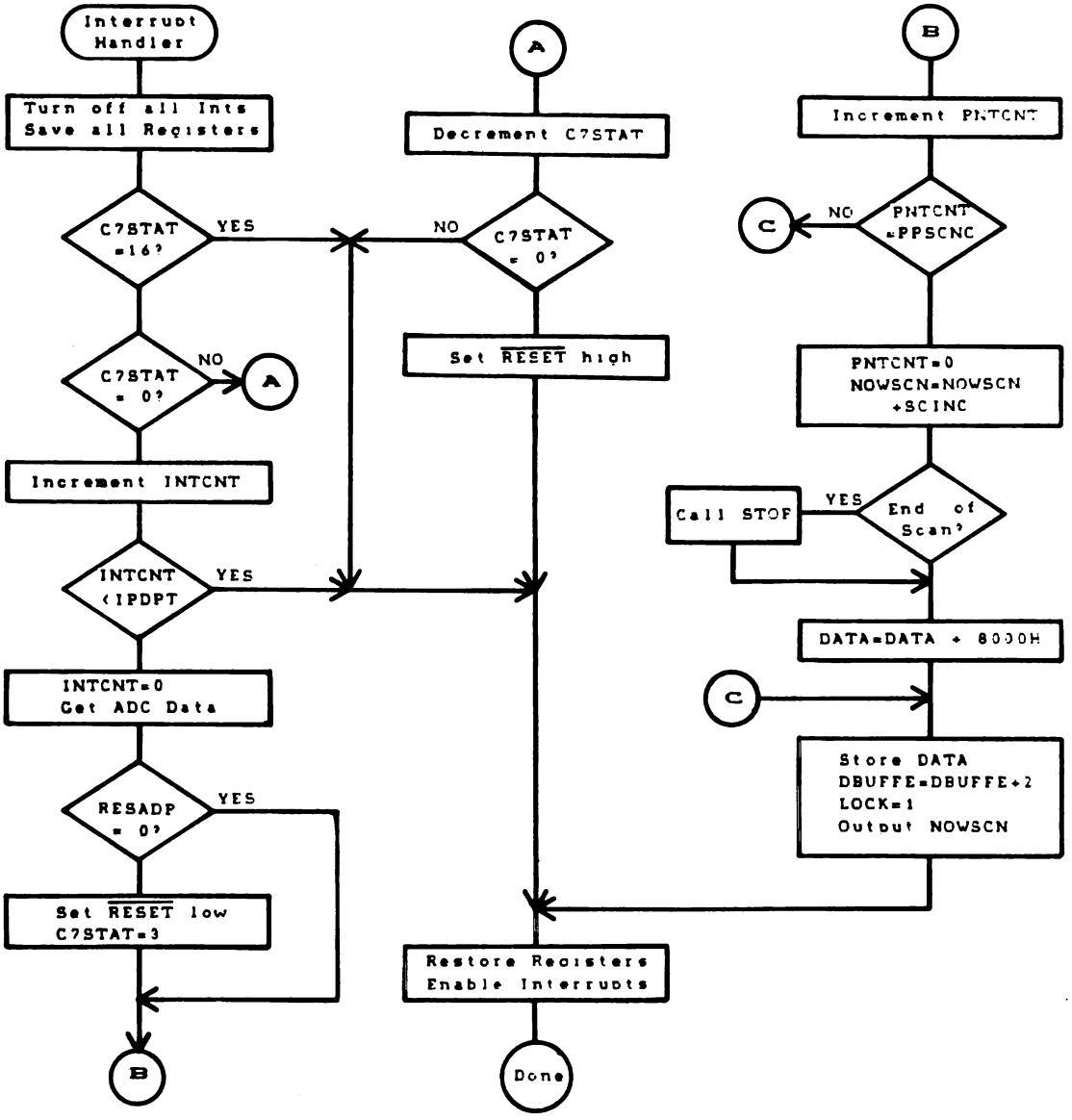


Figure 4-2. The Interrupt Handler

within the boxcar integrator enough time to properly reset.

If C7STAT is currently zero, the boxcar integrator is collecting data. The interrupt counter, which counts the number of interrupts since the last data point was taken, is incremented, and compared to IPDPT, the desired number of interrupts per data point. IPDPT can vary from 1 to 255. If IPDPT equals zero, then there will be 256 interrupts per data point. If the two numbers do not match, the interrupt handler replaces the processor registers, and exits.

If the proper number of interrupts has occured, the interrupt handler gathers a data point. It first clears the interrupt counter, and blinks a light so the operator can tell that the routine is working. A subroutine is called to convert the present analog output from the boxcar integrator into an appropriate digital value. The conversion is accomplished using one 8-bit digital-to-analog converter (DAC) and a comparator. A serial approximation conversion is accomplished in software, and the output of the comparator is sensed using one bit of one input port from the PIO. This conversion takes approximately 400 microseconds using non-optimized assembly language. Even at the fastest repetition rate of the laser (~60 Hz), this speed is sufficient to prevent data overrun.

The data point returned is only 8 bits long, but it is treated as if it were 12 bits; this leaves 4 empty bits in the 16 bit processor word for use as flag bits. Only one flag bit is used presently, and it is discussed later.

After the data point is collected, RESADP is checked. If it's value is zero, no action is taken. If it is any non-zero value, the active LO reset line to the boxcar integrator is set LO, using one bit of one port of the PIO. To insure that this signal is held LO long enough to reset the boxcar integrator, a count of three is placed in C7STAT.

Next, the data point counter is incremented, and compared to the value PPSCNC, the desired number of data points for each scan increment. This value can take the same range as IPDPT. If the two values are the same, then bit 14 of the data point is set HI, which indicates to the software on the LSI-11 that a time scan increment will occur on the next point. When this occurs, the data point counter is also reset to zero. The current scan location is incremented by SCANINC, the amount to increment the scan after PPSCNC is reached. The value of SCANINC is in the range from 0 to 255. An overflow of the current scan location causes a call to STOP. This subroutine performs the same function as the STOP command, which is discussed later.

When the acquired data point is ready to be stored in the data buffer, the value is written to the memory location pointed to by DBUFFE, the end of the data buffer pointer. As a safety precaution, to detect the end of RAM memory, the data point is read back, and compared to the value still in the processor's registers. If the two do not match, usually due to memory overflow, the subroutine STOP is called.

The interrupt handler must now check for the end of a time resolved scan. The value STSCAN, the beginning value of the scan position, is added to LSCAN, the desired length of scan. If the sum of these two overflows 8 bits, the resolution of the scan DAC, STOP is called. The current scan location is compared to the sum, and if the current location is greater, the end of a scan has been reached, and STOP is called.

As data are being taken, the lock on the parameter table is set to prevent accidental parameter changes. At this point the interrupt handler has finished its task. The registers saved at the beginning are restored, after sending the end-of-interrupt (EOI) command to the interrupt controller to enable further interrupts. The interrupt handler also clears a J/K flip flop used as a latch on the interrupt input. The interrupt handler then executes a RETURN instruction, which returns the processor to the task

it was performing before it was interrupted.

Because the data routine is interrupt controlled, SLOPS is available to perform other tasks while data are being taken. One such task is called by the command STOP. The routine STOP masks, or disables, the interrupt input which coincides with the input from the boxcar integrator. Since the computer can no longer 'see' the interrupt, no more data are taken. None of the counters or parameters are affected by STOP, so that data collection can be resumed by re-enabling the interrupt line. This function is performed by the command START. One additional command is available, called NEW, which does the same things TAKE does, with the exception that interrupt 3 is not enabled. TAKE, in reality, performs NEW followed by START.

It is important to discuss the use of the time resolved scan parameters. They were designed originally to control the time function of the boxcar integrator. They are quite useful, though, when no time resolution is desired. For example, if STSCAN is 0, LSCAN is 0, and SCANINC is 100., the computer will take only PPSCNC data points. Once that number of points is taken, the scan location is incremented past STSCAN + LSCAN, stopping the data collection. This is quite useful when one wishes to obtain 50 or 100 points for each of several concentrations of analyte ion, and then

later average all 50 or 100 into one point. Software on the LSI-11 to be covered later automatically averages these points, and calculates their standard deviation.

DUMP

DUMP is a command which causes the current contents of the data buffer to be displayed on the screen. This allows the operator to see if erroneous data (for example, underflow or overflow of the ADC) are being collected, and to perform immediate corrections.

SAVE

Once the data have been collected, they are sent to the LSI-11 for storage and reduction. This function is performed by the routine SAVE. The greater portion of SAVE, called UPLOAD, was written by P. Hoffman[16]. UPLOAD was not originally written to run under SLOPS, and has thus been modified for use in this work.

SAVE passes the begining address of the information to be saved, and the end address to UPLOAD. UPLOAD then asks

for a file name under which to save the information on the LSI-11. UPLOAD causes RSX-11, the operating system in use on the LSI-11, to execute a task called UPCHUCK, which accepts the information sent up the serial line and stores it in a file on the LSI-11's disk.

These routines are the major programs written to handle the boxcar integrator. Several debugging and status aids were also written. Since they are not needed in normal operation, they are not discussed here.

DOWNLD

Because the microcomputer has no mass storage devices and the data collection routines are not in ROM, there must exist some method for loading these programs from some other source. This is done by the routine, stored in the ROM operating system, called DOWNLD. A program on the LSI-11, called DOWNLOAD, sends programs down the serial line to the microcomputer, where DOWNLD places them in the correct memory locations. The programs that are sent to the microcomputer have been written on the LSI-11 in assembly language, and assembled by a cross-assembler, which has been written to use the extremely powerful MACRO-11 assembler and a set of macros defining the 8085 assembler nmemonics. A

screen oriented text editor, VTECO, is also used during program development

LSI-11/23 Software

The LSI-11 is the logical choice for any data manipulation that needs to be done because of its mass storage, its high speed, and the availability of higher level languages. A high level plotting package, MULPLOT[17], makes presentation of the data in graphical form easy.

UPCHUCK

The data sent to the LSI-11 by the micro are received by the program UPCHUCK. This program takes the data, and stores them in a disk file in unformatted binary. That is, each data byte is stored exactly as it looks coming up the serial line. The parameter list is sent as the first 512 bytes, with the actual data making up the remaining file. Each data point is stored as 16 bits, and is the Y value only. The X values are determined later, by RAW2COOK.

File names on the LSI-11 consist of two parts; a 9 letter file name, and a three letter extension. Because of the way the microcomputer software was written, the extension of each of the raw data files is blank.

From this point, two different programs are used to operate on the raw data. The first separates the parameters from the data, and stores them in a file with the extension .DOC. It also assigns an X value to each point. The second program takes the output from the first, does a small amount of statistical treatment of the data, and allows several separate data files to be consolidated into one file.

RAW2COOK

RAW2COOK operates on the raw data file, and creates two output files. One file is the parameter table formatted into a text file. This file has the extension .DOC. The DOC file also has the name of the raw data file saved in it. After processing the data, the minimum and maximum X and Y values are also stored in the .DOC file. A sample DOC file is shown in Figure 4-3. The second file created contains the data, properly formatted for MULPLOT use. A listing of RAW2COOK is in Appendix A.

```
PARENT RAW DATA FILE IS:  8T50KDCE
BUFFER START:  16896  END:  17716
INTERRUPTS PER DATA POINT:     10
RESET AFTER DATA POINT:        NO
START OF SCAN:                  0
LENGTH OF SCAN:               200
SCAN INCREMENT:                 5
POINTS PER SCAN INCREMENT:     10

     >>> Boxcar parameters <<<
  Aperture delay section
  1) %Initial A:
  2) %Initial B: 20%
  3) Range: 10 microseconds
  Aperture section
  4) Duration: 0.5 microseconds
  5) Scan time:
  Time constants section
  6) Boxcar: 0.1 second
  7) Model 164: 10 microseconds
  8) Comments: Time resolved Na New ckt
  9) 50k load DC coupled 1 M Exponential

DATA FILE WRITTEN: 8T50KDCE.DAT
 XMIN:      2.0000  XMAX:      9.8120
 YMIN:      0.0000  YMAX:      8.2812
SD  DATA FILE  8T50KDCE.DAT STDDEV INTO FILE
8T50KDCE.ICE
```

Figure 4-3. A Typical DOC File

When RAW2COOK executes, it asks for three file names: the raw data file, the DOC file, and the output data file. It then asks for the initial X, and X increment values. Now the X values are assigned to the data points. This allows the microcomputer to collect data with many different abcissa's, without the microcomputer having to worry about which was used to take the data. If the ionization signal from a 10 ppm Na solution were recorded, the initial X would be 10, and the X increment would be 0. If a wavelength scan of the Na D lines were done, X would be the starting wavelength and might be 588.5 nm. In the latter case, the X increment would be calculated from the dye laser scan rate. After writing the data to the DAT file, RAW2COOK displays the X and Y extremes, and writes them to the DOC file.

ICING

ICING is the program which takes one or more DAT files and creates from them a single file, with the extension ICE. It also averages all the consecutive data points with the same X value into one point, and optionally calculates the standard deviation of the Y value. When the standard deviation option is chosen, this information is written to the output file in such a way that MULPLOT automatically draws standard deviation bars on the plot that it generates.

When ICING is run, it asks only two questions. The first is for the name of the output data file. The second is whether the standard deviation option is desired. After these questions are answered, the program responds with the prompt 'Ice>'. The data file names which are to be combined to form the output file are then entered, one for each prompt. To end the program, a control-Z (FORTRAN end of file mark) is entered.

All files do not need to be entered at one time. The data written to the output file are appended to the file, so a user can end ICING, then add more data files at a later time. The standard deviation option can even be turned on for portions of the ICE file, and off for others. A listing of ICING is in Appendix B.

It is this software which provides the enormous flexibility in the microprocessor-boxcar integrator experimental system, over the boxcar integrator alone. The number of parameters that the user may vary increases the complexity of data collection, but by the proper selection of these parameters, many hours of manual data reduction may be eliminated.

CHAPTER 5

Boxcar Integrator Noise Reduction

Introduction

A critical test of any data collection system comes in using well behaved data (i.e. data for which the expected results can be predicted) to verify that the expected results are obtained. Because the signals which are observed in laser ionization experiments are pulses that decay exponentially with time, this type of signal was chosen to test the system. An integrated circuit noise generator was used to add random noise to this signal, and the system's ability to discriminate against this noise was determined.

Noise Reduction

One reason for using a boxcar integrator is the signal-to-noise enhancement capability of the integrator. The magnitude of the enhancement can be expressed as the

ratio of the signal-to-noise ratio (S/N) of the output to the S/N of the input. This ratio is called the signal to noise improvement ratio (SNIR). The SNIR for the entire system can be calculated by multiplying the SNIR's of each portion of the system together. This system can be broken down into three sections, for SNIR purposes.

The first SNIR is that due to the electronics in the Model 164 sampling integrator. The amount of improvement in this section is dependent on the integration mode selected on the Model 164. In linear integration mode, the SNIR is simply the square root of the number of samples integrated, since the output value is the linear sum of the input values. This assumes that the noise present is truly random, white noise. In the exponential mode, however, the SNIR does not depend upon the number of points sampled, within the constraint that the total time the aperture is open exceeds 5 times the input filter time constant. The SNIR in this mode is:

$$SNIR = SQRT (2TC/AD) \qquad (5-1)$$

where TC is the input time constant, and AD is the aperture duration[18]. For example, if the time constant is 10 μs,

and the aperture duration is 5 μs, the SNIR of this stage is 2.

The SNIR calculations are based upon the assumption that the noise present is white noise. A constant noise power spectrum allows the simplification of the SNIR equations by removing the boxcar integrator's frequency dependent amplitude transfer coefficient, and using instead the noise equivalent bandpass. When the noise source is not white, however, the variance of the output signal will be a convolution of the true noise power spectrum P(e) and the amplitude transfer function H(f). The equation for the variance is:

$$s_E^2 = \int P(e)\left|H(f)\right|^2 df \qquad (5-2)$$

If the noise power spectrum is constant with respect to frequency, it can be removed from within the integral. The equation then becomes:

$$s_E^2 = P(e) \int H(f)^2 df \qquad (5-3)$$

where the integral is defined as the noise equivalent bandpass (NEB) [19].

The NEB for a simple integrator is given by 1/(2t) where t is the total integration time. For exponential mode integration, the equation for an RC filter holds, and the NEB is given by 1/(4RC), where RC is the filter time constant.

The amplitude transfer function for the boxcar integrator is not readily available, and it's calculation is beyond the scope of this work. The exact quantization of the effect of non-white noise upon the expected SNIR is not possible; an intuitive qualitative treatment is presented later.

The second contribution to the total SNIR is from the output electronics of the Model 162 mainframe. The ratio at this point depends on the sampling rate of the Model 164 and the output time constant. A rough approximation for this is:

$$SNIR = SQRT (f_s / f_{co}) \qquad (5-4)$$

where f_s is the sampling frequency, and f_{co} is the cutoff frequency for the outpnt filter. The cutoff frequency of this filter is related to the time constant by:

$$f_{co} = 1 / (2 \pi TC) \qquad (5-5)$$

If the time constant is 0.1 second, and the sampling frequency is 20 Hz, the cutoff frequency is 1.6 Hz, and the SNIR would be 3.5.

The final contribution to total SNIR comes from the data averaging performed by the microcomputer and minicomputer. When N data points are averaged into one data point, the SNIR is the square root of N. When the system is controlled by the microcomputer, large numbers of data points can be collected, and the major contribution to the final SNIR will come from this data averaging.

The total SNIR will be limited by any noise introduced between the boxcar integrator outpnt and the microcomputer data averaging step. One source for noise at that point is the noise added by the analog-to-digital conversion process. The amount of this noise is determined by the resolution of the ADC, the value being the smallest resolvable voltage of

the ADC divided by the square root of 12 [20]. If the noise contributions from other portions of the system are smaller than this conversion noise, then this will be the limiting factor in increasing the S/N of the system. If the domain conversion is the primary noise source, an ADC of higher resolution can be used to lower the conversion noise.

Noise

Because the statistical treatment of noise depends on that noise being random and white (equal noise power density at al frequencies), it would be nice to have a simple random noise generator. This function is provided by the National MM5837 noise generator chip[21]. The MM5837 is a pseudo-random noise generator, which consists of a 17 bit shift register, and an exclusive-or gate to generate the input to the shift register from two of the bits of the register. As the shift register has only a finite number of states, the output of the 5837 must repeat. The number of states for the 5837 is so large, however, that the minimum repetition period is 1.1 seconds.

Experimental

The schematic for the source of the test signal is

shown in Figure 5-1. The exponential decay is taken from the charging curve of the timing capacitor for an NE555 timer circuit. By use of IC2, IC3, and IC6, this charging curve is converted to an exponential decay that varies from -10 millivolts to 0 millivolts.

The noise generated by the 5837 is first passed through C3 to remove any DC bias, then through a filter formed by IC5 to cut off extremely high frequencies. The amount of noise added to the signal is set by R7, and a variable DC offset is provided by R10. The square wave output of the NE555 is used as the trigger source for the boxcar integrator.

During the course of the experiment the aperture duration was set to 5 μs, the input time constant was 10 μs, and the output time constant was 0.1 second. Both A and B channels were set to external scan mode, and the scan range was 50 ms. The input circuit was set to 50 Ω impedance, DC coupled. The front panel %B delay control was adjusted so that the overlap light remained unlit. The software was set so that 30 repetitions of the input signal occured for each data point taken. One hundred data points were taken before the data were saved on the minicomputer. For the data taken in exponential mode, RESADP was set to zero. For the linear mode, the integrator had to be cleared after each data point

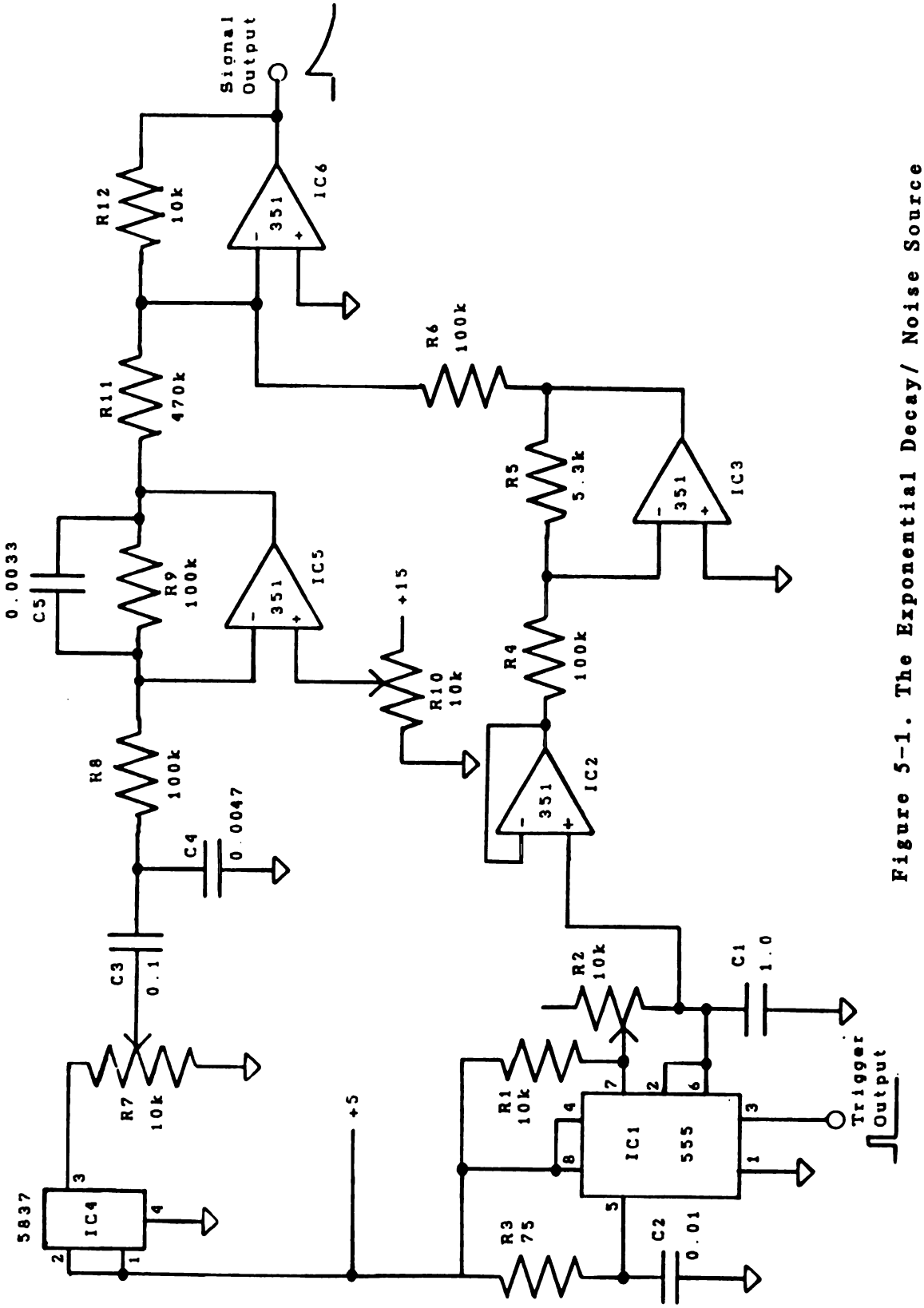


Figure 5-1. The Exponential Decay/ Noise Source

to prevent overflow, and RESADP was set to 1.

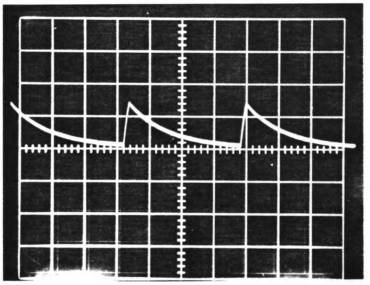
The SNIR of the system was determined for both modes of integration. In each case, 30 repetitions of the signal were integrated to generate one data point, and 100 data points were collected by the microcomputer. This process was repeated 5 times for each of three different measurements. One measurement delayed the aperture until the lowest point of the decay curve, which was used as a baseline value. The input signal was adjusted until this point was at zero volts. The second measurement was at the lowest aperture delay possible on the range that included one entire cycle of input signal. This value was used as the signal level in the S/N calculations. The final measurement was taken at the same aperture delay as the second, with the addition of approximately 50 mV p-p noise. This provided the noise value.

The entire decay curve was also observed, using the time resolved scan function of the boxcar integrator, controlled by the microcomputer.

Results

A photograph of the oscilloscope output of the exponential decay is in Figure 5-2a. Figure 5-2b is the

a)



b)

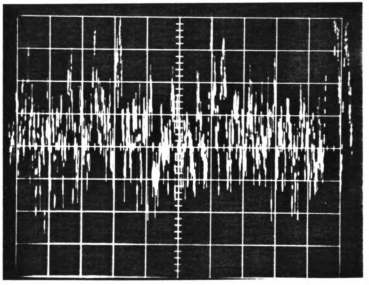


Figure 5-2. The Decay Signal and Noise

same signal, with the addition of the 50 mV p-p noise. As can be seen in Figure 5-2b, the noise almost obliterates the desired signal. Figure 5-3 is the recovered signal for the signal with and without noise.

The earliest available point on the decay curve was used as a test point to determine the system's SNIR. The boxcar integrator limited this point to 9% of the aperture delay range, or 4.5 ms. For the linear integration mode, the SNIR of the input circuit should be $\sqrt{30}$. The repetition rate of the signal was 28 Hz, but because the aperture delay range was longer than one cycle of the signal every other cycle was ignored, making the sampling frequency 14 Hz. The output filter SNIR is then $\sqrt{14/1.6}$. The SNIR due to signal averaging should be 500. The total SNIR should be $\sqrt{30} * \sqrt{14/1.6} * \sqrt{500}$, or 349.

To determine the true SNIR for the system, the following procedure was used. First, each set of 100 data points was averaged, to give an average value and the standard deviation for each point in the set. The five average values obtained were then averaged into one final value, and the standard deviation of each of the five points was calculated from the five points themselves. The standard deviation of the final average value was calculated by improving the standard deviation of the five points by

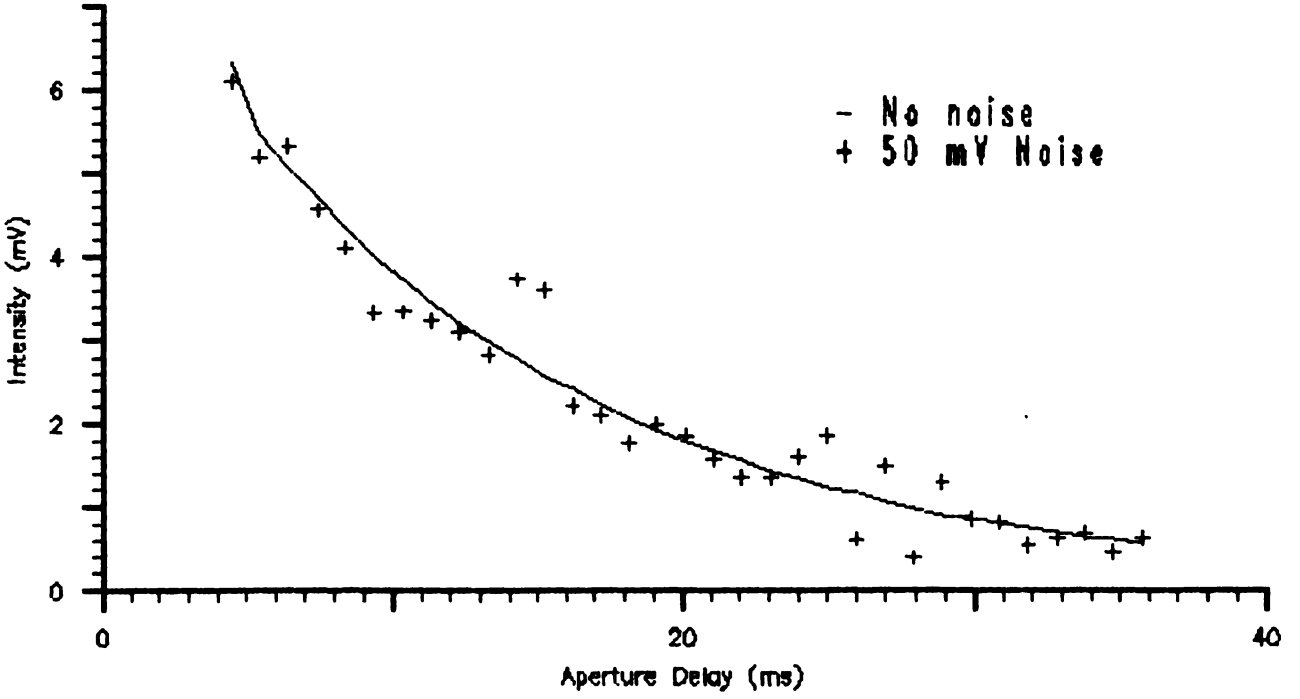


Figure 5–3. Observed Signal and Noise

$\sqrt{5}$.

To help verify the final standard deviation value, the final standard deviation was also calculated from the standard deviations of each of the original data points. This was done by improving the average standard deviation of each of the points by $\sqrt{500}$, since 500 points were averaged overall. In only one case did this give a different value than that derived from the preceding method, and in that case the value provided by the second method was used. The results for the linear and exponential mode are shown in Table 1.

The value of the signal was determined by subtracting the baseline value from the value of the signal without noise. The noise on the signal is the standard deviation of the difference between the values for signal with and without noise. Dividing the signal value by the noise value gives the S/N ratio, which for the linear mode is 217.

The noise on the input signal was approximately 50 mV p-p. Since there is a 99% probability that the RMS noise value is within 2.5 standard deviations of the mean (in this case, zero)[22], the standard deviation (or RMS noise value) is one fifth the peak to peak, or 10 mV. The signal level at the point measured on the decay curve was 10 mV, so the

Table 1. Observed Data and Calculated SNIR Values

	Mode	
	Linear	Exponential
Baseline	0.915 ±.001	0.0884 ±.0003
Signal	6.33 ±.001	1.07 ±.0004
Signal+Noise	6.09 ±.050	1.08 ±.013
S	5.41	0.99
N	0.025	0.0067
S/N	217.	147.
SNIR	217.	147.
SNIR (calc)	349.	127.

S/N ratio for the input signal was 1. For the linear integration mode, the SNIR is then 217/1, or 217. The theoretical value for the SNIR was 349. The possible causes for this discrepancy are discussed later.

The theoretical SNIR for the exponential mode depends upon the time constant of the input, and the aperture duration. For this work, the time constant was kept at 10 μs, and the aperture duration was 5 μs. This leads to an expected SNIR for the input electronics of 2. Combined with the following SNIR's, the total SNIR of the system in exponential mode shoud be 127. Using the same procedure as for the linear mode, the the calculated S/N ratio for the exponential mode was 147/1, giving an SNIR of 147.

Discussion

The first important result is the standard deviation of the points with no noise added. Because the analog signals from the boxcar integrator are first converted to the digital domain, there will be a certain amount of noise added to the signal from the conversion process. The collection of the linear mode data was done by using the 10V full scale range of an 8-bit ADC; the smallest step is 0.04 V. Each point taken had $0.04/\sqrt{12}$, or 0.012V added to the standard deviation, and the final average would have

$0.012/\sqrt{500}$, or 0.0005V added. The observed standard deviation is larger than this, which would indicate that the noise introduced in the system due to the quantization process is not the primary source of noise.

Since the voltage levels observed during the use of the exponential mode were so small, the DAC was switched to 2.56V full scale. The resolution was then 0.01V, and the added noise to the final average should be 0.0001V. Once again, this is smaller than that observed, which leads to the conclusion that the quantization process is still not the largest source of noise. Because the noise from the ADC is smaller than the current system noise, the inclusion of a 12-bit ADC in the system should not greatly improve the S/N of the system. When laser ionization signals are observed, the system noise is expected to increase, due to the radio frequency noise present from the laser discharge.

One early worry in the design of this system was that the use of an 8-bit DAC to drive the external scan of the boxcar integrator might lead to a jitter in the time resolution of the data. The fact that the noise on the signal at an early point on the decay, where the signal changes the fastest, is the same as the noise at a later point on the curve indicates that this jitter is negligible. If the other contributions to the system noise can be

reduced in the future, some jitter may be exposed, but at the present time none is seen.

There are 2 main reasons that the experimental SNIR values vary from the theoretical. The first deals with the method of measuring the noise on the input signal. The amplitude was estimated from an oscilliscope display by observing the maximum peak-to-peak variation in the noise display. The level of noise was adjusted so that it was approximately 50 mV p-p maximum. This level is only a rough approximation to provide a S/N ratio for the input close to 1. It is more important that the noise level remain constant during the course of the experiment, so that any comparison between modes may be accurate. The level control was not adjusted after the level was set; the presence or absence of noise was determined by connecting or disconecting the noise source. If the signal to noise value for the input signal was wrong, it would yield the same relative amount of error in both SNIR's, so clearly this must not be the only source of error.

The second source of error is the noise source itself. The approximations used to calculate the SNIR for the input module depends on the frequency spectrum of the noise having a constant power density. Also, the noise spectrum should not extend beyond 1/(2AD) Hz [18]. For an aperture duration

of 5 μs, the cutoff frequency should be 10 kHz. In order to determine the frequency spectrum of the noise a Federal Scientific Ubiquitous Signal Analyzer Model U-14a was used[23]. The photograph in Figure 5-4 is a picture of the frequency spectrum of the noise source. The x axis is in units of frequency, with the observed trace going from 0 to 10kHz. The y axis displays power density, and the spikes on the signal are markers placed every 1 kHz. Zero output is located on the first division from the bottom. Clearly, the amount of noise above 10 kHz is minimal.

Also it can be readily seen from Figure 5-4 that the noise is not white. A numerical correction to the calculated SNIR is not available, but the direction of the error can be explained. The calculated SNIR for the linear integration mode is a factor of 2.7 larger than that calculated for the exponential mode. The observed SNIR is only 1.5 times as large; the discrepancy arises in the input module.

The signal that passes thru a filter or integrator is proportional to the reciprocal of the noise equivalent bandpass. The noise is inversely proportional to the square root of the NEB. The S/N is then inversely proportional to the square root of the NEB. The NEB for the integrator in linear mode is 1/2t where t is the total integration time.

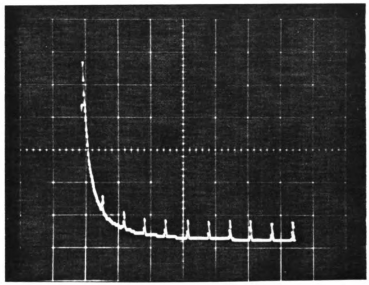


Figure 5-4. The Frequency Spectrum of the Noise

Thirty samples were integrated, each sample was 5 μs long. The NEB_1 is then $1/(300X10^{-6}$ s), or 3.3 kHz. In exponential mode, NEB_e is 1/4TC, where TC is 10 μs. The NEB_e is 25 kHz. The ratio of the S/N ratio for the linear mode to the S/N ratio of the exponential mode is 2.7; it is here that the factor of 2.7 seen in the final SNIR's appears.

The preceeding discussion depends upon the noise source being white, but it is the basis for understanding the pink noise correction. Because there is less total noise in the noise power spectrum than the calculations require, the observed S/N ratios should be higher than those calculated assuming white noise. The fact that the observed SNIR is less than the calculated SNIR for the linear mode is due to the inaccuracy in measuring the initial noise added to the signal. Because the noise bandpass for the linear mode is much smaller than that for the exponential mode, the actual noise power spectrum deviates from white noise less for the linear mode than for the exponential mode. The increase in S/N ratio will not be a great as for the exponential mode, where the noise deviates more from white. The proportionally greater increase in the exponential S/N ratio will cause the observed ratio between exponential and linear SNIR to be less than 2.7. In this work, the observed ratio is 1.5. The observed deviation of the SNIR from the calculated value of 349 in the linear mode is due to the

measurement of the noise superimposed on the input signal, but the relationship between the observed SNIR's is explained.

The noise power spectrum could be corrected by changing the filter circuits, as the output of the 5837 without any filtering has the spectrum shown in Figure 5-5. The peaks at every 5 kHz are produced by the spectrum analyzer as reference points, the peak at 48 kHz is believed to be due to the internal clock of the 5837. The specifications of the 5837 list the half power frequency at a minimum value of 26 kHz; the actual -3 dB point is at approximately 21 kHz.

Conclusions

From the data presented, the microcomputer-boxcar integrator interface and the software to control it perform the functions they were designed to perform. The use of a low resolution DAC as the scan source does not introduce any apparent noise in measuring data that involve high rates of change for small differences in aperture delay. The noise reduction capability of the boxcar integrator, even though not totally quantifiable, is able to extract a signal buried beneath large amounts of noise.

One data collection method in this experiment used the

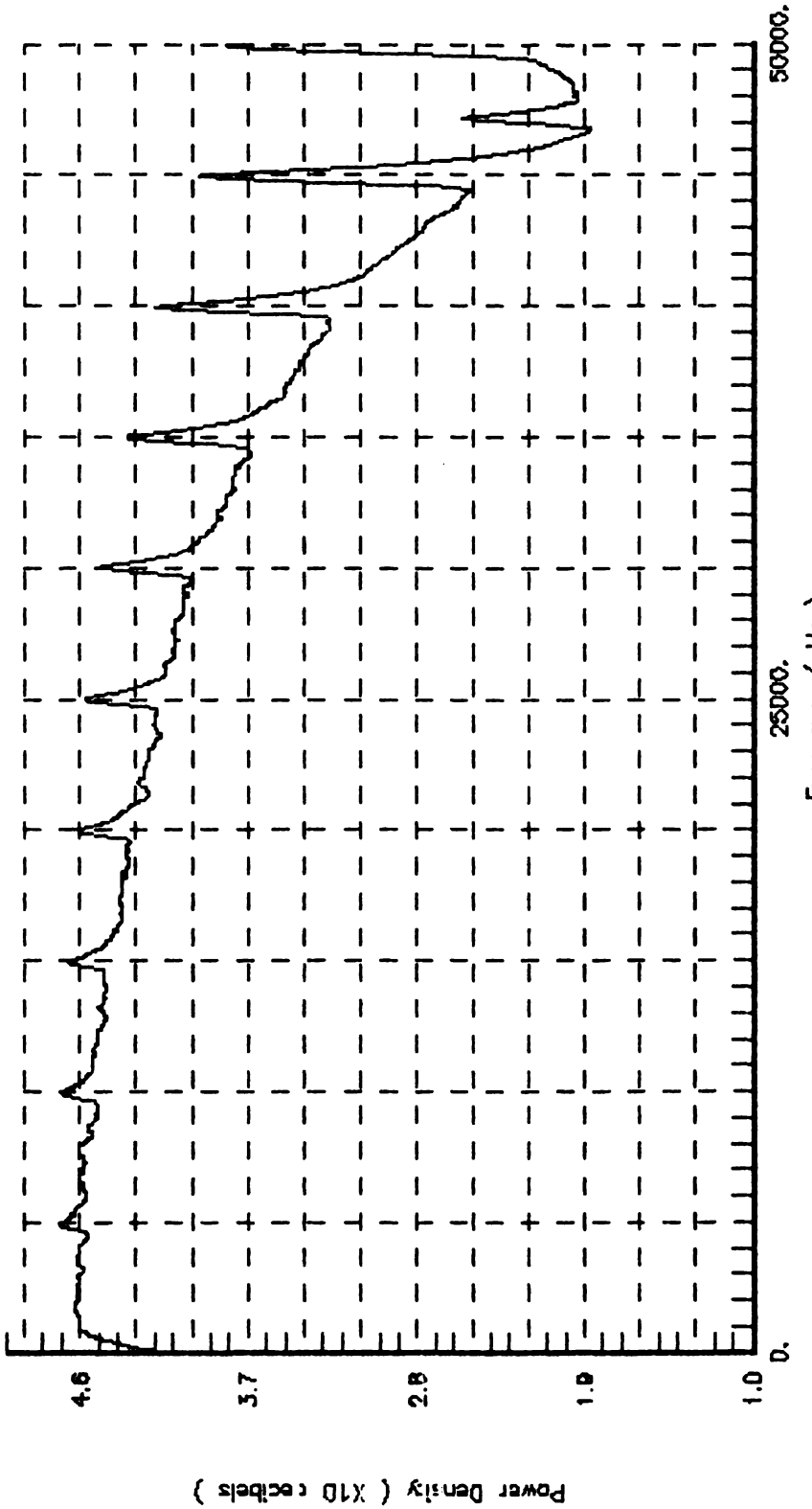


Figure 5–5. MM5837 Noise Spectrum

From 0 to 50 kHz

linear summation mode of the boxcar integrator. Using this mode of the boxcar integrator without the microcomputer interface would be very difficult, since the number of points collected has to be known. The use of this mode with the microcomputer will allow for the observation of lower level signals, with higher resolution.

CHAPTER 6

Decay of Sodium Ion Populations

Introduction

The DLI experimental system consists of four parts: the nitrogen pump laser, a tunable dye laser, the atomization system, and the ion detection system. Probes having a DC bias impressed upon them are placed in the flame to capture the ions and electrons produced by the laser-induced ionization of the analyte atoms. Because the nitrogen laser is pulsed, the tunable dye laser is also pulsed, as is the observed ionization signal. Since the pulse length of both lasers is small (5-10 ns), and the time between pulses is large (10-30 ms) the flame can be treated as a static system, with periodic perturbations provided by the laser pulses.

As with any such relaxation technique, there are two types of measurements that can be made. The first, and probably the easiest, is the measurement of the height of

the peak produced by the perturbation. In the DLI/LEI system, this magnitude depends upon the number of ions formed; the number of ions produced depends upon the concentration of analyte in the sample solution. Other work on the DLI system in this laboratory [24] has shown that the relationship between peak height and analyte concentration is linear over several orders of magnitude.

The peak height does not provide much information about how the system returns to equilibrium. To obtain this information, the signal behavior versus time is studied. A boxcar integrator is used to study the signal intensity at various times after the nitrogen laser pulse, and the time domain behavior of the signal may be observed by plotting signal intensity versus the aperture delay time of the boxcar integrator. Other work being done in this laboratory [25] uses the decay time of the signal to match a theoretical model for ion/electron migration in the flame to experimental results, and to calculate the temperature of the flame.

Ion migration, the movement of ions due to an applied field within the flame, is one measurement that requires the determination of the decay time τ of the ionization signal. The equation relating τ to the voltage applied to the probes, the distance between the probes, and the mobility

is:

$$\tau = L^2 / \mu_i E_p \qquad (6\text{-}1)$$

where L is the distance between the probe and the ionization region, μ_i is the mobility of the ion, and E_p is one half the applied potential to the probes. The mobility of sodium ions in the DLI flame has been shown to be on the order of 20 $cm^2 V^{-1} s^{-1}$ [1].

Other parameters also affect the decay time of the ionization signal. One such parameter is the rate the ions are withdrawn from the flame. As the ions are removed more quickly, the time it takes for the ions to reach the probes (migration rate) will begin to affect the signal. At very low currents, the voltage generated across a resistor in series with the high voltage supply to the probes depends only on the total charge generated within the flame, and the flame acts as a capacitor, when considering the rate of decay of the ionization signal. At relatively high currents, the voltage observed depends only on the amount of charge arriving at the probe at that time; the depletion of charge giving an exponential decay. At currents in between, the observed signal will be a convolution of both effects.

Because the amount of current drawn from the flame depends upon the size of the resistor in series with the power supply, or the effective load resistance of a current-to-voltage (i/V) converter, at low resistances the decay time will be independent of the load resistor. Previous work done in this laboratory [1] using a simple load resistor has shown this to be true.

In order to assist the boxcar integrator in noise reduction, and to reduce DC baseline offsets, the collection circuit was changed from a simple resistor to a 30 kHz high pass filter. A buffer amplifier was also included to allow the signal transmission line from the flame to the boxcar integrator input to be a low impedance line to help reduce noise pickup. Because of these changes, it became necessary to verify the correct operation of the new collection circuit.

Experimental

The DLI experimental system has been described in detail elsewhere [1], and only a brief description will be given here. A nitrogen discharge laser [26] provides 337 nm radiation to pump a Hänsch design tunable dye laser. The output of the dye laser is focused into a laminar flow hydrogen-oxygen-argon flame, which has an outer mantle flame

to provide a constant temperature radially across the flame. A portion of the nitrogen laser pulse is also focused into the flame, colinearly with the dye laser pulse.

Nichrome or iridium wire probes are mounted on translation stages, and positioned in the flame so that the negative probe is directly beneath the region where ionization is to be observed, and the positive probe is one or two cm above that region.

The instrumental parameters are shown in the documentation file in Figure 6-1. The boxcar integrator was set to exponential mode, with DC coupling, and a 1 MΩ input impedance. Because the aperture delay range was 10 μs, the aperture duration was reduced from the usual 5 μs to 0.5 μs.

Results

In the newly designed collection circuit, the resistor of the 30 kHz filter served as the passive i/V converter. To keep the filter cutoff frequency constant, the capacitor was changed as the load resistor was changed. Along with changing the load resistance, this circuit also changed the observed decay times of the signals. The observed times varied from 1 to 12 μs, exhibiting a marked dependence on the load resistor. There was also a very long time constant

```
PARENT RAW DATA FILE IS:  8T50KDCE
BUFFER START:  16896  END:  17716
INTERRUPTS PER DATA POINT:     10
RESET AFTER DATA POINT:        NO
START OF SCAN:                  0
LENGTH OF SCAN:               200
SCAN INCREMENT:                 5
POINTS PER SCAN INCREMENT:     10

     >>> Boxcar parameters <<<
  Aperture delay section
  1) %Initial A:
  2) %Initial B: 20%
  3) Range: 10 microseconds
  Aperture section
  4) Duration: 0.5 microseconds
  5) Scan time:
  Time constants section
  6) Boxcar: 0.1 second
  7) Model 164: 10 microseconds
  8) Comments: Time resolved Na New ckt
  9) 50k load DC coupled 1 M Exponential

DATA FILE WRITTEN: 8T50KDCE.DAT
 XMIN:      2.0000  XMAX:      9.8120
 YMIN:      0.0000  YMAX:      8.2812
SD  DATA FILE  8T50KDCE.DAT STDDEV INTO FILE
8T50KDCE.ICE
```

Figure 6-1. The Experimental Parameters

added to the signal, giving a slow return to the baseline which was not reached even at long aperture delays.

After examining this circuit, a replacement circuit was designed [27]. This circuit is shown in Figure 6-2. The effective load resistance is determined by the selection of the resistor in the voltage divider in the feedback loop. The 30 kHz filter is fixed between the i/V converter and the output buffer. A switch is placed across the capacitor in the filter to allow easy removal of the filter, and DC coupling of the signal path.

The decay noticed at times longer than 5 time constants was not removed by the use of this circuit. Changing the input coupling of the boxcar integrator to DC did remove this last added time constant. The results obtained using the new circuit, with DC coupling, are shown in Figure 6-3. The aperture delay range of all but Figure 6-3e is 10 μs, that of Figure 6-3e is 50 μs.

Discussion

Upon inspection, it is clear that the time constants for the ionization decay for the three lowest load resistances are the same. These constants are:

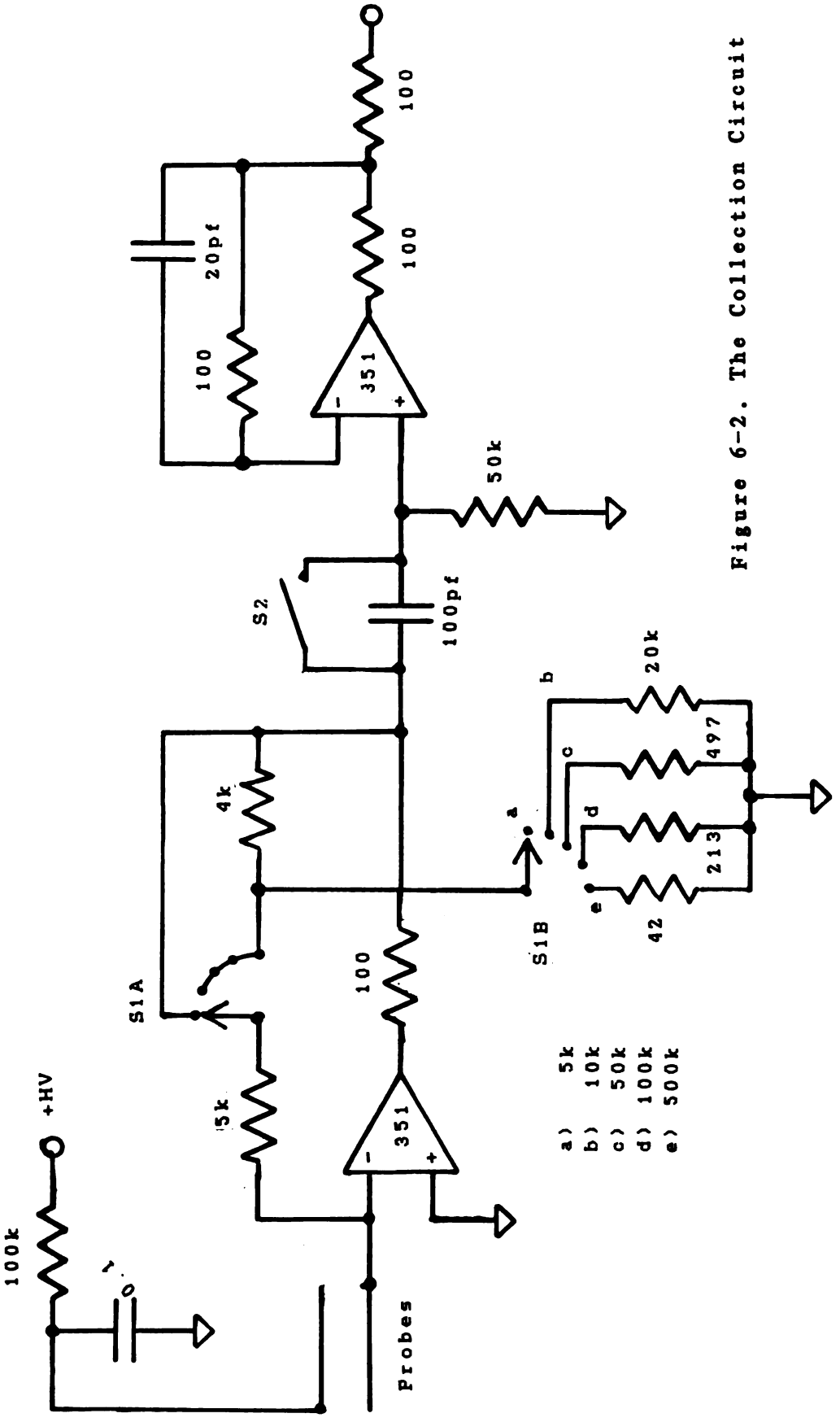


Figure 6-2. The Collection Circuit

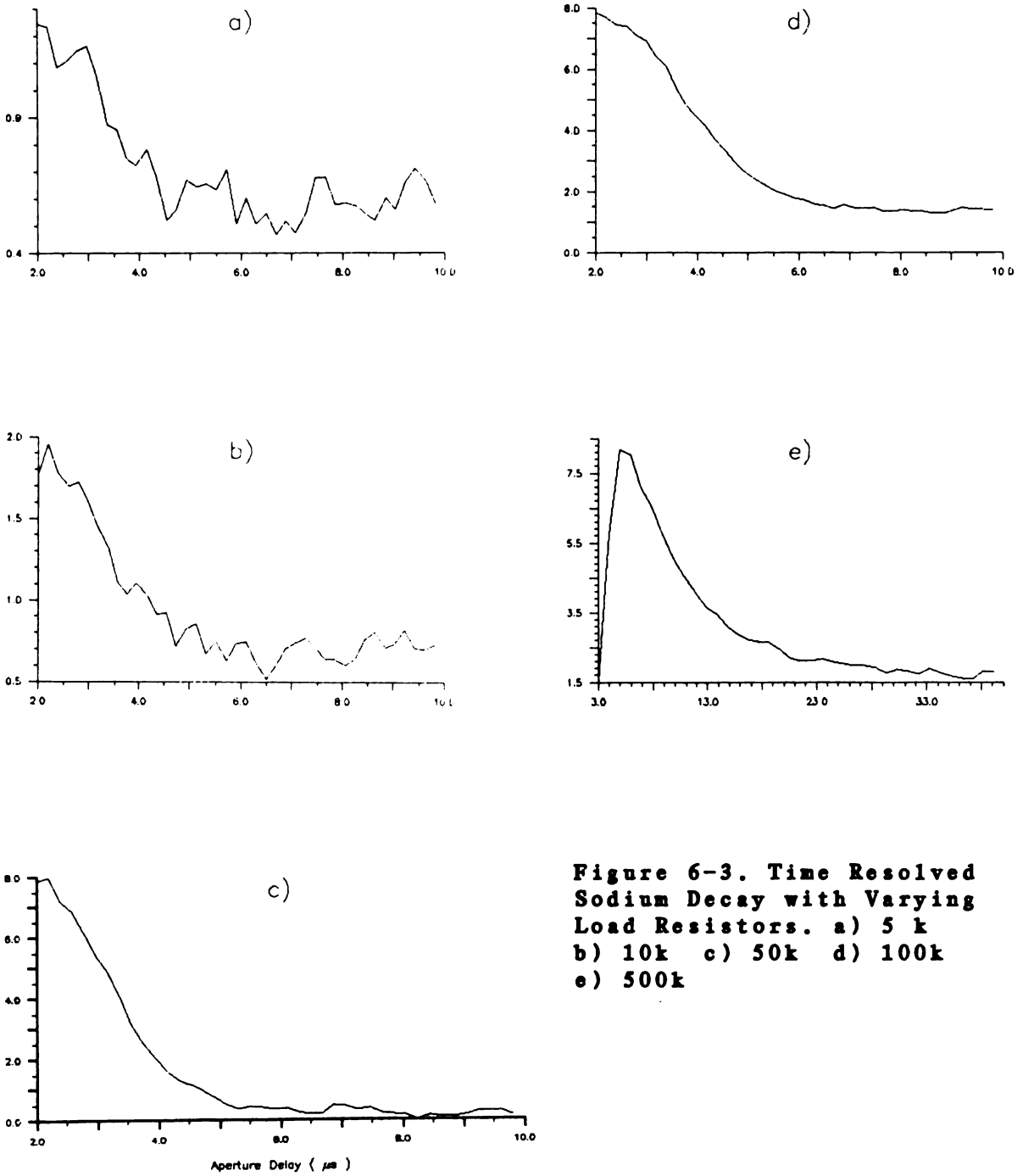


Figure 6-3. Time Resolved Sodium Decay with Varying Load Resistors. a) 5 k b) 10k c) 50k d) 100k e) 500k

Load Resistance	Decay Time
5 kΩ	1.1 μs
10 kΩ	1.1 μs
50 kΩ	1.2 μs
100 kΩ	>1.2 μs
500 kΩ	5.0 μs

It appears that the time constants for the higher load resistances are longer; however, problems begin to occur with 100kΩ and larger load resistances. Measuring the decay time at different portions of the decay curve gives different decay times; the decay time towards the end of the decay approaches 1.2 μs. The problem is more apparent in the decay curve for the 500 kΩ load resistor. What appears to be the beginning of the ionization pulse at 3 μs aperture delay is really a remnant from the noise pulse that appears on the signal at the instant of laser pulsing. This noise pulse generates a damped sine wave signal superimposed upon the ionization signal. As the circuit gain goes up (i.e. larger load resistances) the length of the ringing signal increases. It is this ringing that causes the odd shape of the decay curve for the 100kΩ load resistor, and it's presence also prohibits the observation of times earlier than 2 μs for the lower load resistances.

This ringing is triggered by the discharge in the nitrogen laser, and is generated in the collection circuit electronics. Bypass capacitors, and 100 Ω resistors have

been used in the circuit to try to reduce this ringing, but large amounts are still present. The nitrogen laser uses a spark gap to determine when the proper voltage has been reached on the capacitors within the laser, and cleaning the electrodes in this spark gap helped reduce the ringing, but did not eliminate it.

As the i/V converter gain is reduced, the S/N ratio of the system goes down. This would be observed if the noise has been introduced after the i/V converter. If it were present in the signal directly off the probes, decreasing the gain of the signal would also decrease the noise, and the S/N ratio would be the same. In practice, the noise at longer aperture delays is reduced by operating the boxcar integrator at 50 Ω input impedance, which would point to the cable between the collection circuit and the boxcar integrator as an antenna for RF noise. The use of the 50 Ω input impedance is limited to higher level signals, though, because the introduction of two 100 Ω resistors in the output of the buffer amplifier to limit ringing creates a resistive divider network, and reduces the signal to 1/5 of it's former value.

Equation 6-1 can be rearranged to solve for the mobility of the sodium ions in the flame. The negative probe is positioned by raising the probe towards the

ionization region until the dye laser strikes the probe. The probe is then lowered slightly. This method allows the negative probe to be placed in the closest possible proximity to the ionization region. When adjusted in this manner, the probe is approximately 0.5 mm from the ionization region of the flame. This measurement is only approximate due to the difficulty in forming a perfectly straight probe from the nichrome wire stock. The potential applied to the probes was 100 V. The mobility is then on the order of 40 $cm^2V^{-1}s^{-1}$. This is larger than that obtained previously. One reason for the difference between present and past work is that the gas flow rates for the flame have been changed, and the temperature of the flame will affect the ionic mobility.

Conclusions

The evidence shown herein would indicate that the current data collection electronics do not seriously interfere with the observation of time resolved ionization decay signals from the DLI system. The restriction imposed in DC coupling the entire system does not seem detrimental to the data collection process. Careful attention must be paid to experimental parameters in order to discriminate against the noise generated by the nitrogen laser.

The measurement of peak heights in quantitative analysis with the DLI system is not affected by the factors influencing the time decay measurements. Any effect seen in the peak height should be seen in all of the peaks observed in the same amount. If this were not the case, the high degree of linearity now observed would not be seen.

CHAPTER 7

Future Developments

The use of the microcomputer controlled boxcar integrator, even though much easier than control of the boxcar integrator by hand, has several limitations. These limitations are present in both the software and the hardware.

The software is currently limited by the use of assembly language, and a cross-assembler. Even though assembly language can give code which will execute faster than that generated by higher level languages, the time required to re-assemble and download the code after every minor correction is extremely large. The current operating system (SLOPS) does not allow the creation of functions from the keyboard; that is, every operation must be programmed.

The FORTH language can solve two of these problems. An internal compiler allows the definition of new commands from the keyboard. Minor changes in higher level commands can also be made directly from the keyboard. FORTH is currently

being installed on the DLI microcomputer.

There are also some improvements that can be made to the hardware. Even though the resolution of the ADC used is high enough that the domain conversion process is not the primary source of noise, the cost of a 12-bit ADC is low enough that one should be used. The use of a one-chip ADC will also remove the A-to-D conversion burden from the software, and increase the speed of the conversion.

Experiments also being done involve the wavelength scanning of the dye laser. The tuning element for the dye laser is a Heath monochromator, with the grating mounted so that the laser radiation is diffracted from it. The scanning function is still controlled by the manual controller. If the monochromator were controlled by a microcomputer, experiments that require wavelength scans, or detuning from the central wavelength could be controlled entirely by the microcomputer. One problem anticipated is the large amount of RF noise that will be in the area of the control circuit. The monochromator is placed next to the nitrogen laser, an excellent RF source, and the source of much of the noise in the rest of the system.

The improvements listed here are such that they can be accomplished concurrently with other projects. The highest

priority of these is the change of language, and this change will involve re-writing the microcomputer software.

APPENDICES

APPENDIX A

Listing of RAW2COOK

```
      PROGRAM RAW2COOK
C
C   CONVERTS RAW DATA FROM THE MICRO INTO
C   A MULPLOT COMPATABILE FILE, AND A DOC
C   FILE WHICH CONTAINS INSTRUMENTAL PARA-
C   METERS AND AN AUDIT TRAIL OF DATA
C   MANIPULATIONS.
C
C   7-MAY-1981
C
C      JOHN D STANLEY
C      DEPT OF CHEMISTRY
C      MICH STATE UNIVERSITY
C
C
C
C   SET UP DATA AREAS
C
      BYTE CMDLIN(80),FILES(32,10),FLAGS(10,10)
      BYTE BUFF(512),DOCFIL(32),DATFIL(32)
      BYTE RAWFIL(32),RD(2),SPACE
C
      BYTE PSTR(35,13),IPDPT,RESADP,LSCAN,SCNINC
      BYTE STSCAN,PPSCNC
      BYTE BYT(2)
C
      INTEGER ITABLE(96),DBUFF,BUFFE,CBASE
      INTEGER BUFF2(256),DOC,DATA
C
      REAL*4 X,Y,XINC,XMIN,XMAX,YMIN,YMAX,TEMP,DAC
C
C
C  THESE EQUIVALENCES SORT PARAMS FROM BUFF
C
      EQUIVALENCE (PSTR(1,1),BUFF(23))
      EQUIVALENCE (DBUFF,BUFF(1)),(BUFFE,BUFF(3))
      EQUIVALENCE (IPDPT,BUFF(5)),(RESADP,BUFF(6))
      EQUIVALENCE (CBASE,BUFF(7)),(LSCAN,BUFF(9))
      EQUIVALENCE (SCNINC,BUFF(10)),(BUFF(1),BUFF2(1))
      EQUIVALENCE (PPSCNC,BUFF(13)),(STSCAN,BUFF(11))
```

```
      EQUIVALENCE (IWORD,BYT(1))
C
C ***************************************************************
C
C  BEGIN CODE
C
C   INITIALIZE EXTREMA VARIABLES
C
      XMIN= 100000.
      YMIN= 100000.
      XMAX=-100000.
      YMAX=-100000.
C
C   AND MULPLOT DATA TAGS
C
      RD(1)='R'
      RD(2)='D'
      SPACE=' '
C
C   SET UP LUN INFORMATION
C
      DOC=2
      DATA=3
      TTY=5
C
C  GET MCR COMMAND LINE
C
      CALL GETMCR(CMDLIN,NUM)
C
C  PARSE OF FILE NAMES
C   PARSE RETURNS FILE NAMES FROM CMDLIN TO ARRAY FILES
C
      CALL PARSE(CMDLIN,FILES,FLAGS)
C
C  COPY FIRST FILE PARSED INTO RAWFIL
C
      CALL COPY(FILES(1,4),RAWFIL,32)
C
C  IF NO FILE NAME THERE, ASK FOR ONE
C
      IF (RAWFIL(1) .NE. 0) GO TO 2
    1 WRITE(5,1000)
 1000 FORMAT('∫RAW FILE TO COOK:    ')
      READ (TTY,1010,ERR=1,END=999)RAWFIL
 1010 FORMAT(32A1)
C
C   SIMILAR TO RAWFIL ACTION ABOVE
C
    2 CALL COPY(FILES(1,2),DATFIL,32)
      IF (DATFIL(1) .NE. 0) GO TO 3
      WRITE(TTY,1020)
 1020 FORMAT('∫DATA FILE TO WRITE: ')
      READ (TTY,1010,ERR=2,END=999)DATFIL
```

```
C
    3   CALL COPY(FILES(1,3),DOCFIL,32)
        IF (DOCFIL(1) .NE. 0) GO TO 4
        WRITE(TTY,1030)
 1030   FORMAT('∫DOC FILE TO WRITE:  ')
        READ (TTY,1010,ERR=3,END=999)DOCFIL
C
C   REMOVE CONTROL CHARS AND CR'S
C
    4       CALL ZEREND(DATFIL)
            CALL ZEREND(DOCFIL)
            CALL ZEREND(RAWFIL)
C
C   OPEN THE FILES
C
        CALL UNFORM(0,IERR,ITABLE,RAWFILE,0,3,1)
        IF (IERR .LT. 0) GO TO 800
C
        OPEN (UNIT=DOC,NAME=DOCFIL,TYPE='UNKNOWN',
     1       ACCESS='APPEND',ERR=810)
C
        OPEN (UNIT=DATA,NAME=DATFIL,TYPE='UNKNOWN',
     1        ACCESS='APPEND',FORM='UNFORMATTED',ERR=820)
C
C   GET X AXIS VALUES
C
   10   WRITE(TTY,1100)
 1100   FORMAT('∫X VALUE OF FIRST POINT: ')
        READ(TTY,1110,ERR=10,END=999)X
 1110   FORMAT(F10.4)
C
   11   WRITE(TTY,1120)
 1120   FORMAT('∫X INCREMENT: ')
        READ(TTY,1110,ERR=11,END=999)XINC
C
   12   WRITE(TTY,1125)
 1125   FORMAT('∫DAC Voltage: ')
        READ(TTY,1110,ERR=12,END=999)DAC
C
C   DAC IS CONVERSION OF NUMBERS FROM ADC TO VOLTAGE
C
        DAC=DAC/256.
```

```
C
C    GET FIRST BLOCK OF DATA
C    INSTRUMENT PARAMS
C
        CALL UNFORM(1,IERR,ITABLE,BUFF,512)
C
C   STORE STUFF IN DOC FILE AS ASCII
C
        WRITE(DOC,2010)RAWFIL
 2010   FORMAT(1X,'PARENT RAW DATA FILE IS:   ',32A1)
        WRITE(DOC,2020)DBUFF,BUFFE
 2020   FORMAT(' BUFFER START: ',I6,'  END: ',I6)
        BYT(1)=IPDPT
        BYT(2)=0
        WRITE(DOC,2030)IWORD
 2030   FORMAT(' INTERRUPTS PER DATA POINT: ',I6)
C
        TEMP='NO'
        IF (RESADP .NE. 0) TEMP='YES'
        WRITE(DOC,2040)TEMP
 2040   FORMAT(' RESET AFTER DATA POINT:   ',A4)
        WRITE(DOC,2050)CBASE
 2050   FORMAT(' CURRENT BASELINE:          ',I6)
C
        BYT(1)=STSCAN
        WRITE(DOC,2060)IWORD
 2060   FORMAT(' START OF SCAN:             ',I6)
        BYT(1)=LSCAN
        WRITE(DOC,2070)IWORD
 2070   FORMAT(' LENGTH OF SCAN:            ',I6)
        BYT(1)=SCNINC
        WRITE(DOC,2080)IWORD
 2080   FORMAT(' SCAN INCREMENT:            ',I6)
        BYT(1)=PPSCNC
        WRITE(DOC,2090)IWORD
 2090   FORMAT(' POINTS PER SCAN INCREMENT: ',I6)
        WRITE(DOC,2099)
 2099   FORMAT('  ')
C
C    NOW ADD BOXCAR PARMS
C
        DO 20 I=1,13
           WRITE(DOC,3000)(PSTR(J,I),J=1,32)
   20   CONTINUE
 3000   FORMAT(3X,32A1)
C
        WRITE(DOC,3010)DATFIL
 3010   FORMAT('DDATA FILE WRITTEN: ',32A1)
C
```

```
C ************************************************************
C
C   NOW WRITE DATA FILE
C
C       ICOUNT IS NUMBER OF POINTS WRITTEN
C       INUM IS NUMBER TO WRITE
C
        INUM=((BUFFE-DBUFF)/2)
        IF ( INUM .LE. 0 ) GO TO 600
        ICOUNT=0
C
C   SAVE DOCFIL NAME IN DATA FILE
C
        WRITE(DATA)SPACE,DOCFIL
C
C   GET BLOCK OF DATA
C
  100   CALL UNFORM(1,IERR,ITABLE,BUFF,512)
        IF (IERR .NE. 0) GO TO 700
C
        DO 110 I=1,256
C
C   LOOK AT TWO BYTES OF DATA AS INTEGER*2 WITH BUFF2
C    THEN CONVERT TO REAL NUMBER
C
            Y=FLOAT(BUFF2(I))
C
C   CORRECT FOR FLAGS
C
            DOXINC=0.
C
C   IF Y < 0 THEN BASELINE FLAG SET
C
            IF (Y .LT. 0.) Y=Y+32768.
C
C   IF Y > 2**15 THEN SCAN INCREMENT
C
            IF (Y .LT. 16384.) GO TO 105
C
C  INCREMENT X NEXT POINT
C
            Y=Y-16384.
            DOXINC=1.

C
C   CORRECT NUMBER TO VOLTAGE
C
  105       Y=DAC*Y
C
C   CHECK EXTREMES
C
            IF(X .GT. XMAX) XMAX=X
            IF(X .LT. XMIN) XMIN=X
```

```
            IF(Y .GT. YMAX) YMAX=Y
            IF(Y .LT. YMIN) YMIN=Y
C
C   WRITE UNFORMATTED DATA
C
            WRITE(DATA) RD,X,Y
C
            ICOUNT=ICOUNT+1
            IF(ICOUNT .EQ. INUM) GO TO 700
            X=X+(XINC*DOXINC)
C
  110   CONTINUE
        GO TO 100
C
  600   WRITE(TTY,6000)
 6000   FORMAT(' Warning - empty data file ...')
```

```
C
C    NOW EXIT GRACEFULLY
C
  700   CLOSE (UNIT=DATA)
C
        WRITE(DOC,4000)XMIN,XMAX
 4000   FORMAT('E XMIN: ',F10.4,'  XMAX: ',F10.4)
        WRITE(DOC,4010)YMIN,YMAX
 4010   FORMAT('E YMIN: ',F10.4,'  YMAX: ',F10.4)
C
        CLOSE (UNIT=DOC)
C
        WRITE(TTY,4000)XMIN,XMAX
        WRITE(TTY,4010)YMIN,YMAX
C
C   ALL DONE, BYE BYE
C
        CALL EXST(1)
```

```
C ************************************************************
C
C   ERROR EXITS
C
  800   WRITE(TTY,8000)RAWFILE
 8000   FORMAT(' ERROR IN OPENING : ',32A1)
        STOP
C
  810   WRITE(TTY,8000)DOCFIL
        STOP
C
  820   WRITE(TTY,8000)DATFIL
        STOP
C
C   BYE, EOF DURING TTY READ
C
  999   CALL EXST(0)
        END
```

APPENDIX B

Listing of ICING

```
      PROGRAM ICING
C
C   A PROGRAM TO TAKE DAT FILES AND AVERAGE
C   THE DATA POINTS WITH THE SAME X VALUES
C   INTO ONE X VALUE, AND GIVE SD INFO TOO.
C
C   PROVIDE <1001 DATA POINTS
C
      REAL ARRAY(1000)
      LOGICAL EOF,NODOC,SDOFF,CMDLIN
      BYTE DATFIL(32),DOCFIL(32),OUTFIL(32),GRUNGE,SDINP
      BYTE INLINE(80)
      INTEGER RD,FN,DM,COUNT,TTY,DOC,DAT,OUT
C
C
C   DEFINE LUNS
C
      TTY=5
      DOC=3
      OUT=2
      DAT=1
C
C   DATA TAGS
C
      RD='RD'
      FN='FN'
C
C   SET NO INPUT COMMAND LINE
C
      CMDLIN=.FALSE.
C
      CALL GETMCR(INLINE,ILEN)
      IF (ILEN .LT. 1) GO TO 1
      CMDLIN=.TRUE.
C
C   NOW PUT INLINE INTO ALL THREE FILE NAMES
C
      DO 6 I=1,ILEN
         J=I+4
         DATFIL(I)=INLINE(J)
```

```
          OUTFIL(I)=INLINE(J)
    6   CONTINUE
C
C   STRADD ADDS STR1 TO STR2 GIVING STR3
C
       CALL STRADD(DATFIL,'.DAT',DATFIL)
       CALL STRADD(OUTFIL,'.ICE',OUTFIL)
C
       IF (CMDLIN) GO TO 7
C
    1  WRITE(TTY,1000)
 1000  FORMAT(1X,'Data icing program, ver 1.3'/
     1     '/Output file name:')
C
C   GET OUTPUT FILE NAME
C

       READ(TTY,1010,END=900)OUTFIL
 1010  FORMAT(32A1)
C
    7  CALL ZEREND(OUTFIL)
C
C   OPEN OUTPUT FILE, IF NOT PRESENT, CREATE
C
       OPEN(UNIT=OUT,
     1      NAME=OUTFIL,
     2      TYPE='UNKNOWN',
     3      ACCESS='APPEND',
     4      ERR=910)
C
       SDOFF=.TRUE.
C
C   IF CMDLINE ENTERED, SKIP THIS QUESTION
C
       IF (CMDLIN) GO TO 11
C   ASK IF SD BARS WANTED.
C
       WRITE(TTY,1011)
 1011  FORMAT('/Standard deviation feature ',
     1         'desired [Y/N] ?')
       READ(5,1012)SDINP
 1012  FORMAT(A1)
       IF ((SDINP .EQ. 'Y') .OR. (SDINP .EQ. 'y'))
     1      SDOFF= .FALSE.
C
C   SKIP SPECIAL FEATURE IF SDOFF = TRUE
C
       IF (SDOFF) GO TO 10
C
C   SET UP OUTPUT FILE FOR SPECIAL FEATURES
C   STD DEV BARS
C
       WRITE(OUT,5090)
```

```
 5090  FORMAT('..LNM03')
C
C    TELL USER I'M READY FOR FILE TO ICE
C
   10  WRITE(TTY,1020)
 1020  FORMAT('∫Ice>')
C
C    GET FILE TO ICE, IF Z NO MORE FILES
C
       EOF = .FALSE.
       READ(TTY,1010,END=920)DATFIL
   11  CALL ZEREND(DATFIL)
C
C    OPEN DATA FILE
C
       OPEN(UNIT=DAT,
     1       NAME=DATFIL,
     2       TYPE='OLD',
     3       ACCESS='SEQUENTIAL',
     4       FORM='UNFORMATTED',
     5       ERR=930,READONLY)
C
C    READ DOCFIL NAME FROM DATFIL, GRUNGE IS ONE CHARACTER
C     FIX FOR CARRIAGE CONTROL BLANK IN DATFIL
C
       READ(DAT,ERR=940)GRUNGE,DOCFIL
C
C    INITIALIZE NODOC IN CASE NO DOCFILE EXISTS
C
       NODOC=.FALSE.
       CALL ZEREND(DOCFIL)
C
C    OPEN DOCFILE, IF NONE EXISTS, ERR
C
       OPEN(UNIT=DOC,
     1       NAME=DOCFIL,
     2       TYPE='OLD',
     3       ACCESS='APPEND',
     4       ERR=950)
C
C    SAVE DATFILE NAME IN NEW DATAFILE FOR AUDIT TRAIL
C
   20  WRITE(OUT,5000)FN,DATFIL
 5000  FORMAT(A2,32A1)
C
C    GET FIRST DATA POINT, IF NOT RD TAG - IGNORE
C
   30  READ(DAT)DM,XLAST,ARRAY(1)
       IF(DM .NE. 'RD') GO TO 30
C
       COUNT=1
C
C    GET NEXT POINT
```

```
C
  100   READ(DAT,END=150)DM,X,ARRAY(COUNT+1)
        IF ( DM .NE. 'RD') GO TO 100
C
C    IF THE NEXT POINT'S X VALUE DIFFERENT THAN LAST,
C     THEN WE HAVE AN X INCREMENT, TIME TO AVERAGE AND
C      STD DEV LAST DATA SET, KEEP NEW POINT IN ARRAY
C       ONE HIGHER THAN LAST POINT THIS SET.
C
           IF ( X .NE. XLAST ) GO TO 200
           COUNT=COUNT+1
        GO TO 100
C
C    IF END OF FILE, SET EOF FLAG, TEST LAST
C     DATA POINT FOR RD TAG
C    Last point read eof, bad data
C
  150   EOF = .TRUE.
C
C    TIME TO AVERAGE DATA SET WE HAVE
C
  200   SUM=0.
C
        DO 210 I=1,COUNT
  210   SUM=SUM+ARRAY(I)
C
        AVERAG=SUM/FLOAT(COUNT)
C
C    NEW SUM USED IN STD DEV PART
C
        SUM=0.
C
C    SKIP THIS PART IF NO STD DEV WANTED
C
        IF (SDOFF) GO TO 223
C
```

```
C ************************************************************
C
C   BEGIN STANDARD DEVIATION PART OF PROGRAM
C
C   CORRECT FOR N<6 TO BE N-1 IN STDDEV
C
       N=COUNT
       IF (N .GT. 5) GO TO 215
          N=N-1
          IF (N .LE. 0) N=1
C
C   SUM THE SQUARES OF THE ERRORS
C
  215  DO 220 I=1,COUNT
  220  SUM=SUM+(ARRAY(I)-AVERAG)**2
C
C   DIVIDE BY N
C
C      This gives sd of averaged point, not sd of
C           each point in average
C
       STDDEV=SQRT(SUM)/FLOAT(N)
C
C   END STANDARD DEVIATION PART
C
C ************************************************************
```

```
C
C    NOW OUTPUT TO FILE
C
  223   WRITE(OUT,5010)RD,XLAST,AVERAG
 5010   FORMAT(A2,E14.6,',',E14.6)
C
C    SKIP IF STD DEV TURNED OFF
C
        IF (SDOFF) GO TO 225
C
        WRITE(OUT,5020)XLAST,AVERAG,0.,STDDEV
 5020   FORMAT('..LNDR ',4(E14.6,','))
C
C
C    GET FIRST VALUE OF THIS SET INTO ARRAY(1),
C     SET NEW XLAST VALUE
C
  225   ARRAY(1)=ARRAY(COUNT+1)
        COUNT=1
        XLAST=X
C
C    NOW BACK TO GET MORE, IF NOT END OF FILE
C
        IF ( .NOT. EOF ) GO TO 100
C
        IF (NODOC) GO TO 240
           WRITE(DOC,2000)DATFIL,OUTFIL
 2000      FORMAT('SD  DATA FILE  ',32A1,' STDDEV ',
     +             'INTO FILE ',32A1)
           CLOSE(UNIT=DOC)
C
  240   CLOSE(UNIT=DAT)
C
C    BACK FOR MORE INPUT FILES
C
        IF (CMDLIN) STOP 'ONE FILE DONE'
        GO TO 10
```

```
C
C ************************************************************
C
C  ERROR ROUTINES
C
C   END OF FILE DURING OUTPUT FILE NAME READ, GRACIOUS EXIT
C
  900   CALL EXST(1)
C
C   ERROR IN OPENING OUTFIL, OR READING FILE NAME ON INPUT
C
  910   WRITE(TTY,9100)OUTFIL
 9100   FORMAT(1X,'I don''t understand file ',32A1)
        GO TO 1
C
C   END OF FILE ON INPUT OF DATA FILE NAME, GRACIUOS EXIT
C
  920   CLOSE(UNIT=OUT)
        CLOSE(UNIT=TTY)
        GO TO 1
C
C   ERROR OPENING DATA FILE, TRY AGAIN
C
  930   WRITE(TTY,9300)DATFIL
 9300   FORMAT(1X,32A1,' not found. Try again.')
        GO TO 10
C
C   PROBLEM READING DOCFILE NAME IN DATA FILE
C
  940   WRITE(TTY,9400)DATFIL
 9400   FORMAT(1X,'Error on docfil read in ',
     +             32A1,', file abort.')
        CLOSE(UNIT=DAT)
        GO TO 10
C
C   ERROR ON DOCFILE NAME, OR NO DOCFILE
C    MAY CREATE, OR SET NO DOCFILE FLAG TO FORGET
C     ABOUT DOCFILE ALL TOGETHER
C
  950   WRITE(TTY,9500)DOCFIL
 9500   FORMAT('∫',32A1,' not found. Create?')
        READ(TTY,9510)TEMP
 9510   FORMAT(A1)
        IF(TEMP .NE. 'Y') NODOC = .TRUE.
        IF( .NOT. NODOC)
     1     OPEN(UNIT=DOC,NAME=DOCFIL,TYPE='NEW',
     1          ACCESS='APPEND')
        GO TO 20
C
C   END ERROR ROUTINES
C
C ************************************************************
        END
```

BIBLIOGRAPHY

BIBLIOGRAPHY

1. C.A. vanDijk, F.M. Curran, K.C. Lin, S.R. Crouch, Anal.Chem., 53, 1275 (1981)

2. D. Larsen, Anal. Chem., 45, 217 (1973)

3. R. Lum, R. Jones, Rev. Sci. Ins., 51, 954 (1980)

4. C. Calkin, Ph.D Thesis, Michigan State University, 1981

5. D. Christmann, Anal. Chem., 53, 276 (1981)

6. J.P. Avery, Ph.D thesis, University of Illinois, 1978

7. J. Perry, M. Bryant, H. Malmstadt, Anal. Chem., 49, 1702 (1977)

8. B. Newcome, C. Enke, 'A Modular Microprocessor System for Laboratory Automation with a Novel Twin Bus Construction.' , work in progress.

9. H. Gregg, P. Hoffman, Ph.D research in progress.

10. H.V. Malmstadt, C.G. Enke, S.R. Crouch, Electronics and Instrumentation for Scientists, Benjamin/Cummings, Menlo Park, Ca., 1981

11. P. Dryden, Ph.D Thesis, Univ. of Ill., 1975

12. F.J. Holler, J.P. Avery, S.R. Crouch, C.G. Enke, Experiments in Electronics, Instrumentation and Microcomputers, Benjamin/Cummings, Menlo Park, CA, 1981

13. Princeton Applied Research, Princeton, NJ

14. E. Ratzlaff, Ph.D research in progress

15. H. Gregg, Ph.D research in progress.

16. P.Hoffman, Ph.D research in progress

17. T. Atkinson, MULPLT - A Multiple Data Set, File Based Data Plotting System, Chemistry Department, Michigan State University, 1981

18. Model 162 Boxcar Integrator Operating Manual, Princeton Applied Reasearch, Princeton N.J.

19. J. D. Ingle, S. R. Crouch, Optical Spectrochemical Methods of Analysis, being written, Prentice Hall, 1985

20. H.V. Malmstadt, C.G. Enke, S.R. Crouch, Electronic Measurements for Scientists, Benjamin, Menlo Park, CA, 1974, p852

21. National Semiconductor Corporation, Santa Clara, CA

22. H.V. Malmstadt, C.G. Enke, S.R. Crouch, G. Horlick, Optimization of Electronic Measurements, Benjamin, Menlo Park, CA, 1974

23. Federal Scientific Corporation, New York, NY

24. F. Curran, Ph.D research in progress

25. K. Lin, Ph.D research in progress

26. National Research Group, Madison, WI, 53705

27. M. Rabb, Electronic Engineer, Michigan State University