Instrumental design requirements for measurements of electrochemical reduction rates of haloorganics by Richard Allen Hughes

A thesis submitted in partial fulfillment of the requirements for the degree of Master of Science in Chemistry
Montana State University

Abstract:
The degradation rates of environmental contaminants are of potential use in many fields. As such, a methodology for the rapid determination of the rates of degradation of electroactive Species could prove valuable.

An attempt was put forth to construct a micro-computer controlled interface and electrochemical instrumentation.

It was hoped that a highly flexible instrument capable of high speed, high resolution data transfer would aid in the investigation of the rates of degradation of selected haloorganics.

A commercial instrument (EG&G Par 273A) was utilized in a preliminary investigation of the degradation rates of carbon tetrachloride and carbon tetrabromide.

The instrumentation which was constructed was not utilized in this study due to difficulties with data transfer capabilities and other problems with the electronics. However, utilizing the information gathered with the commercial instrumentation, preliminary values for the rate constant of the aqueous degradation of carbon tetrachloride were determined.

INSTRUMENTAL DESIGN REQUIREMENTS FOR MEASUREMENTS OF ELECTROCHEMICAL REDUCTION RATES OF HALOORGANICS

by

Richard Allen Hughes

A thesis submitted in partial fulfillment
of the requirements for the degree

of

Master of Science

in

Chemistry

MONTANA STATE UNIVERSITY
Bozeman, Montana

May 1995

APPROVAL

of a thesis submitted by

Richard Allen Hughes

This thesis has been read by each member of the thesis committee and has been found to be satisfactory regarding content, English usage, format, citations, bibliographic style, and consistency, and is ready for submission to the College of Graduate Studies.

6-15-95
Date

Richard D Geer
Chairperson, Graduate Committee

Approved for the Major Department

6/15/95
Date

David M Doty
Head, Major Department

Approved for the College of Graduate Studies

7/18/95
Date

R K Brown
Graduate Dean

STATEMENT OF PERMISSION TO USE

In presenting this thesis in partial fulfillment of the requirements for a Master of Science degree at Montana State University, I agree that the Library shall make it available to borrowers under rules of the Library.

If I have indicated my intention to copyright this thesis by including a copyright notice page, copying is allowable only for scholarly purposes, consistent with "fair use" as prescribed in the U.S. Copyright Law. Requests for permission for extended quotation from or reproduction of this thesis in whole or in parts may be granted only by the copyright holder.

Signature Paul A Hyles

Date 6/13/95

To Tammy, Allyson, Dad and Mom

TABLE OF CONTENTS

LIST OF FIGURES

TABLE OF CONSTANTS

A = area of electrode

α = transfer coefficient

C_o^* = Bulk concentration of oxidized species

D_o = diffusion coefficient

E = potential of an electrode vs a reference

E^o = standard potential of an electrode

$E^{o\prime}$ = formal potential of an electrode

E_p = peak potential

E^* = potential at which reduction rate is to be measured

F = Faraday constant

i_p = peak current

k_f = heterogeneous rate constant for reduction

k_{homo} = rate constant for a homogenous reaction

k^o = standard (intrinsic) heterogeneous rate constant

n = electrons per molecule oxidized or reduced

n_a = number of electrons involved in the rate determining step

R = gas constant

T = temperature

ν = potential sweep rate

ABSTRACT

The degradation rates of environmental contaminants are of potential use in many fields. As such, a methodology for the rapid determination of the rates of degradation of electroactive species could prove valuable.

An attempt was put forth to construct a micro-computer controlled interface and electrochemical instrumentation. It was hoped that a highly flexible instrument capable of high speed, high resolution data transfer would aid in the investigation of the rates of degradation of selected haloorganics.

A commercial instrument (EG&G Par 273A) was utilized in a preliminary investigation of the degradation rates of carbon tetrachloride and carbon tetrabromide.

The instrumentation which was constructed was not utilized in this study due to difficulties with data transfer capabilities and other problems with the electronics. However, utilizing the information gathered with the commercial instrumentation, preliminary values for the rate constant of the aqueous degradation of carbon tetrachloride were determined.

INTRODUCTION

Degradation Pathways and Kinetics

Since the advent of pesticides, herbicides and other chemicals used to control pests, there has been concern about the longevity of these compounds within the environment. Many of these compounds are halogenated organics[1] and as such may be susceptible to oxidative or reductive destruction. It is known that there are non-biological conditions in localized environments that can cause reduction or oxidation of electro-active organic substances. For instance in ground water or flooded soil systems the reduction or oxidation of an electro-active species may be governed by the presence of soluble, thermodynamically reversible redox couples. However, the potential at which these redox couples exist may be mediated by microbial or other biological systems and the contaminants themselves may be degraded by these biological systems.[1,2,3]

Utilizing certain electrochemical techniques it may be possible to predict the fate and minimum rate of environmental degradation for many of these halogenated organic chemicals from their electrochemical redox kinetics and other measurable redox properties in the environment. These procedures could prove to be a viable method for predicting the environmental fate and minimal rate of

degradation of a variety of newly proposed pesticides, chemicals that may have intentional exposure to the environment and to provide quick and reasonable predictions of degradation rates in areas of environmental concern.

For the purposes of this investigation only those reduction processes which do not involve direct biological intervention will be considered. The rates for these processes are minimal rates, uncatalyzed by biological activity. The half life for these rates can range from minutes to centuries, depending upon the substrate and the concentration of the reversible redox couples in the environment. There are several underlying assumptions and principals which will support the use of electrochemical techniques for investigating the minimum rates of localized degradation of environmental contaminants.

The existence of an oxidation reduction potential(ORP) in the environment is confirmed by a simple potentiometric measurement, with an inert and reference electrode pair. From this measurement the ratio of the concentrations of reduced to oxidized species can be determined. This is according to the well known Nernst equation (Eq 2). So, for example , using iron species as the redox couple(Eq 1):

$$Fe^{3+} + e^{-} \rightleftharpoons Fe^{2+} \quad E^{\circ} = 0.771V \qquad 1$$

$$E = E^{\circ} - \frac{RT}{nF} \ln \frac{[Fe^{2+}]}{[Fe^{3+}]} \qquad 2$$

Therefore, the ratio of the species present sets the measured potential of the local environment. Again it is understood that certain biological systems can potentially alter the observed potential, but for the purpose of this study it is assumed that experimental steps will be taken to minimize this so that the desired information will be obtained.

Carbon tetrachloride is a common organic solvent and was widely used in the past. Although use has declined due to its carcinogenicity, it has been the source of isolated but serious environmental contamination problems.[4] For this reason and for ease of explanation, the discussion of determining the degradation rates of electro-active halogenated organics will focus on carbon tetrachloride (CCl_4). The degradation rate of CCl_4 depends on three main factors. First, the concentrations of the reduced and oxidized species of the major reversible redox couple(s), and therefore the redox potential. Second, the concentration of CCl_4, the electro-active species of interest. And third, the intrinsic homogeneous rate constant, k_{homo}, for the acceptance of an electron by CCl_4 at the measured redox potential.

The reaction to consider would then be

$$CCl_4 + Fe^{2+} \overset{k}{\rightleftharpoons} CCl_4^{\cdot-} + Fe^{3+} \quad (3)$$

The radical anion produced is unstable and rapidly dissociates to form free chloride and the reactive radical $CCl_3^{\cdot}$.

$$CCl_4^{\cdot-} \overset{fast}{\rightarrow} CCl_3^{\cdot} + Cl^- \quad (4)$$

The radical will react via one of several possible reaction sequences, depending upon the relative concentrations of reactants.

1. Abstract a hydrogen atom from an Environmental Source (HES) to give a more stable radical ($ES^{\cdot}$) and a less halogenated compound. (Note, abstraction of a hydrogen atom from water is not likely as the hydroxy radical is more reactive and would reverse the process.)

$$CCl_3^{\cdot} + HES \rightarrow HCCl_3 + ES^{\cdot} \quad (5)$$

2. Abstract an electron from the reduced form of a reversible redox couple to form a very basic anion that reacts with water.

$$CCl_3^{\cdot} + Fe^{2+} \rightarrow CCl_3^- + Fe^{3+} \quad (6)$$

$$CCl_3^- + H_2O \rightarrow HCCl_3 + OH^- \quad (7)$$

3. Addition to a pi bond system, like >C=C<, >C=O, or -N=O, again giving rise to a new radical.

$$CCl_3^{\cdot} + R\text{-}HC{=}CH\text{-}R \rightarrow Cl_3C\text{-}\underset{\displaystyle R}{\underset{|}{HC}}\text{-}\dot{C}CH\text{-}R \quad (8)$$

4. Coupling with relatively stable radicals in the environment, like triplet dioxygen or perhaps the

nitric oxide (·NO) radical from the further reduction of nitrite.

$$CCl_3^{\cdot} + \dot{N}O \rightarrow Cl_3C\text{-}NO \qquad 9$$

It should be noted that sequences 1 and 2 would lead to the replacement of a halogen with hydrogen, a frequently observed anaerobic process.[5,6] Sequences similar to 3 and 4, however, would lead to a variety of products with diverse chemistry that will not be considered in this study. Also note that sequences 1 and 3 would generate a free radical chain reactions in addition to their other pathway.

Since reaction 3 is rate limiting and the same for all these reaction sequences, the overall rate of disappearance of CCl_4 is controlled by its intrinsic rate constant, k_{homo}.

$$\frac{-d(CCl_4)}{dt} = k_{homo}[CCl_4][Fe^{2+}] \qquad 10$$

Electrochemical reduction of organohalogen compounds, like CCl_4, follow the sequence of Reactions 3, 4, 6 and 7 of the pathways discussed above with the electrode taking the place of the reversible redox couple and acting as an infinite source of electrons. However a key difference is that the rate constant is based on a heterogeneous reactions at the electrode surface.

A critical factor within this study will be the ability to directly relate heterogeneous and homogeneous rate constants for the reactions that are being investigated. One possible solution to this dilemma would be to consider

the homogeneous system at the molecular level and consider each interaction of analyte and reduced redox species as an independent heterogeneous reaction. It may be possible to derive a mathematical equation that will allow this comparison. To verify this, it would be necessary to obtain data via two independent experimental techniques[7,8].

Methodology for Determining the Heterogeneous and Homogeneous Degradation Rates of CCl_4

Investigating Homogeneous Degradation Kinetics

Consider the following system; where CCl_4 degradation in a sterile, sealed, reversible redox system is possible over a reasonable range of ORP values. Considering only redox systems that transfer a single electron it would be possible to determine the degradation kinetics and compare them to Equation 10. Such an experimental environment might be envisioned to consist of an inert electrolyte (perhaps 0.1 M potassium chloride) and a metal ion complex poised at the desired ORP, either by chemical or electrochemical titration. To ensure pseudo first order kinetics for the disappearance of CCl_4, the reduced metal ion complex concentration would be >10 times that of CCl_4 initially. To maintain the ORP relatively constant during the experiment, the oxidized to reduced species ratio of the redox couple must not change significantly during the course of the reaction. This requirement limits the usefulness of the environmentally important iron(II)/(III) couple for this

study under simulated anaerobic experimental conditions since the ORP would tend to be quite positive under standard conditions. An ORP of less than ≈-0.10 volts vs. SCE is much more practical. The cobalt(II/III) ammonia complex couple with a E° = -0.133 volts[9] could be a useful surrogate for these first tests. Working with a sealed system avoids loss of volatile CCl_4 or $HCCl_3$ during the course of the experiment.

The progress of the CCl_4 degradation could be followed both by *in situ* electrochemistry and by removing samples for analytical analysis. With the electrodes sealed into the system the ORP could be monitored continually by potentiometry to ensure a stable ORP. The concentrations of redox species and carbon tetrachloride could be measured periodically by cyclic voltammetry. Other electrochemically active products may be observed when concentrations reach detectable limits. These experiments would not appreciably perturb the system.

Results of these experiments should give insight into several aspects of the degradation system. It should show whether the degradation rate is first order in CCl_4 as predicted. The reduction capacity of the system could be determined if the degradation rate is first order with respect to the reduced species of the reversible redox couples. The log of the intrinsic homogeneous rate constant (k_{homo}) for the CCl_4 degradation could be compared to the

measured ORP value in order to investigate if they are in fact proportional. Lastly, the mass balance between observable products and CCl_4 disappearance could be monitored to verify its completeness and also allow a check on major routes of degradation.

Investigating Heterogeneous Degradation Kinetics

The previous example relied on indirect determination of the rate based on the change in concentration over time. The goal of this study is to demonstrate that by utilizing quality instrumentation and the known electrochemical technique of cyclic voltammetry it is possible to predict the degradation rates at virtually any potential within a short period of time. This method would allow the determination of the intrinsic heterogeneous rate constant (k^o) for the reduction of CCl_4 at a metal electrode over a range of potentials.

To briefly review cyclic voltammetry, a compound of interest, in a solvent containing electrolyte, is subjected to a linear potential change. Then at the limiting voltage the potential sweep is reversed until the chosen end potential is reached. The resultant current which flows is recorded relative to this applied potential. Generally, the features of interest are the potential where the cell current peaks, E_p, and the value of the peak current, i_p. The equations which explain the relationship of peak current

and potential in irreversible systems are shown in equations 11 and 12 below.[10]

$$E_p = E^{o\prime} - \frac{RT}{\alpha n_a F}\left[0.780 + \ln\left(\frac{D_o^{1/2}}{k^o}\right) + \ln\left(\frac{\alpha n_a F \nu}{RT}\right)^{1/2}\right] \qquad 11$$

$$i_p = 0.4958 n F A C_o^* D_o^{1/2} \nu^{1/2} \left(\frac{\alpha n_a F}{RT}\right)^{1/2} \qquad 12$$

Equations 11 and 12 can be combined to give

$$i_p = 0.227 n F A C_o^* k^o \exp\left[-\left(\frac{\alpha n_a F}{RT}\right)(E_p - E^{o\prime})\right] \qquad 13$$

or, after rearrangement

$$\ln i_p - \ln n = \ln(0.227 F A C_o^*) + \ln k^o - \left[\left(\frac{\alpha n_a F}{RT}\right)(E_p - E^{o\prime})\right] \qquad 14$$

Thus a plot of ($\ln i_p - \ln n$) vs ($E_p - E^{o\prime}$) for data collected over a wide range of sweep rates would give a slope of $-\alpha n_a F/RT$ and an intercept of $\ln(0.227 F A C_o^*) + \ln k^o$.[10]

In the above equations there are two terms that cannot actually be used as defined, the formal electrode potential, $E^{o\prime}$, and the standard rate constant, k^o.

The formal electrode potential, $E^{o\prime}$, is defined as the measured half-cell potential where the ratio of oxidized to reduced species is unity and the other solution conditions are in their standard state.[10] For an irreversible reaction $E^{o\prime}$ cannot be defined. This is because if a reduction process is irreversible perhaps due to a subsequent chemical

reaction, then there is no reverse reaction and unity, even though it may exist, cannot be maintained. It is therefore impossible to determine the point of unity for a species in an irreversible reaction when the reactant can be transformed into product just by making a measurement on the system. Energy applied to the system in an attempt to force unity will ultimately cause either degradation of the species present or will cause the species present to find alternative pathways to new products not previously considered part of the reaction. A remedy to this dilemma will be discussed shortly.

The second term that must be considered is the standard rate constant k°. The standard rate is defined as the rate of the forward or reverse reaction rate at the formal potential.[10] Considering this relationship and remembering that $E^{\circ\prime}$ cannot be defined for an irreversible reaction, k° is then, also not strictly definable.

There is a convenient solution to this problem. If $E^{\circ\prime}$ is redefined to be the potential at which the rate is to be determined, E^{*}, then the rate constant of the reaction under consideration, k_f, would be the rate constant at this potential. So if we reconsider equation 14 and substitute the new values, the following equation results,

$$\ln i_p - \ln n = \ln(0.227FAC_o^*) + \ln k_f - \left[\left(\frac{\alpha n_{aF}}{RT}\right)(E_p - E^*)\right] \quad 15$$

Cyclic voltammetry experiments on CCl_4 in a supporting electrolyte solution done over the widest possible range of sweep rates would give a set of peak currents, i_p, and potentials, E_p. Thus a plot of ($\ln i_p - \ln n$) vs ($E_p - E^*$) would give a slope of $-\alpha n_a F/RT$ and an intercept of $\ln(0.227FAC_o^*) + \ln k_f$.

Equation 15 shows how k_f can be determined at any desired E_{ORP} from these data by utilizing electrochemical techniques. These k_f values could be correlated with k_{homo} values determined using the methodology discussed earlier. This will permit an empirical correction factor between these two types of rate constants to be determined and enable rate data obtained by these two methods to be compared. However, the recordable range of values of E_p is limited by the instrumentation and determination of k_f at the desired potential, E_{ORP}, may require extensive extrapolation of the data collected. Thus it is imperative that the instrument provide very accurate and precise peak potential and current measurements.

Instrumental Considerations

Computer controlled electrochemical instrumentation has been in existence and use for more than 20 years.[11,12] Instrumentation for investigating electrochemical phenomena

are essentially available from three sources. First, a few complete commercial instruments are available that are capable of an array of electrochemical experiments. This equipment is usually capable of interfacing directly with a personal computer thus allowing computer control of the instrument and computerized data workup. Second, commercial data acquisition equipment is available that would allow conversion of analog data to digital data, which would be stored and retrieved by computer. Also possible with this data acquisition equipment is event timing and analog output for use in controlling remote circuitry. Information from separate electrochemical instrumentation could be digitized and transferred into a personal computer via the data acquisition system. The final option is to build an entire data conversion, storage and instrumentation system. This would allow for maximum flexibility, yielding a customized and upgradable system. It would be very similar to the second option but all components would be designed and constructed from scratch.

STATEMENT OF PROBLEM

The goal of this work was to develop improved microcomputer controlled electrochemical instrumentation that would be versatile for use in a wide variety of electrochemical experiments. Initially this instrumentation was to be used in the investigation of the reduction rates of various halogenated organics. A brief investigation was eventually performed using commercial instrumentation.

Funding for this work was limited and the possibility of purchasing commercial electrochemical instruments was not an option. At the initiation of this work the option of utilizing commercial data acquisition equipment was only a remote possibility for two reasons. First, at the inception of this project, the capabilities and availability of most of this equipment was limited. Second, if the desired equipment was available the purchase cost kept it from being a workable option. The final alternative for obtaining new and improved electrochemical instrumentation was to design and construct the desired system. It was felt that it was possible and practical to develop a system, which would be as good as, if not better than, that which was commercially available. This is the avenue that was initially pursued in this study.

There have been several other instruments designed and built by the Geer group since 1974 that were used in a variety of electrochemical investigations.[7,13] The most

recent instrument was an Apple IIe based cyclic voltammeter. This instrument greatly improved the capabilities of the Geer group to record and store data but the instrument was cumbersome, data retrieval was time consuming and tedious and overall the instrument was inflexible with respect to its upgradeability.[14] Also, the overall capabilities of this instrument would have been the limiting factor based on the quality of the data that is needed to estimate degradation rates. The proposed instrument is IBM PC based, and is designed to be capable of data acquisition from a variety of electrochemical instruments. It is designed such that upgrades are not only possible but expected with each major function existing as a separate module. These modules are largely independent and can be changed without significant effect to the rest of the system.

Brief studies into the nature of aqueous reductions of carbon tetrachloride and carbon tetrabromide are of interest in this thesis. The Geer group has done extensive reduction studies on haloorganics,[5,6,7,8,15,16] but little has been done in aqueous systems concerning the rates of these reductions. It is felt that this information could eventually aid in the determination of degradation rates in the environment of a variety of halogenated organic molecules.

EXPERIMENTAL

Computer Interface and Instrumentation

For this study the instruments built by the Geer group not only lack the flexibility, but the precision and accuracy necessary. In this work, a modular computer interface was designed and built with instrumentation circuitry separate from the interface circuitry. This design however would enable future users to upgrade the system as better semiconductor devices and instrumentation designs become available. Separate instrumentation circuitry allows different instruments to be easily utilized and critical circuitry to be close to the electrochemical cell, which can aid in the reduction of noise. Also, if a new instrument or interface card is being developed the old card can still be used, minimizing the down time of the instrument. Within the interface there are currently three independent cards. These include an analog to digital converter card, a digital to analog converter card and an I/O and clock timer card. As previously mentioned the interface is connected to and controlled by an IBM PC clone computer. Within the computer, a decode card, which is essentially an extension of the computer bus, allows the computer software to communicate with the interface and acts as a buffer for incoming data. Explanation of each of the modules and instrumentation cards follow.

Decode Card

The decode card provides a means by which information can be sent to, or received from, the remote interface. Within that realm it serves 5 essential functions. It controls address and data lines, supplies timing and control signals to the interface, provides temporary data storage and shuttles data from the interface into computer for final storage. A schematic of the decode card is shown in Figure 1. Nomenclature for bus, card and chip assignments are shown in Appendix A. The basic circuitry for the decode(control) of address lines and data lines was predesigned by the manufacturer of the card. This board was purchased from and manufactured for JDR Microdevices, catalog number JDR-PR10. The circuitry supplied by the manufacturer can be divided into three specific areas. First, four bi-directional data bus buffers control 16 bits of I/O data lines and 16 bits of memory. Second, four uni-directional buffers drive the address bus and several other control bus signals. Third, two PAL16L8's programmable chips which are programmed by the manufacturer, are used to assign the decode card an address and allow the decode card to communicate with the computer and visa versa. This decode card is preprogrammed to communicate at the hex 300 base address of the computer.[17]

Design and construction of the remaining circuits on this card were done jointly with Mr. Mike Morrison. This

was necessary since Mr. Morrison wrote the software that allows the computer to communicate with the interface and instrumentation. Communication between the computer and the decode card is a critical link and joint design was of utmost importance here. The program for running the interface is provided in Appendix B.

The circuitry that was installed onto the decode card can be divided into 3 specific areas: Port Address Enable(PAEN), first in first out 2048 x 9 bit data input buffering(FIFO) and address line buffering.

The PAEN signal line is generated by utilizing a 74LS04(quad inverter) to invert address line signals A5, A6 and A7 and to invert the computer bus signal Address Enable(AEN). These four inverted signals and address lines A8 and A9 are then input into a 74LS30(8 input and gate) to create the final signal, PAEN. This signal is later used to give each interface card a specific address allowing each interface card to be communicated with independently and enabling data transfer to and from that card. This addressing scheme will be discussed later.[18,19,20]

First in first out is a form of memory that allows data to be temporarily stored in such a manner that the first data stored is the first data removed when the computer retrieves the data from the chip.[21] This configuration allows data generated by the interface and instrumentation, which is sent to the computer, to be temporarily stored by

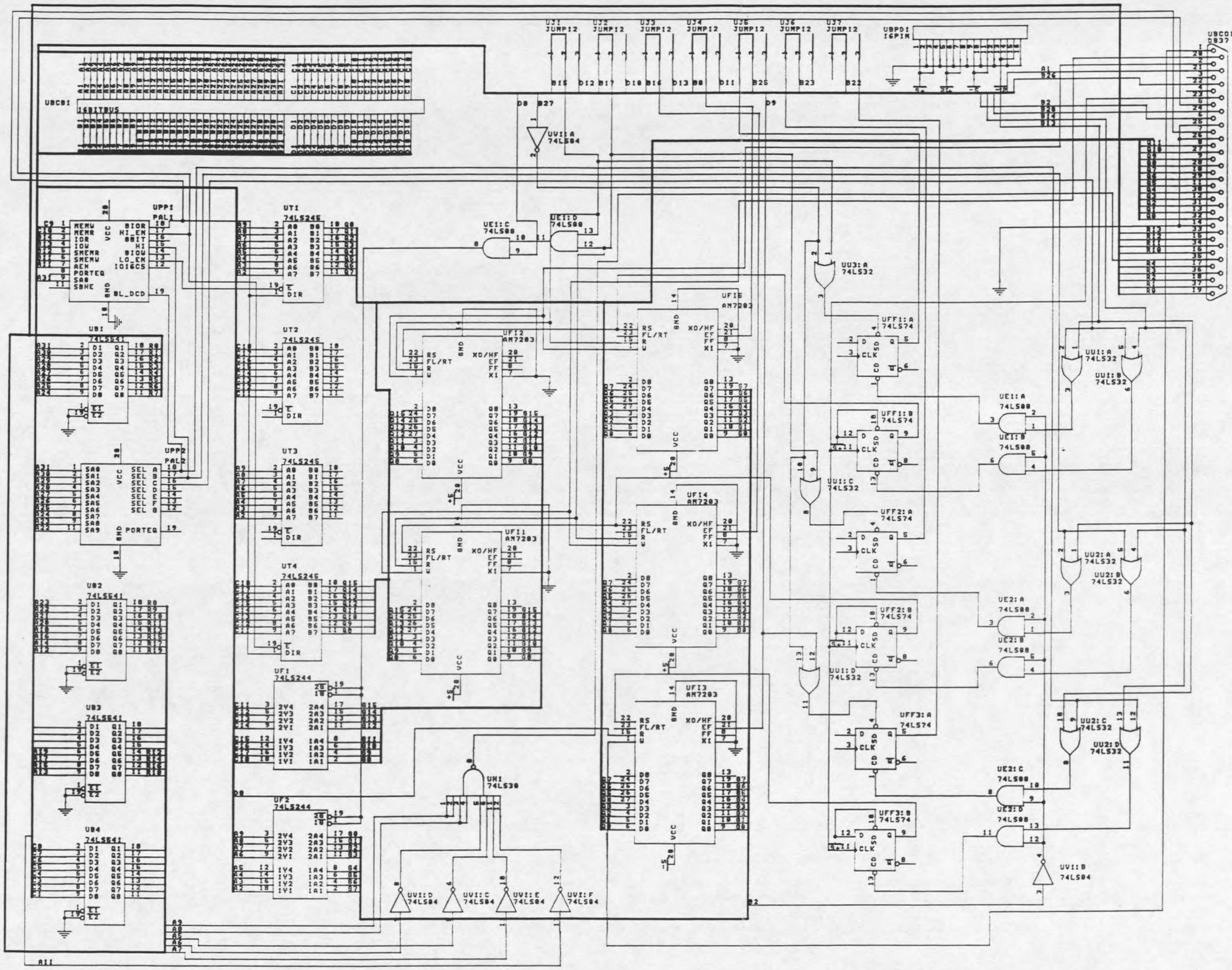

Figure 1 Schematic of the computer decode card circuitry.

the decode card. Once the analog to digital converters start converting data generated by the instrument, it is sent to the FIFO chips where it is temporarily stored. As soon as data is detected in the buffer the computer direct memory access (DMA) is activated allowing data transfer from the FIFO directly to computer memory. This process allows the fastest method of real time retrieval and storage of data with the minimal amount of extra circuitry and cost. Direct memory access bypasses the central processor and accesses the computer memory directly.[18] In this way the function of the computer and interface are independent for the retrieval and storage of data. The method of timing, storage and retrieval will be discussed shortly.

The third circuit that was necessary on the decode card is the address line buffering. Having the data and address lines in one cable between the computer and interface can cause significant noise in the data being collected. There is also no reason to have the address lines available to the interface at all times since they are only used a small portion of the time. This was accomplished by utilizing two 74LS244 bi-directional tri-state buffers. The address lines, which go to the interface, are turned on by the computer when needed and off when not in use.

Address, data and computer signals leave the decode card via a 37-pin d-sub connector. Power and ground signals leave the decode card via a 16 pin edge connector. These

connectors are attached to a 50 conductor cable which brings the signals and power lines to the interface. Here, it is connected to the interface via a 50 pin d-sub connector to the interface back-plane. The back-plane consists of six 44 pin edge connectors attached in series. It is in these slots that the interface cards are connected.

Computer Interface

The back-plane of the interface allows each interface card to be accessed independently. This independent access refers not only to the direct physical aspect of each card but also to the independent access the computer software has to each card. Each of the interface cards utilizes a circuit such that the computer can communicate with that card without interference from, or with the other cards.

Eight signals from the computer or decode card are used to achieve independent access.[18] These are A2, A3, A4, A10, A11, A12, A13 and PAEN. The circuit in which these signals are utilized is shown in Figure 2. A10 through A13 each connect to a separate input of a 74LS04 hex inverter. The four corresponding output signals from the inverter and the original non-inverted signals are attached to a series of four triple jumper blocks, with each center pin as the selected signal. The four selected signals are connected to the input of a 74LS20, a four input nand gate. The output of this gate, now defined as CRDS, along with PAEN are connected to the enable inputs of a 74LS138, a 3 to 8 line

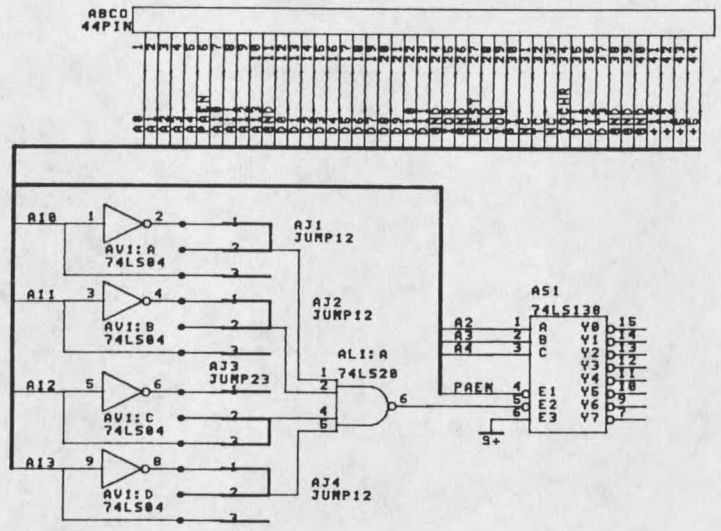

Figure 2 Schematic of the card select circuitry.

decoder. Address lines A2, A3 and A4 are connected to the three select lines of the 74LS138. If the CRDS and PAEN signals are true, high or +5 volts, then the 74LS138 is enabled. It is under this condition that a particular interface card is selected and can be communicated with independently. The state, high or low, of the three select lines is translated into one of eight possible conditions at the output of the 74LS138. These eight lines are used as chip select lines that subsequently allow a specific chip to be enabled. All of the chips on the interface cards that utilize the data and address lines of the computer have chip select ports, which must be activated in order for the chip to accept the data and address information from the computer. Again, this circuit is duplicated on each of the three interface cards. By changing the configuration of the jumpers on the jumper block, each card can be assigned a

separate address. There are 16 available addresses, allowing for up to 16 separate interface cards.

Power

There are two sources of power to the interface, one for digital circuitry and one for analog circuitry. The digital power is supplied by the computer via the decode card. This power is used by digital circuitry, which has relatively high noise immunity or where potential noise from a switching power supply would not be a problem. Low noise analog power is supplied by an external linear power supply manufactured by Standard Power Inc.. It is rated at 7.0 amps for +5 volts and 3.0 amps for +/-12 volts. An op amp was installed to provide 25 milliamps of -5 volt power. Analog power is utilized by the A/D and D/A cards in the interface and by the external instrumentation. Connection of the power supply to these devices is accomplished through the use of ribbon cable and edge connectors. Chips converting between analog and digital signals require both forms of power to function properly. Grounds are tied common between the two power supplies.

I/O Timer Card

The I/O timer card provides two main functions. First, utilizing two 82C55A CMOS programmable peripheral interface chips,[22] this card provides 48 I/O lines that can be used to interface peripheral equipment to the interface. These chips are controlled by the computer software and each of

the 48 lines can be independently manipulated. Under current design the 48 lines will only be used as logic outputs. Later design changes could utilize this chip as an input buffer device. The second function of this card, utilizing two 82C54-2 CMOS programmable interval timers,[22] provides a total of 6 independent 16-bit counters which can be used to control internal and external timed events. These chips are also controlled by the computer software and are independently programmable. A figure of the I/O time card is shown in Figure 3.

The 48 I/O lines can be used to provide logic control to other cards in the interface or to control external devices, such as stir plates, heaters and solenoids. The counter timers can be used as event timers, wave form generators, and real time clocks. Connection of the outputs of one of the I/O interface chips to other cards and to the instrument and peripherals was accomplished through the use of a 26 conductor ribbon cable and the appropriate edge connectors. This cable is currently connected to the A/D converter card and to the cyclic voltammeter instrument card. The clock chips and one of the I/O interface chips are unused at this time and connections to other devices have not been provided.

Digital to Analog Converter Card

The digital to analog (D/A) card provides one main function. It provides analog voltages, which are computer

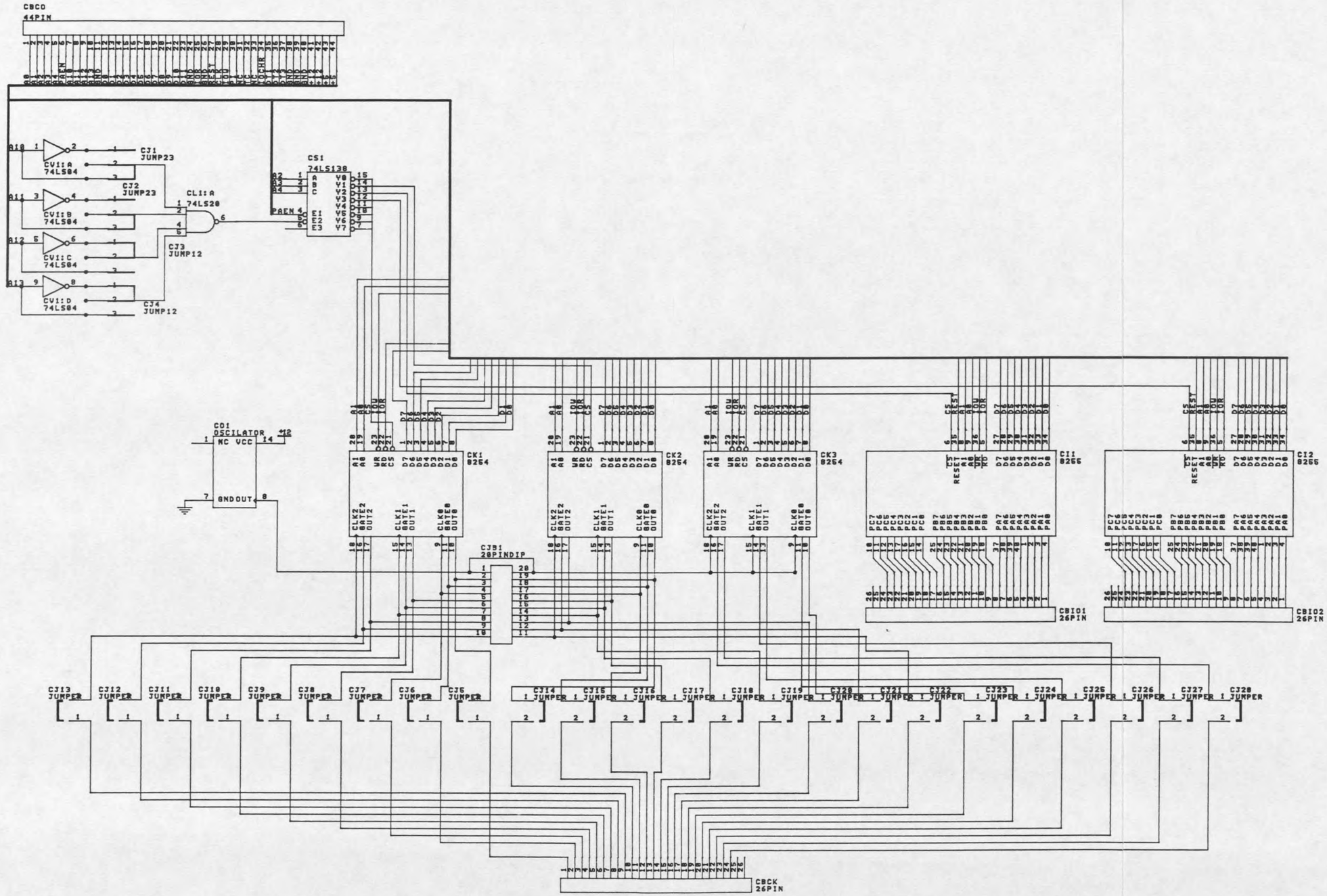

Figure 3 Schematic of the I/O timer card circuitry

software controlled, to outside instrumentation and to circuits within the interface. These voltages are provided by means of 7 DAC0830 8-bit microprocessor compatible,[23] double-buffered digital to analog converters. A schematic of the D/A card circuit is shown in Figure 4. The card is configured such that four of the output voltages are 0 volts to 2.5 volts and three of the outputs are +/-2.5 volts. The method by which the chips are used to produce these voltages was suggested in the manufacturer's literature. All 7 voltages can be independently turned on/off, changed and set via the computer software.

Probable uses for these outputs include setting of high-low set points on instrumentation cards, providing voltage for an analog sweep rate generator, as used in voltammetry, and as a source of voltages for calibration purposes. Connection of all 7 D/A outputs to other cards and to the instruments and peripherals was accomplished through the use of a 20 conductor ribbon cable and the appropriate edge connectors. The cable is currently connected to the A/D card and to the cyclic voltammeter instrument card.

Analog to Digital Converter Card

The analog to digital converter (A/D) card is the most complicated card and the most critical card in the interface. A schematic of the A/D circuitry is shown in Figure 5. The A/D card has one main function, to convert

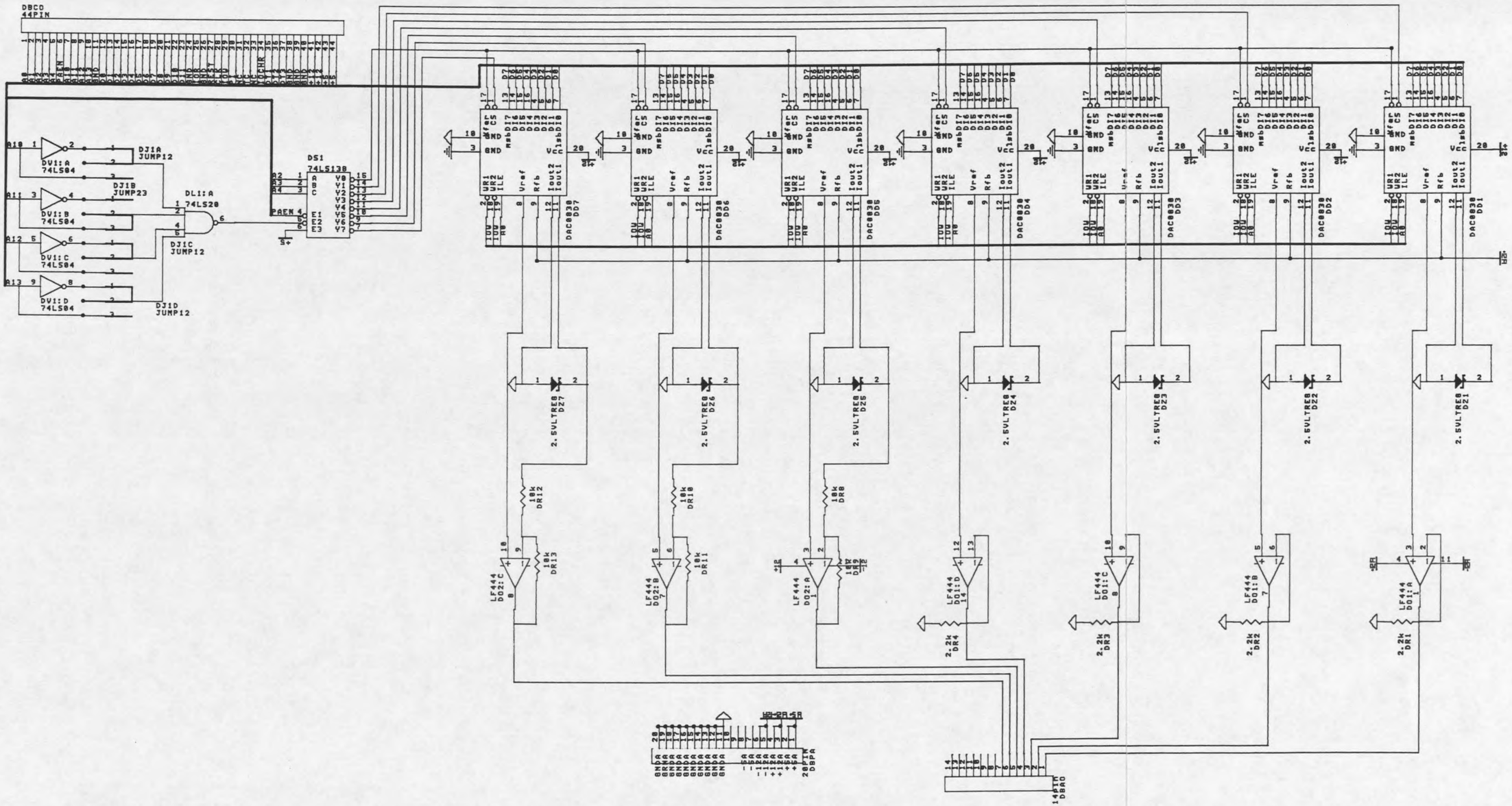

Figure 4 Schematic of the digital to analog converter card circuitry.

analog data produced by the instrument card during an experiment to digital data that can be transferred into the computer. There are, however, several other sub-functions within the A/D card that must be explained.

The analog to digital conversion is accomplished on two channels simultaneously by two ADC0820 8-bit high speed microprocessor compatible A/D converters with track/hold function.[23] There was, however a need for better resolution(12-bit) over a relatively wide voltage range(+/- 2.5 volts). Eight bits divided over 5 volts gives a resolution of 19.5mv. Utilization of 12-bits increases the resolution to 1.2mv. The use 12-bit A/D converters directly was not initially practical since it was difficult to utilize the computers capabilities of transferring 12-bits simultaneously. Transferring 12-bit data in two pieces, as is common with many 12-bit converters, was also not practical as this significantly reduced the data transfer speed, which was one major reason for construction of this interface. An auto-ranging system was designed which allowed 12-bit resolution over the entire 5 volt range to be achieved. A schematic of the auto-range circuit is shown in Figure 6.

The summarize function of the auto-range circuit: if the input voltage from the instrument goes to high or to low with respect to the set points on a high/low limit detector, a pulse sequence is initiated which causes a binary up/down

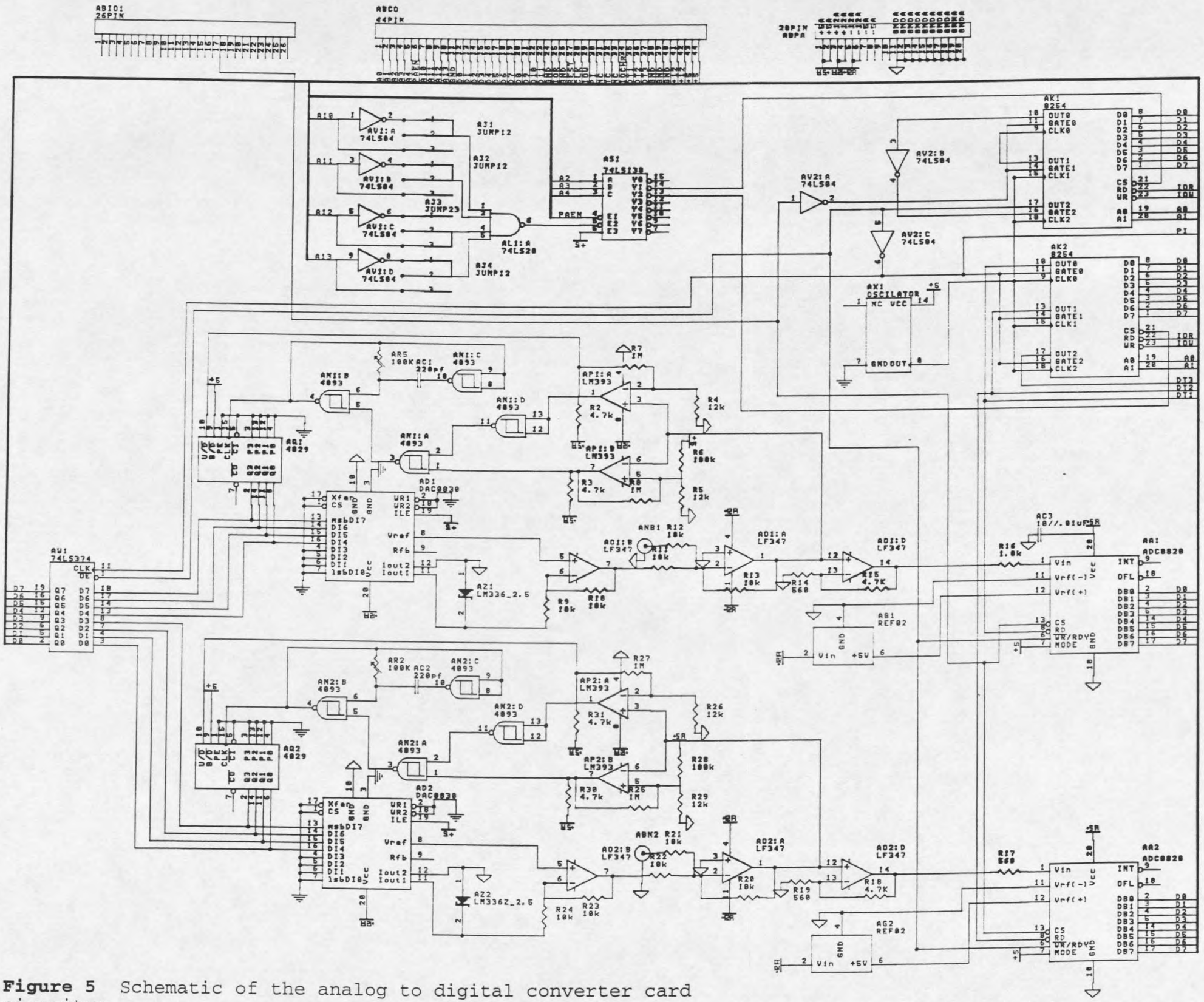

Figure 5 Schematic of the analog to digital converter card circuitry.

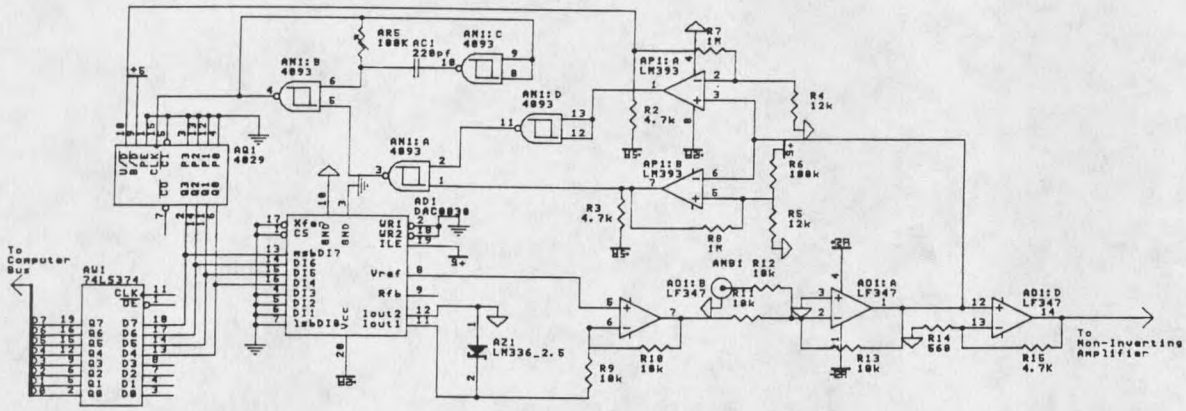

Figure 6 Schematic of the auto-range circuitry.

counter to increment or decrement a 4-bit binary nibble. A change in the nibble, increments or decrements the voltage output of a D/A converter that is subsequently summed with the input voltage. This process loop continues until the summed input voltage is within the predetermined high/low set points of the limit detector. This added significant complexity to the A/D board circuitry.

The auto-range circuit can be divided into five functional areas for which a more complete explanation follows. In the first area, the instrument signal voltage is summed with the make up voltage. See Figure 7. No amplification of the input or the auto-range signal is performed. Precision, matched resistors are utilized to

ensure signal trueness. The output of the summing amplifier is fed into the input of the high/low limit detector.

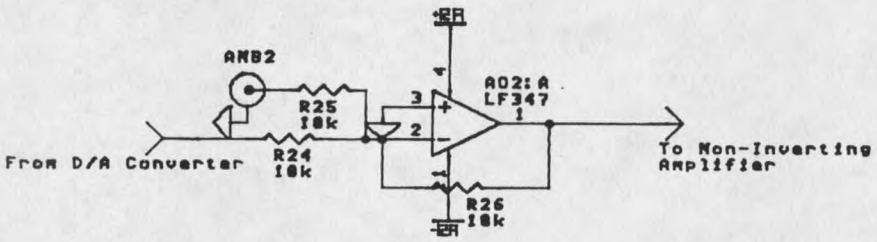

Figure 7 Schematic of summing amplifier circuitry.

The second function within the auto-range circuit is high/low limit detection. The high/low limit detector, shown in Figure 8, consists of two LM393 voltage comparitors. They monitor the output of the summing amplifier and change state when the input voltage goes beyond the preset voltages, approximately 0.0 to 0.320 volts. This portion of the circuit works on the premise that if the summed input voltage is within the set points, the output of the low side comparitor is low and the output of the high side comparitor is high, and the system is stable. If the summed voltage goes outside these set points, high or low, the state of the respective comparitor changes. These outputs are then utilized by the logic gate pulse generator in the next section. In order to inhibit oscillations within this circuit when the voltage is very near the high or low limit, 60 mV of positive feedback hysteresis is utilized.

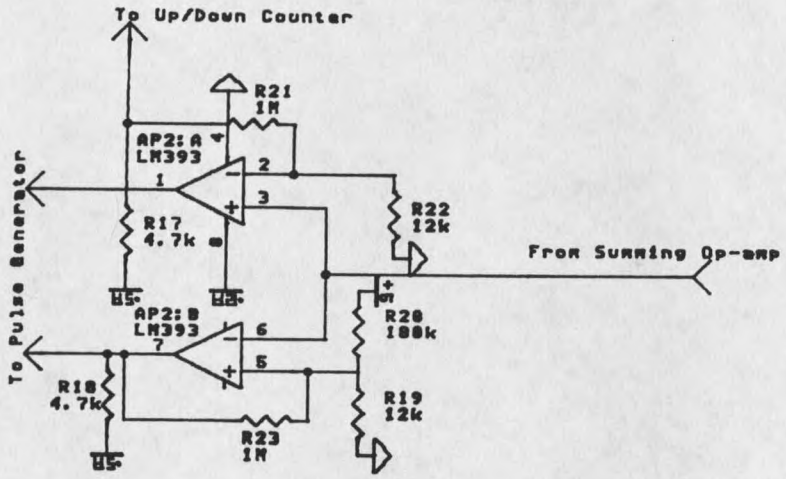

Figure 8 Schematic of limit detector circuitry.

In the third section of the auto-range circuit, if the comparitor circuit changes state then a combination of four nand gates produces a square wave pulse pattern with a period of 9.0 μs and a pulse width of 4.5 μs. This is a gated square wave generator and is shown in Figure 9. The output of this generator is utilized as the clock pulse input in the up/down counter.

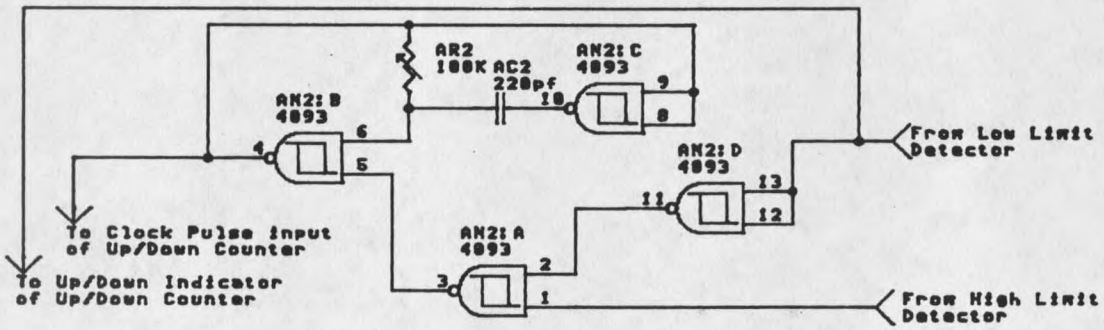

Figure 9 Schematic of the pulse generator circuitry.

The fourth function of the auto-range circuit employs a LM4029 synchronous up/down counter as a step counter which has the capability to count 16 binary units, 8 up and 8 down.[24] The counter, shown in Figure 10, counts up or down based on the logic, high or low, of the low side comparitor of the limit detector previously discussed. For each pulse of the pulse generator the counter counts up or down one binary unit. The output of the counter consists of a 4-bit nibble which produces the 16 possible combinations and is input into a digital to analog converter.

The fifth and final component, also shown in Figure 10, within the auto-range circuit utilizes the 4029 output as an input into a DAC0830 D/A converter. It is set up as

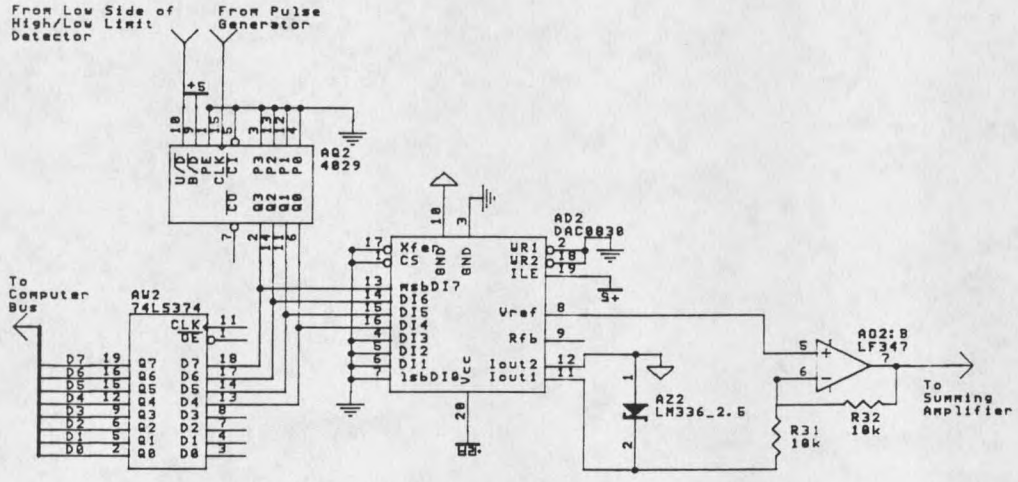

Figure 10 Schematic of the 4029 up/down counter and D/A converter circuitry. Also shown is the 74LS374 which is utilized to control computer acquisition of the output from the 4029.

recommended by the manufacturer in a bipolar output configuration with a fixed reference and a single op amp.

In this configuration it can be employed as a variable voltage source that is capable of a +/- 2.5 volts range in 312 mV increments. The output of this D/A becomes the input to the summing amplifier which was discussed first, thus completing the auto-range circuit loop. The adjustment loop continues to function until the summed voltage is within the high/low limit of the limit detector.

The output of the summing amplifier in the auto-range circuit provides an additional function. This involves preparation of the 0 to 320 mv signal for input into the A/D converter. In order to exploit the A/D to its full extent the signal input should range from 0 to 5 volts.

This smaller signal must therefore be amplified such that the 0 to 320 mv change from the auto-range circuit will be 0 to 5 volts. This is accomplished by non-inverting amplification prior to the input into the A/D converter. Thus the 320 mv signal is amplified to a 5 volt signal. This circuit is shown in Figure 11. This signal is then

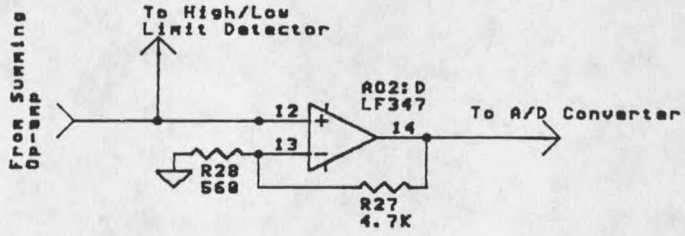

Figure 11 Schematic of the non-inverting amplifier circuitry.

used as the analog input into the ADC0820 converter which

will convert it to an 8-bit digital word that represents the 0 to 320mv signal.

The purpose of the auto-range system was to achieve 12-bit resolution using an 8-bit analog to digital converter. These extra four bits are a result of the output from the 4029 up/down counter in the auto-range circuit. The 74LS374 utilized in the auto-range circuit, see Figure 10, controls when this data is put onto the interface bus. Once the data is on the bus it can be stored in the computer. The method of timing, data generation, conversion, storage and retrieval will be discussed shortly. The 4-bit nibble is directly related to the voltage output of the D/A converter in the auto-range circuit. It represents the amount of voltage that was added to or subtracted from the instrument signal. This information is saved separately from the 8-bit A/D conversion data with which it is combined later. It should be noted here that since two data points are created simultaneously there are two 4-bit nibbles, one corresponding to each of the A/D 8-bit words. These nibbles are combined into one 8-bit word for ease of transfer into the computer and separated later when used to produce the 12-bit data.

Data Transfer and Conversion

Now that conversion of an analog signal into an 8-bit digital signal and a corresponding 4-bit nibble is possible, it is necessary to transmit that information to the computer

in a controlled fashion. Based on previous discussion it can be seen that there are three 8-bit words to transfer. Timing of data acquisition is controlled by a pair of 82C54 clock timers identical to those on the I/O clock card. Signals from the timers initiate data conversion within the A/D converter and the time between these initiations. The timers also facilitate transfer of this data from the A/D card into the computer. Time between data points can be adjusted from 5.0×10^{-6} to 537 seconds. This then makes it possible to collect a maximum of 200,000 data points per second with 1.4 mv (12-bit) resolution.

The following text describes in the briefest and clearest possible manner, the methodology used to transfer data from the A/D card to the decode card and finally to the computer. Inspiration for this methodology was acquired from an article written by Dobbs.[19]

First within the software, the number of data points to be taken is set, then based on initial, apex and ending voltages and sweep rate, the time between data points is determined. Next, the 82C54 counters on the A/D card are programmed to provide the necessary timing signals to each chip involved with data conversion and data transfer. The counters have not been initiated yet. The DMA portion of program is initiated and the DMA chip in the computer is programmed to prepare for data transfer. The computer memory is set up to prepare for arrival of data. Within the

computer this memory space is designated page 6, 7 and 8. Pages 1 through 5 are utilized by the computer for storage of programs like DOS etc. The program mode value is set and the channels are unmasked on the computer DMA chip. All of the switches on the instrument that may cause noise if sufficient settling time is not provided are set. The sweep direction of the sweep rate generator on the instrument card is checked and set. All remaining hardware functions such as switches to begin potential sweep and start counters 0,0 and 0,1 are engaged. Address lines are turned off. The I/O line from the I/O clock card changes state. This initiates the count down of counter 0,0 and 0,1. These counters are cascaded and count the amount of time between each data point. As this count reaches zero, clock 0,2 outputs a strobe to seven separate places. Count down on clock (1,0), (1,1) and (1,2) is initiated. An inverted and non inverted clock 0,2 signal is used to "open" the 74LS374 and allow the data from the auto-range circuits onto the interface bus and available to the FIFO chips. Simultaneously the A/D converters are signaled to initiate conversion of data. As each of the counters ((1,0), (1,1) and (1,2)) count down to zero a strobe causes the A/D converter to put its converted data on the bus. Simultaneously the FIFO receives the same signal, which causes it to write the data that is on the bus, at that time, into its memory. This process takes place continuously until the appropriate number of data points

have been taken at which time this particular process ceases. As currently set, timer 1,2 counts to zero in six clock cycles, clock 1,0 counts down to zero in 20 clock cycles and clock 1,1 counts to zero in 23 clock cycles, after the signal from clock 0,2 initially changes state. Each time these counters count to zero a data point is written into the FIFO memory and they are reset and wait for the next appropriate signal from clock 0,2. The total time for these three pieces of data to be written to the FIFO is 8 μs.

On the decode card other data transfer processes are taking place simultaneously. As soon as the FIFO senses data in the buffer, the empty flag(EF) output on the FIFO changes state causing a DMA request (DRQ) signal. As long as there is data in the FIFO there will be DMA requests on that channel. As a result of this request, the corresponding DMA acknowledge (DACK) signal strobes the read port of the FIFO causing it to put the data on the computer bus. Once the data is on the computer bus the DMA chip controls data transfer into computer memory. Data transfer continues on each channel until that particular FIFO is empty. The process then proceeds to the next DMA request and transfers all of the data from that channel. This process continues until all data, on all channels has been transferred. Data from an experiment has now been stored in the computer and the DMA process is complete.[19]

Upon initiating these transfers the software calculates the total amount of time for the run plus one second. The program then delays this amount of time. After this time has elapsed the address lines are turned on and the counter timers are stopped. Control of the computer is then returned to the computer software.

There are now three 8-bit words, 0 to 255 decimal, for each two data points, stored in the computer. It is necessary to convert this data to two 12-bit words, 0 to 4096 decimal. This is accomplished in the computer software. First, the 8-bit word corresponding to the auto-range data is divided into its two respective 4-bit nibbles. This is accomplished by dividing this number by 16. The decimal portion of the number is then separated from the integer portion. The decimal portion is multiplied by 16. Both remaining numbers, which will range from 0 to 15, are then multiplied by 256. These numbers, which will range from 0 to 3840, are then added to their corresponding 8-bit decimal number from each A/D conversion. The resultant numbers, which will range from 0 to 4096, can now be converted to the appropriate voltage or current which they represent.

Cyclic Voltammeter Instrument Card

The cyclic voltammeter (CV) instrument is of basic design modeled after standard instrument designs discussed in many texts[10,12,25]. This circuit consists of four major

segments. The first section is the sweep rate generator. The second section is a high/low limit detector. The third section is an uncompensated resistance adjustment circuit. The final segment is a basic circuit for a three electrode cell. Control for switches utilized in the circuit is provided by the I/O timer card in the interface. Control voltages, such as initial potential and apex voltages, are provided by the interface D/A card. A schematic of the cyclic voltammeter instrument card is shown in Figure 13.

In order for the instrument to operate properly there are five voltages which must initially be set. These are the initial potential, the final potential, high apex potential, low apex potential and lastly the sweep rate. There may be a sixth potential that is used for uncompensated resistance adjustment. The low and high apex voltages are set into a high/low limit detector which is used to determine the potential at which the scanning potential changes direction. The desired initial voltage is set and applied to the summing input of an amplifier whose output is connected to the counter electrode of the electrochemical cell. The sweep rate and direction is set by utilizing a switch selectable sign changer and an op amp integrator with seven digitally selectable RC combinations. Under this configuration the sweep rate equals $-E_{app}/RC$.[26] The output of the sweep rate generator is summed along with the initial potential setting and this voltage is applied to

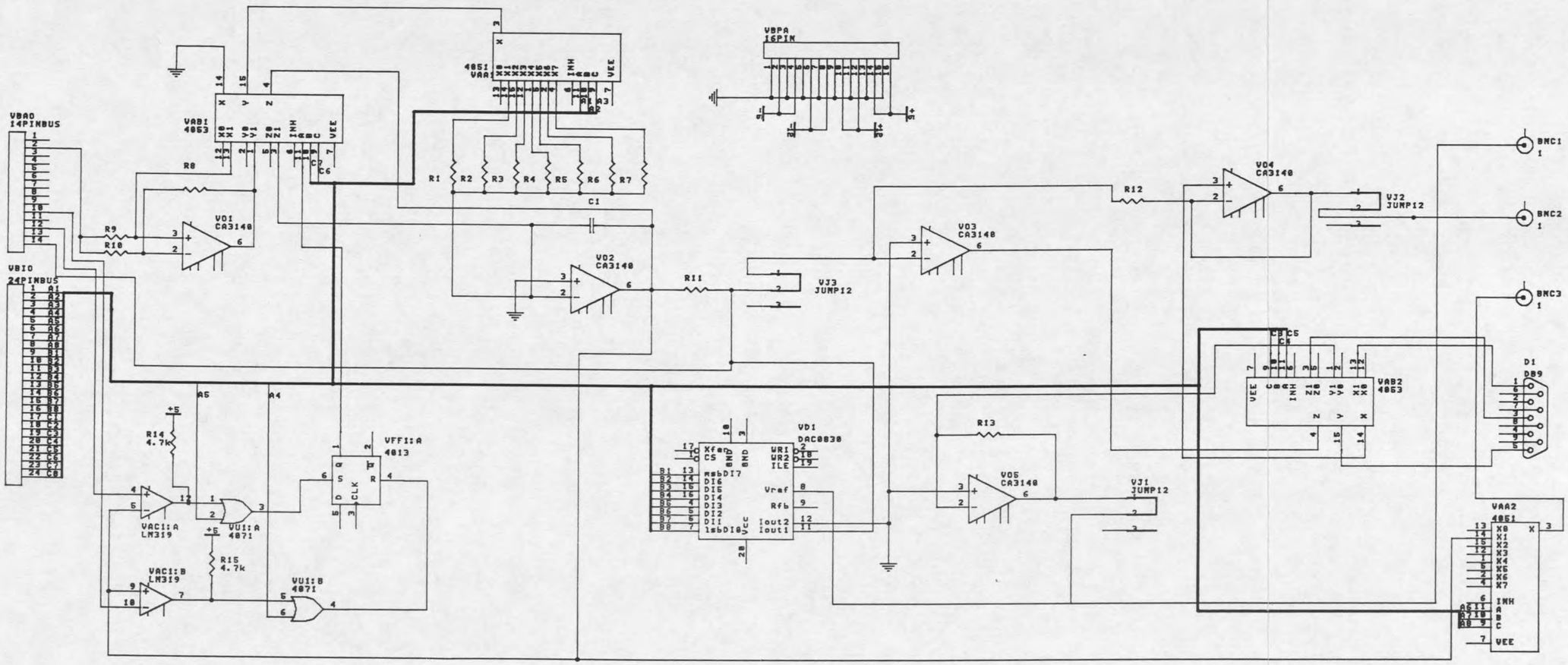

Figure 12 Schematic of the cyclic voltammeter circuitry.

the counter electrode of the electrochemical cell. The potentiometer is of basic design and is shown in Figure 13. The output of the applied potential and the output of the resultant current of the working electrode are input into the A/D converter card in the interface.

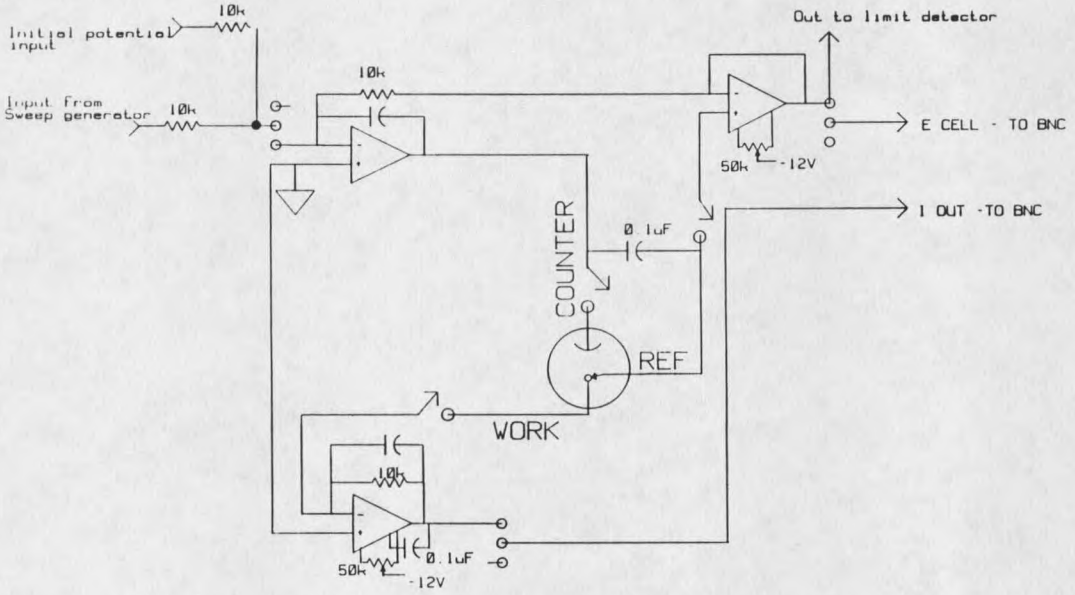

Figure 13 Potentiometer portion of the cyclic voltammeter circuit.

The uncompensated resistance section is not currently set up to function but is installed and requires only programming changes to bring it on line. It is designed to provide positive feedback and compensate for solution resistance effects often observed at higher sweep rates and in solutions with poor conductance. A small amount of cell output voltage is taken from the output amplifier of the cell and fed back into a digitally controlled DAC 0830. This software controlled D/A allows a small portion of the voltage to be converted to a current and fed back in a

positive sense to the counter electrode. This then acts as an offset voltage to compensate for the IR drop due to solvent resistance between the working and reference electrode.

Experiments Utilizing Commercial Instrumentation

Through previous contacts it was known that the Engineering Research Center at Montana State University had one EG&G PAR 273 Potentiostat and two EG&G PAR 273A Potentiostats. These commercial electrochemical instruments are capable of a wide range of electrochemical experiments and are widely used and accepted in the electrochemical community. Through the generosity of the Lewandowski group at the ERC, four months of instrument time was granted on the PAR 273A. The PAR 273A was used in conjunction with a Model 92 electometer and Model 352 Soft Corr^{tm} II corrosion measurement and analysis software. Within the software, the cyclic polarization scheme was used exclusively for running experiments since it most closely approximates a standard cyclic voltammetry regime. The built in data analysis software was used for preliminary inspection of data.

Electrochemical Cell and Electrodes

The electrochemical cell, utilized in all experiments, consisted of a 20ml scintillation vial with the cap drilled to accommodate the three necessary electrodes, a degassing tube and a vent hole. The three electrodes consisted of a

working electrode, a counter electrode and a reference electrode. The working electrode was a mercury coated platinum flag with an approximate area of 0.5 cm^2. The platinum electrode was plated with mercury using the procedure describe by Enke and coworkers.[27] The platinum flag was attached to platinum wire and then sealed in the end of 4 mm glass tube. The counter electrode was a platinum flag electrode of approximately the same area as the working electrode. Three types of reference electrodes were utilized in this study. The first reference electrode was a Ag/AgCl reference electrode, which was constructed by attaching a small piece of Porous Vycor rod, obtained from Corning Glassworks, to the end of a 5 cm piece of 4 mm glass tubing. The second reference electrode was also a Ag/AgCl electrode, it was constructed by pulling various diameters of glass or polycarbonate tubing to a fine capillary point. Tip diameters ranged from 5 to 50 µm. This design approximates the well known cracked bead type junction discussed in many texts.[12] Both of the Ag/AgCl electrodes were filled with 1-3M KCl and a AgCl coated Ag wire was inserted into the tube. The silver wire was coated with silver chloride using the procedure described by D.T. Sawyer.[12] The third reference electrode was a commercial saturated calomel electrode with a porous ceramic junction, purchased from Fisher Scientific, catalog number 1362051.

Cell volumes were typically 10 ml of electrolyte solution, usually 0.1 M KCl. The cell also contained a teflon coated spin bar. All solutions were deaerated by bubbling water saturated argon through them for 10 to 30 minutes. Argon flow over the top of the solution was continued throughout the duration of the experiment.

Determination of Working Electrode Area

Actual area of the working electrode was determined experimentally using aqueous cadmium chloride. The diffusion coefficient of Cd^{2+} is well known at 25°C and the peak current for the reduction of Cd^{2+} to Cd^{0} is determined by cyclic voltammetry. The following equation was used to determine the area:[10]

$$i_p = (2.69 \times 10^5)\ n^{3/2}\ A\ D_o^{1/2}\ \nu^{1/2}\ C_o^*$$

In order to determine the electrode area , a solution of 1.0×10^{-4} M cadmium chloride was prepared in 0.1 M potassium chloride. A 10 ml sample was deaerated with wet argon for 10 to 30 min and the temperature was maintained at 25°C. Using the three electrode system described above, the solution was subjected to a sweeping potential. The potential was swept from -0.1 to -0.8 volts vs Ag/AgCl or SCE. The starting and stopping potential was -0.1 volts, and seven different sweep rates ranging from 20 to 1280 mv/sec were utilized. Resolution was 1.0 mv/data point up to 320 mv/sec, 2 mv/data point for higher sweep rates.

Eight hundred data points were taken for each sweep sequence. Potassium chloride blanks were run under identical conditions to facilitate background subtraction. This would assure that the peak current, i_p, used to determine area was due only to the reduction of the cadmium ion. This determination was done on a regular basis to assure stability of the electrode surface and consistent results.

Study of Carbon Tetrachloride

Samples of HPLC grade CCl_4 were purchased from Fisher Scientific and were used as delivered. Amounts of CCl_4 added to the 10 ml of electrolyte varied from 0.1 to 100 microliters. Carbon tetrachloride has a solubility of 1ml in 2 1 water at 25°C.[28] Addition of 0.1 ml to a 10ml electrolyte solution of 0.1 M KCl would give a 20 fold excess of CCl_4. Samples were subjected to a potential sweep of -0.1 to -1.7 volts vs. Ag/AgCl or SCE, with the starting and stopping potential of -0.1 volts, at seven different sweep rates ranging from 20 to 1280 mv/sec. The electrolyte used throughout the course of these experiments was 0.1 M KCL. Solutions were stirred between each run for 10 seconds. All solutions were deaerated with wet argon for at least 10 minutes.

Study of Carbon Tetrabromide

Amounts of CBr_4 added to 10 ml of 0.1 M KCl electrolyte varied from .0017 to 0.010 grams. Samples of CBr_4 used were

purchased from Lancaster Chemical and were used as delivered. Samples were subjected to a potential sweep of -0.1 to -1.7 volts or -0.1 to -0.8 volts vs. Ag/AgCl or SCE, with the starting and stopping potential of -0.1 volts, at seven different sweep rates ranging from 20 to 1280 mv/sec. The electrolyte used throughout the course of these experiments was 0.1 M KCL. Solutions were stirred between each run for 10 seconds. All solutions were deaerated with wet argon for at least 10 minutes.

Investigating the Effects of Oxygen

Oxygen is a common artifact in cyclic votammograms when the solution which is being studied is not deaerated properly or sufficiently. For this study, the exact potential at which oxygen would be reduced might be valuable since there is a possibility that other species to be studied will be reduced in the same region. In order to determine this potential a series of experiments were performed in which oxygen was not degassed from an electrolyte solution. This sample was then subjected to a sweeping potential. The potential was swept from -0.1 to -1.7 volts, with the starting and stopping potential of -0.1 volts, at seven different sweep rates ranging from 20 to 1280 mv/sec.

RESULTS AND DISCUSSION

The presentation of results and discussion will be divided into two major areas. The first is a description of the attempt to construct a computer interface and associated electronics. The second area of discussion will include an explanation of the results obtained in preliminary work, utilizing commercial instrumentation borrowed from the Montana State University Engineering Research Center, for the study of the degradation of CCl_4 and CBr_4 in aqueous systems.

Interface Construction and Testing

At its inception, the interface was being designed at the state of the art. The most common microcomputer was based on the Intel 80286 microprocessor with the Intel 8088 microprocessor still quite common. Shortly thereafter the Intel 80386 microprocessor based computer was introduced and as prices began to fall it was realized that the 8-bit technology of the Intel 8088 was going to be obsolete very soon. As such, the interface was designed so that future upgrades could take advantage of the 16-bit technology in these newer systems.

Also at the beginning of the project, the literature, knowledge and the use of DMA for outside acquisition of data was very limited. DMA was originally designed to transfer data from the disk drives and other peripherals to the

processors and memory, within the computer.[18] Only by calling Intel directly were problems with DMA aided data collection advanced.

Initially, design and construction progressed at a very rapid rate. Construction of the I/O clock card and D/A card were straight forward and proceeded with only minor problems. Some difficulty was encountered when construction of the decode and A/D cards began.

Testing of circuits was performed as construction progressed but due to the integral nature of the system, complete testing was not possible until near or total completion construction of the interface circuits. Upon completion, the interface did function and was capable of acquiring data over a wide range of data transfer rates. It was possible to calibrate the D/A converters and the A/D converters. The I/O clock card functioned as expected and functioned predictably. The CV instrument card functioned as planned. However, noise and the lack of reliable data transfer represented serious problems. Whether data was collected from the CV instrument card, a sweep rate generator or from a battery the A/D, auto-range, FIFO network seemed to lose data or save extra data intermittently. See Figure 14. This most likely arose from two sources. First from noise on the signal being read which resulted in the wrong information being saved. Second, from drift or miscalculations in the timing of data

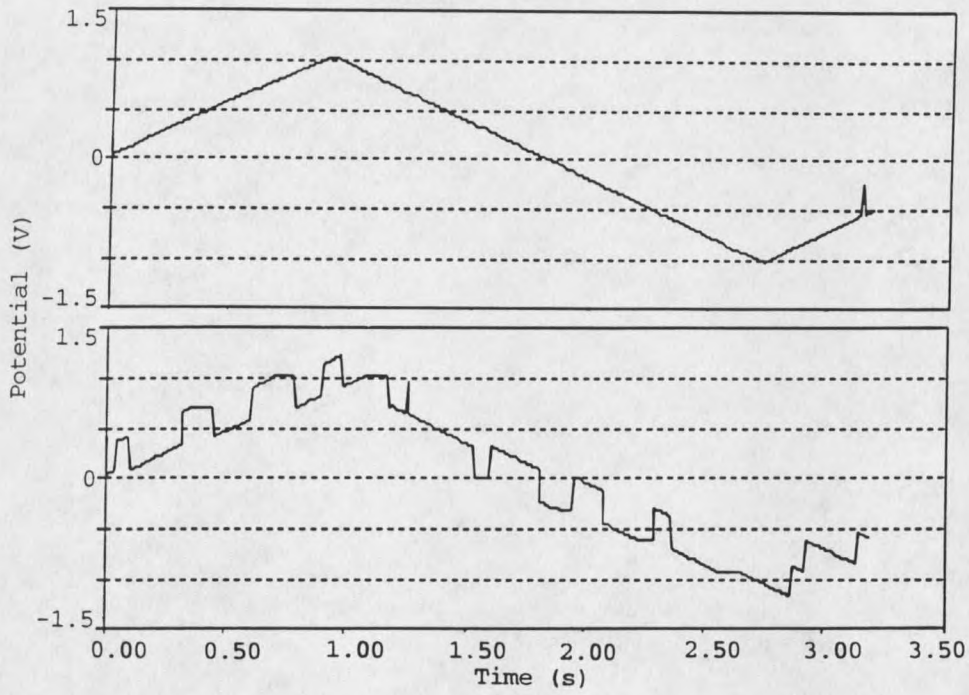

Figure 14 Example of two sets of data acquired utilizing the interface. Each represents the output of the sweep rate generator on the cyclic voltammeter instrument card.

acquisitions or data transfer which would cause information of the data bus to be read at inappropriate times. The cause of this noise or timing errors could not be traced and attempts to correct these problems proved to be futile. Most likely the number one cause of noise can be attributed to the nature of construction of all of the cards built. The method of construction used throughout was wire wrap. This is a standard method for the construction of prototypes and simple circuits. However, as the complexity of the circuit increases the probability of interference becomes significant. Printed circuit boards would, without a doubt, significantly reduce this problem. But to be clear this was

not the only problem. As a chemistry oriented group we are capable of building rather complex instruments as was demonstrated in the past. However, this interface and instrument seemed to exceed the limit of our knowledge in the area of high speed analog to digital conversion and the engineering problems associated with it.

Construction and completion of software and instruments to include other techniques such as uncompensated resistance, potentiometry, chronocoulometry, bipolar pulse conductometry and other techniques discussed were not completed since the interface did not function well enough to justify building this circuitry and writing the software code to support these techniques.

After several months of testing it was decided that outside help would be beneficial. First, we attempted to acquire the help of the Electrical Engineering Department through their Senior Project Class. Under this plan, a team of four senior Electrical Engineering students would agree to work together with an outside group or business to design a new circuit or improve an existing circuit. The team was to redesign our A/D card and improve the overall data transfer design and consider shielding techniques. Unfortunately, no team was interested in the project.

It became evident, based on testing and the inability to receive outside help, that the usefulness of the interface for the purposes of this study would be limited.

It would have been advantageous to remedy the problems with the interface but it was unmistakable that without outside assistance this would not be a timely solution. It was for these reasons that alternative methods for performing the preliminary investigation of the aqueous degradation of CCl_4 and CBr_4 were sought out.

Experimental Results

Study of Carbon Tetrachloride

Initial experiments investigated the reduction of CCl_4. Figure 15 below shows that the results of these experiments looked promising. This voltammogram represents the

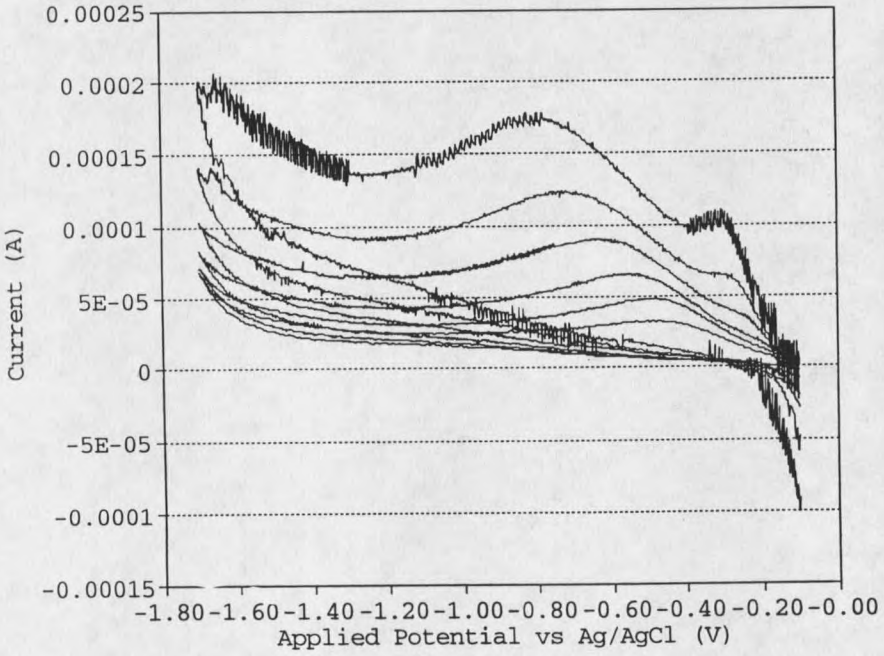

Figure 15 Voltammogram of 1µl of CCl_4 in 10 ml of 0.1 M KCl at sweep rates ranging from 40 to 1280 mv/s.

attempted electrochemical reduction of 1.0 µl of CCl_4 added to 10 ml of 0.1 M KCL at sweep rates ranging from 40 to 1280 mv/s. However, as depicted in Figure 16, subsequent experiments showed shifts in peak potentials, decreases in peak currents and sometimes total loss of all signal.

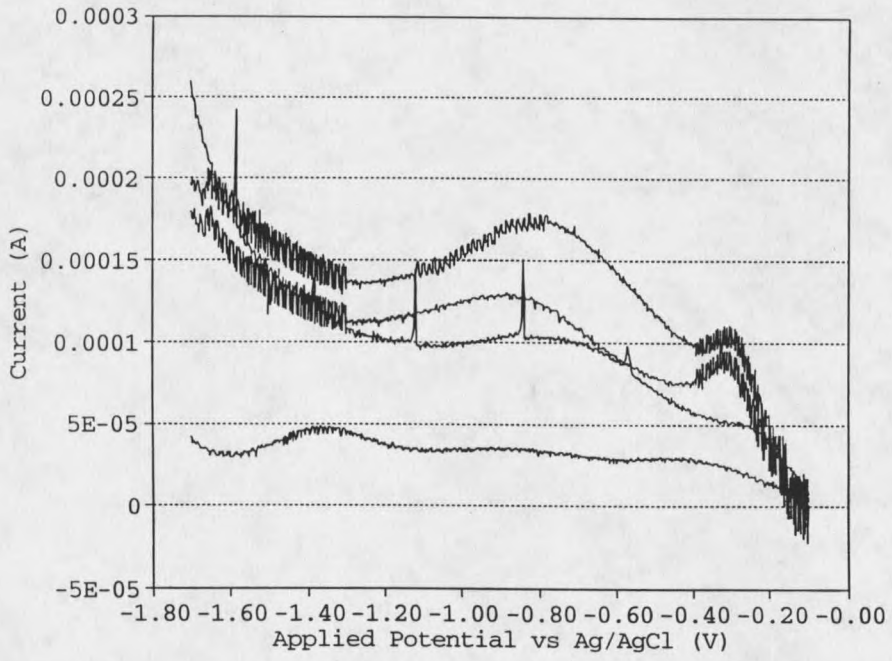

Figure 16 Voltammograms of four different sets of data utilizing Porous Vycor tipped Ag/AgCl reference electrode. All samples consist of 1µl of CCl_4 in 10 ml 0.1 M KCl at a sweep rate of 1280 mv/s.

It was determined that the main factor contributing to this inconsistency was a change in the reference electrode. It is known that Porous Vycor glass, which is used as the junction for the Ag/AgCl electrode, can be prone to interference under certain circumstances.[12] It is therefore possible that the CCl_4 was actually clogging the Porous

Vycor or inhibiting the flow of ions and therefore changed the junction resistance and hence the measured potentials. It is possible to justify this based on the limited solubility of CCl_4 in aqueous solution. The CCl_4 in solution may be drawn to the non-polar environment within the Porous Vycor and thereby affect the ability of the junction to function predictably. As a result, the use of alternative reference electrodes was investigated.

Reference Electrodes

In order to verify the malfunction of the Porous Vycor, the Ag/AgCl electrode was tested utilizing a simple potentiometric technique. In this experiment, the potential at which no current flows is measured between two electrodes. First, two Ag/AgCl Porous Vycor tipped reference electrodes were compared, one "old" electrode that had been exposed to CCl_4 and one "new" electrode that had not been exposed to CCl_4. Theoretically the potential should have been near zero volts if both electrodes were working properly. The potential measured was instead -0.173 volts. Next, measurements were made between a commercial saturated calomel electrode (SCE) and a "new" Ag/AgCl electrode and then between the SCE and an "old" Ag/AgCl electrode. A voltage of -0.045 volts was expected between the SCE and the "new" Ag/AgCl electrode. The measured potential was -0.041 volts, well with in the expected range. However, the voltage measured between the SCE electrode and

the "used" Ag/AgCl electrode was -0.01 volts. Clearly a problem existed and an alternative reference electrode would be necessary.

The SCE was tried first, although for experimental purposes the size of the SCE electrode was inhibiting while utilizing the setup described earlier. This electrode has the porous ceramic type junction and initially seemed to work for calibration purposes. However, the results that were obtained with this electrode when the reduction of CCl_4 was investigated were disappointing. Figure 17 shows the result of one such experiment. The behavior of this

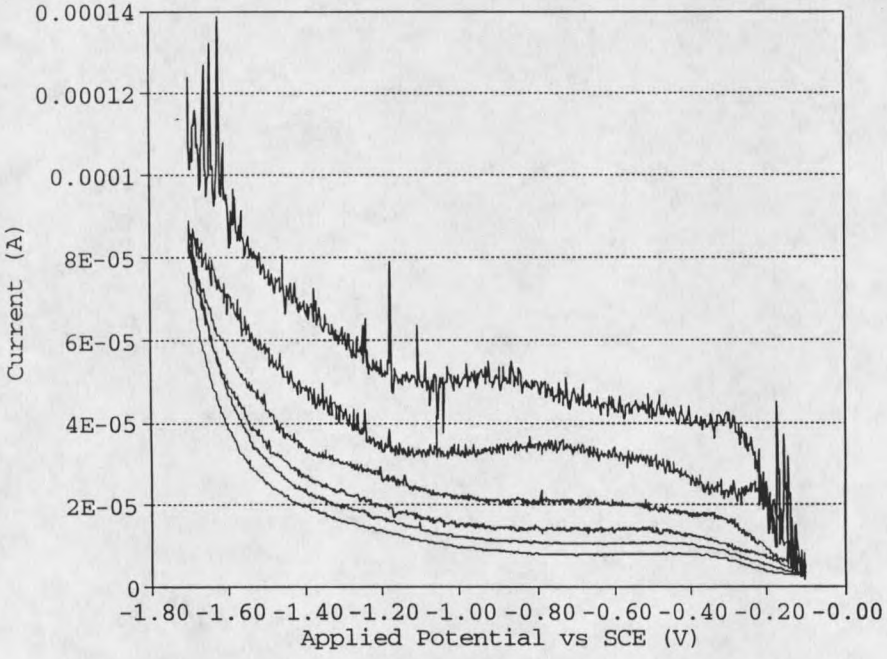

Figure 17 Voltammogram of 1.0 µl of CCl_4 in 10 ml of 0.1 M KCL utilizing SCE as the reference electrode.

electrode did not seem to be terminally affected as was the Porous Vycor tipped Ag/AgCl electrode. It just did not respond well to these conditions.

The polycarbonate or glass electrode that was manufactured by pulling a short length of tubing down to a fine capillary point proved to be troublesome to work with. After constructing several of these electrodes with varying tip diameters a brief study of their potential use in this investigation was performed.

Electrodes with smaller diameter tips worked but were very prone to clog at unpredictable times. If a small tip that did not clog could have been devised, it may have been an ideal reference to use under these conditions. Its small size would have allowed the tip to be placed close to the working electrode and ion transport using this "leak" type design would not have likely been affected by solution properties. The electrodes made of polycarbonate were quite durable and could withstand repeated washing and vacuum aspiration in attempts to clean them.

Electrodes with larger diameter tips were not prone to clog but rather "leaked" unreasonable quantities of reference electrolyte into the test solution. Large quantities of reference electrode solution added to the main solution could lead to a change in the concentration of the electrolyte in the cell. This could possibly affect the

solubility of the analyte of interest or inadvertently affect the voltammogram itself.

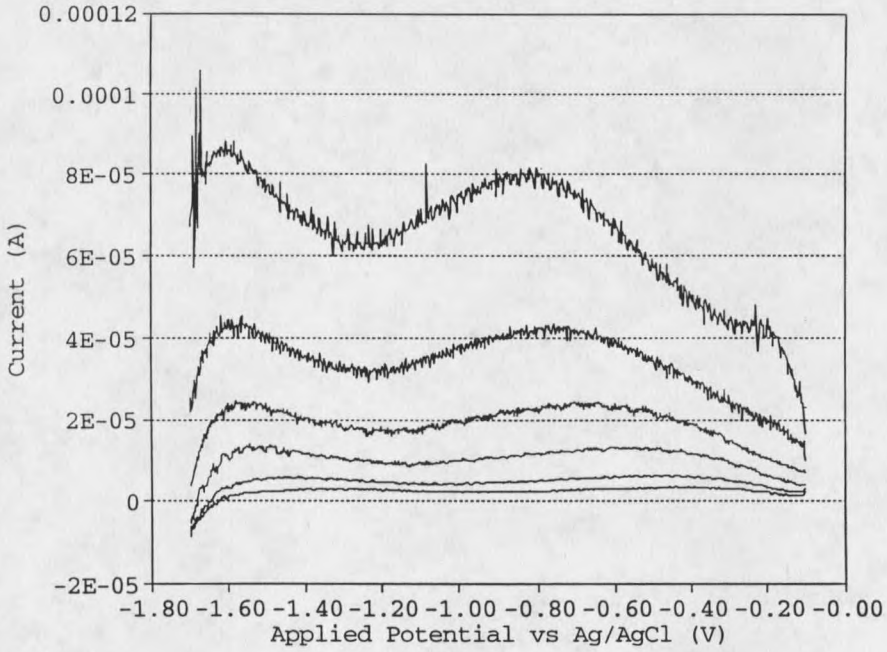

Figure 18 Voltammogram of 1.0 µl of CCl_4 in 10 ml 0.1 M KCl utilizing capillary tip Ag/AgCl reference electrode.

Study of Carbon Tetrabromide

Carbon tetrabromide (CBr_4) was studied under the conditions previously discussed. It is known that CBr_4 has a more positive reduction potential in non-aqueous systems than CCl_4.[29] Investigation showed that experiments using a Ag/AgCl Porous Vycor tipped reference electrode gave a good signal for what was initially thought to be a result of the reduction of CBr_4, Figure 19. However, this system was subject to the same interference as with CCl_4. Initial experiments showed a significant peak at approximately -0.20

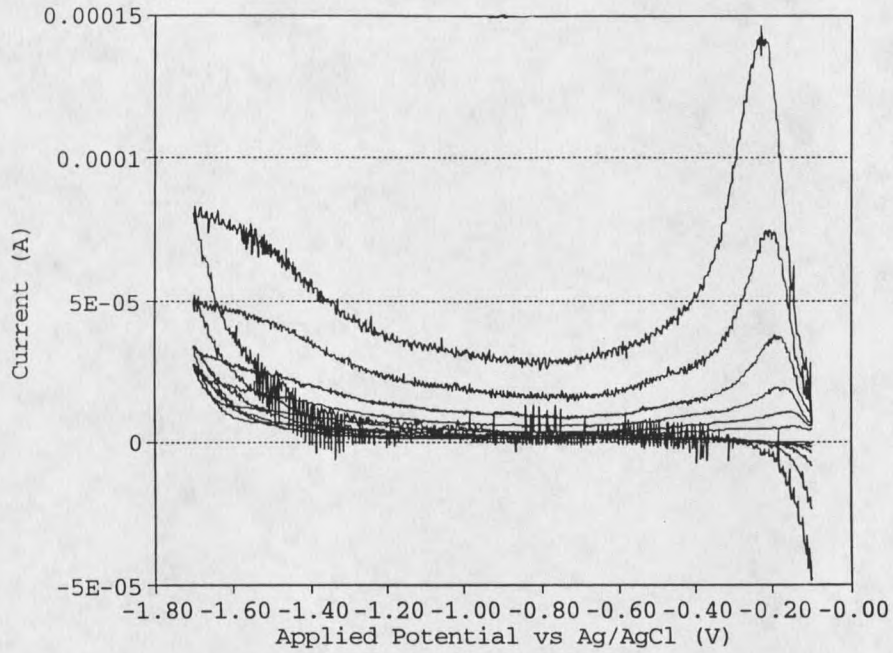

Figure 19 Voltammograms of two different samples of CBr_4 in 0.1 M KCl utilizing Porous Vycor tipped Ag/AgCl reference electrode.

volts vs. Ag/AgCl with the expected shift in peak potential to more negative potentials at higher sweep rates. However upon inspection it was determined that these peaks were likely due to the absorption and desorption of Cl^-, from the electrolyte, into and out of the mercury layer on the working electrode. Lending to this assumption, it was noticed that the current response for this system was unexpectedly strong, given the assumed limited solubility of CBr_4 in water. Also, a second peak for the reduction of bromoform was conspicuously absent.

Further studies utilizing a SCE as the reference electrode produced results similar to experiments performed employing the Porous Vycor tipped Ag/AgCl reference

electrode. Figure 20 shows that there is a progression of E_p to more negative potential at higher sweep rates as before but these signals also show the characteristics of absorbed Cl^-.

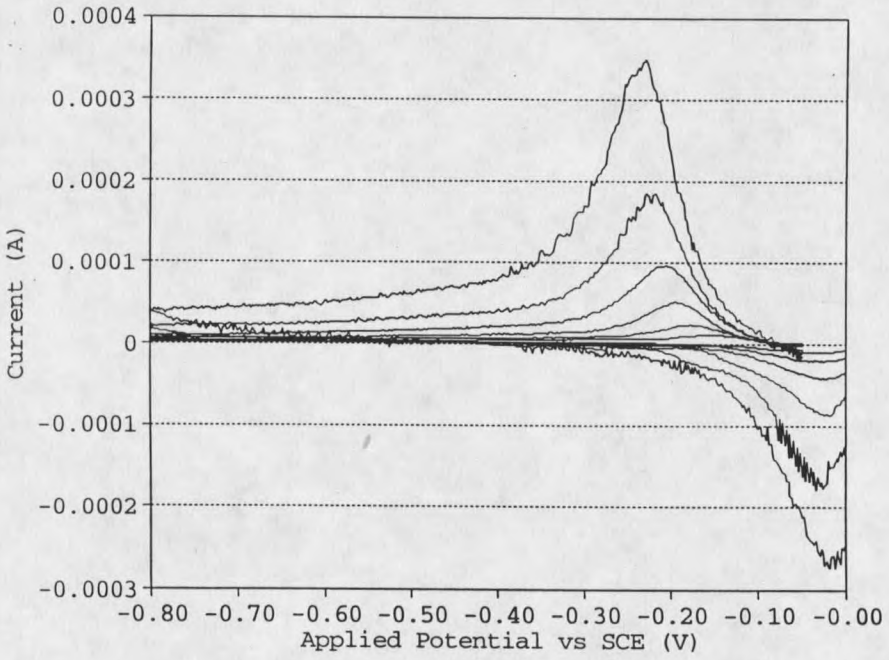

Figure 20 Voltammogram of CBr_4 in 0.1 M KCl utilizing a SCE reference electrode.

Experiments performed employing the Ag/AgCl capillary tip reference electrodes showed disappointing and inconsistent results. Figure 21 shows the nature of the data obtained. It is thought that these experiments may have been affected by the high "leak" rate of the capillary tip Ag/AgCl electrode used. There was unfortunately a limited amount of time available to continue with this portion of the investigation and no further experiments were performed.

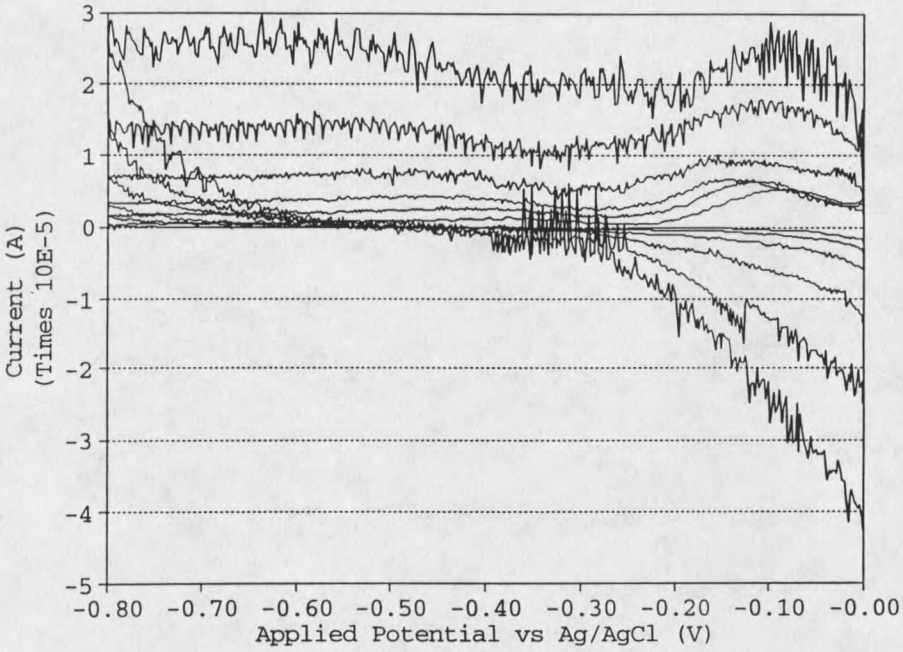

Figure 21 Voltammogram of CBr_4 in 0.1 M KCl utilizing a capillary tip Ag/AgCl reference electrode.

Preliminary Determination of Rate Constants

The carbon tetrachloride data sets did afford an opportunity to demonstrate the methodology for determining the degradation rates and the need for quality instrumentation. If the E_p and i_p data obtained from the data shown in Figure 15 is utilized as discussed in the introduction and a plot of $(E_p - E^*)$ vs $(\ln i_p - \ln n)$ is constructed, it can be seen in Figure 22 that the slopes and the intercepts that result are different.

Utilizing the values of the intercept for E_{ORP} = 0 volts for these systems, the rate constants were determined to be 3.0×10^{-8} cm^2/s and 4.5×10^{-8} cm^2/s. Evaluating the rate for the degradation of CCl_4 at E_{ORP} = 0 volts and at a concentration of 5.17×10^{-3}M would give a maximum rate of 2.4

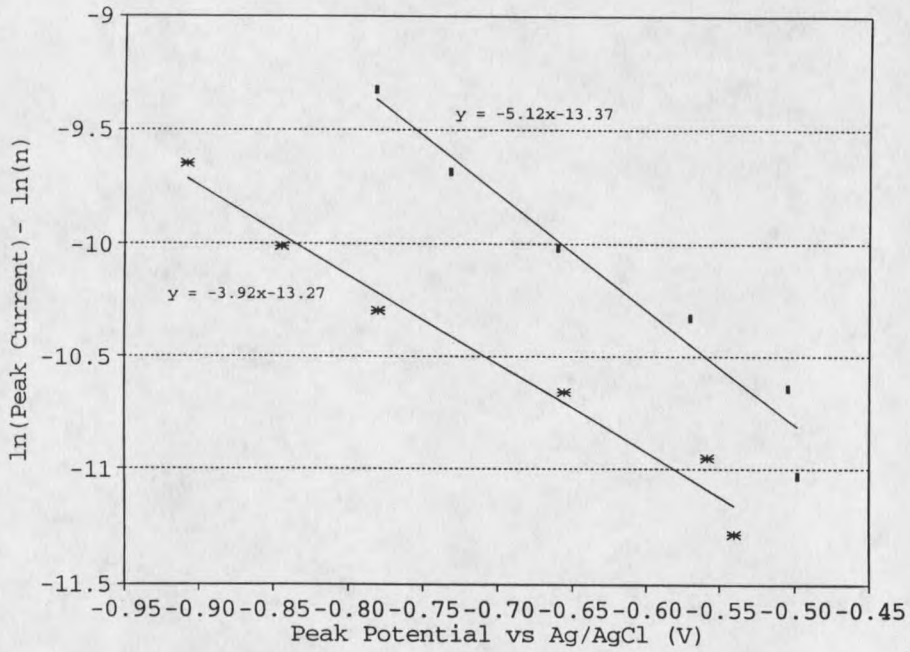

Figure 22 (E_p - E^*) vs. (ln i_p - ln n) for CCl_4 utilizing two sets of data.

x 10^{-10} mol/cm^2s^{-1} and a minimum rate of 1.6 x 10^{-15} mol/cm^2s^{-1} for these data sets. Comparing these rates to known degradation rates for other known compounds, these values seem to be well within justifiable values.

SUMMARY

Through this work it has been shown that there is a great potential for the improvement of electrochemical instrumentation. However, due to the increased complexity of the data acquisition circuits it will most likely be necessary to employ the experience of electrical engineers to further this particular interface. As alluded to briefly there is now an abundance of data acquisition hardware available in the commercial market. By utilizing this market it would be possible to refocus efforts on improving the actual electrochemical portion of the system. It is believed that the system, as designed, is viable and the overall design concept is sound. However, given time and money constrains, it is not possible to redesign the affected circuitry and complete this portion of the project. If a similar system became necessary in the future it may be more practical to purchase commercial data acquisition hardware and programs and begin the design from this stage.

Initial experiments on aqueous systems have shown that care must be taken with the choice of components used. Extreme interference of CCl_4 with Porous Vycor rendered this type of reference useless for these investigations and subsequent experiments impossible. Investigation of CBr_4 utilizing the Porous Vycor Ag/AgCl reference electrode produced some results but interference was also evident with this system.

Investigation of the reduction of CCl_4 and CBr_4 utilizing the commercial SCE reference electrode and the capillary tipped Ag/AgCl reference electrode gave mixed results as with the Porous Vycor Ag/AgCl electrode. Commercial electrodes come in several other designs and any problems with the reference could be readily solved upon further experimentation.

Due to this interference actual investigation of the degradation of CCl_4 and CBr_4 was only briefly investigated. Even with interference problems, initial results look promising. These limited sets of data allowed the concept of determination of degradation rates to be demonstrated. These brief studies also demonstrated the necessity of quality instrumentation and careful interpretation of data.

LITERATURE CITED

LITERATURE CITED

1. Geer, R.D.; Montana Agricultural Experiment Station: Annual Report of the Cooperative Regional Research Project W-45, Sept. 30, 1982.

2. Geer, R.D.; Montana Agricultural Experiment Station: Annual Report of the Cooperative Regional Research Project W-45, Oct. 1, 1979.

3. Geer, R.D.; Montana Agricultural Experiment Station: Annual Report of the Cooperative Regional Research Project W-45, Oct. 1, 1978.

4. Departmental Seminar presented by Dr. Rodney Skeen of Battelle Northwest Laboratory, Clean-up of carbon tetrachloride in Hanford ground water, 1993.

5. Beland, Frederick A.; Farwell, S.O.; Robocker, A. Elaine and Geer, Richard D. *J. Agric. Food Chem.*, **1976**, 24(4), 753-756.

6. Beland, Frederick A.; Farwell, S.O.; Callis, Patrik R. and Geer, Richard D. *J. Electroanal. Chem.*, **1977**, 78, 145-159.

7. Farwell, Sherry Owen *Interactive voltammetric instrumentation for identification of chlorinated hydrocarbons and the determination of reduction pathways for selected aromatic chlorine compounds*, Thesis for Doctor of Philosophy in Chemistry, Montana State University, 1973.

8. Byker, Harlan Jay *I. Electrochemical reduction of chlorinated nitro and nitrosobenzene in dimethylsulfoxide II. Power series representation in the high temperature phase equilibria of several metal oxide systems.*, Thesis for Doctor of Philosophy in Chemistry, Montana State University, 1979.

9. Weast, Robert C. *CRC Handbook of Chemistry and Physics*, CRC Press Inc.: Florida, 1984, p D156.

10. Bard, A.J. and Faulkner, L.R. *Electrochemical Methods. Fundamentals and Application*; Wiley & Sons New York, **1980**, Ch. 1-6.

11. Sawyer, Donald T.; Roberts, Julian L. *Experimental Electrochemistry For Chemists*; Wiley & Sons, Inc.: New York, **1974**; pp 236-281.

12. Lauer, George; Abel, Roger and Anson, Fred C. *Anal. Chem.*, 1967, 39(7), 765-769.

13. Bonsteel, Russell Allen *A micro computer controlled high speed high resolution cyclic voltammeter*, Thesis for Masters of Science in Chemistry, Montana State University, 1986.

14. O'Connor, Penny Maureen *Investigation of the structure of interfaces and transport mechanisms for adsorbed alkylsulfates*, Thesis for Doctor of Philosophy in Chemistry, Montana State University, 1991.

15. Farwell, S.O.; Beland, F.A. and Geer, R.D. *Anal. Chem.*, **1975**, 47(6), 895-903.

16. Brunke, Kathleen Elizabeth *The effects of a proton source on the electrochemical reduction of the chlorinated nitrobenzenes in dimethylsulfoxide, dimethylformamide and acetonitrile*, Thesis for Doctor of Philosophy in Chemistry, Montana State University, 1982.

17. *JDR Microdevices User's Manual, 16-bit Prototype Board with I/O Decode Logic for the IBM PC-AT*, Catalog Number JDR-PR10; JDR Microdevices, 1988.

18. Royer, Jeffery P. *Handbook of Software & Hardware Interfacing for IBM PCs*; Prentice-Hall, Inc.: New Jersey, 1987; pp 187-222.

19. Nolan, Tom *Real-Time Data Acquisition Using DMA*, Dr. Dobb's Journal, **1990**, 28.

20. Motorola Semiconductor Products Inc., *Schottky TTL Data*, 1983.

21. Advanced Micro Devices, 1989, pp 2-36 - 2-45

22. Intel Corporation, *Microprocessor and Peripheral Handbook*, Vol. II-Peripheral, 1988, pp 6-46 - 6-109.

23. National Semiconductor, *Data Acquisition Databook*, 1993, pp 3-23 - 3-40 and 2-168 - 2-183.

24. Fairchild Semiconductor *MOS/CCD Data Book*, 1975, pp 4-87 - 4-92.

25. Kissinger, P.T.; Heineman, W.R. *Laboratory Techniques in Electroanalytical Chemistry*, Marcel Dekker, New York, **1984**, Ch. 1-3.

26. Jung, Walter G. *IC OP-Amp Cookbook*, 3rd ed.; SAMS, A Division of Macmillan, Inc.: Indiana, 1986; pp 281-284, 425-440.

27. Enke, G.G.; Ramaley, L.; Brubaker, R.L. *Anal. Chem.* **1963**, *35*, 1088.

28. Budavari, Susan *The Merck Index*, 11th ed.; Merck & Co., Inc.: New Jersey, 1989; p 276.

29. Mann, Charles K. and Barnes, Karen K. *Electrochemical Reactions in Nonaqueous Systems*, Marcel Dekker, Inc.: New York, 1970, pp 201-243.

APPENDICES

APPENDIX A

Bus, Card and Chip Assignment Codes

Card Codes:

C = Clock and I/O Board
D = Digital to Analog converter board
A = Analog to Digital converter board
V = Cyclic Voltammetry instrument board
U = Computer/Decode board

Bus Codes:

Nomenclature - Four or Five symbols defining: 1, card on which the bus exists, using Card Codes above; 2, the letter B indicating that it is a bus; 3 and 4, define the function and type of bus; 5, a number representing the number of the bus if there is more than one identical bus on a card. Example: CBIO1 - Clock card, Bus, I/O, for chip 1

Codes: where "__" represents the possible Card Code symbol,

__ BPD = Power bus for digital circuits
__ BPA = Power bus for analog circuits
__ BAO = Analog output bus
__ BCO = Decode/Interface input output bus
__ BIO = Input output bus
__ BCK = Clock signal bus
__ BCB = Computer Main Bus

Chip Codes:

Nomenclature - Four symbols defining: 1, the card on which the chip exists (using the above codes); 2, the chip (codes given below); 3, the number of the chip on the card if there is more than one identical chip on the card; 4, A letter to distinguish between like components with in the same chip. Example: AV2:B = A to D card, hex inverter chip number two, inverter B (pins 3 and 4).

Codes:

A = ADC0820 Analog to Digital Converter
B = 74LS541 Octal Buffer/Line Driver
C = Capacitor
D = DAC0830 Digital to Analog Converter
E = 74LS08 Quad dual input and gate
F = 74LS244 Octal Buffer/Line Driver
G = REF02 Voltage reference
H = 74LS30 Eight input nand gate
I = 82C55 I/O
J = Jumper block
K = 82C54-2 Clock
L = 74LS20 Dual four input nand
M = 74LS85 Four bit magnitude comparitor
N = 74LS00 Quad dual input nand gate

O = Op amp
P = LM339 Quad comparitor
Q = LM4029 Up/Down counter
R = Resistor
S = 74LS138 1-of-8 decoder/demultiplexer
T = 74LS245 Octal bus transceiver
U = 74LS32 or 4071 Quad OR gate
V = 74LS04 Hex inverter
W = 74LS374 Octal d-type flip flop
X = 8 MHz Oscillator
Y = Sample and Hold
Z = LM336 Zener voltage reference
AA = 4051 8-channel analog multiplexer/demultiplexer
AB = 4053 Triple 2-channel analog multiplexer/demultiplexer
AC = LM319 Dual comparitor
FF = 74LS74 or 4013 Dual d-type pos. edge trig. flip flop
FI = AM7203 FIFO
PP = PAL16L8 Manufactures programmable chip

APPENDIX B

C++ Computer Code for Operation of Interface

```
#include<dos.h>
#include<mat4h.h>
#include<conio.h>
#include<stdio.h>
#include<stdlib.h>
#include<stdarg.h>
#include<ctype.h>
#include<graphics.h>


void clrscr(void);
void delay();
FILE *fopen(),*in_file,*test_file;
float a;
int *done,did,c,c1,c2,er,flag1,flag2,flag3,flag4,temp,r;
int PTS,BUFFER,VOLT,CURRENT,COUNTER;
long dac1,dac2,dac3,dac4,dac5,dac6,dac7,setdac;
unsigned char n1,n2,N1,N2;
long cw1,cw2,ga1,ga2,gb1,gb2,gc1,gc2;
unsigned char a1,a2,a3,a4,a5,a6,updn;
long ccw1,ct11,ct12,ct13,ccw2,ct21,ct22,ct23;
long accw1,act11,act12,act13,accw2,act21,act22,act23;
long adc3;
long int offset[11];
float ch_swr,aswr[10],t1,a,an,vtg,v,f1,x;
int
h_dl,l_dl,init,c,c1,c2,i_o1[4],add[5],s,dn,hv,lv,hd,ld,f,f2,
PTS;
char f_nam[22];
char mtf[161],ch1[1],pau;
char *y,*w;
unsigned char far tp1,tp2,tp3;
unsigned char swrv[10],swrng[10],tn1,tn2;
float cal[16][3],x1;
long float
def[32],ip[10],hl[10],ll[10],sr[10],gv[10],rs[10];
FILE *c1;
FILE *de,*out;
void gotoxy(int x,int y);
float g[24][3];
float ax[25];
float r1,w1,u1,tm[10];
int j,l;
char *f_name[1];
int s1[5];

main()
{
```

```
  dac1=2821;dac2=2825;dac3=2829;dac4=2833;
  dac5=2837;dac6=2841;dac7=2845;setdac=2816;
  cw1=3843;ga1=3840;gb1=3841;gc1=3842;
  cw2=3847;ga2=3844;gb2=3845;gc2=3846;
  ccw1=3851;ct11=3848;ct12=3849;ct13=3850;
  ccw2=3855;ct21=3852;ct22=3853;ct23=3854;
  accw1=4871;act11=4868;act12=4869;act13=4870;
  accw2=4875;act21=4872;act22=4873;act23=4874;
  adc3=1800;
  VOLT=28672;
  CURRENT=32768;
  COUNTER=24576;
  BUFFER=0;
  N1=84;N2=160;
  flag1=0;flag4=0;pau='N';temp=0;

initial();
main_menu();
return 0;
}
/* initialization program for the interface */
initial()
{
  outportb(772,0); /*open address and i/o lines out to the
        interface */
  /* intitialize all dacs to 0 volts */
  outportb(dac1,0);
  outportb(dac2,0);
  outportb(dac3,0);
  outportb(dac4,0);
  outportb(dac5,140);
  outportb(dac6,115);
  outportb(dac7,128);
  outportb(setdac,0);

  /* initialize the clocks and I/O chips */
  /*initialize all I/O chips to mode 0, out high */
  /* I/O port  1 */

  outportb(cw1,128);
  outportb(ga1,128);    /* set to have all 4051 switches open
          and set */
  outportb(gb1,0);      /* ga3 and ga4 lo for direction set*/
  outportb(gc1,160);    /* set to have pc7 at 0 to insure
          clocks are stopped and to insure
          that all switches on cell and sw
          gen are open except sw3 which is
          closed. */
  /* I/O port 2 */
  outportb(cw2,128);
  outportb(ga2,255);
  outportb(gb2,255);
```

```
  outportb(gc2,255);
  set_arrays();
  r=64000;
  return 0;
}
int getch(void);
void_exit(int status);

limit_chkf(float hlm,float llm,int xval,int yval,int zval)
{
  if ((f1<=hlm) && (f1>= llm))
    {
    def[zval]=f1;
    }
    else
    {
     gotoxy(yval,xval);
     printf("The Range is %e to %e.",llm,hlm);
     delay(5000);
    }
return (0);
}
limit_chki(int hlm,int llm,int xval,int yval,int zval)
{
  if ((f1<=hlm) && (f1>= llm))
    {
    def[zval]=f1;
    }
    else
    {
     gotoxy(yval,xval);
     printf("The Range is %d to %d.",llm,hlm);
     delay(5000);
    }
return (0);
}
float fnd_t1()
{
  /* to find the time between data points for the CV sweep
   calibration routine */
  float tswr;
  v=(((cal[0][2]-cal[0][0])/256.0)*f2);
  tswr=v/(def[c]*def[c+3]); /* to find the theoretical
              sweep rate */
  t1=0.05/tswr;           /* to find time between data
           points */
  return tswr;
}
main_menu()
{
    char cmd;
   int *p;
```

```
    *done=0;
   pt_main_mess();
   do {
  cmd=toupper(getch());
  switch (cmd) {
      case 'A': cal_menu(); break;
      case 'B': cv_menu(); break;
      case 'C': bicon_menu(); break;
      case 'D': amp_menu(); break;
      case 'E': chro_menu(); break;
      case 'Q': clrscr(); _exit(0);break;
      default: error_mess();  break;
      }pt_main_mess();
  } while (!*done);
  return 0;
   }
cal_menu()
{
   char cmd0;
   pt_cal_mess();
   did=1;
   do {
       cmd0=toupper(getch());
       switch (cmd0) {
         case 'A': all();break;
      case 'B': dacs();break;
      case 'C': adcs();break;
      case 'Q': did=0;break;
      default: error_mess(); break;
      }pt_cal_mess();
     } while (did);
     initial();
     return 0;
}
cv_menu()
{
   char cmd1;
   pt_cv_mess();
   did=1;
   do {
       cmd1=toupper(getch());
       switch (cmd1) {
      case 'A': main1();did=0;break;
      case 'B': step();did=0;break;
      case 'Q': did=0;break;
      default: error_mess(); break;
     }
     } while(did);
     return 0;
}
bicon_menu()
{
```

```
    char cmd2;
    pt_bicon_mess();
    did=1;
    do {
        cmd2=toupper(getch());
        switch (cmd2) {
         case 'A': amp();break;
       case 'B': volt();break;
       case 'Q': did=0;break;
       default: error_mess(); break;
       } pt_bicon_mess();
      } while(did);
      return 0;
}
amp_menu()
{
    char cmd3;
   pt_amp_mess();
   did=1;
   do {
        cmd3=toupper(getch());
        switch (cmd3) {
          case 'A': ed();break;
     case 'B': did=0;break;
      default: error_mess(); break;
      } pt_amp_mess();
     } while(did);
     return 0;
}
chro_menu()
{
   clrscr();
   printf("not done yet\n");
   delay(3000);
   return 0;
}
all()
{
   clrscr();
   cal_dacs(in_file);
   cal_adcs(in_file);
   mess_cal();
   return 0;
}
dacs()
{
  cal_dacs(in_file);
  mess_cal();
  return 0;
}
adcs()
{
```

```
  cal_adcs(in_file);
  mess_cal();
  return 0;
}
step()
{
   clrscr();
   chro_menu();
   return 0;
}
amp()
{
   clrscr();
   chro_menu();
   return 0;
}
volt()
{
   clrscr();
   chro_menu();
   return 0;
}
ed()
{
   return 0;
}

/* calibration of the inteface section */
cal_dacs()
{
   /* program section to calibrate the interface
    designed by the Geer group 1991 */
   char cmd4;
   *f_name="calib.dat";
   done=0;
   file_in();
   do
   {
    done=0;
    clrscr();
    printf("DAC CALIBRATION PROGRAM\n");
    printf("DO YOU WANT TO CALIBRATE ALL (A) DACS OR
       JUST ONE (O). EXIT (E): ");
      cmd4=toupper(getch());
    switch (cmd4)
     {
     case 'A': dac_all(); break;
     case 'O': dac_one(done); break;
     case 'E': pt_cal_mess(); break;
     default : error_mess(); *done=1; break;
     }
      }while (done);
```

```
    wr_file();
    fclose(in_file);
    pt_main_mess();
    return 0;
}
dac_all()
{
   int i;
   s1[3]=2821;
   n1=128;
   n2=255;
   clrscr();
   printf("CALIBRATION OF ALL THE DACS\n");
   printf("ASSUMING ALL THE DACS ARE 8 BIT. TO CALIBRATE A
     12 BIT DAC USE THE SINGLE DAC PROGRAM.");
   for (i=0;i<7;++i)
   {
    c=i;
    get_dac_dat();
    c2=i+1;
    f2=1;
    l=1;
    ud_array();
    s1[3]=s1[3]+4;
    }
   printf("\n COMPLETED CALIBRATION OF DACS.");
   delay(3000);
   return 0;
}
dac_one(done)
{
  char dac_num,bit_num;
  int bit;
  done=0;
  clrscr();
  s1[4]=1;
  printf("SINGLE DAC CALIBRATION PROGRAM.");
  printf("\nENTER THE DAC NUMBER (1 thru 7): ");
  do{
     dac_num=getch();
    switch (dac_num)
    {
    case'1': s1[3]=2821; bit=1; c=0; break;
    case'2': s1[3]=2825; bit=2; c=1; break;
    case'3': s1[3]=2829; bit=3; c=2; break;
    case'4': s1[3]=2833; bit=4; c=3; break;
    case'5': s1[3]=2837; bit=5; c=4; break;
    case'6': s1[3]=2841; bit=6; c=5; break;
    case'7': s1[3]=2845; bit=7; c=6; break;
    default: err3_mess(); dac_one(); break;
    }
     }while(done);
```

```
   printf("\nIS THIS AN 8 OR 12 BIT DAC? ");
   do{
      bit_num=(getch());
      switch(bit_num)
      {
      case'8': s1[0]=0;
           n1=128;
           n2=255;
           break;
      case'1': s1[0]=0;
           n1=2048;
           n2=4096;
           break;
      default: err4_mess();
           dac_one();
           break;
     }
      }while(done);
   s1[4]=bit;
   get_dac_dat();
   c2=c+1;
   f2=1;
   l=1;
   ud_array();
   printf("\nCOMPLETED SINGLE DAC CALIBRATION.");
   delay(3000);
   return 0;
}
get_dac_dat()
{
   float f1;
   int z;
   z=c+1;
   clrscr();
   printf("ENTER THE MEASURED VOLTAGE VALUE FOR EACH DAC
      OUTPUT AS REQUESTED.");
   outportb(s1[3],0);
   outportb(2816,0);
   printf("\n\n\nENTER THE ZERO VOLTAGE OUTPUT
      FOR DAC %d: ",z);
   scanf("%e",&f1);
   ax[0]=f1;
   outportb(s1[3],n1);
   outportb(2816,0);
   printf("\nENTER THE MIDDLE VOLTAGE OUTPUT: ");
   scanf("%e",&f1);
   ax[1]=f1;
   outportb(s1[3],n2);
   outportb(2816,0);
   printf("\nENTER THE HIGH VOLTAGE OUTPUT: ");
   scanf("%e",&f1);
   ax[2]=f1;
```

```
  return 0;
}
cal_adcs()
{
  char cmd5;
  ld_def();
  *f_name="calib.dat";
  file_in();
  clrscr();
  printf("ADC CALIBRATION PROGRAM.");
  printf("CALIBRATES ALL ADCs AT ONCE.");
  adc_all();
  wr_file();
  fclose(in_file);
  return 0;
}
adc_all()
{
  clrscr();
  printf("CALIBRATION OF ALL ADCS.");
  for(c=7;c<9;++c)
     {
     get_adc_dat();
     }
   printf("\nCOMPLETED CALIBRATION OF ADCS.");
    delay(3000);
   return 0;
}
get_adc_dat()
{
  float f1;
  int n,m;
  clrscr();
  printf("ENTER THE MEASURED VOLTAGE VALUE FOR EACH ADC
     INPUT AS REQUESTED.");
  printf("\n\nHOOK UP ALL THE ADCs TO -2.5 VOLTS.");
  printf("\nENTER VOLTAGE: ");
  scanf("%f",&f1);
  ax[0]=ax[9]=ax[18]=f1;
  n=3;m=6;
  dat(n,m);
  f2=3;
  clrscr();
  printf("\nHOOK UP ALL THE ADCs TO GROUND.");
  printf("\nENTER VOLTAGE: ");
  scanf("%f",&f1);
  ax[1]=ax[10]=ax[19]=f1;
  n=4;m=7;
  dat(n,m);
  clrscr();
  printf("\nHOOK UP ALL THE ADCs TO +2.5 VOLTS ");
  printf("\nENTER VOLTAGE: ");
```

```
  scanf("%f",&f1);
  ax[2]=ax[11]=ax[20]=f1;
  n=5;m=8;
  dat(n,m);
  c=7;
  c2=15;
  ud_array();
  return 0;
}
ud_array()      /* up dates the array after changes have been
       made */
{
  int i;
  i=0;
  for (c=c;c<c2;++c)
  {
  for (c1=0;c1<3;++c1)
  {
   u1=ax[i];
   g[c][c1]=u1;
   i=i+1;
  }
  }
  return 0;
}
dat(n,m)    /* function to collect the calibration
      data for the ADCs */
{
  int q,q1;
  int offset;
  int t[20],t1[20],tn1;
  float tn,kf;
  unsigned char tn2,tn3;
  long ki;
  ki=0;
  kf=0;

  /* ADC 3 */
  for (q=0;q<20;++q)        /* aquires 20 readings from the
            adc and averages them*/
  {
   outportb(adc3,0);
   delay(1);
   t[q]=inportb(adc3);
   ki=ki+t[q];
  }
  kf=ki;
  ax[n+18]=kf/20;
  ki=0;
  for (q=0;q<20;++q)       /* finds the standard deviation */
  {
   ki=ki+(ax[n+18]-t[q]);
```

```
}
kf=ki;
ax[m]=kf/19;

/* ADC 1*/
   /* parameters to set the DMA chip */
PTS=20;
N1=84;
N2=160;
BUFFER=0;
tm[flag2]=0.001;
counter_su();
DMA();
offset=0x0000;
ki=0;
for (q=0;q<20;++q)        /* aquires 20 readings from the
          adc and averages them*/
{
 tn2=peekb(0x7000,offset);
 tn3=peekb(0x6000,offset);
 t[q]=tn2;
   t1[q]=tn3;
 tn=t1[q];
 tn=tn/16;
 tn1=tn;
 tn=tn-tn1;
 tn=tn*16*256;
 t[q]=tn+t[q];
 ki=ki+t[q];
 ++offset;
}
kf=ki;
ax[n]=kf/20;
kf=0;
for (q=0;q<20;++q)       /* finds the standard deviation */
{
 kf=kf+(ax[n]-t[q]);
}
ax[m]=kf/19;

if (n==3)
{
 fclose(test_file);
 test_file=fopen("test1.val","w");
 fprintf(test_file,"%d \n",c);
 for (c1=0;c1<20;++c1)
  {
   fprintf(test_file,",%d, \n",t[c1]);
  }
}
else
{
```

```
 fclose(test_file);
 test_file=fopen("test1.val","at");
 fclose(test_file);
 test_file=fopen("test1.val","at");
 fprintf(test_file,"%d \n",n/3);
 for (c1=0;c1<20;++c1)
  {
   fprintf(test_file,",%d,\n",t[c1]);
  }
}
fclose(test_file);
/* ADC 2 */

offset=0x0000;
ki=0;
for (q=0;q<20;++q)        /* aquires 20 readings from the
        adc and averages them*/
{
 tn2=peekb(0x8000,offset);
 t[q]=tn2;
 tn=t1[q];
 tn=tn/16;
 tn1=tn;
 tn1=tn1*256;
 t[q]=tn1+t[q];
 ki=ki+t[q];
 ++offset;
}
kf=ki;
ax[n+9]=kf/20;
kf=0;
for (q=0;q<20;++q)       /* finds the standard deviation */
{
 kf=kf+(ax[n+9]-t[q]);
}
ax[m+9]=kf/19;
if (n==3)
{
 fclose(test_file);
 test_file=fopen("test2.val","w");
 fprintf(test_file,"%d \n",n/3);
 for (c1=0;c1<20;++c1)
  {
   fprintf(test_file,",%d, \n",t[c1]);
  }
}
else
{
 fclose(test_file);
 test_file=fopen("test2.val","at");
 fclose(test_file);
 test_file=fopen("test2.val","at");
```

```
    fprintf(test_file,"%d \n",n/3);
    for (c1=0;c1<20;++c1)
     {
      fprintf(test_file,",%d,\n",t[c1]);
     }
   }
   fclose(test_file);
   return 0;
}
file_in()
{
    if((in_file=fopen(*f_name,"rt")) == NULL)
    {
     err2_mess();
     cal_menu(0);
    }
    else
    {
     fclose(in_file);
     in_file=fopen(*f_name,"rt");
     rd_file(in_file);
    }
    fclose(in_file);
    return 0;
}

wr_file()  /* saves the calibration data */
{
    in_file=fopen(*f_name,"wt");
    for (c=0;c<16;++c)
    {
     for (c1=0;c1<3;++c1)
     {
      w1=g[c][c1];
      fprintf(in_file,"%e ",w1);
     } fprintf(in_file,"\n");
    }
    fclose(in_file);
    return 0;
}
rd_file()         /* gets the calibration file and
         loads it into g array */
{
   for (c=0;c<16;++c)
    {
     for(c1=0;c1<3;++c1)
     {
      fscanf(in_file,"%e",&r1);
      g[c][c1]=r1;
     }
    }
   return 0;
```

```
 }

/* CV portion of the program */

main1()
{
  done=0;
  an=0;
  ld_cal();
  ld_def();
  cv_men();
  return 0;
}
ld_cal()  /* load calibration file data */
{
  float a;
  if((cl=fopen("calib.dat","rt"))==NULL)
  {
   clrscr();
   printf("Calib.dat File Not Found.");
  }
  else
  {
   for(c=0;c<16;++c)
   {
    for (c1=0;c1<3;++c1)
    {
     fscanf(cl,"%e ",&a);
     cal[c][c1]=a;
    }
   }
  }
  fclose(cl);
  return 0;
}
ld_def()  /* load default settings for parameters */
{
  float sag;
  if((de=fopen("set.dft","rt"))==NULL)
  {
   clrscr();
   printf("SET.DFT was not found.");
   printf("\nNo default values loaded. You must set up all
      parameters before you run the CV.");
   delay(3000);
  }
  else
  {
   for (c=0;c<32;++c)
   {
    fscanf(de,"%e ",&sag);
    def[c]=sag;
```

```
    }
  }
  fclose(de);
  *f_name="def.dat";
  updn=def[31];
  set_swp();
  set_arrays();
  return 0;
}
cv_men()
{
  char cmd;
  *done=1;
  do {
    cv_men_ms();
    cmd=toupper(getch());
    switch(cmd)
    {
     case'A': sw_cal();break;
     case'B': cv_par();break;
     case'C': cv_run();break;
     case'Q': *done=0;break;
     default: err_ms();break;
    }gotoxy(46,18);
  }while(*done);
  return 0;
}
cv_par()   /* get and store the run parameters. */
{
  char cmd;
  int don,t;
  char m1[24],dump;
  find_PTS(0);
  if(flag1==0)
   {
    set_swp();
   }
  par_mes();
  don=1;
  do {
    f1=0;
    cmd=toupper(getch());
    switch (cmd)
     {
     case'A': gotoxy(23,3);
     /* gets file name */
      gets(f_nam);
      *f_name=f_nam;
      break;
     case'B': gotoxy(31,4);
     /* gets initial potential */
      scanf("%e",&f1);
```

```
 limit_chkf(2.5,-2.5,23,10,0);
 break;
case'C': gotoxy(34,5);
/* gets high limit value */
 scanf("%e",&f1);
 limit_chkf(2.5,-2.5,23,10,1);
 break;
case'D': gotoxy(33,6);
/* gets low limit value */
 scanf("%e",&f1);
 limit_chkf(def[1],-2.5,23,10,2);
 break;
case'E': gotoxy(24,7);
/* gets sweep rate */
 scanf("%e",&f1);
 limit_chkf(def[28],def[23],23,10,3);
 break;
case'F': gotoxy(30,8);
      /* gets number of sweeps */
 scanf("%e",&f1);
 limit_chki(10,1,23,10,4);
 break;
case'G': set_swr();
 break;
case'H': gotoxy(18,10);
     /* gets gain value */
 scanf("%e",&f1);
 limit_chki(6,0,23,10,5);
 break;
case'I': gotoxy(24,11);
 /* gets resolution */
 scanf("%e",&f1);
 limit_chki(100,1,23,10,6);
 break;
case'J': gotoxy(21,12);
      /* gets message */
 gets(mtf);
 break;
case'K': gotoxy(23,13);
      /* gets initial direction */
 updn=toupper(getch());
 if(updn=='U')
  {
  def[31]=updn;
  break;
  }
   else
   {
    if(updn=='D')
    {
     def[31]=updn;
     break;
```

```
        }
        else
         {
         gotoxy(28,13);
         printf("Only a U or D will
            be accepted.");
         delay(1500);
            updn=def[31];
         break;
         }
        }
case'L': gotoxy(34,14);
     /* get pause data  */
pau=toupper(getch());
if(pau=='Y')
 {
  flag4=1;
  break;
 }
 else
 {
  if(pau=='N')
  {
   flag4=0;
   break;
  }
  else
  {
   gotoxy(38,14);
   printf("Only a Y or N will be
      accepted.");
   delay(1500);
   break;
  }
 }
case'M': set_ind_swp();
   break;
case'Q': don=0;
   break;
default: err_ms();
   break;
 }
  if (flag1==0)
  {
   set_arrays();
  }

  find_PTS();
  par_mes();
  gotoxy(46,21);
}
```

```
    while(don);
    clrscr();
    printf("Updating Default File.");
    if(flag1==0)
     {
     set_swp();
     }
    sv_def();/* saves default values for later use */
    pt_main_mess();
    return 0;
}
sw_cal()
{
   /* function to calibrate the sweep rate generator */
  int don;
  char cmd1;
  don=1;
  sw_ms();
  do{
    cmd1=toupper(getch());
    switch(cmd1)
    {
    case'A': gotoxy(12,4); /* a-g are resistor values, h
                is the capacitor value */
          scanf("%e",&f1);
          limit_chkf(1e8,1,20,10,7);
          break;
    case'B': gotoxy(12,5);
          scanf("%e",&f1);
          limit_chkf(1e8,1,20,10,8);
          break;
    case'C': gotoxy(12,6);
          scanf("%e",&f1);
          limit_chkf(1e8,1,20,10,9);
          break;
    case'D': gotoxy(12,7);
          scanf("%e",&f1);
          limit_chkf(1e-5,1e-12,20,10,10);
          break;
    case'E': gotoxy(12,8);
          scanf("%e",&f1);
          limit_chkf(1e-5,1e-12,20,10,11);
          break;
    case'F': gotoxy(12,9);
          scanf("%e",&f1);
          limit_chkf(1e-5,1e-12,20,10,12);
          break;
    case'I': gotoxy(12,12); /* gets initial direction */
          updn=toupper(getch());
          if(updn=='U')
           {
           def[31]=updn;
```

```
                break;
                }
                else
                 {
                  if(updn=='D')
                  {
                   def[31]=updn;
                   break;
                  }
                  else
                   {
                    gotoxy(16,12);
                    printf("Only a U or D
                    will be accepted.");
                    delay(1500);
                    updn=def[31];
                    break;
                   }
                 }
       case'J': don=0;
                break;
       case'Q': cv_men_ms();
                return (0);
       default: err_ms();
                break;
       } sw_ms();
        gotoxy(56,18);
     } while(don);
     sv_def();
     clrscr();
     printf("RUNNING SWEEP RATE CALIBRATION
        PROGRAM.\n\nPLEASE WAIT.");
     outportb(adc3,0);
     x=23;tn1=1;
     test_file=fopen("test.val","r");
     er=7;
     for(c=7;c<10;++c)
     {
      a=0;f2=1;
      fnd_swr();
      ++x;++er;
      a=1;f2=254;
      fnd_swr();
      ++x;++tn1;
     }
     sv_def();
     fclose(test_file);
     pt_main_mess();
     return 0;
}
cv_run()
{
```

```
  char v;
  clrscr();
  flag3=1;
  BUFFER=0x0000;
  for (flag2=0;flag2<def[4];++flag2)
  {
   clrscr();
    pause();
   set_up();
   if (flag3==1)
   {
   counter_su();
   printf("SWEEP RUN %d.\n",flag2+1);
   printf("SWEEP RATE %e\n",sr[flag2]);
   DMA();
   offset[flag2]=BUFFER;
   BUFFER=BUFFER+PTS;
   }
  }
  if(flag3==1)
  {
  sv_dat();
  cv_men();
  }
  flag2=0;
  return 0;
}
set_ind_swp()
{
  for(f=0;f<def[4];++f)
  {
   cv_par_menu();
  }
  return (0);
}
cv_par_menu()
{
  char cmd;
  int don,t;
  find_PTS();
  cv_run_mess();
  don=1;
  do {
    f1=0;
    cmd=toupper(getch());
    switch (cmd)
     {
      case'A': def[29]=ip[f];
       gotoxy(31,6);/* gets initial potential */
       scanf("%e",&f1);
       limit_chkf(2.5,-2.5,20,10,29);
       ip[f]=def[29];
```

```
          break;
        case'B': def[29]=hl[f];
          gotoxy(34,7);/* gets high limit value */
          scanf("%e",&f1);
          limit_chkf(2.5,-2.5,10,10,29);
          hl[f]=def[29];
          break;
        case'C': def[29]=ll[f];
          gotoxy(33,8);/* gets low limit value */
          scanf("%e",&f1);
          limit_chkf(2.5,-2.5,20,10,29);
          ll[f]=def[29];
          break;
        case'D': def[29]=sr[f];
          gotoxy(24,9); /* gets sweep rate */
          scanf("%e",&f1);
          limit_chkf(def[28],def[23],20,10,29);
          sr[f]=def[29];
          break;
        case'E': def[29]=gv[f];
          gotoxy(18,10);    /* gets gain value */
          scanf("%e",&f1);
          limit_chki(6,0,20,10,29);
          gv[f]=def[29];
          break;
        case'F': def[29]=rs[f];
          gotoxy(24,11);  /* gets resolution */
          scanf("%e",&f1);
          limit_chki(100,1,20,10,29);
          rs[f]=def[29];
          break;
        case'Q': don=0;
          break;
        default: err_ms();
          break;
       }
          find_PTS();
          cv_run_mess();
          gotoxy(46,17);
    } while(don);
    flag1=1;
    return (0);
}
set_swr()  /* main function for sweep rate change setup */
{
  char buff;

  flag1=1;
  clrscr();
  printf("This routine is for the setup of the amount of
     change in \nsweep rate between sweeps.");
  printf("\nEnter all amounts in volts.");
```

```
  printf("\n\nDo you wish to enter each change manually? ");
  buff=toupper(getch());
  if(buff=='Y')   msw_ch();
  else sw_ch();
  clrscr();
  par_mes();
  return 0;
}
msw_ch()   /* for manual entry of the sweep rates */
{
  clrscr();
  printf("Enter the value for each change in the sweep rate
     as they will be run,(max of 10).");
  printf("Enter values in volts.");
  for(c=0;c<def[4];++c)
  {
   f2=c+1;
   printf("\nEnter sweep %d value: ",f2);
   scanf("%e",&x);
   sr[c]=x;
  }
  return 0;
}
sw_ch() /* gets the information from the operator to set
     even sweeprates */
{
  clrscr();
  printf("Enter the value of the change in the sweep rate in
     volts: ");
  scanf("%e",&f1);
  def[30]=f1;
  printf("\nCalculating and saving sweep rates.");
  sr[0]=def[3];
  for (c=1;c<def[4];++c)
  {
   f2=c-1;
   sr[c]=sr[f2]+f1;
  }
  return 0;
}

set_swp() /* calculates the sweep rates for evenly spaced
      sweeps */
{
  f1=def[30];
  sr[0]=def[3];
  for (c=1;c<def[4];++c)
  {
   f2=c-1;
   sr[c]=sr[f2]+f1;
  }
  return 0;
```

```
}
cv_cal()
{
  return 0;
}
set_up()    /* sets up the I/O ports */
{
  int seta,setb,setc;
  float ed;
  if((def[24]>sr[flag2]) && (sr[flag2]>def[23]))
   {
   seta=1;swrng[flag2]=seta;

swrv[flag2]=(((sr[flag2]-def[23])/(def[24]-def[23]))*253);
   ed=swrv[flag2];
   aswr[flag2]=((ed*(def[24]-def[23]))/253);
   }
   else
   if((def[26]>sr[flag2])&&(sr[flag2]>def[25]))
    {
    seta=2;swrng[flag2]=seta;

swrv[flag2]=(((sr[flag2]-def[25])/(def[26]-def[25]))*253);
    ed=swrv[flag2];
    aswr[flag2]=((ed*(def[26]-def[25]))/253);
    }
    else
    if((def[28]>sr[flag2])&&(sr[flag2]>def[27]))
     {
     seta=3;swrng[flag2]=seta;

swrv[flag2]=(((sr[flag2]-def[27])/(def[28]-def[27]))*253);
     ed=swrv[flag2];
     aswr[flag2]=((ed*(def[28]-def[27]))/253);
     }

     else
      {
      clrscr();
      printf("\n\n\nValue entered for sweep rate
        falls outside calibration parameters.");
      printf("\nReturn to parameter setup routine
        and choose a new sweep rate.");
      flag3=0;
      flag2=def[4];
      delay(3000);
      }
  if (flag3==1)
  {
  if (gv[flag2]==0)
   {
   setb=32;
```

```
 }
 else
 if (gv[flag2]==1)
  {
  setb=64;
  }
  else
  if (gv[flag2]==2)
   {
   setb=96;
   }
   else
   if (gv[flag2]==3)
    {
    setb=128;
    }
    else
    if (gv[flag2]==4)
     {
     setb=160;
     }
     else
     if (gv[flag2]==5)
      {
      setb=192;
      }
      else
      if (gv[flag2]==6)
       {
       setb=224;
       }


an=def[2];
condw();
l_dl=f2;
an=def[1];
cond();
h_dl=f2;
an=def[0];
cond7();
init=f2;
temp=seta+setb;
/* outportb(ga1,(char) set[c]); */
N1=84;
N2=160;
chkupdn();

fnd_tm();
output();
delay(10);
}
```

```
  return 0;
}
counter_su()
/* calculates the number of counts for
   each counter and sets them */
/* this function is for 8254 chips */

{
  int lclb,lchb,hclb,hchb;
  float cy,hc,tn;
  cy=8e6;
   *done=1;
   hc=1;
   tn=tm[flag2];
  tn=cy*tn;
  do
   {
    if (tn<65536)
    {
     hc=hc+1;
     tn=tn/2;
     *done=0;
    }
    else
    {
     do{
     if (tn>65536)
     {
      hc=hc*2;
      tn=tn/2;
     }
     else
     {
      *done=0;
     }
     }while(*done);
    }
   }while(*done);
  tn=tn/(256);
  lchb=tn;
  tn=tn-lchb;
  lclb=tn*256;
  tn=hc/256;
  hchb=tn;
  tn=tn-hchb;
  hclb=tn*256;
  outportb(accw1,52);     /* setup counter 0 as rate
            generator */
  outportb(act11,hclb);
  outportb(act11,hchb);
  outportb(accw1,146);    /* setup counter 2 as hardware
            retriggerable one shot */
```

```
  outportb(act13,8);
  outportb(accw1,116);    /* setup counter 1 as rate
           generator */
  outportb(act12,lclb);
  outportb(act12,lchb);
  outportb(accw2,26);     /* these are setup to be hardware
           triggered strobes */
  outportb(act21,23);
  outportb(accw2,90);
  outportb(act22,20);
  outportb(accw2,154);
  outportb(act23,6);

  return 0;
}
DMA()
{
  int single_mask,done,N1X;
  unsigned long int tmn,xa;
  int mode,flip_flop,mode_value[3],mode_value1[3];
  int clr_mask[3],set_mask[3];
  int pg[3],base[3],chad[3],chcnt[3]
  int pgad[3],ca[3][2],cc[3][2];
  unsigned char a,x,y,z,cz,dz,xas;
  float tn1,tn2;
  int aw[8][2],k,k1,offset,tme,did2,dw,yq,de;
  unsigned char r,k2;

  int chad[0]=2;chcnt[0]=3;chad[1]=0;
  int chcnt[1]=1;chad[2]=6;chcnt[2]=7;flip_flop=12;
  mode=11;mode_value[0]=69;
  mode_value[1]=68;mode_value[2]=71;
  mode_value1[0]=85;mode_value1[1]=84;mode_value1[2]=87;
  single_mask=10;
  clr_mask[0]=1;clr_mask[1]=0;clr_mask[2]=3;
  set_mask[0]=5;set_mask[1]=4;set_mask[2]=7;
  pgad[0]=131;pgad[1]=135;pgad[2]=130;
  pg[0]=6;pg[1]=8;pg[2]=7;

  printf("Running DMA portion. Please wait.\n");
  for(k=0;k<3;++k)
  {
   tn1=BUFFER;
   tn1=tn1/256;
   ca[k][0]=tn1;
   tn1=tn1-ca[k][0];
   ca[k][1]=tn1*256;
   tn1=PTS-1;
   tn1=tn1/256;
   cc[k][0]=tn1;
   tn1=tn1-cc[k][0];
   cc[k][1]=tn1*256;
```

```
}


/* set mask bits channels 1,2,and 3 */

outportb(single_mask,set_mask[0]);
outportb(single_mask,set_mask[1]);
outportb(single_mask,set_mask[2]);

outportb(flip_flop,0);

/* program dma address,count,and page registers */

outportb(2,ca[0][1]);
outportb(2,ca[0][0]);
outportb(3,cc[0][1]);
outportb(3,cc[0][0]);
outportb(131,pg[0]);     /* for page 6 */
outportb(0,ca[1][1]);
outportb(0,ca[1][0]);
outportb(1,cc[1][1]);
outportb(1,cc[1][0]);
outportb(135,pg[1]);     /* for page 8 */
outportb(6,ca[2][1]);
outportb(6,ca[2][0]);
outportb(7,cc[2][1]);
outportb(7,cc[2][0]);
outportb(130,pg[2]);     /* for page 7 */

/* fifo reset and interupts */
outportb(796,0);
inportb(796);

/* program the mode value and unmask the channels */

  outportb(mode,69);
  outportb(mode,68);
  outportb(mode,71);
  outportb(single_mask,clr_mask[0]);
  outportb(single_mask,clr_mask[1]);
  outportb(single_mask,clr_mask[2]);
N1X=N1+56;
outportb(gc1,N1X);  /* start counters */
delay(200 );
chkupdn();
outportb(gc1,N1);  /* start counters */

outportb(768,0);
tn2=((tm[flag2]*PTS*1000)+1000);
tmn=tn2;
printf("\nTIME OF RUN: %d SECs.",tmn/1000);
xa=tmn;yq=1;dw=1;
```

```
  if(tmn<65535)
   {
    yq=1;
   }
  else
  {
  do{
   xa=xa/2;
   yq=yq*2;
    if(xa<65535) dw=0;
   }while(dw);
  }
  _setcursortype(_NOCURSOR);
  xa=xa/400;
  for(cz=0;cz<yq;++cz) /* to display a changing character on
         the screen for operation purposes */
   {
    for(dz=0;dz<xa;++dz)
    {
    gotoxy(40,5);
    printf("|");
    delay(100);
    gotoxy(40,5);
    printf("/");
    delay(100);
    gotoxy(40,5);
    printf("-");
    delay(100);
    gotoxy(40,5);
    printf("\\");
    delay(100);
    }
   }
  _setcursortype(_NORMALCURSOR);
  gotoxy(1,10);
  printf("\nDelay completed. \n");

/* shut down process */

  outportb(772,0);
  outportb(single_mask,set_mask[0]);
  outportb(mode,mode_value1[0]);
  outportb(mode,mode_value1[1]);
  outportb(mode,mode_value1[2]);
  outportb(gc1,N2);  /* stop counters */
  outportb(796,0);
  inportb(796);

  printf("\nCompleted DMA.");
  delay(2000);
 return(0);
}
```

```
output()  /* sets up the DACs for each run */
{
  outportb(dac1,swrv[flag2]);
  outportb(dac6,l_dl);
  outportb(dac5,h_dl);
  outportb(dac7,init);
  outportb(setdac,0);
  return 0;
}
/* section to calculate the parameters required. */
fnd_swr()
{
  float k,time,twsr;
  int tn11,offset,t[20],t11[20],t111[20],tz,tx;
  float tn,conv[20],k1;
  unsigned char tn2,tn3,r;

  k=0;k1=0;

  twsr=fnd_t1();  /* gets the time between data points */
  outportb(2821,f2);    /* sets the dac to the
          desired voltage */
  outportb(2837,240);   /* set high limit to max
          to prevent interference */
  outportb(2841,10);    /* set low limit to min to prevent
          interference */
  outportb(2816,0);     /* reset dacs to new value */
  delay(1);  /* delay to insure dac has stabilized */
  temp=tn1+0;
  chkupdn();  /* to set direction of sweep */
  delay(1);
  tm[flag2]=t1;
  N1=84;
  N2=160;
  PTS=40;
  BUFFER=0;
  counter_su();
   clrscr();
   printf("SWEEP RUN %d.\n",c-6);
   printf("SWEEP RATE %e %e\n",twsr,t1);
  DMA();
 offset=0;
 for (c1=0;c1<20;++c1)
  /* aquires 20 readings and averages them*/
  {
   t[c1]=0;t11[c1]=0;
   tn2=peekb(0x7000,offset);
   tn3=peekb(0x6000,offset);
   t[c1]=tn2;
   t111[c1]=tn2;
   t11[c1]=tn3;
   tn=t11[c1];
```

```
    tn=tn/16;
    tn11=tn;
    tn=tn11;
    tn=tn*256;
    t[c1]=tn+((t[c1]-16)*1.438);
    dn=t[c1];
    cona(dn);
    conv[c1]=f1;
    ++offset;
   }
  for (c1=1;c1<20;++c1)
   {
    k=0;
    k=conv[c1]-conv[0];
    k1=k1+(k/(t1*c1));
    def[x]=k1/19;
   }
   r=0;
   pk(r);
   delay(5000);

   if (er==7)
   {
    fclose(test_file);
    test_file=fopen("test.val","w");
    fprintf(test_file,"%d \n",c);
    for (c1=0;c1<20;++c1)
     {
fprintf(test_file,",%d,%d,%d,\n",t[c1],t11[c1],t111[c1]);
     }
   }
   else
   {
    fclose(test_file);
    test_file=fopen("test.val","at");
    fclose(test_file);
    test_file=fopen("test.val","at");
    fprintf(test_file,"%d \n",c);
    for (c1=0;c1<20;++c1)
     {
fprintf(test_file,",%d,%d,%d,\n",t[c1],t11[c1],t111[c1]);
     }
   }
   /*outportb (ga1 ,0);*/
   return 0;

}
fnd_tm()
{
   /* to find the time between data points
    for the CV run routine */
   float t,trn;
```

```
    t=find_PTS();
    trn=t/aswr[flag2];    /* to find the total time
            of run in seconds */
    trn=trn*1.00;
    tm[flag2]=trn/PTS;    /* to find the time required
            between data points */

  return 0;
}
sv_def()    /* saves default values to the disk */
{
  de=fopen("set.dft","wt");
  for (c=0;c<32;++c)
  {
   x=def[c];
   fprintf(de,"%e \n",x);
  }
  fclose(de);
  return 0;
}
sv_dat()
{
  /* to save data from the CV routine */

  clrscr();
  printf("LOADING DATA ON DISK.");
  out=fopen(*f_name,"wt");
  fprintf(out,"%s",mtf);
  fprintf(out,"\nADC1
voltage:,,,%f,%f,%f,",cal[7][0],cal[7][1],cal[7][2]);
  fprintf(out,"\nADC1
digital:,,,%f,%f,%f,",cal[8][0],cal[8][1],cal[8][2]);
  fprintf(out,"\nADC1 stand
devi:,,,%f,%f,%f,",cal[9][0],cal[9][1],cal[9][2]);
  fprintf(out,"\nADC2
voltage:,,,%f,%f,%f,",cal[10][0],cal[10][1],cal[10][2]);
  fprintf(out,"\nADC2
digital:,,,%f,%f,%f,",cal[11][0],cal[11][1],cal[11][2]);
  fprintf(out,"\nADC2 stand
devi:,,,%f,%f,%f,",cal[12][0],cal[12][1],cal[12][2]);
  f2=0;
  for (c=23;c<29;++c)
  {
   ++f2;
  fprintf(out,"\nSWEEP RATE RANGE
    %d:,,,%f,%f",f2,def[c],def[c+1]);
   ++c;
  }
  fprintf(out,"\n");
  for (c1=0;c1<def[4];++c1)
  {
   fprintf(out,"SWEEP,%d,,",c1+1);
```

```
  }
  fprintf(out,"\n");
  for (c1=0;c1<def[4];++c1)
  {
   fprintf(out,"SWEEP RATE,%e,,",sr[c1]);
  }
  fprintf(out,"\n");
  for (c1=0;c1<def[4];++c1)
  {
   fprintf(out,"SWEEP VALUE,%d,,",swrv[c1]);
  }
  fprintf(out,"\n");
  for (c1=0;c1<def[4];++c1)
  {
   fprintf(out,"SWEEP RANGE,%d,,",swrng[c1]);
  }
  fprintf(out,"\n");
  fprintf(out,"\n");
  for (c1=0;c1<PTS;++c1)
  {
   for (c=0;c<def[4];++c)
   {
     tp1=peekb(0x7000,offset[c]);
     tp2=peekb(0x8000,offset[c]);
     tp3=peekb(0x6000,offset[c]);
     fprintf(out,"%d,",tp1);
     fprintf(out,"%d,",tp2);
     fprintf(out,"%d,",tp3);
     ++offset[c];
   }
   fprintf(out,"\n");
  }
  fclose(out);
  r=PTS*def[4];
  return 0;
}

cona()
{
   /* to convert a digital value to analog value */

f1=((((cal[7][2]-cal[7][0])/(cal[8][0]-cal[8][2]))*dn)-2.5);
   return 0;
}

cond()
{
/* to convert an analog value to a digital value for dac5 */

  float n;
  n=an+2.5;
  f2=(255/(cal[4][2]-cal[4][0]))*n;
```

```
  return 0;
}
cond7()
{
/* to convert an analog value to a digital value for dac7 */

  float n;
  n=an+2.5;
  f2=(255/(cal[6][2]-cal[6][0]))*n;
  return 0;
}

condw() /* to conver an analog value to a digital value for
dac 6 */
{
  float n;
  n=an+2.5;
  f2=(255/(cal[5][2]-cal[5][0]))*n;
  return 0;
}
pt_main_mess()
{
   clrscr();
   printf("MAIN MENU");
   printf("\n\nSelect An Option By Entering The
     Letter.\n\n");
   printf("\tA) Calibration\n\tB) Cyclic Voltammerty\n\tC)
     BICON\n");
   printf("\tD) Amperommetry\n\tE) Chronocoulometry\n\tQ)
     Exit\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n");
   printf("Enter An Option: ");
   return 0;
}
pt_cal_mess()
{
  clrscr();
  printf("CALIBRATION MENU\n\n");
  printf("Select An Option By Entering The Letter.\n\n");
  printf("\tA) Calibrate The Interface.\n");
  printf("\tB) Calibrate The DAC's Only.\n");
  printf("\tC) Calibrate The ADC's Only.\n\tQ) Main
     Menu.\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n");
  printf("\tEnter An Option: ");
  return 0;
}
pt_cv_mess()
{
   clrscr();
  printf("CYCLIC VOLTAMMETRY MENU.\n\n");
  printf("Select An Option By Entering The Letter.\n");
  printf("\n\n\tA) Analog\n\tB) Step\n\tQ) Main
     Menu\n\n\n\n\n\n\n\n\n\n\n\n\n\n");
```

```
  printf("Enter An Option: ");
  return 0;
}
pt_bicon_mess()
{
  clrscr();
  printf("BICON MENU.\n\n");
  printf("Select An Option By Entering The Letter.");
  printf("\n\n\tA) Constant Current\n\tB) Constant
  Voltage\n\tQ) Main Menu\n\n\n\n\n\n\n\n\n\n\n");
  printf("\n\n\n\n\n\tEnter An Option: ");
  return 0;
}
pt_amp_mess()
{
   clrscr();
   chro_menu();
   main_menu(0);
   return 0;
}
mess_cal()
{
  printf("\nCompleted Calibration Procedure.\n");
  delay(2000);
  return 0;
}
par_mes()
{
  int ns,gn,res;
  ns=def[4];gn=def[5];res=def[6];
  clrscr();
  printf("Cyclic Voltammeter setup parameters.");
  printf("\n\n\tA) File name: %s",*f_name);
  printf("\n\tB) Initial Potential: %e V",def[0]);
  printf("\n\tC) High Limit Potential: %e V",def[1]);
  printf("\n\tD) Low Limit Potential: %e V",def[2]);
  printf("\n\tE) Sweep rate: %e V/s",def[3]);
  printf("\n\tF) Number of Sweeps: %d",ns);
  printf("\n\tG) Set the Sweep Rate Change Parameters.");
  printf("\n\tH) Gain: %d Pick any intiger between 0 and
     6",gn);
  printf("\n\tI) Resolution: %d mV Data Points:
     %d",res,PTS);
  printf("\n\tJ) Message: %s",mtf);
  printf("\n\tK) Direction: %c",updn);
  printf("\n\tL) Pause between sweeps: %c",pau);
  printf("\n\tM) Set individual sweep parameters.");
  printf("\n\n\tQ) Return to CV menu.");
  printf("\n\n\n\n\tENTER THE LETTER OF YOUR SELECTION: ");
  return 0;
}
cv_run_mess()
```

```
{
  int ns,gn,res;
  ns=def[4];gn=gv[f];res=rs[f];

  clrscr();
  printf("Individual sweep setup parameters.");
  printf("\n\nSweep %d of %d sweeps.\n\n",f+1,ns);
  printf("\n\tA) Initial Potential: %e V",ip[f]);
  printf("\n\tB) High Limit Potential: %e V",hl[f]);
  printf("\n\tC) Low Limit Potential: %e V",ll[f]);
  printf("\n\tD) Sweep rate: %e V/s",sr[f]);
  printf("\n\tE) Gain: %d",gn);
  printf("\n\tF) Resolution: %d mV     Data Points:
     %d",res,PTS);
  printf("\n\n\tQ) Next sweep.");
  printf("\n\n\n\n\tENTER THE LETTER OF YOUR SELECTION: ");
  return 0;
}
cv_men_ms()
{
  clrscr();
  printf("Cyclic Voltametry Main Menu.");
  printf("\n\n\n\tA) Calibrate Sweep Rate.\n");
  printf("\tB) Set CV Parameters.\n");
  printf("\tC) Run CV Experiment.");
  printf("\n\tQ) Return To Main Menu.");
  gotoxy(10,18);
  printf("Enter The Letter Of Your Selection:  ");
  return 0;
}
sw_ms()
{
  clrscr();
  printf("ENTER THE VALUE OF THE RESISTERS IN OHMS AND
     CAPACITOR IN FARADS USED IN THE MODUAL.");
  printf("\n\n\tA) %e\n\tB) %e\n\tC) %e\n\tD) %e\n\tE)
%e\n\tF) %e ",def[7],def[8],def[9],def[10],def[11],def[12]);
  printf("\n\tI) %c\n\tJ) ACCEPT THE VALUES.\n\tQ)
     QUIT",updn);
  gotoxy(2,18);
  printf("ENTER THE LETTER OF THE COMPONENT YOU WANT TO
     CHANGE: ");
  return 0;
}
err_ms()
{
  gotoxy(2,17);
  printf("Only The Letters Designated Will Be Accepted.
     Please Try Again.");
  return 0;
}
error_mess()
```

```
{
   clrscr();
   printf("\nOnly The Marked Letters Will Be Accepted. Try
     Again.\n\n");
   delay(3000);
   return 0;
}
err2_mess()
{
   clrscr();
   printf("\nThe Calibration File Was Not Found In This
     Directory.");
   delay(3000);
   return 0;
}
err3_mess()
{
  clrscr();
  printf("Only The Numbers Indicated Will Be Accepted.
     Please Try Again.");
  delay(3000);
  return 0;
}
err4_mess()
{
 clrscr();
 printf("You Must Enter An 8 Or 12. Please Try Again.");
 delay(3000);
 return 0;
}
pk(r)
{
/* to look at what is loaded into the
   mem by the DMA routine */

  unsigned char a,b;
  int buf,count;
  buf=r;

  clrscr();
  for(count=0;count<20;++count)
  {
   a=peekb(0x6000,buf);
   b=peekb(0x7000,buf);
   printf("%d %d \n",a,b);
   ++buf;
  }
  return 0;
}
chkupdn()
{
  /* this routine is for the direction
```

```
   determination of the CV sweep. */

  int temp1;
  unsigned char r1,r2;
  r1='U';r2='D';

  if(def[31]==r1)
  {
  temp1=temp+16;
  outportb(3840,temp1);
  }
  else
   {
    if(def[31]==r2)
    {
    temp1=temp+8;
    outportb(3840,temp1);
    }
    else
    printf("Error in default file. Unable to determine
       direction.");

   }
  outportb(3840,temp);
  return 0;
}
set_arrays()
{
  int v1;
  for(v1=0;v1<10;++v1)
  {
   ip[v1]=def[0];
   hl[v1]=def[1];
   ll[v1]=def[2];
   sr[v1]=def[3];
   gv[v1]=def[5];
   rs[v1]=def[6];
  }
  return (0);
}
find_PTS()
{
  float t;

  t=((hl[f]-ll[f])+(hl[f]-ip[f])+(ip[f]-ll[f]));
  /* find total voltage range */
  PTS=(t/rs[f])*1000;
  /* to find number of data points */
  return (t);
}
pause()
{
```

```
    if (flag4==1)
     {
      printf("\nA pause between runs was selected.\n");
      printf("Hit return to start the next run.\n");
      scanf("%c",&v);
     }
      return (0);
}
```

HOUCHEN
BINDERY LTD
UTICA/OMAHA