To the Graduate Council:

I am submitting herewith a dissertation written by Martin Todd Mansfield entitled "Microcomputer Applications in Polymer Film Electrochemistry." I have examined the final copy of this dissertation for form and content and recommend that it be accepted in partial fulfillment of the requirements for the degree of Doctor of Philosophy, with a major in Chemistry.

J. Q. Chambers, Major Professor

We have read this dissertation
and recommend its acceptance:

Accepted for the Council:

Vice Provost
and Dean of The Graduate School

MICROCOMPUTER APPLICATIONS IN POLYMER FILM ELECTROCHEMISTRY

A Dissertation

Presented for the

Doctor of Philosophy

Degree

The University of Tennessee, Knoxville

Martin Todd Mansfield

March 1988

DEDICATION

This dissertation is dedicated to my grandfathers, Harold M. Heiser, Captain, U.S.N.(Ret.), 1908-1985, and Brook S. Mansfield, Captain, U.S.N.(Ret.), 1898-present. Their outstanding moral character, sacrifice for their family and country, and intellectual endeavors taught me the true meaning of perseverance and commitment.

ACKNOWLEDGEMENTS

The author wishes to thank his parents, Malcolm and Mary Ann. Their love and family devotion cannot be expressed in words. Without their faith and encouragement, this educational endeavor would not have been possible.

Special thanks go to Jim Chambers, who read and corrected this dissertation. Additional thanks go to Dr. G. Mamantov, Dr. C. A. Lane, and Dr. K. J. Monty who also read and corrected this manuscript. Additional thanks go to Aggie Scott and Ann Lacava, who helped with thesis formatting and figure layout.

To my colleagues, Dr. G. Inzelt, Dr. Roger Day, Dr. Houshang Karimi, Dr. Cecil Francis, and Dr. Ken Muschelewicz, I am grateful for their contributions to the synthetic and electrochemical aspects of polymers.

I also wish to thank Dr. Mike Moshell, Dr. Doyt Perry, and Dr. Hedley Bond of The University of Tennessee, Computer Science Department for their instruction in Pascal and assembly language programming. Thanks go to Dr. Don Hinton and Jim Vargo of The University of Tennessee, Mathematics Department for their instruction in numerical methods.

Finally, I would like to acknowledge the financial support of The University of Tennessee, Chemistry Department and the National Science Foundation.

ABSTRACT

Polymer modification of electrode surfaces has been a topic of great interest in the past few years. Deliberate chemical modification of electrode surfaces has been used to suppress photocorrosion of semiconductor electrodes, electrocatalyze the conversion of light to energy, improve fuel cell performance, and sense specific ions in complex biological matrices. In our laboratory, different electrochemistry has been observed for polymer modified electrodes(PMEs) by varying the nature of the polymer backbone, the redox moiety itself, the concentration of the redox sites, film thickness, temperature, the supporting electrolyte, and the solvent.

This research studied the effects of temperature, supporting electrolyte, and solvent on PME electrochemistry by comparing cyclic voltammetric results to two PME, electrochemical models. Cyclic voltammetry and chronoamperometry were automated by interfacing an Apple II+, 6502-based, microcomputer to a Bioanalytical Systems CV-27 potentiostat with Interactive Structures A/D and D/A converters. Real-time data acquisition software was written in 6502 assembly language; file management, data reduction, and numerical analysis software was written in UCSD Pascal. Application of the electrochemical models to the digitally acquired data was facilitated by the Pascal programming environment. In addition to useful electrochemical information, conclusions were drawn from the microcomputer interface and programming design to evaluate the potential of other microcomputers and high-level programming languages to electrochemical methods.

Both models suggest that mechanical effects in PMEs arising from the forced intrusion of charge compensating counterion and solvent during oxidation or reduction should be included in the electrochemical theory. Elasticity changes in thin films of a tetracyanoquinodimethane terpolymer and poly(vinylferrocene) were observed as the temperature, supporting electrolyte, and solvent were varied. Neither model alone explained the electrochemical results. The concepts of stress-strain and measurement time invoked in the mechanical/electrochemical model explained the interaction parameter results. The results indicated that solvent-polymer interactions neglected in the mechanical/electrochemical model should be included in the derivation and that polymer elasticity as a function of these three variables should be included among the factors influencing charge transport through thin polymer films.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER I

INTRODUCTION

A. Computers in Analytical Chemistry

Since its introduction in the 1970s, the microcomputer has had a significant impact on analytical chemistry. Prior to then, the use of computers was limited essentially to the simulation of response curves and the statistical treatment of manually entered data points. The introduction of the first minicomputers such as the PDP-8 by Digital Equipment Corporation signalled an era of change in analytical chemistry.[1] Since then, advances in solid state design of computers have been made at an alarming rate, with both advantageous and deleterious effects. The advantages have included memory compaction, an increase in speed, and reduction in cost; thus, capabilities once associated with bulky, expensive minicomputers have become associated with microcomputers. They are now common, off-the-shelf weapons in the analytical laboratory.[2]

While the development of inexpensive microcomputers has given analytical chemists a new and powerful tool for use in research and teaching, computers have not seen as widespread use as one would expect. Each microcomputer has its own microprocessor instruction set, hardware configuration, and bus structure making it difficult to decide which microcomputer and software are best suited for the job in hand. There are also different languages available for programming and a wide selection of commercially available peripheral devices from which to

choose.[1-5] Most systems now include the necessary peripherals in a single package for word processing needs, freeing the user from these potential pitfalls. The situation is reversed in the analytical laboratory because the microcomputer must be capable of monitoring and interacting with the instrumental environment through an interface to its bus structure.[1] The interfacing problems and solutions discussed here attest to these facts.

Many of the important decisions the chemist must make to solve interfacing problems in the laboratory can now be gleaned from the chemical literature.[1,2] Since 1980, numerous examples of different microprocessors, operating systems, hardware configurations, and bus structures have been published to guide the chemist in his/her interfacing decisions.[2,6-13] Some of the more common microprocessors in daily use in the analytical laboratory include the 8088 of the IBM-PC/XT which executes in the PC-DOS, MS-DOS, and UCSD p-operating systems; the 6502 of the Apple II/II+/IIe and Commodore-64 which executes in the DOS 3.3, PRODOS, and UCSD p-operating systems; and, the Z80 of the TRS-80(Radio Shack) and others which executes in the CP/M, TRSDOS, and UCSD p-operating systems.[6-13] These operating systems perform all of the tedious, unglamorous jobs necessary to make the computer work. They monitor the keyboard and branch to subroutines indicated by keyboard entries. They also operate the monitor, read and write information to disk in the proper format, and perform many other housekeeping chores.[14] For any interfacing problem, the first step is understanding how to use and traverse the operating system of the chosen microcomputer.

A microcomputer is of little value to the analytical chemist without a programming language written for its operating system. Most microprocessors now support high-level languages such as Basic, Fortran, Pascal, Algol, and Forth and a native assembly language environment for the particular microcomputer.[1,14] Less familiar languages such as C and Modula-2 are gaining in popularity also.[14,15] In assembly language, short verbal mnemonics, such as LDA for load accumulator or SBC for subtract with carry, are used to manipulate binary values in the native code of the microprocessor. Assembly language programming accesses the true capabilities of the machine, produces the most efficient code, and gives the best control over timing. Unfortunately, most assembly language instruction sets do not include multiply or divide instructions and reading or writing a message to the monitor requires more than a single command. Another problem is that real number arithmetic and numerical analysis are too difficult to handle. Therefore, high-level languages are used to process chemical data. The real utility of assembly language lies in its ability to control peripheral devices such as analog-to-digital and digital-to-analog converters, ADCs and DACs, respectively, in time-critical applications. Assembly language programming is most often concerned with signal domain conversion between a computer and chemical instrumentation.[1]

High-level languages use simple statements to accomplish what would otherwise require several assembly language statements to implement. Programs require more memory and execution time than their assembly language counterparts but far less time to write. Of the

high-level languages, Basic is most often interpreter-based while Fortran, Pascal, and Algol are compiler-based. Basic program instructions are decoded one character at a time, interpreted and executed. Compiler-based languages translate a program into a machine language version of the original which is stored in memory for execution. High-level languages are also more conducive to numerical programming and more complex data manipulation tasks such as graphics, memory management, and input/output(I/O).[1]

Of the operating systems, languages, and microcomputers mentioned, we were interested in applications of the 6502-based, Apple II/II+/IIe/IIGS to electroanalytical instrumentation. To a lesser extent, other microcomputer-based data acquisition systems which automated electrochemical techniques also proved helpful.[16-18] A plethora of publications have appeared which demonstrate the rich diversity of hardware and software available.[1-18] However, none of the Apple interfacing approaches have used the combination of Pascal and 6502 assembly languages for data acquisition.[10,19,20]

Since the first appearance of on-line computers in nuclear chemistry,[21] many electrochemical methods have been automated. Lauer and coworkers[22] described a computerized chronocoulometric experiment as early as 1967. Computer-based data acquisition and control systems for cyclic voltammetry,[23,24] AC/pulse polarography,[25] anodic stripping voltammetry,[26] ion-selective electrode measurements,[27] potentiometric titrations,[28] and conductance[29] have appeared since then. These early acquisition systems had to be dedicated to one technique because the interfacing hardware was complicated and inflexible.[21-29] Recently,

much effort has been expended in the area of electrochemical development systems which will automate several techniques in one microcomputer or minicomputer facility. Britz,[16] Ramaley,[30] Li et al.,[31] and Fanelli et al.,[17] have made significant contributions in this area. These development systems attempt to avoid considerable programming effort and free the user from formidable interfacing tasks.

Another approach that has become more prevalent is the use of two computers for data acquisition and control. Examples of mini-to-mini, mini-to-micro, and micro-to-micro designs have appeared.[27,30-32] In these systems, a satellite computer is dedicated to controlling electrochemical instrumentation. The satellite computer communicates directly with a host computer via a bidirectional, serial data path, permitting rapid on-line transfer of data and commands. Electrochemical experiments are supervised by the satellite while the host does what it is best suited for, namely data processing, storage, and plotting. The satellite merely acts as a peripheral device to the host computer and frees the host computer to perform other tasks while the measurements are performed. Inexpensive microcomputers are more likely to be used as the satellite computer.[30-32]

Ideally, a 6502-based, Apple II series interface to electrochemical instrumentation would be flexible, inexpensive, and programmable. A highly flexible interface at the hardware and software level would allow some of the more sensitive pulsing methods such as staircase voltammetry and pulse polarography to be implemented without recourse to a host/satellite or complex analog circuit design.[31,33] For the Apple II series microcomputers, Basic is already in ROM, and Fortran,

Pascal, and Modula-2 compilers are also available.[14,15] Literature on 6502 assembly language programming is also diverse.[34] Peripheral expansion of memory for the Apple II series is easily accomplished in these languages, and a variety of hardware is available for data acquisition and control.[14] We investigated these possibilities on an Apple II+ microcomputer interfaced to a potentiostat with commercial peripherals. Automation of cyclic voltammetry and chronoamperometry was achieved in the UCSD Pascal and 6502 assembly language programming environments. We were also interested in evaluating the 6502-based, Apple II series microcomputers as stand-alone, development systems in the electrochemical laboratory.

B. Polymer Modified Electrodes

In any analytical laboratory, there are several advantages to having a microcomputer interfaced to instrumentation. Data can be digitized and stored on magnetic disks for subsequent calculations; calculations can be performed in a high-level programming language for comparison to theory; graphics, output, and interaction with the experiment can be simplified; and, all of these features can be customized or modified to suit a specific application in a short period of time. Such features are of particular importance in electrochemistry because conclusions always come from a comparison of the experimental data to a theoretical response.

The research we conducted on polymer modified electrodes(PMEs) exemplifies this situation. Initially, the interaction parameter model was applied to the cyclic voltammetric response(i-E) of PMEs in various

supporting electrolytes and solvents.[35-37] The relative effects of temperature, supporting electrolyte, and solvent on polymer, thin film electrochemistry were ascertained by interaction parameter calculations. Whereas the interaction parameter model has existed for some time, the mechanical/electrochemical(ME) model has not. The ME model, which we also invoked to explain PME behavior, was recently proposed by Evans and coworkers[38-40]. Both models are discussed in CHAPTER II.

Electrochemical studies of PMEs are part of a larger class of electrochemical research dealing with chemically modified electrodes(CMEs).[35,41-44] The electrochemist deliberately seeks to immobilize a molecule on an electrode surface so that it will thereafter display some desirable property. CMEs can be fabricated by chemisorption, covalent binding, and film deposition. They are dubbed chemically modified, derivatized, functionalized, electrostatically trapped, and polymer coated electrodes. Adsorption of molecules or ions on the electrode surface may involve multilayer of monolayer quantities, and a variety of electrochemical and spectroscopic methods are used to study and characterize their surface behavior.[35] Deliberate chemical modification of electrode surfaces has been used to suppress photocorrosion of semiconductor electrodes,[45,46] electrocatalyze the conversion of light to energy,[45,47] improve fuel cell performance,[45,48] control the release of chemically active reagents,[45,49] enhance electrochromic displays,[45,50] and preconcentrate electroactive species in solution.[45,51] Thus, the electrode can be tailor-made to control or improve an electrochemical process.

The first publication to articulate and demonstrate the notion of immobilizing a redox functionality was that of Hubbard and Lane.[52,53] Electroactive quinone-bearing olefins were chemisorbed on Pt electrodes by some covalent bond-like interaction between the olefin and Pt surface. Although chemisorption can seldom be predictably invoked,[35] it was the first adsorption process to incorporate the interaction parameter in its theoretical treatment.[54] The interaction parameter model has since been used to explain PME voltammetric behavior.[37,55-57]

Covalent bonding refers to a covalent linkage between the electroactive species and the electrode surface, which is usually a metal oxide.[41] SnO_2,[48] RuO_2,[59] TiO_2,[60] Pt/PtO,[61] Si/Si(oxide),[62] Ge,[63] and oxidized carbon surfaces have been functionalized with a variety of organosilanes in a versatile immobilization strategy. Unfortunately, both chemisorption and covalent bonding require special chemical pretreatment of the electrode surface and sophisticated analytical techniques are required to characterize the small, monolayer amounts of material involved. These CMEs lack the robust character necessary for the applications cited previously, because the layers are usually lost too quickly to be effective.[42,64]

Emphasis then shifted to incorporation of electroactive moieties into polymer matrices.[65] These polymers can be deposited as thin films in multilayer quantities by dip,[66] spin[65] and evaporation[67] coating, organosilane bonding,[68] electrochemical precipitation[56,69] and polymerization,[70] electrostatic binding,[71] and plasma discharge polymerization.[39,67] The advantages of PMEs include, ease of preparation, larger electrochemical responses since multilayers of redox sites are

involved, and stability from the polymer matrix.[35,65] PMEs studied by the electroprecipitation of thin films of poly(vinylferrocene)(PVF)[56,69,72] and evaporation coated tetracyanoquinodimethane-terpolyester(TCNQ-TPE)[73] were the subject of this study. Electroprecipitation of PVF thin films is based on the fact that a polymer's solubility depends on its ionic state.[35,56,69] By oxidizing PVF to its less soluble and more adsorbable ferricenium form, thin films were adsorbed at the electrode surface.[69,74] Evaporation coating on the other hand involves evaporating a few microliters of a dilute solution of the polymer in its neutral state on the electrode for modest electrode coverages. The electroprecipitated PVF films adsorb with the supporting electrolyte in place. These films must be annealed for geometric uniformity.[40] Film preparation by evaporation or dip coating is widely used to study PME behavior in electrocatalysis applications.[35,44] A detailed discussion of electrocatalysis is beyond the scope of this manuscript. It is worthy of some comment, however, since electrocatalysis deals with improving the kinetics of electrode processes.[35,41,75]

In electrocatalysis, reduced or oxidized constituents of a polymer film act as electron transfer mediators(catalysts) to drive electrochemical reactions of solution components(catalyst substrates).[35,41,42] Slow electrochemical reactions at naked electrodes might be improved kinetically by suitable choice of a redox polymer. The kinetics of a slow electron transfer process at a bare electrode/solution interface might require extreme negative or positive potentials for reduction or oxidation, respectively. Thus, one might

have to work near the overpotential limit of the solvent/electrolyte system. Andrieux and Saveant[76] have pointed out that PMEs can be used to circumvent solution kinetic problems, and that the high concentration of redox sites in a multilayer PME leads to better catalytic efficiency than a derivatized monolayer.

Selecting a PME for a particular application will depend on the general electrochemical properties of the modified surface. For example, the electroactivity of redox polymer films is a function of the film's "ionic" conductivity.[35] For high ionic conductivity, enough supporting electrolyte ions must penetrate the film through pores, or the film must contain many fixed electroactive sites with charge compensating counterions. Ionic conductivity is necessary to create a sufficient potential gradient at the electrode/polymer interface to drive electron transfer between the electrode and adjacent redox centers. Solvent penetration into the polymer(i.e. swelling) also facilitates adequate ionic mobility, penetration, and conductivity. Ion-free or "neutral" polymer films containing electroactive groups can be less than fully electroactive because of film ohmic resistance or poor solvation. A film of low "ionic" conductivity may ,however, possess sufficient "electronic" conductivity for electron transfer at the polymer/solution interface.[35] Polyacetylene, polyphenylene, and polypyrrole films are examples of good electronic conductors.[77] Electron transport through these polymers occurs by a delocalized band structure which is different from redox conductivity by site-site hopping in most polymer films.[35,65]

Another factor related to counterion and solvent penetration is the film's permeability. This is important in electrocatalysis because the solution species must penetrate the polymer film for electron mediation. In other polymer films, the electroactive solution constituent must partition into and diffuse through the film to reach the electrode and express its intrinsic electroactivity. Cases are known of full and partial electroactivity,[67] indicating inability to penetrate or slow diffusion through the film. The charge and size of the solution species, and the thickness and solvent swelling of the polymer film are relevant factors.

Most redox centers in an ionically conductive polymer film cannot directly contact the electrode for electron exchange because they are fixed in the polymer matrix. Therefore, Kaufman et al.[65] proposed oxidation or reduction to occur by electron self-exchange reactions between neighboring redox sites. In addition to the electron self-exchange process, other possible rate-limiting phenomena exist.[35] They include the flow of counterions and associated solvent,[81] segmental motions[35] and elasticity of the polymer,[38] interactions between similarly charged redox sites,[36,37,67,82] and the intrinsic thermal barrier to electron transfer.[37] Migration of electrons in PMEs is coined charge transport.[67,83]

Charge transport for fixed-site redox polymers seems to follow Fickian diffusion[64,76] and can be studied by chronocoulometry as Karimi and Chambers[73] have shown. In this research, D_{ct}, the effective diffusion coefficient for charge transport,[84,85] is important from the standpoint of its effect on the peak symmetry common to voltammetric

surface waves.[37] Charge transport problems in voltammetric surface waves are indicated by diffusional tailing, $\Delta E_p \neq 0$, and $i_p \alpha v^{1/2}$ instead of v.[35,65] According to Dahms[84] and Ruff,[86] Chambers and Inzelt,[37] and the work of Murray and coworkers[87] on bipyridyl copolymer films, one can avoid this problem to a great extent by increasing the temperature, increasing the concentration of redox sites, preparing thinner films, and sweeping in the thin film domain.[37,65,87]

Charge transport studies alone do not explain the cyclic voltammetric peak-shape changes often observed for PMEs. Multiple waves are sometimes observed for one redox center;[56,65,82] current spikes can appear;[69,71,82] and, cathodic and anodic waves may be asymmetric or broader than expected.[65,67,80] Such phenomena seem very sensitive to the solvent and supporting electrolyte ions employed.[56,67,82,88,89] These waveshapes have been attributed to attractive or repulsive, site-site interactions in the interaction parameter model.[56,67,69,82,89] At the limits of this model, phaselike[67,82,89] or site dimerization[65] interactions prevail. The purpose of this work was to investigate temperature, solvent, and supporting electrolyte effects on the redox behavior of PMEs by invoking the interaction parameter and ME models.[37,38]

After ion-exchanging polymers were described by Oyama and Anson,[90] a series of closely spaced publications appeared describing the electroactivity of other "ion-exchanging" and "preformed" redox

PMEs.[35,42,44] In ion-exchanging polymer films, the redox ion is in-partitioned by soaking the PME in a dilute solution of the ion,[91]

$$\underset{\text{(polymer)}}{Pt/[-CHCH_2-]_x}\Big| \atop PhSO_3^-Na^+ \quad + \quad \underset{\text{(soln)}}{[Ru(bpy)_3]^{2+}} \longrightarrow \underset{\text{(polymer)}}{Pt/[-CHCH_2-]_x}\Big| \atop PhSO_3^-[Ru(bpy)_3]^{2+} \quad + \quad \underset{\text{(soln)}}{Na^+}$$

After in-partitioning, the electrode is transferred to a fresh supporting electrolyte solution for electrochemical study.[44] A couple of examples of ion-exchanging polymers are the evaporation coated Nafion PMEs of Bard[71] and Ferguson et al.,[96] and the evaporation coated poly(vinylpyridine) PMEs of Anson and coworkers.[97] In preformed polymers, the redox polymer is most straightforwardly synthesized by polymerizing the appropriate electroactive monomer.[44] Ferrocene,[69,72,83,89] alkylated 4,4'-bipyridinium(viologens),[92,93] nitrostyrene,[80] tetrathiafulvalene,[56,94] and TCNQ[95] can be incorporated into polymers in this way.

Two redox, preformed polymers, which exhibit significantly different cyclic voltammetric waveshapes depending on polymer structure, solvent, supporting electrolyte, and temperature are based on the electroactive monomers, ferrocene[39,40,56,67,82] and tetracyanoquinodimethane(TCNQ).[37,73,98,99] Polymers of the former are electron donors and of the latter are electron acceptors.[101,102] Symmetric, broadened cyclic voltammetric waveshapes have been observed for ferrocene polymer films in CH_3CN solvent which are not too thick($\Gamma_T < 10^{-8}$ mol/cm^2),[56] are based on organosilane linkages,[41,45,89] contain oxygenated sites

from plasma damage during polymerization,[39,83,103] are diluted by an inert copolymer such as acrylonitrile,[56,104] and are examined at slow scan rates.[56] When such films are thicker or are studied at faster scan rates or lower temperatures, charge transport rates become important and voltammetric waveshapes exhibit diffusional tailing.[37,55,65] In aqueous solvents, the ferrocene polymer films sometimes exhibit spiked anodic peaks and either broad or spiked cathodic peaks even at slow scan rates.[72,82,89,94] Sharpness of the peaks can be electrolyte dependent and multiple peaks may appear. Such behavior has been attributed to strong attractive site-site interactions(i.e. activity is independent of redox site concentration) caused by poor solvation of the film.[67,82]

Similar, diverse cyclic voltammetric waveshapes have been observed for TCNQ polymers.[37,73,98,99,102,105] Since Day's synthesis of the TCNQ polyester,[95,106] this electroactive monomer has been studied as a polyurethane[107] and copolymer,[73] at different temperatures,[36] and in different supporting electrolytes[102,105] and solvents.[98] In addition to temperature, solvent, and supporting electrolyte effects on TCNQ-PME voltammetry, these waveshapes are complicated by dimerization,[101] ion association,[102] and protonation equilibria,[99] coupled to the charge transport process. Whereas, ferrocene polymers show classic one electron behavior,[56] TCNQ polymers are reduced in two successive one electron steps. The electron hopping charge transport process through TCNQ polymer films is markedly affected by chemical reactions in the film phase making application of the interaction parameter model to these experiments difficult.[37] Two notable studies were reported by Inzelt,

Joo and Chambers from this laboratory.[37,98] Temperature was shown to have a significant effect on the TCNQ polyester voltammetry in 0.5M phosphate buffer. Waves for the TCNQ/TCNQ$^{\cdot-}$ and TCNQ$^{\cdot-}$/TCNQH^{-} couple at pH 7 became sharper and narrower at higher temperatures. In a combined scan rate and temperature analysis of these films, it was shown that voltammetric response enters the diffusion regime at higher scan rates and/or lower temperatures and that dimerization ($2TCNQ^{\cdot-}/TCNQ_2^{2-}$) is favored at lower temperatures.[37] Studies of TCNQ-PMEs in 0.1M LiCl/ROH where R = CH_3, $CH_3(CH_2)_2$, and $CH_3(CH_2)_3$ have also shown marked effects on the voltammetric $TCNQ^{o/-}$ and $TCNQ^{-/2-}$ waveshapes, D_{ct}, the film "break-in" process, and the stability of the film in the reduced state.[98] The TCNQ polymer studied here is a TCNQ, 1,4-bisoxyethanol benzene copolymer, which was originally synthesized for charge transport studies.[73] The second wave $TCNQ^{\cdot-/2-}$ has been well characterized by spectroelectrochemistry and electron spin resonance[101,102] and was the subject of our study of TCNQ-PME voltammetric behavior.

CHAPTER II

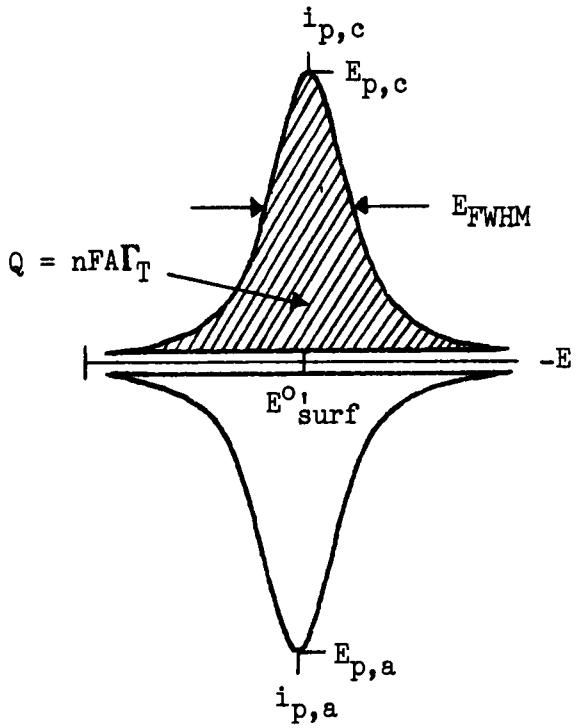
THEORY

A. The Interaction Parameter Model

Cyclic Voltammetry of the Ideal Thin Film

Cyclic voltammetry(CV) has been used extensively to characterize the electroactivity of monolayer and multilayer films of redox species irreversibly adsorbed on solid electrodes.[59,74,93] Certain quantities of interest in a voltammetric surface wave are illustrated in Figure 1.[108,109] Eqs. (2-4) in this figure describe the voltammetric characteristics of a reversible, immobilized, redox monolayer(more or less). In this ideal case, the surface activities of O and R are both directly proportional to their surface coverages Γ_O and Γ_R, respectively, and Γ_T is the total electroactive coverage($\Gamma_O + \Gamma_R$). The significant properties of a reversible reaction are that a proportionality exists between i_p and scan rate v, that the peak potentials and waveshapes for the cathodic and anodic surface waves are identical($\Delta E_p=0$), and that $E_{FWHM} = 3.53RT/nF$.[55,110b]

Electrodes with immobilized monolayers regularly yield i_p-v proportionality. It follows for these electrodes that the charge under the current-potential(time) curve gives the total quantity Γ_T in mol/cm^2 of attached electroactive sites, which is independent of v. Charge and Γ_T measurements which use a simple current baseline, as in Figure 1, assume that any dispersion in double-layer capacitance which



$$[O + ne^- \rightleftharpoons R]_{surf} \qquad (1)$$

$$i = \frac{-n^2F^2\Gamma_T v}{RT} \quad \frac{\varepsilon}{(1+\varepsilon)^2} \qquad (2)$$

$$i_p = \frac{n^2F^2\Gamma_T v}{4RT} \qquad (3)$$

$$\varepsilon = \exp[(nF/RT)(E-E^{o'}_{surf})] \qquad (4)$$

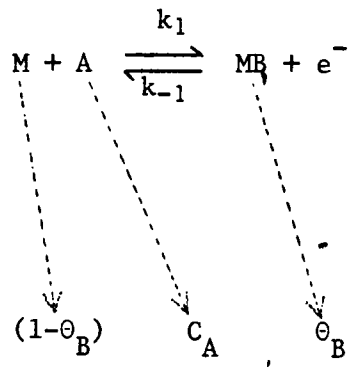
Figure 1. Equations for an ideal CV surface wave.

may accompany alterations in surface ionic charge caused by reduction or oxidation is insignificant.[35]

When the kinetics of electron transfer are rapid and equilibrium of the redox sites is established instantaneously with respect to the change in electrode potential, $\Delta E_p = 0$ at any scan rate, the Nernst equation is applicable in principle, and eqs. (1-4) in Figure 1 describe the electrochemical process for all practical purposes. If the potential of the electrode is swept at a rate which far exceeds the rate of electron transfer in a thin film, ΔE_p deviates from zero and waveshapes change.[111-115] The above equations describing ideal i-E behavior do not apply because we are removed from the equilibrium case. The fact that ΔE_p is scan rate dependent if electron transfer is slow tells the experimentalist to reduce the scan rate and allow more time for electron transfer as the applied potential is changed. This brings the CV experiment into the thin film domain where i_p is proportional to v and waveshapes are symmetrical.

Interaction Parameter Theory and Background

Long before electrochemists began modifying electrodes with redox polymers, they were studying the adsorption isotherms of species such as HCOO•, H, and O at different solid electrode surfaces by both simulation and experiment.[54,116] The isotherm equations developed in simulation attempted to relate the amount of substance adsorbed on the electrode per unit area, Γ_T, the activity of the substance in bulk solution, and the electrical state of the system.[110] An example of a set of simulation isotherm equations is shown in Figure 2.[112,116]



(1)

$$i = Qd\Theta_B/dt = Q[k_1 C_A(1-\Theta_B)\exp\{\Delta V/b-(1-\beta)g\Theta_B\}-$$

$$k_{-1}\exp(-\Delta V/b + \beta g\Theta_B)] \qquad (2)$$

where $k_1 = k_{-1}$, $\Delta V = V_i + st - E^\circ$,

$b = RT/\beta F$, $\beta = 0.5$

$Q = 220\mu C\ cm^{-2}$(monolayer coverage)

$$\Theta_B/(1 - \Theta_B) = (k_1/k_{-1})\exp(-g\Theta_B)\cdot C_A\exp(VF/RT) \qquad (3)$$

Figure 2. Langmuir-type equations which first used the interaction parameter to describe chemisorption.

Eq. (1) in Figure 2 reveals that a chemisorption process is being simulated and that the species is not irreversibly adsorbed on the metal substrate at the beginning of the voltammetric experiment. A further complication is that the fractional coverage, θ_B, at any time in the sweep, depends on the sweep rate s and rate constants, k_{-1} and k_1, as eq. (3)(Figure 2) demonstrates. Some simulated results for eq. (2)(Figure 2) are shown in Figure 3. The pseudocapacitance, $Qd\theta_B/dt$, is plotted versus potential for different log(s/k) values. From corresponding plots of $\Delta V_{1/2}$ versus log(s/k), it was possible to extract kinetic and reversibility information.[112-115]

These curves assumed an absence of lateral interactions so that g = 0 and the fractional coverage at any time was not affected by changes in activity in the adsorbed monolayer. This is equivalent to a Langmuir isotherm which is described by eq. (3) in Figure 2. Some isotherms produce sharper and narrower i-E behavior as the sweep rate, s, is varied while others produce attenuated and broader i-E curves. This ultimately means the surface coverage is being altered by a change in activity as well as sweep rate and kinetics. Therefore, simulated curves for the interaction parameter, $g \neq 0$, were introduced to explain this(Figure 4A).[112,116] The consequence of this simulation was that E_{FWHM} was found to be almost linearly related to the interaction parameter, g(Figure 4B). Hence, one could measure E_{FWHM} from the curve, compute g directly, and access the family of simulated curves to determine kinetic, reversibility, and mechanistic information.

Gileadi and Conway[117] extended the calculation of the interaction parameter for the electrodeposition and ionization of chemisorbed atoms

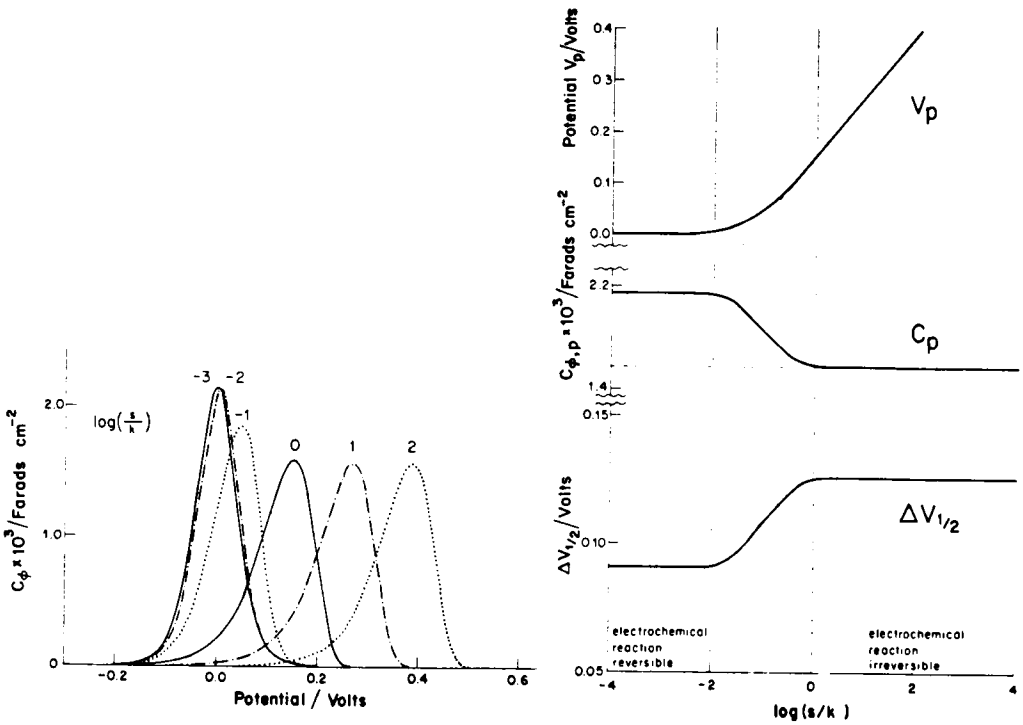


Figure 3. Pseudocapacitance(i-E) and kinetic plots of Langmuir isotherm when g = 0.

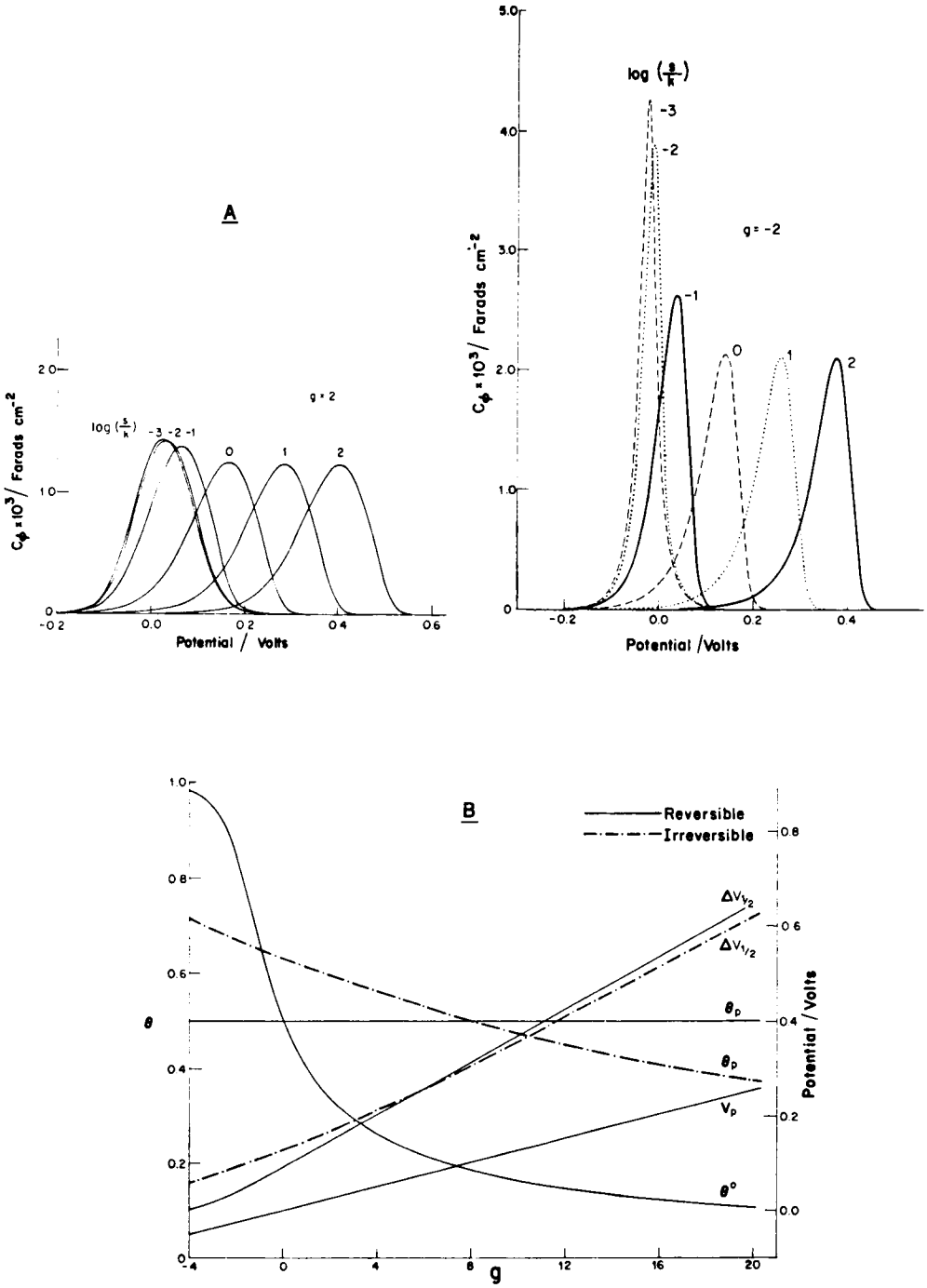


Figure 4. (A) Family of curves when $g \neq 0$, and (B) near linear dependence of $E_{FWHM}(\Delta V_{1/2})$ on the interaction parameter, g.

to EC and CE electrochemical mechanisms. Conway[118] treated irreversibility in surface electrochemical reactions when rate constants are small compared to sweep rate, and Srinivasan and Gileadi[116] treated the transition from reversible to irreversible behavior. More recently, the chemical kinetics of surface processes under linear potential sweep control has been discussed by Laviron.[119] Thus, in one of these papers, Laviron[120] repeated the conclusion of Conway et al.[54] regarding the linear dependence of E_{FWHM} on g(G according to Laviron) without evidently being aware of their earlier work.

During the 1970s, the theory of Laviron[119,120] gained more attention due to the relevance to experimental cases. Electrochemists began to modify electrodes by irreversible adsorption, coating with polymers, and covalent bonding so that both the oxidized and reduced species were strongly adsorbed,[35,41] $C_A = 0$ in eq. (1) of Figure 2, and the solution kinetic processes were effectively eliminated from the adsorption isotherm equations.[121] In many cases, these modified electrodes were monolayer or submonolayer in thickness and the kinetics of electron transfer appeared rapid because ΔE_p was close to zero and cathodic and anodic peaks were symmetrical.[36,122] These results meant that the earlier equations of Figure 1 ought to agree in theory with the experiment in terms of i-E behavior, recalling that we could also describe the surface coverage as function of time by an equilibrium isotherm as Conway[54,112] and Laviron[119,120] have suggested. Conway and his colleagues[112-115] continued in their theoretical treatments where only one form, O or R, was strongly adsorbed from solution, whereas Laviron[57,121] continued theoretical treatments where both O and R are

strongly adsorbed. For that reason, the theory of Laviron[55,57,120,121] is considered more important in studies of PMEs.

Determination of the Interaction Parameter

The previous discussion explains the use of the interaction parameter in the context of this study. Brown and Anson[36] studied 9,10-phenanthrenequinone, benzo[c]cinnoline, iron protoporphyrin IX, and other redox molecules(no polymers) as irreversibly adsorbed monolayer on pyrolytic graphite electrodes. They compared the CVs of these CMEs to the theoretical results obtained from eqs. (2,4) in Figure 1 and found that the experimental curves were flatter and E_{FWHM}s larger than expected. The waves were symmetrical and E_ps near zero so that slow electron transfer could not explain the deviations.[112]

Based on the theoretical treatment of Laviron[120] which explained this result as lateral interactions in a Langmuir isotherm, Brown and Anson[36] decided to modify eqs. (2) and (4) in Figure 1 by a Frumkin-like[35,110] isotherm which changes the activities of O and R through an interaction parameter, r. (Note that r, g, and G are equivalent terms.) Thus, lateral interactions between O and R could cause a change in wave shape depending on the value of the interaction parameter. A summary of this derivation is shown in Figure 5, where a, γ, Γ, r_{OO}, r_{OR}, r_{RR}, and r_{RO} are the surface activities, activity coefficients, concentrations, perturbing influences between oxidant-oxidant, oxidant-reductant, reductant-reductant, and reductant-oxidant molecules, respectively. The fraction of molecules on the surface in their oxidized form is denoted f, $r_O = r_{OO}-r_{OR}$, and $r_R = r_{RR}-r_{RO}$, so

$$i = \frac{-n^2F^2\Gamma_T v}{RT} \quad \frac{\varepsilon}{(1+\varepsilon)^2} \tag{1}$$

$$\varepsilon = \exp[(nF/RT)(E-E^\circ)] \tag{2}$$

$$\varepsilon = \frac{a_O}{a_R} = \frac{\gamma_O\Gamma_O}{\gamma_R\Gamma_R} \tag{3}$$

$$\gamma_O = \exp-(r_{OO}\Gamma_O + r_{OR}\Gamma_R) \tag{4}$$

$$\gamma_R = \exp-(r_{RR}\Gamma_R + r_{RO}\Gamma_O) \tag{5}$$

$$\varepsilon = \frac{f}{1-f} \exp-\{\Gamma_T[f(r_O + r_R) - r_R]\} \tag{6}$$

$$i = \frac{-n^2F^2\Gamma_T vf(1-f)}{RT[1 - 2f\Gamma_T r(1-f)]} \tag{7}$$

$$i_P = \frac{n^2F^2\Gamma_T v}{RT[4 - 2\Gamma_T r]} \tag{8}$$

Figure 5. Outline of the Brown and Anson derivation of the interaction parameter equations.

that eq. (6) in Figure 5 can be obtained from substitution of eqs. (4) and (5) into eq. (3); a final simplification is made by setting $r = (r_O + r_R)/2$ to obtain eq. (7). Entire CVs have been fit to eq. (7) in Figure 5 to obtain a measure of ideality. When r = 0, eq. (7) reduces to eq. (1) and the film is ideal. Negative r values indicate that some process is causing instability among neighboring redox sites and waveshapes are flatter and broader. Narrower, sharper waves indicate processes that are causing stability or less repulsion among neighboring sites. This interaction parameter approach proved to be an excellent method for comparing film ideality among different molecules on different electrode surfaces.[36,122] It is helpful here to make the analogy between $r\Gamma_T$ and the $g\theta_B$ term which was used in the early literature to explain slow electron transfer.[116] Another powerful interpretation of the Brown and Anson model is that we could compare the ideality of similarly prepared films in a study where temperature, supporting electrolyte, and solvent are varied. Thus, it should be possible to gauge the effect of these variables on the redox processes in thin films.

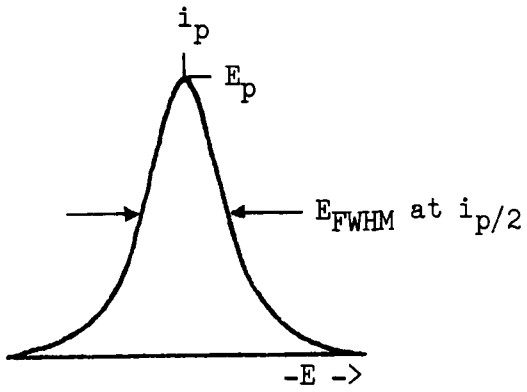
Eq. (7) of Figure 5 fits experimental waveshapes fairly well for chemisorbed and covalently bound monolayers.[59,61,122,123] It seems that the product $r\Gamma_T$ is more characteristic of a particular surface species than r itself, since E_{FWHM} does not vary systematically with Γ_T and in fact seems from data available to be independent of it.[61,123] It is not surprising that E_{FWHM} depends linearly on the product $r\Gamma_T$.[122] This is a consequence of E_{FWHM} being linearly dependent on g if an isotherm is obeyed. Smith et al.[122] showed this relation graphically

for E_{FWHM}s at 25°C. It is worth noting that the only term in eqs. (7) and (8) in Figure 5 that will be affected by swelling of the film is the interaction parameter. When r = 0, this model predicts that the peak current will not be influenced by swelling. If r values are negative and repulsive interactions persist, one would expect swelling to reduce these repulsive interactions and make r less negative.[37]

To determine $\Gamma_T r$ values from our experiments, we have taken advantage of the linear relation between $\Gamma_T r$ and E_{FWHM}. Others have fit entire CVs to theoretical ones to solve for r directly. Mathematically, this approach calculated the minimum deviation from the ideal curve using a Milne Predictor-Corrector method.[122] Unfortunately, convergence to the minimum deviation required as long as 20 minutes per CV.[122] Our method[37] has the advantage of simplicity and efficiency.

The method used here begins by dividing eq. (7) by (8) in Figure 5 to obtain eq. (1) of Figure 6. The fraction electrolyzed at a current equal to the peak current divided by two, $i_{p,a/2}$ is $f_{1/2}$ and the $\pm$ operator refers to the side of the curve $i_{p,a/2}$ is on. At $i_{p,a}$, f = 1/2, and one must add to or subtract from that value to find the fraction electrolyzed.[37] This subtle point bears significance in PROGRAM PEAKFIT where $E_{p,c/2}$ or $E_{p,a/2}$ was computed. In both cases, the more negative potentials were determined and E_{FWHM} was found by doubling the absolute value of $E_{p,c/2}-E_{p,c}$ or $E_{p,a/2}-E_{p,a}$.

To refine the $_T r$ calculation, eqs. (2) and (6) of Figure 5 were equated and cast in the form of the Nernst equation due to Brown and Anson.[36] The result is eq. (2) of Figure 6, where $E = E_{p/2}$ at $i = i_{p/2}$ and $E_p = E^o$. Eqs. (1) and (2) of Figure 6 were used to compute a



$$f_{1/2} = \tfrac{1}{2} \pm 0.5(1 - 4/B)^{1/2} \qquad (1)$$

where $B = 8 - 2\Gamma_T r$

$$(E_{p/2} - E_p)nF/RT = \ln\left(\frac{f_{1/2}}{1-f_{1/2}}\right) - \Gamma_T r(2f_{1/2}-1) \qquad (2)$$

Figure 6. Equations from Brown and Anson treatment used to compute $\Gamma_T r$.

working curve of $\Delta E_{1/2}nF/RT$ versus $\Gamma_T r$. Values of $\Gamma_T r$ between -2.0 and 1.0 were used to calculate $f_{1/2}$ values(with the - operator). The $f_{1/2}$ and $\Gamma_T r$ values were then substituted into eq. (2)(Figure 6) to calculate $\Delta E_{1/2}nF/RT$ values.[37]

CVBROWN, a specific version of PROGRAM CVOUTPUT, implements the working curve as a line whose slope and intercept were determined by the linear regression of fifteen points. CVBROWN computes $\Delta E_{1/2}nF/RT$ by doubling and taking the absolute value of $E_{p/2}-E_p$. An initial value of $\Gamma_T r$ is found from the working curve, substituted into eq. (1)(Figure 6) to find $f_{1/2}$, and finally substituted into eq. (2)(Figure 6) to compute a refined value of $\Gamma_T r$. E_{FWHM} was found by doubling the absolute value of $E_{p,c/2}-E_{p,c}$ or $E_{p,a/2}-E_{p,a}$.

Limitations and Applications

While the interaction parameter calculation would be in principle an excellent method for comparing solvent/electrolyte effects in a multiexperiment study of a PME, previous literature studies reveal that many intrinsic experimental factors can and do affect their cyclic voltammetric surface waves. The kinetics of electron transfer and its scan rate dependent effect on CV waves has already been discussed, for example, and the original theory was designed for monolayers of species where $\Delta E_p = 0$. The $\Gamma_T r$ calculation will be a composite of all these factors which cause nonideality. Therefore, we must have prior knowledge of these factors to gauge their relative effects in PME experiments.

In most immobilized monolayers, ΔE_ps are small but seldom actually zero. Some carefully examined cases show that these nonzero ΔE_ps are not scan rate dependent and speculation about solvation, molecular conformation, and kinetic problems has been made.[61,123] These problems in monolayer films are even further magnified in polymer, multilayer films on electrodes and cause surface waves to become even more complex.

Mechanisms for electron exchange between sites where they are too remote to contact the electrode directly, polymer chain motion, counterion uptake, and solvation are collectively described by charge transport in polymer films. Charge transport is generally thought to occur by electron self-exchange and is represented mathematically by the charge transport diffusion coefficient, D_{ct}, which is a strong function of the experiment duration, τ, and film thickness, d.[65,83,103] Experimentally, one can reduce the sweep rate and/or decrease the film thickness to avoid charge transport complications. Thus, i_p becomes proportional to v and $\Delta E_p \longrightarrow 0$. In cases where D_{ct} is very small,[83] diffusional tailing appears, i_p is proportional to $v^{1/2}$, and the Randles-Sevcik equation applies,

$$i_p = 2.69 \times 10^5 \, n^{3/2} A D_{ct} v^{1/2} C_o \qquad (1)$$

These ideas were introduced and illustrated by Daum et al.,[82,83] Nowak et al.,[103] and Oyama and Anson[124] for PMEs. Diffusional behavior is promoted by small D_{ct}(low temperature),[82] polymer film crosslinking,[103] fast potential scan rate,[83] and thick films.[124] The important conse-

quence of this discussion is that D_{ct} is affected by temperature. Although our $\Gamma_T r$ calculation takes temperature in the equilibrium case into account, D_{ct} could become small enough at low temperature to overwhelm ideal thin film behavior. Chambers[37] addressed this problem in PMEs and demonstrated the need to sweep slowly and use thin films. In this research, a series of control experiments were performed to observe the combined effect of temperature and film thickness(i.e. D_{ct}) on the interaction parameter calculation; the sweep rate was maintained at 20mV/s throughout.

The interaction parameter model has already been applied in some notable cases to explain the electrochemical behavior of PMEs. Peerce and Bard[56] incorporated the interaction parameter in their digital simulation of poly(vinylferrocene)(PVF) and poly(vinylferroceneacrylonitrile)(PVFAN) in 0.1M tetrabutylammonium perchlorate(TBAP)/CH_3CN and 0.1M tetraethylammonium p-toluenesulfonate(TEApTS)/CH_3CN. They also simulated electrochemically non-equivalent sites in the films to explain shoulders on the more positive potential side of the main $E^{o\prime}_{surf}$. The presence of the shoulder was related to the main wave in a scheme containing coupled equilibrium(square scheme) with equilibrium constants which appeared to change with the polymers and supporting electrolytes.

Daum and Murray[67,82] have studied plasma-polymerized PVF modified electrodes in 0.1M $LiClO_4/CH_3CN$, 50/50 ethanol/H_2O, and H_2O. They observed very different anodic peak shapes in the CVs and showed on the basis of interaction parameter theory that voltammetric behavior in the film was heavily dependent on the type of solvent. When H_2O was the

solvent, an anodic current spike($r >> 0$) was observed which they attributed to activities becoming independent of surface coverages. Laviron and Roullier[57] have simulated and confirmed this experiment with an interaction parameter modified Frumkin isotherm. The simulation indicated irreversibility due to slow electron exchange with the electrode.

In two recent publications, the interaction parameter was revisited for kinetic analysis of charge transport in PMEs. Ikeda et al.[44] employed an interaction parameter, G, based on redox site concentration, to explain deviations in surface voltammetric waves of Pt/poly$[Ru(vbpy)_3]^{3+/2+}$. They used an E_{FWHM} working curve to determine G and found that it was independent of Γ_T = 2 x 10^{-10} to 10^{-8} mol/cm^2.[122,125] Their equations were equivalent to (1) and (2) of Figure 5. Daifuku et al.[126] analyzed surface waves of PMEs with E_p vs. ln v plots which were the same in theory as early simulation studies of chemisorption processes(Figure 3). Extrapolation of the irreversible region of these curves was used to determine the critical scan rate, v_i. The slope of the irreversible portion of these Tafel-like curves is proportional to the transfer coefficient. Interaction parameter values were calculated from a working curve of E_{FWHM} nF/RT vs. v_i.[126] With the transfer coefficients, interaction parameter, and v_i, heterogeneous rate constants can be calculated for these films.

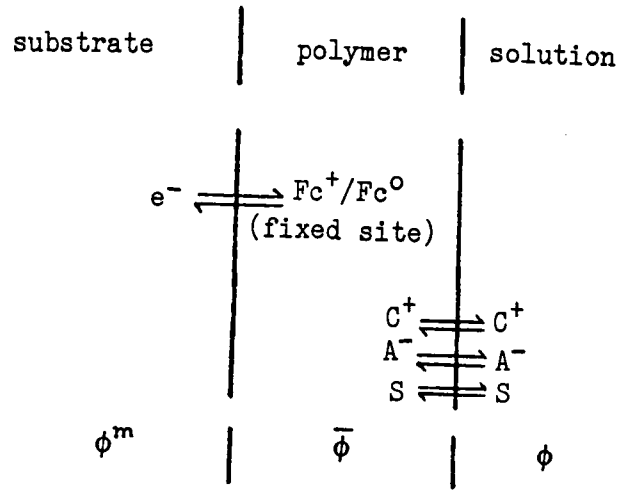
B. The Mechanical/Electrochemical Model

The Mechanical/Electrochemical Concept

Another model was recently proposed to explain the redox properties of PMEs.[38] It couples the mechanical and electrochemical thermodynamics associated with a change in the charge per redox site during conversion. Mechanical effects in these films result from the forced intrusion of charge compensating counterions which changes the film's void volume. This forced swelling of the film requires energy in excess of that needed to carry out simple electron transfer to/from redox sites and may be coupled to the thermodynamic properties of the solvent. The model relates this mechanical energy to the chemical energy as a work term, where energy is consumed to expand the polymer matrix. The mechanical effect can be considered thermodynamically reversible if film deformation is elastic and irreversible if deformation is plastic.

The mechanical/electrochemical model we will review here suggests that mechanical Gibbs energy terms which are neglected in analyses of most electrochemical reactions should be included in the overall energy expressions of PME behavior. The inclusion of a mechanical, Gibbs energy, work term can provide an alternative explanation to chemical models which have been invoked to explain nonideal PME behavior.

A summmary of the thermodynamic derivation is given in Figure 7. Eqs. (1) and (3) in Figure 7 represent equilibrium between the ferricenium cation, Fc^+, and neutral species, Fc^o, the univalent cation, C, and univalent anion, A, of a strong electrolyte, CA, the solvent mole-



$$\bar{Fc}^+ + e^{-,m} \rightleftharpoons \bar{Fc}^\circ \tag{1}$$

$$\bar{\mu}_{Fc^+} + \bar{\mu}_{e^-}{}^m = \bar{\mu}_{Fc^\circ} \tag{2}$$

$$\bar{A} \rightleftharpoons A,\ \bar{C} \rightleftharpoons C,\ \bar{S} \rightleftharpoons S \tag{3}$$

$$\bar{\mu}_A = \mu_A,\ \bar{\mu}_C = \mu_C,\ \bar{\mu}_S = \mu_S \tag{4}$$

$$\mu_i = \mu_i^\circ + RT \ln \gamma_i c_i + Z_i F\phi + PV_i \tag{5}$$

$$\Delta\phi = \phi^m - \phi = (\bar{\mu}^\circ_{Fc^+} - \bar{\mu}^\circ_{Fc^\circ} + \mu^{\circ,m}_{e^-})/F + (RT/F)\ln[\bar{\gamma}_{Fc^+}\bar{c}_{Fc^+}/\bar{\gamma}_{Fc^\circ}\bar{c}_{Fc^\circ}] + (\bar{\mu}^\circ_A - \mu^\circ_A)/F + (RT/F)\ln[\bar{\gamma}_A\bar{c}_A/\gamma_A c_A] + \bar{P}(\bar{V}_{Fc^+} - \bar{V}_{Fc^\circ})/F + V_A(\bar{P} - P)/F \tag{6}$$

$$E = \hat{E}^{\circ\prime} + (RT/nF)\ln[\frac{f}{(1-f)}] + nV_A{}^2 cK(f-0.5)/(10^4)FZ^2 \tag{7}$$

Figure 7. Thermodynamic equations for the mechanical/electrochemical model.

cule, S, and the electron, e^-, in the different phases. A bar over a species denotes the polymer film phase and superscript m the substrate phase.[127,128] By equating the chemical potentials for each species in the model such as eqs. (2) and (4) and developing an expression such as eq. (5) for each species i, eq. (6) in Figure 7 can be derived. Substitution of eq. (5) into eqs. (2) and (4) in Figure 7 for the anion only gives two equations which can be rearranged for the electrostatic potential drops, $\phi^m - \bar{\phi}$, and $\bar{\phi} - \phi$, respectively(V_i is the partial molar volume, c_i the concentration, P the pressure relative to P^o, and γ_i the activity coefficient).

On the basis of eq. (6) in Figure 7, there are a number of terms contributing to the overall electrode potential. The first term on the right side of eq. (6) adds to the standard electrode potential, E^o, measured against a reference electrode. The second term is the Nernstian activity ratio term. The third and fourth terms are the constant and concentration dependent contributions resulting from a Donnan ionic equilibrium. The fifth term is a mechanical work term which is important when the partial molar volumes of Fc^+ and Fc^o in the film differ substantially from each other under conditions where the internal pressure of the film is significant. The sixth and final term is the mechanical work term which depends on the pressure drop across the film/solution interface and the partial molar volume of the incorporated counterion. Thermodynamic contributions to eq. (6) in Figure 7 from the solvent, μ_s, have not been included.

Bowden et al.[38] chose to ignore Donnan contributions to electrode potential to emphasize the mechanical aspects of the model. Although

Braun et al.[129] have presented an instructive analysis of a polymer film as an anion exchanger, an experimentally high background electrolyte concentration introduced a negligible Donnan potential difference. It is possible to assess the contributions of the Donnan term by experimentally varying the supporting electrolyte concentration. The final term in eq. (6) of Figure 7 is the term of interest in this model. It can be expected to be temperature sensitive. Pressure at the film/solution interface has two origins, osmosis of the solvent and expansion of the polymer network to accommodate counterions for redox conversion.[127,128] The fifth term is assumed to be negligible based on crystallographic evidence.[130]

Relating eq. (6) of Figure 7 to our experiments requires further thermodynamic treatment, particularly when considering the sixth mechanical term. The important Nernstian-modified equation which recasts eq. (6) for experimental analysis is eq. (7) of Figure 7, where E is in mV for K in dyn/cm^2, V_A is the molar volume of the anion in cm^3/mol, and c is the total concentration of ferrocenic sites in mol/cm^3. The fraction of redox sites oxidized is labelled f. The meaning of standard potentials(E^{o}) or formal potentials($E^{o\prime}$) has been changed to an operational formal potential, $\hat{E}^{o\prime}$, for the discussion of experimental data. It is defined as the equilibrium potential of a PME when one half of the charge required for total film conversion has been passed. The bulk modulus term, K, describes the elasticity of the polymer film. By varying the value of the third term on the right side of eq. (7) in Figure 7, simulated plots of $E-E^{o\prime}$ versus $\log[f/(1-f)]$ were constructed for comparison to experiment. The mechanical contri-

bution to electrode potential is predicted to increase as the bulk modulus of the polymer film increases.

Assumptions and Limitations

One major problem of this model is that deconvolution of mechanical effects from one another, as well as chemical effects from one another, cannot be achieved without independent measurements other than current and voltage. Measurement of the volume of a film as a function of its redox state for example would aid immensely in characterizing thermodynamic behavior.

Two other major assumptions are that the film activity and bulk modulus have constant values irrespective of the film redox state. We know that the bulk modulus changes with potential as the film becomes more ionic since amorphous polymers and ionic crystals have K values of $1\text{-}5 \times 10^{10}$ and $0.1\text{-}4 \times 10^{12}$ dyn/cm^2, respectively.[131,132]

To derive eq. (7) of Figure 7, it was assumed that volumetric strain was homogeneous. This may not be true since the electrode surface constrains one film face. Also, distribution of redox sites may depend on electrode potential and the assumption of an isotropic distribution of redox sites may be invalid. It is also possible that the solvation sheath of the counterions changes with the level of internal film stress, changing the film volume as applied potential is varied. The point here is that separation of variables in the model may or may not be valid, but we must do so to make comparison with experiments. Clearly, there is overlap in the voltammetric experiment

between the Donnan, site-aggregation, and mechanical regions of the Nernst plots.[131,132]

Applications

Plasma-polymerized vinylferrocene(PPVF) electrodes have been studied by this model. It is a highly crosslinked polymer which may be deposited on virtually any substrate through the use of a radio frequency, nonequilibrium discharge carried out in the presence of monomeric vinylferrocene.[82,103] The ferrocene moieties retain their structure in these films. In addition to the usual considerations of the electron transfer reaction, the mechanical/electrochemical model was used to interpret nonideal thermodynamic behavior. Nonidealities were attributed to the mechanical Gibbs energy expended in the process of transferring an anion from solution to the polymer phase, forcing the film to expand.[39]

These PPVF films were studied on glassy carbon(GC), pyrolytic graphite(PG), and Pt electrodes in the presence of 0.1M tetrabutylammonium perchlorate(TBAP)/CH_3CN, 0.1M tetrabutylammonium hexafluorophosphate($TBAPF_6$)/CH_3CN, 0.1M $LiClO_4/H_2O$, 0.1M KCl/H_2O, 0.1M KNO_3/H_2O, and 0.1M KF/H_2O. Film thicknesses were measured by ellipsometry[110] and polymer film morphology was examined by scanning electron microscopy. Acquisition of plots of electrode potential vs. $\log(c_{Fc}+/c_{Fc}o)$ was the primary experimental objective. This log term is most easily measured by controlled potential coulometry(CPC) or, as was done here, by numerically integrating successive portions of the voltammetric wave. A specific version of PROGRAM CVOUTPUT, called CVTHERM, performs this

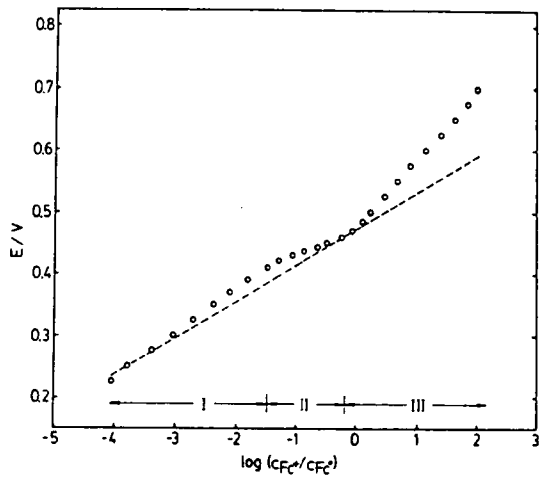
calculation and computes values of $E-E_{p,a}$ vs. $\log[f/(1-f)]$. Coulometry consisted of first applying a desired potential for sufficient duration to achieve equilibrium between the film, electrode, and solution, and then measuring the charge passed upon subsequent reduction of the film, Q_{Fc^+}. The log term then becomes, $\log[Q_{Fc}{}^+/(Q_T-Q_{Fc}{}^+)]$. This procedure was then repeated for a set of potentials spanning the redox region of the Fc^+/Fc^o couple within the film. Measured values of Q included contributions from both double layer charging and extraneous faradaic processes, as well as the faradaic process of interest. The time dependent loss of redox sites via ferrocenium decay,[123] uncertainty with regard to establishment of equilibrium, and hysteresis effects were other complications. Step potentials were routinely incremented from 0.0 to 0.9V at 50mV intervals. The authors explained a numerical method for ferrocenium decay correction in their study of PPVF.[39]

The PPVF exhibited a high degree of crosslinking between alkyl chains and less permeability as the films became thicker. Waves in slow scan CVs of PPVF were symmetrical, but broader than expected(E_{FWHM} = 150mV as opposed to 90.6mV), and the secondary redox system resulted from a small population of ferrocene pendant groups having carbonyl moieties bonded to the carbon atom α to the ferrocene. This shifted those ferrocene $\widehat{E}^o$'s by ca. 250mV in the positive direction.[94,133] The $\widehat{E}^o$'s for PPVF in the various supporting electrolyte/solvent systems ranged between 0.304 and 0.430V vs. SSE in CVs recorded at 10mV/s. Redox thermodynamic behavior showed no apparent dependence on film thickness or electrode substrate for PPVF films in

the various systems either. Interpretations of PPVF behavior by the mechanical term in the model will be discussed later.

PVF of the type used in this research was also studied by Bowden et al.[40] using CPC. Films were prepared by spin-coating a few drops of the polymer dissolved in CH_2Cl_2(20mg/ml) onto GC and Pt electrodes. Film thickness and morphology was determined as previously described.[38] These neutral films were studied in 0.1M TBAP/CH_3CN vs. Ag/AgCl, and analyzed by regions as shown in Figure 8A.

Region I corresponds to the Donnan equilibrium term where the potential is governed by the thermodynamics of the Fc^o/Fc^+ couple as well as the anion exchange properties of the film.[128] Slopes of the curves in Region I were found to be greatly influenced by the supporting electrolyte concentration. Region II is interpreted by a redox site aggregation model. Site aggregation is expected to occur in PVF polymers as they become oxidized. From Eisenberg's[134] theory, it is also expected that the ferricenium cations and counterions exist as ion pairs rather than separate ions. This theory predicts that as the concentration of ionic sites increases, ion pairs associate into larger clusters of considerable size.[135] Cluster formation has also been called microphase separation.[30] Quantitative application of the theory in an electrochemical environment requires estimation of the cluster decomposition temperature which is unavailable at present. Also, the solvent and electrolyte partition into the film to a certain extent; different solvents will cause changes in the dielectric constants of the film. Film heterogeneity could also be a problem. Therefore, the environment inside a film is very complex and difficult to quantitate.



A

$$E = E^{\circ\prime} + (RT/nF)[\ln(2f/(1-f)) - \ln(1+(1+8fcK_{do})^{\frac{1}{2}})] \qquad \text{B}$$

$$E = E^{\circ\prime} - (RT/nF)[\ln(2(1-f/f)) + \ln(1+(1+8(1-f)cK_{dr})^{\frac{1}{2}})] \qquad \text{C}$$

$$E = E^{\circ\prime} + (RT/nF)\ln\left[\frac{(1 + 8\ fcK_{do})^{\frac{1}{2}}-1}{(1 + 8(1-f)cK_{dr})^{\frac{1}{2}}-1}\right] \qquad \text{D}$$

Figure 8. (A) Sample Nernst plot, (B) oxidant dimerization equation, (C) reductant dimerization equation, and (D) both types of dimerization equation.

The qualitative picture that was developed to explain Region II behavior is a Nernst equation modified by equilibrium constants for ionic-oxidant and reductant dimerization, K_{do} and K_{dr}, respectively. An ionic dimer would be comprised of two Fc^+A^- ion pairs, called a quartet by Eisenberg.[39] This type of dimerization shifts the operational formal potential to more negative values and causes a decrease in Nernst slope for $f < 0.5$. Reductant dimerization($2Fc^o = Fc_2^o$) shifts the operational formal potential to more positive values and causes a decrease in Nernst slope for $f > 0.5$. Equations for oxidant, reductant, and both types of dimerization are summarized in Figure 8B-D. The total concentration of ferrocene sites(Fc^+ and Fc^o) is denoted c. Using the oxidant dimerization equation and a cK_{do} value of 100, the simulated Nernst plot fit the 0.1M TBAP/CH_3CN results quite well. Region II data can also be explained by the interaction parameter model. Attractive interactions between redox sites and narrow, sharp waves provide a similar interpretation. A major difference is that the aggregate model accounts for bulk interactions,[139] whereas the interaction parameter model applies to lateral interactions of adsorbed species in a monolayer.

The 3-dimensional, site aggregation model also favors the argument which explains thermodynamic behavior of PVF in Region III by the mechanical/electrochemical coupling model. In Region III of the results in Figure 8A for PVF in TBAP/CH_3CN, more energy is required for oxidation than predicted for a simple one electron transfer. The anomalous behavior in Region III is attributed to the elastic expansion/contraction of the polymer network to incorporate or expel

counterions. This mechanical process is coupled to the electrode potential by the electrostatic potential differences as shown previously. Similar deviations in Region III of the PPVF Nernst plots were also observed.

To observe mechanical contributions to electrode potential, there must be a sufficient density of appropriately distributed ionic crosslinks. Also, elastic stress must not dissipate on the experimental time scale, and there must be an absence of internal "free" solvent, since osmotic sources of internal stress were not accounted for.[36] Eq. (7) of Figure 7 was fit to Region III of the PVF Nernst plot to determine a bulk modulus of $1.4 \times 10^{10} dyn/cm^2$. It is of the right order of magnitude for these polymers.[131] These results were based on the molecular theory of the mechanical/electrochemical model. In contrast to the interaction parameter model, it can be experimentally tested and provide insight at the molecular level into PME behavior.[38]

Proposed Research

With these two models, an investigation of PMEs was undertaken. The interaction parameter model requires CVs to perform calculations whereas the mechanical/electrochemical model requires coulometric data for Nernst plots. We chose to perform slow sweep CV to analyze electroprecipitated PVF thin films in various supporting electrolyte/solvent systems over a 100°C range. The effect of film thickness on $\Gamma_T r$ calculations was carried out by varying electroprecipitation times in a control study. The interaction parameter calculation was

also extended to the study of a terpolymer modified electrode described in CHAPTER III.

To understand the molecular behavior of our PMEs, voltammetric data were numerically converted to charge versus time data for comparison to the mechanical/electrochemical model. Thus, it should be possible to determine the role of temperature, supporting electrolyte, and solvent on their electrochemistry. In particular, we will be interested in differences in Regions II and III of the model as a function of these three variables.

Such a study obviates the need to automate CV and to a lesser extent chronoamperometry. A microcomputing facility with advanced numerical capabilities was ideally suited to such a study. For that reason, research will also require mastery of the following microcomputer areas:data acquisition, triggering, timing, software design, hardware selection, data smoothing and manipulation, graphics, numerical and nonnumerical analysis, library and file organization techniques, memory management, assessment of errors, and firm control of the microcomputer and its associated hardware by both a high-level and machine language.

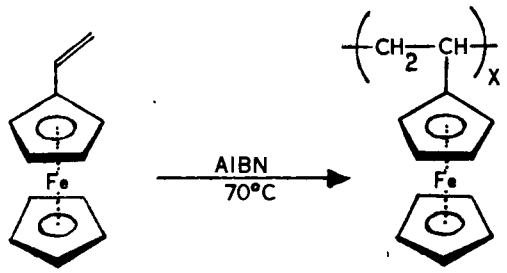
CHAPTER III

EXPERIMENTAL

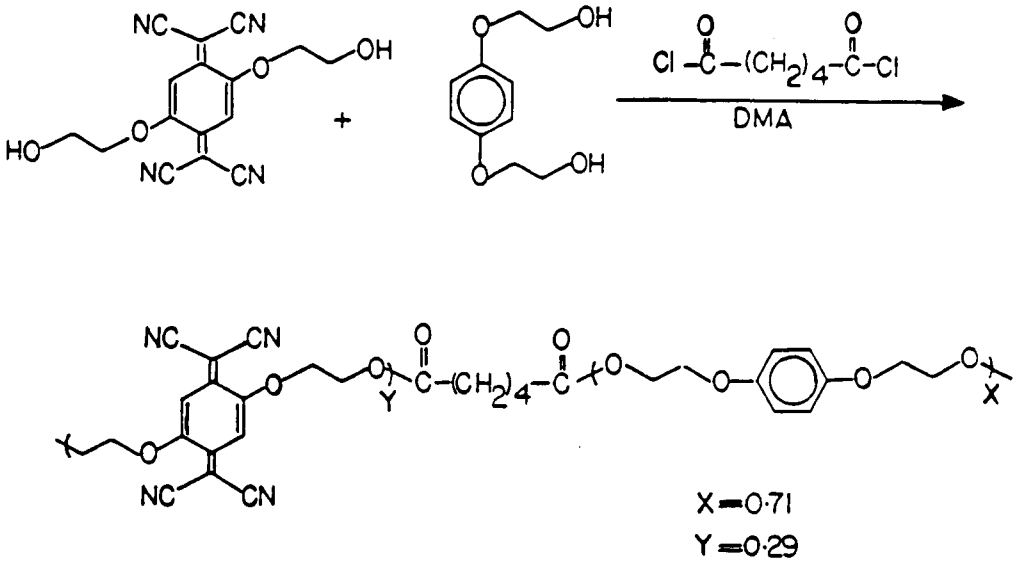
A. Synthesis

Poly(vinylferrocene)(PVF)

Vinylferrocene(Alfa) was recrystallized from distilled hexane and dried under vacuum. Azobis(butyronitrile), AIBN(Eastman), was recrystallized from methanol and vacuum dried. A 50-ml aliquot of benzene(Aldrich, 99+%) was vacuum distilled and stored over 0.3nm molecular sieves. Methanol(Aldrich, 99+%) was also stored over 0.3nm molecular sieves and used without further purification. To synthesize PVF, $\bar{M}_n$ = 26,300, 4.24g of vinylferrocene, 5.0ml of benzene, and 33.6mg of AIBN were charged to a 40-ml Carius tube. The resulting solution was subjected to several freeze-thaw cycles and lyophilized to remove the benzene. During this process, the tube was attached to a high vacuum pump apparatus(Sargent-Welch) to degas the solution and trap the benzene. Freeze drying was continued for 12 hours to exhaust the tube of any benzene; then, the tube was sealed in vacuo and placed in an oven at 70°C for 20 hours to carry out the polymerization. The resulting polymer was dissolved in a minimal amount of benzene, reprecipitated in methanol, filtered, and dried under vacuum overnight at 25°C. This process was carried out three times to remove impurities, yielding 1.7148g(40.4%) of PVF.[136] In Figure 9A are structures of vinylferrocene and PVF.[44]



A



B

Figure 9. (A) Synthetic scheme for PVF, and (B) terpolymer.

TCNQ-BOEB Copolymer(TCNQ-TPE)

The TCNQ polymer used in this study was synthesized by solution polycondensation of a mixture of 2,5-bis-(2'-hydroxyethoxy)-7,7,8,8-tetracyanoquinodimethane(TCNQ-diol), 1,4-bis-(2'-hydroxyethoxy)-benzene(BOEB-diol), and a stoichiometric amount of adipoyl dichloride in dimethylacetamide(DMA) at low temperature. Details of the TCNQ-diol and BOEB-diol synthesis are given elsewhere.[106,137] DMA(Fisher Certified) was fractionally distilled under reduced pressure and stored over 0.3nm molecular sieves. Under the dry nitrogen atmosphere of a glove bag, 0.4375g(2.587mmol) of twice distilled adipoyl dichloride, weighed into a syringe, was added to a frozen solution of 0.3254g(1.6434mmol) of recrystallized BOEB-diol and 0.3058g(0.9438mmol) of TCNQ-diol in ca. 30ml of DMA. After melting, the mixture was stirred at ca. 4°C. One day later, the reaction mixture solidified into a gel, but stirring was continued for 5 days; the gel was then transferred to a beaker and dispersed in 120 ml of absolute ethanol. The solid was collected by vacuum filtration, washed exhaustively with ethanol, and dried overnight under vacuum in an oven. The product was a brown powder, 0.5365g(58.6%). A reaction scheme for this synthesis is shown in Figure 9B.[73,107]

Characterization

Cyclic voltammetry of the polymers cast as thin films was compared to literature results.[56,69,72,73] In Figure 10A, $i_{p,c}/i_{p,a}$ was ca. 0.7 and ΔE_p was ca. 35.0 mV, entirely consistent with the results of Merz and Bard[69] for thin films of PVF prepared in this way. Peak potentials

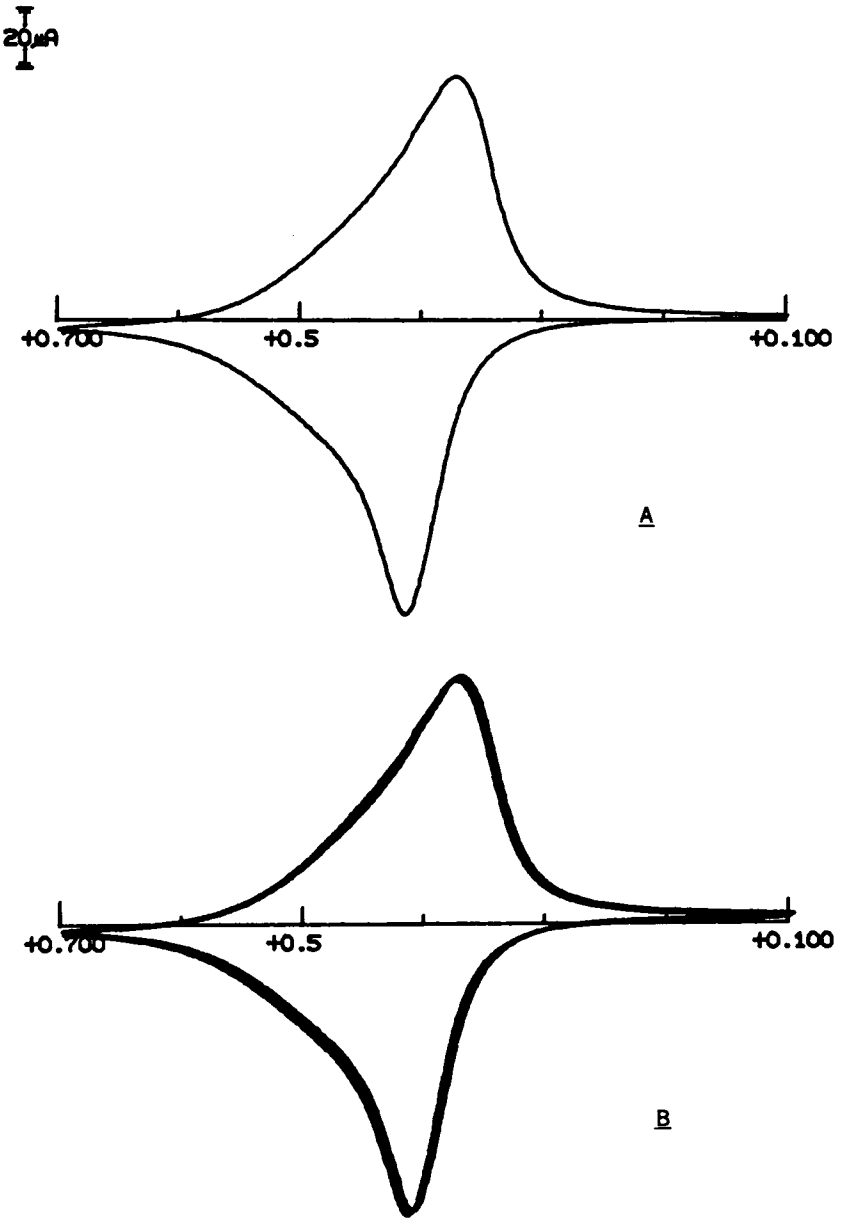


Figure 10. (A) One cycle CV, and (B) ten cycle CV of an electroprecipitated PVF thin film on Pt in 0.1M TEAP/CH_3CN. Scan rate = 20mV/s, initial scan direction = negative.

shifted to more positive potentials because a Ag/AgCl reference electrode was used instead of a Ag wire.[69] Also, at these scan rates, the cathodic peak was broader than the anodic peak and neither $i_{p,c}$ nor $i_{p,a}$ attenuated significantly after several cycles.

On the same Pt electrode used to cast the PVF thin film, a thin film of the TCNQ-TPE was cast. Its CV is shown in Figure 11A. The second wave, which was the redox process of interest in this study is shown in 11B. The E_p and i_p values of the second wave($A^{\cdot-}/A^{2-}$) were consistent with literature results.[73] The mole fraction of TCNQ in the polymer backbone(Figure 9B) was calculated by comparing the relative absorptivities of the TCNQ-diol and BOEB-diol in a UV-visible spectrum. The value 0.294 was in excellent agreement with the elemental analysis(Galbraith, Knoxville, TN); calc:%C = 61.7, %H = 5.52, %N = 5.33; found:%C = 61.3, %H = 5.59, %N = 5.29.[73]

B. Cyclic Voltammetric Analysis

Introduction

An Apple II plus microcomputer was interfaced to BioAnalytical Systems(BAS) CV-27 for real-time acquisition of cyclic voltammetric data. The voltammograms of PVF and TCNQ-TPE were recorded through this interface and subjected to a sophisticated set of Pascal programs which calculated interaction parameter and thermodynamic values. Voltammograms were recorded in various supporting electrolyte/solvent systems over ca. $\pm 50^{o}C$ except where the redox process was shut down by the the freezing point of the solvent. PVF was studied in the various

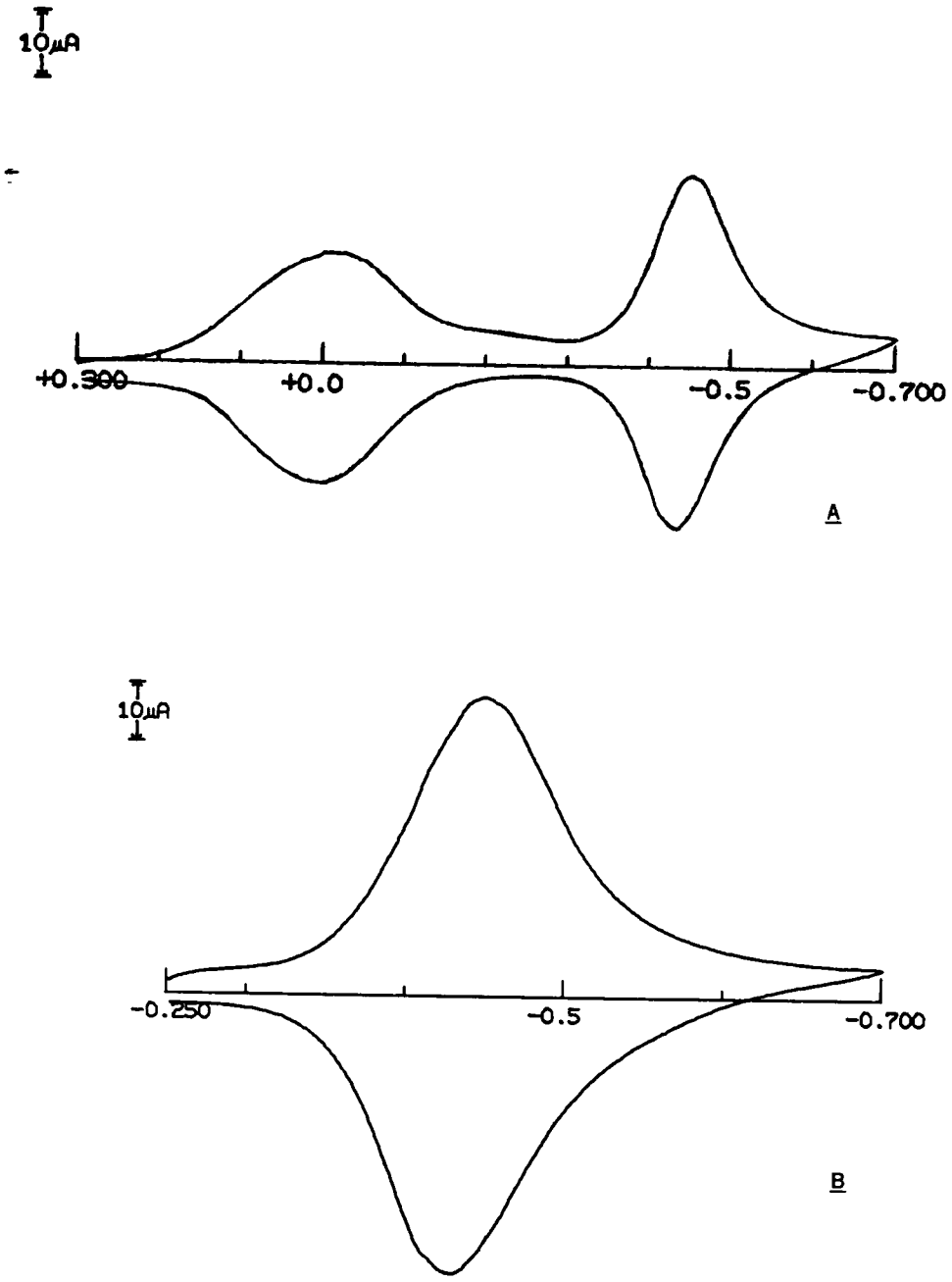


Figure 11. (A) Complete CV, and (B) second wave CV of an evaporation-coated terpolymer thin film on Pt in 0.1M TBAP/CH_3CN. Scan rate = 20 m V/s, initial scan direction = negative.

systems described in Table I; TCNQ-TPE was studied in 0.1M TEAP/CH_3CN. The full chemical names of the supporting electrolytes are given in the following section. All of the voltammetric data were permanently stored on floppy disk.

Chemicals

Tetraethylammonium perchlorate(TEAP), (Fluka, >99%), tetrabutylammonium perchlorate(TBAP), (Kodak, 99%), and tetrabutylammonium tetrafluoroborate($TBABF_4$), (Kodak, 99%) were recrystallized twice in distilled, deionized(DD) water. After each recrystallization, the supporting electrolytes were filtered and washed with 2, 10-ml aliquots of cold DD water and vacuum-oven dried overnight at ca. 30°C. TEAP, TBAP, and $TBABF_4$ were held under desiccation until use. Acetonitrile(CH_3CN), (Aldrich, 99+%), butyronitrile(C_3H_7CN), (Aldrich, 99+%), benzonitrile(C_6H_5CN), (Aldrich, 99+%), and vacuum distilled dichloromethane($MeCl_2$), (Fisher, ACS grade), were stored over 0.3nm molecular sieves and used without further purification.

Film Casting and Electrode Preparation

The usual three electrode configuration was used to acquire cyclic voltammetric data;[139] the working electrode was a Corning, Pt-button Inlay(No. 476060) of area 0.351cm^2; the reference was a Ag/AgCl electrode, prepared by poising a Ag wire at +1.00V in 1N HCl for ten seconds. In all of the solvent systems studied, the potential of the Ag/AgCl electrode was measured relative to a saturated calomel electrode(SCE). The counter electrode was a Pt-grid. For electropre-

Table I. Summary of experimental studies performed on PVF.

ELECTROPRECIPITATION TIME/SEC	NUMBER OF CVs RECORDED	CONC/SUPPORTING ELECTROLYTE/SOLVENT	TEMPERATURE RANGE/^{o}C
		(SCAN RATE EXPERIMENTS)	
15	5	0.1M/$TBABF_4$/CH_3CN	Ambient for all
		(FILM LOADING EXPERIMENTS)	
10	10	0.1M/TBAP/CH_3CN	-40.10 to 46.82
15	12	0.1M/TBAP/CH_3CN	-40.00 to 46.82
20	12	0.1M/TBAP/CH_3CN	-39.90 to 48.52
		(SUPPORTING ELECTROLYTE/SOLVENT EXPERIMENTS)	
5	12	0.1M/TEAP/CH_3CN	-41.21 to 48.31
15	12	0.1M/$TBABF_4$/CH_3CN	-39.95 to 48.61
15	9	0.1M/TBAP/C_6H_5CN	-11.90 to 46.21
30	12	0.1M/$TBABF_4$/C_6H_5CN	-11.60 to 48.44
5	11	0.1M/TBAP/C_3H_7CN	-45.31 to 48.52
10	12	0.1M/$TBABF_4$/C_3H_7CN	-46.92 to 49.67

cipitation of PVF films, a Ag wire was used as the reference and a Pt-foil(dia.=2.5cm, len.=5.0cm) was used as the counter electrode. PVF thin films were prepared as follows. The Pt-button electrode was polished on felt containing a slurry of DD water and Buehler 1.0um alumina; rinsed in DD water; polished again on felt containing DD water and Buehler 0.3um alumina; rinsed again in DD water; soaked in concentrated HNO_3 for ca. ten minutes; rinsed finally in DD water; and air dried.[139] Before precipitating a film, the Pt-foil electrode was also soaked in HNO_3; rinsed in DD water; and air dried. PVF was electroprecipitated in its oxidized form as poly(vinylferrocenium) perchlorate or tetrafluoroborate onto the Pt-button from a stirred solution of 25.0ml $MeCl_2$ containing 5.0mg PVF and the same concentration of supporting electrolyte as that under study in Table I.[140] Deposition was accomplished by maintaining the potential of the Pt-button electrode at 0.7V without iR_u compensation. To achieve more uniform film deposition and enhance symmetric current distribution, the Pt-button was surrounded by the cylindrical Pt-foil to a depth of ca. 1cm in the $MeCl_2$ solution. The quantity of polymer deposited was controlled by the electroprecipitation time(Table I).[72] Following electroprecipitation, the Pt-button was cleansed of excess $MeCl_2$ by gently touching it to a soft tissue and baked at 120°C for 20 minutes. The electrode was allowed to cool to room temperature before any voltammetry was performed. Prior to the electroprecipitation of any film, the polishing-rinsing-HNO_3 treatment was performed.

On the same Pt-button, cleaned as described above, thin films of the TCNQ-TPE were cast. Two drops of a solution(5.0mg TCNQ-TPE/5.0ml

$MeCl_2$) were pipetted onto the surface of the electrode held in an inverted position and the solvent vacuum evaporated. This is in contrast to the DMA/acetone solvent used by Karimi,[73] but the results were the same(Figure 10). These films were then baked at 125°C for 30 minutes. For both PVF and the TCNQ-TPE, the baking process improved markedly the adherence and stability of the films for subsequent electrochemical studies. Even so, some peak height attenuation could be observed upon multicycling. To avoid the problem of faradaic current loss and possible complications in the calculations, no more than four or five CVs were recorded on the same film. Careful attention to electrode cleaning and duplication of film casting procedures gave reproducible electrochemical results among different films. From Table I, it is evident that more than one film had to be cast in each solvent system. To compare results over the stated temperature ranges, the films had to be reproducible($\pm$5%).

Apparatus

Two electrochemical cells were fabricated for cyclic voltammetric studies. They were made of thick-wall, Pyrex glass of i.d. 3.2cm, len. 10cm, and had glass rods for sidearms. One cell was used to precipitate PVF thin films and the other contained the supporting electrolyte/solvent system under study. The sidearms were necessary to immerse the one cell in a Dewar flask containing absolute ethanol and to hold the other cell stationary relative to the motion of the stir bar. Teflon caps added additional stability to the electrodes and were outfitted with four holes, three to accommodate the working, reference,

and counter electrodes and the fourth an NMR tube which contained the thermistor probe assembly for temperature measurement. Of course, only one cell was used for the TCNQ-TPE studies. Cold temperature experiments were performed by gently placing small chunks of dry ice in the ethanol with forceps and waiting at least five minutes for temperature equilibration; similarly, a temperature controller and heating bar were used to obtain the data for high temperature studies. It should be mentioned here the importance of temperature equilibration. Cyclic voltammetry is a quiet technique;[138] temperature gradients from the ethanol surrounding the cell could have induced convection in the solvent. Dry ice bubbling in the ethanol and brushing the sides of the cell could have caused the same problem.

C. Chronoamperometric Analysis

Introduction

An Apple II plus microcomputer interfaced to the BAS CV-27, and another set of Pascal programs were used to collect and analyze chronoamperometric data. In this application, the area of the Pt-button electrode described above was determined; the electrode area was needed to calculate surface coverages.[37] An additional software option was included to compute diffusion coefficients if the electrode area was known. The same cell, Teflon cap, electrodes, electrode cleaning procedures, and electrode configuration were used here as in the CV studies. The data were permanently stored on disk file for subsequent analysis with Pascal software.

Chemicals

Potassium ferrocyanide trihydrate, $K_4Fe(CN)_6 \cdot 3H_2O$, (Merck, ACS grade), and potassium ferricyanide, $K_3Fe(CN)_6$, (Merck, ACS grade), were recrystallized from DD water, filtered, and rinsed again with 2, 10-ml aliquots of cold, DD water. The crystals were then crushed in a mortar and pestle and vacuum oven dried at 30°C overnight. Potassium chloride, KCl, (Fisher, ACS grade), was recrystallized in absolute ethanol, filtered, washed, and vacuum oven dried at 30°C also. All chemicals were held under desiccation until use.

Procedures

To calculate the electrode area, 50.0ml of 0.1M KCl/DD water were placed in a cell and thoroughly degassed. A background CV was run on a BAS-100 electrochemical analyzer to be sure dissolved oxygen had been purged. The background chronoamperogram was recorded by the Pascal program CADATA, then $K_4Fe(CN)_6 \cdot 3H_2O$(4.0mM) was added to the solution, which was subsequently stirred and degassed. CV was performed again to assure no current response from oxygen. The current versus time response of ferrocyanide was recorded. From the current, time data and the known value of the diffusion coefficient, the electrode area was calculated.[141,142] As an additional check on the area obtained, the electrode was used to calculate the diffusion coefficient of ferricyanide and compared to literature results. In both analyses three separate experiments were performed.

D. Instrumentation

Potentiostats

Two potentiostats, a BioAnalytical Systems(BAS) CV-100 and CV-27(Series MF9030), were used in this study. The BAS CV-100 automates several electrochemical experiments including cyclic voltammetry and can provide rapid analysis and output of electrochemical data.[143] For that reason, it was used in initial studies of PVF, $K_4Fe(CN)_6$, $K_3Fe(CN)_6$, and the TCNQ-TPE to determine i_ps and E_ps of these solutes in their respective electrochemical systems. In the case of PVF, it was necessary to ascertain optimum electroprecipitation times so that Γ_Ts near 10^{-9}mol/cm^2 at 25°C could be obtained and ΔE_ps maintained as close to zero as possible.[35-37] These initial studies also provided information on reversibility, scan rate, and potential limits in the various systems. They were also done without iR_u compensation, owing to the fact that instrumental iR_u compensation was not possible on the BAS CV-27 where final experiments were performed. CV of the TCNQ-TPE films was performed on the BAS-100 to ascertain the same experimental parameters, among them the optimum number of drops to evaporation-coat for ideal, thin film behavior. Finally, the BAS-100 was used to determine the step potentials for oxidation of $K_4Fe(CN)_6$ or reduction of $K_3Fe(CN)_6$ in chronoamperometry.

With this information in hand, final experiments were performed on the CV-27 and data acquired through the Apple II plus interface. For cyclic voltammetry, the front panel knobs, DISPLAY, INITIAL E, and SCAN RATE were used to set potential limits and the sweep

rate(Figure 12).[144] The FUNCTION knob was rotated to E1 which was set to the initial potential prior to each scan and the CELL MODE left in STBY before the PME was immersed in the electrochemical cell. The GAIN(sensitivity) knob was set to the appropriate mA/V setting depending on current values from the BAS-100 experiments and the amplifier setting choices of the ADC.

The rear panel was used to connect the CV-27 to the electrochemical cell, power source, and interface hardware. During CV, the Filter switch was set to 2, I Polarity was set to +oxn., and E Polarity was set to +. Shielded coaxial cables were used to connect the I Out and App. E ports to two of 16 possible input channels of the ADC. The importance of using shielded cables and a common ground between the computer interface and potentiostat cannot be understated. For accurate measurement, the voltage output of the CV-27 must be accurately reflected at the ADC inputs. This method of differential wiring maintains voltages that are constant and in phase, preventing the superimposition of interference noise on the signal.[4] This is completely analogous to using a differential amplifier to remove 60Hz interference noise from an output signal.[22]

The rear panel of the CV-27 also has a 15-pin female connector called the Remote Timing Input. By contact closure of any pin(2-8,15) to digital ground(pin 1), functions SCAN, CELL, E2, E1, +DIR, -DIR, INTG, and/or RESET can be initiated.[144] To synchronize data acquisition by the computer with the start of the experiment, a DAC was used to actuate a MOSFET switch. This triggering device consisted of a small BUD box, 15-pin male connector(Radio Shack), N-channel enhance-

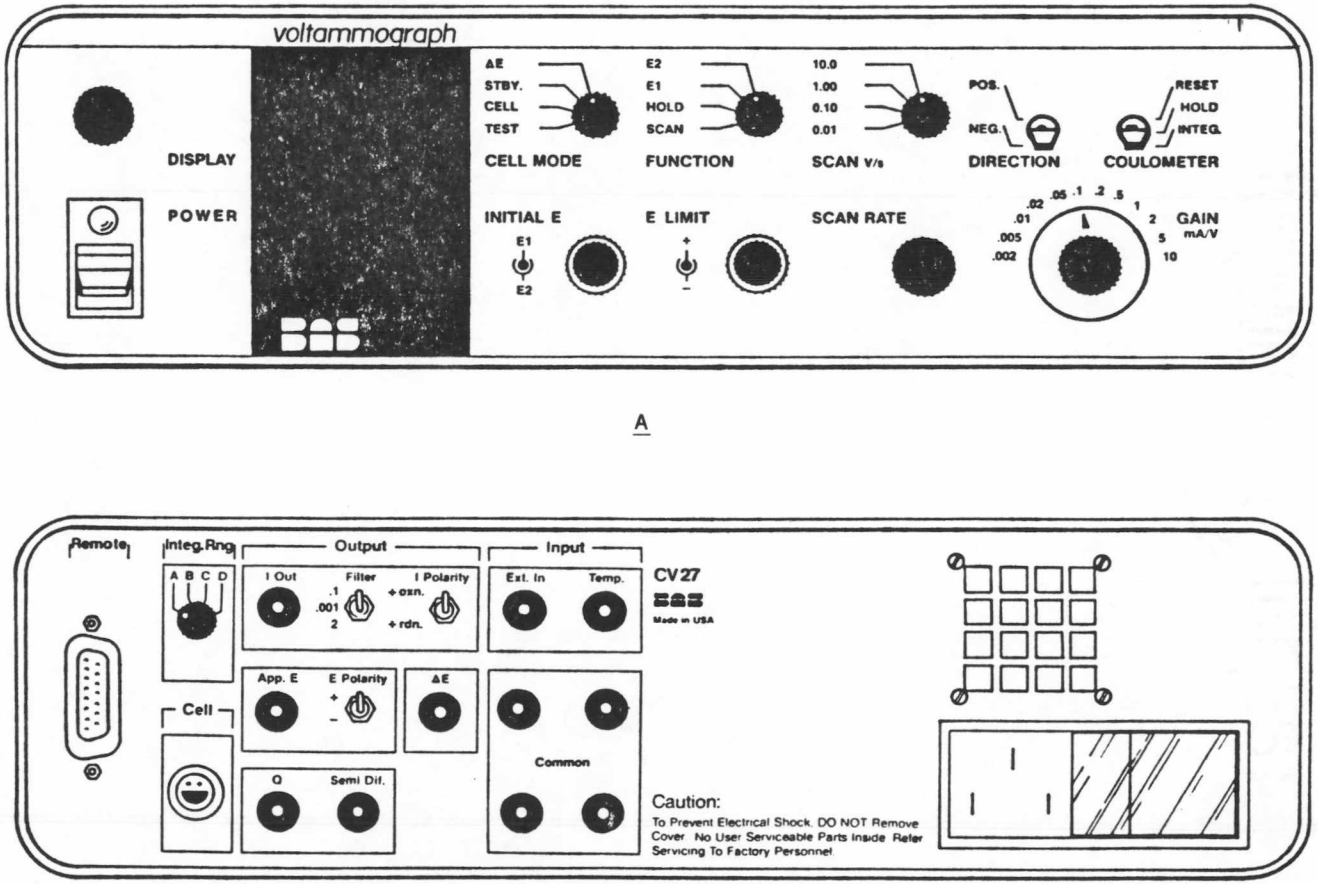
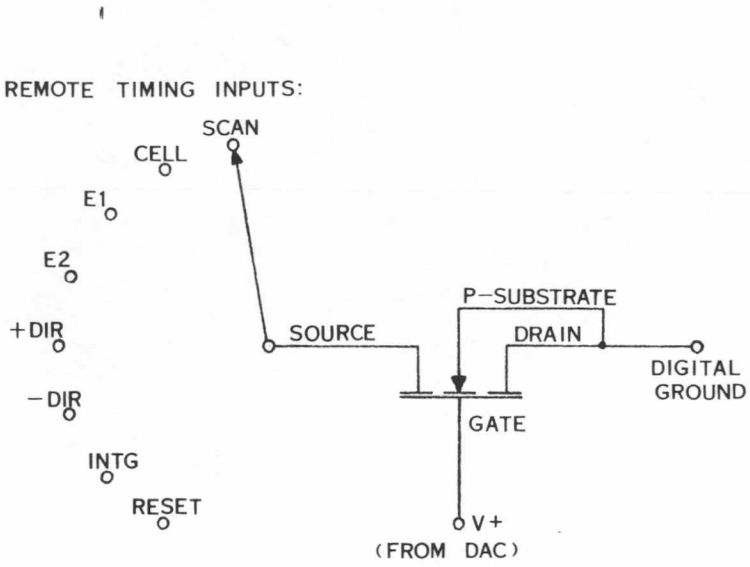


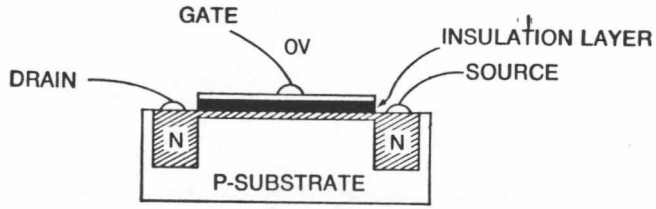
Figure 12. (A) Front panel, and (B) rear panel of the BAS CV-27.

ment MOSFET(RCA, SK9158/465), 10-position rotary switch, one female BNC, and some simple wiring.[145,146] A schematic of the circuit which was constructed is shown in Figure 13. During CV, the rotary switch was positioned at SCAN. Application of OV at the gate restricted the N-channel in ca. 60 nanoseconds so that no current could flow from source to drain. Conversely, application of 5V at the gate opened the N-channel in ca. 45 nanoseconds permitting current to flow and connecting the pin to digital ground. In Figure 12, one can also observe an input port labelled Ext. In. During chronoamperometry, the potential step from the DAC was transmitted to the working electrode through this port. In the event sequence, the initial potential(E1) was applied as soon as the user had input the value, the cell was switched from STANDBY to CELL, the final potential(E2) was applied, and current data immediately acquired by the ADC. The I Polarity switch was set at +oxn. for $K_4Fe(CN)_6$ experiments and +rdn. for $K_3Fe(CN)_6$ experiments; the Filter switch was set at 0.001.

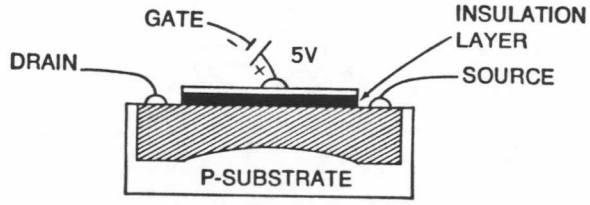
The Apple II Plus

Architecture of the 6502. The Apple II plus computer has a 6502-type of microprocessor integrated circuit(IC) at the heart of its central processing unit(CPU). In the CPU, the actual mathematical, logical, decision-making, and timing operations take place.[147] The II plus-board is specified as Revision 1 by Apple Computer because of its electronic upgrade from the older Apple II, Revision 0 board. Synertek Corp.(Santa Clara, CA) and MOS Technology(Norristown, PA) manufacture





OFF CONDITION



ON CONDITION

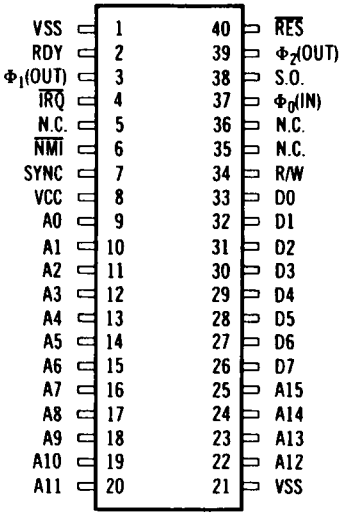
Figure 13. The application of an N-channel enhancement MOSFET as a computer controlled switch.

the 6502 microprocessor and other related ICs which are used on the Revision 1 board.[146,147]

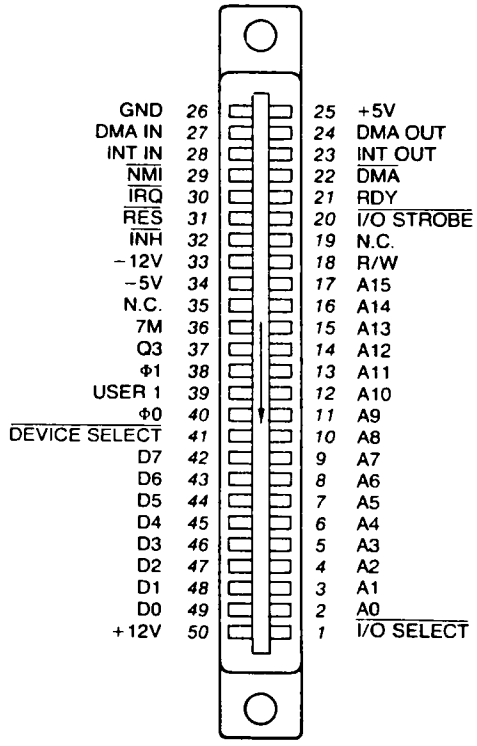
The 6502 is an 8-bit microprocessor; thus, all of the mathematical, logical, data transfer, input and output(I/O) operations manipulate eight binary bits, or one byte, at a time and at a rate of 1,023,000 clock cycles per second(1.023MHz). It has an address range of 65,536 bytes(64K), and its repertory includes 56 instructions with 13 addressing modes. On the board itself, there are about eighty ICs and a handful of other components. In front of the eight slots at the rear of the board is the largest IC, the 6502, whose pin configuration is shown in Figure 14A.[146,147] Sixteen pins on the 6502 are dedicated to addressing permitting control of the two possible logic levels on each of the 16 lines that form the address bus. The address pins on the 6502, A15...A0, determine a 16-bit binary number called the address of a memory location. Lines A7...A0 are referred to as the LO address, while lines A15...A8 represent the HI address. The HI address is also called the page address, since the memory may be arbitrarily divided into 256 pages of 256 bytes per page. Sometimes, the HI and LO order byte are each written as two hexadecimal digits from the set 0...9, A...F and preceded by a $. Then, the entire 16-bit address can be expressed in four and only four hexadecimal digits.[148]

Eight pins on the 6502 in Figure 14A are connected to the data bus and are labelled D0...D7. The READ/WRITE(R/W) pin on the 6502 is connected to a line of the control bus called the R/W line. A READ operation(R/W line is at logic one) causes eight bits of information to be transferred over the data bus from the memory location specified by

A



B



DATA BUS	D7-D0	An 8-bit bidirectional set of lines for transfer of information between the CPU and I/O devices.
ADDRESS BUS	A15-A0	A 16-bit unidirectional address bus used to address both memory and I/O devices.
	A15-A8	HI address bus, most-significant eight address bits.
	A7-A0	LO address bus, least-significant eight address bits.
CONTROL SIGNAL	R/$\overline{W}$	Read/write control signal.

NOTES: The "bar" notation, i.e., $\overline{W}$, indicates a logic zero is the "active" state, the state that causes the corresponding action to take place.

In each case in which a signal is enumerated, the numbers increase as the significance of the bits increases, i.e., A15 = most-significant address bit (MSB).

Figure 14. (A) 6502 microprocessor pin configuration, and (B) peripheral connector pinout.

the address bus to an 8-bit register in the microprocessor. Conversely, a WRITE operation(R/W line is at logic zero) causes eight bits of information to be transferred from a register to the memory location specified on the address bus. The words "load" and "store" are frequently used synonymously with the words READ and WRITE, respectively.[148]

A register is an 8-bit storage location in the microprocessor. It is used to store data for subsequent calculation by the microprocessor. The contents of a register may also control the operation of the microprocessor itself. For programming in 6502 Assembly language, the most commonly used register is the accumulator. Other registers in the 6502 include the index registers, X and Y; the processor status register(P); the stack pointer(SP); and a pair of registers called the program counter high(PCH) and the program counter low(PCL).[34,149] More comment about the registers and instruction set will be made in the programming sections which use them.

Memory Allocation. All computer systems have some memory associated with them. In general, the memory is used to store both a program (or set of programs) that will control the operation of the computer, as well as information that is to be processed.[146] Some of the ICs on the Apple board already contain programs including the System Monitor, the Apple Autostart Monitor, Apple Integer Basic, and Applesoft II Basic, and are available as soon as the power is turned on.[147] They are contained in what are collectively called read-only memory(ROM) and cannot be altered by a write instruction. The purpose

of having ROM locations is to store frequently used programs and data that the user does not want to be altered. For example, the programs which control interaction between the keyboard, monitor, and 6502 are located in ROM. The Autostart Monitor program initiates the process when power is first applied so that the user can immediately interact with the computer via the keyboard.[147]

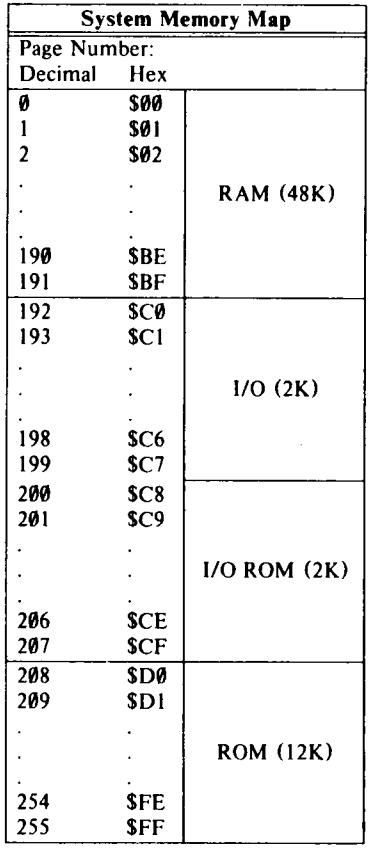
The other type of memory is called R/W and data can be written to or read from it. This type of memory is used for the storage of data that will be changed or updated.[146] This is also where information such as program code is written so that it can be read back by the 6502. The R/W memories may be either static or dynamic. Static memory chips will maintain the values stored in them until they are changed. Dynamic memories require refreshing by external hardware every few milliseconds or they will forget data stored in them.[1] The II plus' memories are dynamic with the necessary refreshing circuitry contained on the main board. Sometimes the notation RAM is used to incorrectly signify R/W memory.

Even with the associated memory, most microcomputers would be worthless without some attached I/O devices.[146] These devices may be standard peripherals such as a printer, disk drive, or console which communicate with the outside world or they may be devices such as R/W memories(RAM expansion) or 80-column cards which expand the internal capabilities of the microcomputer. Other I/O devices can be added to the Apple depending on application needs by adding printed circuit cards to slots 0...7 in the rear of the case. In this particular application, standard, commercially available devices were purchased

that were directly compatible with the Apple. To the 6502 microprocessor, these peripherals appear as individual memory locations and are connected in parallel to the data and address buses so that they may be uniquely addressed by the microprocessor.[146-148] In the peripheral connector pinout of Figure 14B the R/W pin synchronizes the flow of data to and from a peripheral. There is no distinction between memory addresses and I/O devices in 6502-based microcomputers. Any location can be accessed by placing its address on the address bus. This technique is called memory-mapped I/O and allows peripheral devices to be activated by a unique address.[1]

An excellent summary of memory use in the Apple is shown in Figure 15.[147] The basic layout is shown by both page and hexadecimal notation. For consistency, R/W memory will be referred to as RAM from now on. The RAM area in 15A is further subdivided in 15B to perform user-defined tasks. It starts at page zero, extends to page 191, and spans 49,152 bytes(48K) of memory. This RAM area can be expanded to 64K by adding at least 16K more memory to peripheral slot 0. This extra 16K of RAM takes the place of the Apple's ROM, with two 4K segments of RAM sharing the 4K range from $D000 to $DFFF.[147]

Most of the Apple's RAM is available for the storage of programs and data. Some locations however, are reserved for use by the System Monitor, various operating systems, and other system functions. When the additional 16K bytes of RAM are added to the Apple and the Pascal operating system loaded, the memory appears as shown in Figure 16. The lower 48K of Figure 15B coincides exactly with the allocation in Pascal, but the upper 16K of the language card contains two additional



System Memory Map		
Page Number: Decimal	Hex	
0	$00	RAM (48K)
1	$01	
2	$02	
.	.	
.	.	
.	.	
190	$BE	
191	$BF	
192	$C0	I/O (2K)
193	$C1	
.	.	
.	.	
.	.	
198	$C6	
199	$C7	
200	$C8	I/O ROM (2K)
201	$C9	
.	.	
.	.	
.	.	
206	$CE	
207	$CF	
208	$D0	ROM (12K)
209	$D1	
.	.	
.	.	
.	.	
254	$FE	
255	$FF	

A

Page Number: Decimal	Hex	Used For:	
0	$00	System Programs	
1	$01	System Stack	
2	$02	GETLN Input Buffer	
3	$03	Monitor Vector Locations	
4	$04	Text and Lo-Res Graphics Primary Page Storage	
5	$05		
6	$06		
7	$07		
8	$08	Text and Lo-Res Graphics Secondary Page Storage	FREE RAM
9	$09		
10	$0A		
11	$0B		
12 through 31	$0C through $1F		
32 through 63	$20 through $3F	Hi-Res Graphics Primary Page Storage	
64 through 95	$40 through $5F	Hi-Res Graphics Secondary Page Storage	
96 through 191	$60 through $BF		

B

Figure 15. (A) The Apple's basic memory map, and (B) reserved RAM locations.

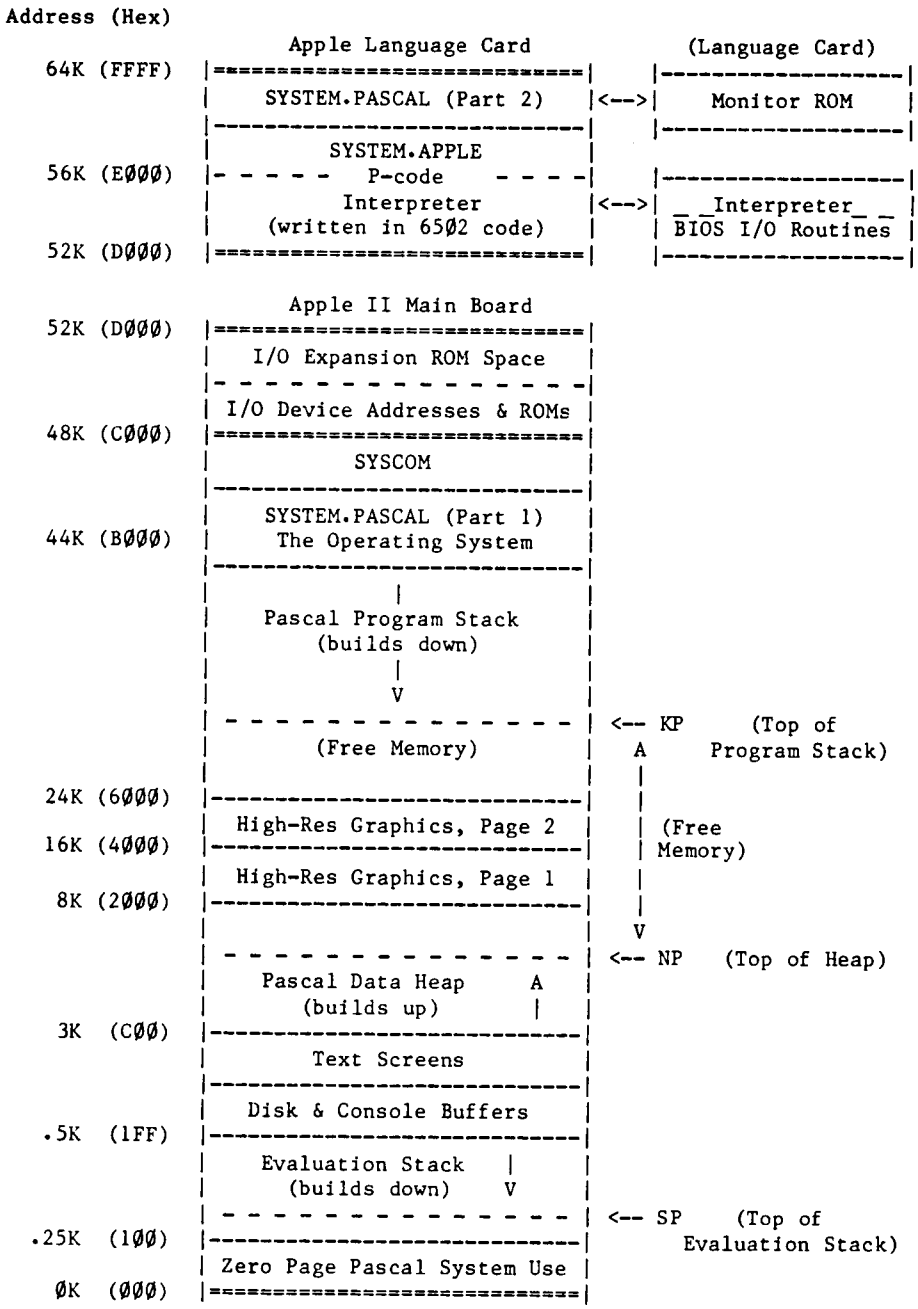


Figure 16. Memory map of the Apple II plus when using Apple Pascal 1.1.

system files which interpret Pascal code and display the command level of the operating system when it is first booted.[150]

Peripheral Devices

Introduction. When choosing peripheral devices for an experimental I/O system in a laboratory, questions concerning cost, size, speed, storage capacity, noise production and compatibility must be asked. This is of particular concern in the case of ADCs and DACs.[1,14,145] For standard I/O devices, communication between the computer and peripheral device is handled by a hardware interface, carefully designed to match the Apple's peripheral pinout. Table II shows the slot location of the various peripherals used in this laboratory application.

The ADC and DAC are designed to communicate with an analog world. In conjunction with this RAM board, the ADC and DAC formed an I/O system that differentiated it from other computer systems. Since the electrochemist is normally interested in monitoring cell parameters such as current, potential, resistance, or capacitance, these devices were destined to be implementation specific and require more attention from the programmer to meet application needs.[1]

The 80-Column Card and External Clock. In microcomputers manufactured after 1980, 80-column displays and high-resolution monitors became a de facto standard. The Revision 1 board of the II plus, manufactured in 1979, was designed for a 40-column screen of 24

Table II. Peripheral device configuration of the Apple II plus microcomputer.

SLOT	PERIPHERAL DEVICE	PURPOSE
0	Titan Technologies, 128K RAM board	Emulation of a disk drive for rapid execution of PASCAL programs.
1	Interactive Structures, PKASO printer interface	Control of graphics and text output to an Epson MX-100 dot matrix printer.
2	Interactive Structures, AI13, A/D converter	Acquisition and conversion of analog data from chemical instrumentation.
3	Applied Engineering, Viewmaster-80	Provide 80-column display.
4	MicroPro StarCard	Contains Z80-B microprocessor for execution of CP/M and WordStar.
5	Thunderware, Thunderclock	Keep track of the real time and date even when the system is powered down.
6	Apple II, disk drive controller	Receive and transmit data between the disk drives and microprocessor.
7	Interactive Structures, A003 D/A converter	Convert and transmit a digital signal to chemical instrumentation.

lines.[147] To upgrade our system, a Viewmaster 80-column card was selected.[151] It had to be compatible with the Pascal and CP/M operating systems. CP/M is the operating system WordStar required to produce this manuscript. The Viewmaster-80 provides 24 lines of 80 characters with a relatively sharp 7 * 9 dot matrix display. Installation involved simple cable changes between the computer, video monitor, and Viewmaster. When initialized in Pascal, the Viewmaster defaults to upper case only mode. Typing <CTRL-A> gives lower case characters; upper case characters are then typed by holding down the shift key. Additional on-board RAM characters are available in combination with the <ESC> key.[151]

Another peripheral which has become a standard addition to microcomputers is an external clock. The Thunderclock Plus(Thunderware Inc.) installed in this system keeps track of the real-time and date even when the computer is powered down by drawing power from two, 1.5V alkaline "N" cells mounted on the board itself.[152] It has three software-selectable interrupt rates of 64, 256, or 2048 events per second and an on-card 1K byte EPROM which contains all of the firmware necessary to interpret Pascal and Assembly commands for reading and setting the clock time. The same EPROM is fully compatible with Integer and Applesoft Basic commands. In this application, the clock was read with Pascal and Assembly code to ascertain the time and date experiments were performed and to automatically set the Pascal system date whenever Pascal was booted.[152,153]

The Printer and Interface. Because of the diverse communication protocols among microcomputers, the installation of printers has presented as many problems as any other peripheral.[3] Printers can be classified by their output quality, dot matrix or draft quality(DQ), near letter quality(NLQ), or letter quality(LQ).[154] Dot matrix printers are necessary for graphics output when pages of screen memory are dumped to the printer. They are cheaper and capable of higher print speeds or baud rate than their LQ counterpart at the expense of lower quality output. Technology has in part solved the problem of poorer quality output for DQ printers by manufacturing NLQ ROM chips which represent each text character in a higher resolution 7 * 9 dot matrix.[154]

In contrast, LQ printers typically have slower baud rates, cannot and were not meant to produce graphics output, and are more expensive. They print with a daisywheel instead of a print head for a very clean and legible output.[154] In some dedicated word processor systems, the cost of an LQ printer alone exceeds all of the other components and software combined. The output quality of a daisy wheel can be rivaled by state-of-the-art LaserWriter printers or DQ printers with an NLQ ROM chip.[154,155] Letter quality printers are also capable of incremental motion so that standard word processing features like boldfacing, superscript, subscript, double strike, and proportional spacing can be implemented.

When purchasing a printer, one must specify serial or parallel, the method of transmission from the computer port to the printer. Accordingly, the hardware on the interface card must understand this

method of transmission. The printer and interface used here for Pascal graphics and text output was a parallel Epson MX-100, dot matrix printer and PKASO EP12-80/100 interface(Interactive Structures). In the less standard, serial interface, data are sent along one wire in eight bit patterns, one bit at a time, in what is known as serial transmission. The computer is then capable of some respectable bit per second(bps) speeds ranging from 150 to 19,200 bps. Serial boards in the Epson can be adjusted to receive the data at similar speeds. At 19,200 bps, the Epson obviously cannot print the data as it is received because the printer plods along at a slower print speed. Therefore, a buffer in the Epson must catch the data overflow and then tell the computer to wait until it has more room. The serial interface is just a holdover from the teletype days of the computer's infancy. It has more important applications in the routing of telecommunications with modems.[1,3-5]

The more accepted method of transmission to a printer is parallel. The printer can accept one byte(8 bits) at a time across the data bus, D0...D7. Efficiency of transmission is greater, but transmission speed is fixed by the print speed of the Epson because only a minimal buffer is available at its parallel port. The parallel port's merit lies in its direct software application and elimination of the need for any major hardware modification. The PKASO EP12-80/100 was chosen for its full compatibility with all of the popular operating systems including Pascal.[156] It was one of the first graphics dump cards available that was completely controllable from software with no DIP switches. The PKASO interface uses an EPROM(EP12-80/100) to interpret simple commands

in Pascal and perform diverse printing functions on Epson printers such as HiRes screen dumps and changing the character size. This hardware combination facilitated output of voltammograms and the mathematical analysis of chronoamperometric and voltammetric data. [156,157]

Disk Drives and Memory Boards. No other peripheral device has had as significant an impact on the microcomputer revolution as the Apple II disk drive and controller card engineered by Steve Wozniak in 1977. There were several ideas which made the microcomputer revolution happen. Color, high-resolution graphics, BASIC in ROM, 48K of RAM, sound, and the use of a TV screen as the output device had never been implemented in a low-cost computer. The highlight of the Apple II project however was the completion of the Apple II, disk drive, controller card which permitted mass storage and retrieval of information from cheap, magnetically-coated, Mylar disks. The Disk II drives used in this study were the original drives designed for the Apple II and use 5 1/4-inch diameter disks that can store ca. 143K bytes of information. These disks are divided into 35 tracks, each containing 16 sectors.[159] Under different operating systems these sectors are grouped into units which are managed by directories, files, and other attributes so that I/O can be easily performed. This file division on the disk is accomplished by formatting programs.[150]

Technology has improved greatly on the original design of the Disk II drive by decreasing storage and retrieval time, compressing more data on yet smaller diameter disks, and including the controller card on the main board as a port. Because this port is not built into the

main board of the II, II plus, or IIe, a disk controller card with a pair of 20-pin connectors for the disk drive cables must be placed in slot 6.[160] Disks are memory devices broken down into files of various formats in different operating systems and may contain numerical data, text or executable program code. In Pascal, a series of files on one disk form the operating system when Pascal is booted. As a result, the heads of the disk drive must move from file to file to read information and perform various tasks such as editing, compiling, filing, linking, and assembling.[150] Physical movement of the disk and alignment of the heads is time consuming. An approach to circumventing this speed problem is to use a RAM card as a disk emulator.[161,162] A Titan Technologies, 128K RAM card was used for this purpose and since it occupied slot 0, one 16K bank was reserved to fulfill the 64K Pascal language requirement. Software was written to install the RAM card as an additional disk drive and automatically transfer operating system files such as, SYSTEM.PASCAL, SYSTEM.LIBRARY, and SYSTEM.EDITOR to it. In a similar manner, the programs written for cyclic voltammetry and chronoamperometry calculations were transferred to the RAM drive for faster execution speeds. At the same time this freed one of the Apple drives for storage of cyclic voltammetric and chronoamperometric data.[161,162]

DACs and ADCs. Because the microcomputer only works in the digital domain, it is necessary to convert analog signals to digital signals. It may also be necessary to convert the digital signals of the computer back to the analog domain to output the stored data on an

analog recorder. Similarly, both analog and/or digital signals may be needed for instrument control and this requires additional peripheral hardware.[1,14]

An understanding of how ADCs function begins with a brief look at how DACs function. DACs are used to convert an n-bit binary number into a corresponding analog signal. DACs can be used to generate waveforms, move the pen of a chart recorder, or set the potential of an electrode. However, they also serve as the heart of some of the most common ADCs.[4,14]

The principal components of a DAC include an analog reference source, a resistor network and output amplifier(Figure 17). Those DACs in which the reference source is external are called multiplying DACs. All DACs multiply to produce an output, but the designation, multiplying DAC, always refers to a model in which the user can apply an external reference.[1,145] When the data latch is loaded from the data bus of the computer's parallel port, the latch holds the binary input value until instructed to change. The binary values control the current produced from a resistor ladder network. There are two types of resistor ladders, binary weighted and R-2R. The binary weighted ladder sums a series of binary weighted currents at the input of an inverting amplifier which thus produces a voltage output proportional to the sum of the currents turned on. This ladder consists of a paralled network of resistors, R, 2R, 4R, 8R...$2^{(n-1)}$R, that can be electronically

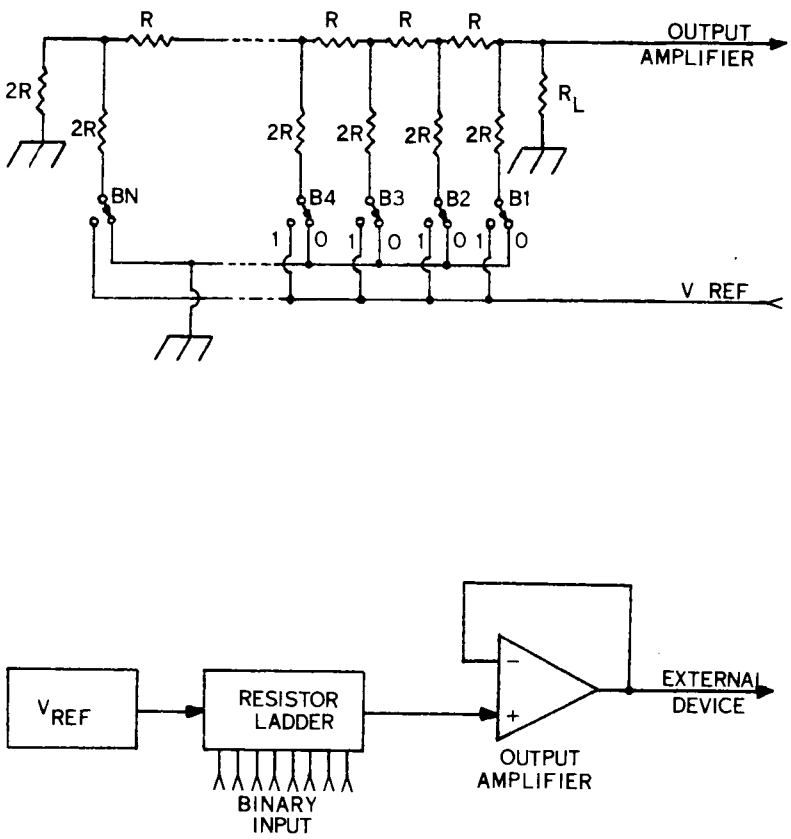


Figure 17. (A) Schematic of R-2R resistor ladder, and (B) block diagram of voltage output DAC.

switched to either V_{REF} if the digit is one, or ground if the digit is zero. Expressed mathematically,

$$I_{OUT} = V_{REF} \sum_{i=1}^{n} a_i/2^{(i-1)}R \qquad (2)$$

where n = number of bits and a = logic 1 or 0 for that bit. If the binary number is incremented, the analog output steps to a new potential. The number of possible steps is called the resolution(2^n) and is determined by the number of binary bits.[4,145]

Resolution above 8 bits is impractical because the resistor values at the ends of the ladder will become out of line, and tolerance matching is difficult. For example, if R is 10K ohms, bit 8 will be $2^{(8-1)}$ * 10K ohms, or 1.28 megohms. For a V_{REF} of +10.00V, current I_8 will be 10.00/1.28 megohms, or 7.8 microamperes. Finding operational amplifier outputs that will respond to input currents of this level is difficult and expensive. Similarly, making R too small produces currents that are too large.[145]

A solution to this problem is the R-2R ladder of Figure 17. The network maintains a binary weighting of bit currents but has only two resistor values.[145] If the load resistor(R_L) has a resistance so much

higher than 2R that its contribution to dropping the ladder output voltage is negligible, then E_{out} is given by eq. (3),

$$E_{OUT} = V_{REF} \sum_{i=1}^{n} a_i/2^i \qquad (3)$$

where a and n were previously defined.[145] In the Thevenin equivalent circuit, it can be seen that B1 makes a contribution $V_{REF}/2$ to the open circuit voltage, B2 a contribution of $V_{REF}/4$, B3 a contribution of $V_{REF}/8$, and so on.[145,163]

Besides resolution, a couple of other parameters which determine a DAC's suitability are linearity and settling time.[1,145] The linearity refers to the worst case difference between the actual output voltage and the straight line drawn through the output states. If the non-linearity exceeds $\pm 1/2$ of the least significant bit(LSB), the output will not be monotonic and there will be one or more codes where the output signal does not increase as the code increases. The settling time is the time required for the DAC to settle to its final value when the digital input is changed. This lag determines the maximum operating frequency of the DAC and is the worst when all bits change from '0' to '1'. When the input changes, the output voltage of the amplifier increases or decreases rapidly to the correct value but is limited by the amplifier's slew rate. The voltage overshoots the mark, then settles down to the correct value which will be a small voltage band clustered around the desired output potential.[145]

These basic principles explain the operating characteristics of the DAC used in this study. An 8-bit, four channel, A0-03, analog output system(Interactive Structures) with an R-2R resistor ladder was used to produce bipolar voltages of ±5V. The DAC had an LSB value of 0.039V, a linearity of ±1/2 LSB(monotonic), adjustable range and offset, output impedance of 500 ohms, and a 3 microsecond settling time. As shipped, the A0-03 was calibrated to a unipolar, 0 to 10 volt range on four channels, 0 through 3, using the Apple's internal voltage as V_{REF}. Jumpers were soldered across B0...B3 for each channel to obtain bipolar outputs. The DAC was then recalibrated by adjusting the zero offset trim pots Z0...Z3 when a code of 0.0V was applied to each channel. Output voltages were monitored with a Beckman 310, digital voltmeter. Then, the range was adjusted using trim pots F0...F3 when -5.000V, zero in binary, and +4.961V, 255 in binary, were applied to each channel. This process was repeated once due to a small amount of crosstalk between range and offset circuitry. The A0-03 can be further altered with jumpers at J0...J7 to accommodate external references and upgraded to an 8-channel output device.[164]

One of the most important components of the instrumention in this research was the ADC because it interfaced the digitally based 6502 to the analog outputs of the BAS CV-27. There are several basic approaches to ADC design including integrating, ramp, parallel, and successive approximation, each having certain advantages over the other.[4,145]

Integration ADC circuits use an operational amplifier integrator to create a ramp voltage from the unknown voltage being measured. Of

the three basic types, single-slope, dual-slope, and multiple slope, the dual-slope(DSI) is the most common. The principal components of the DSI are the integrator, comparator, main gate, counter, control logic section(CLS), reference voltage, and input switch. After integrating the unknown potential for 16 2/3 milliseconds, the CLS switches the input to a stable internal reference voltage of opposite polarity. At the same time, the CLS sends a reset pulse to the counter to reset it to zero. At the beginning of the ramp-down, the counter is started. It stops when the integrator reaches zero because the comparator is ground referenced. The digital value held in the counter is proportional to the unknown analog potential. Although a DSI's acquisition rate is too slow for our application, it has the advantage of rejecting 60 and 120Hz noise because it integrates the unknown over the period of the AC line voltage.[22,54]

The binary ramp, ADC circuit makes use of a DAC in a feedback loop to make a comparison. The major components in this ADC are a DAC, comparator, main gate, binary counter, reference source, and CLS. When a start pulse is received by the CLS, the binary counter is reset and a gate-on signal is applied to the main gate. The binary outputs of the counter are connected to the digital inputs of the DAC, so that the gate can cause clock pulses to ramp the DAC voltage upwards from zero to the unknown voltage. When the DAC's voltage equals the unknown voltage, the comparator drops low turning off the main gate and stopping the counter. The binary count is then proportional to the value of the analog input voltage. While the binary ramp ADC is faster than most integration methods, the time required to make a conversion de-

pends upon the unknown voltage. The worst case is when the voltage is close to full scale and usually the benchmark for measuring ADC performance. The conversion time, T_c, equals 2^n/clock speed, where n is the bit length of the counter. The variable conversion time made this ADC inappropriate for our application.[4,145]

Very few ADC circuits are as fast as the parallel converter. It operates at such blazing speeds that it is sometimes called the flash ADC. It consists of a stack of voltage comparators; the non-inverting input of each comparator is biased one LSB higher than the next one down the chain with a precision resistor network tied to a DC reference source.[145] In this approach, there are 2^n separate comparison amplifiers for an n-bit converter and the conversion is limited by the speed of these amplifiers and the decoding logic that follows it. Unfortunately, normal problems with real voltage comparators limit the value of the LSB and in turn the total bit length. Because the output is not encoded in any of the standard binary codes, a decoder logic circuit at the output of the comparators is needed to make the unit compatible with the outside world. This slows the conversion time considerably.[4,145]

One ADC circuit that is relatively fast and has a fixed conversion time is based on the trial and error method of successive approximation(SA). In the block diagram of Figure 18, the control logic, shift register, and output latches are contained in the successive approximation-register(SAR). When the SAR receives a start pulse, register B1 is set to '1'. The most significant bit(MSB) sets the DAC to half of its full-scale output voltage. If the comparator indicates

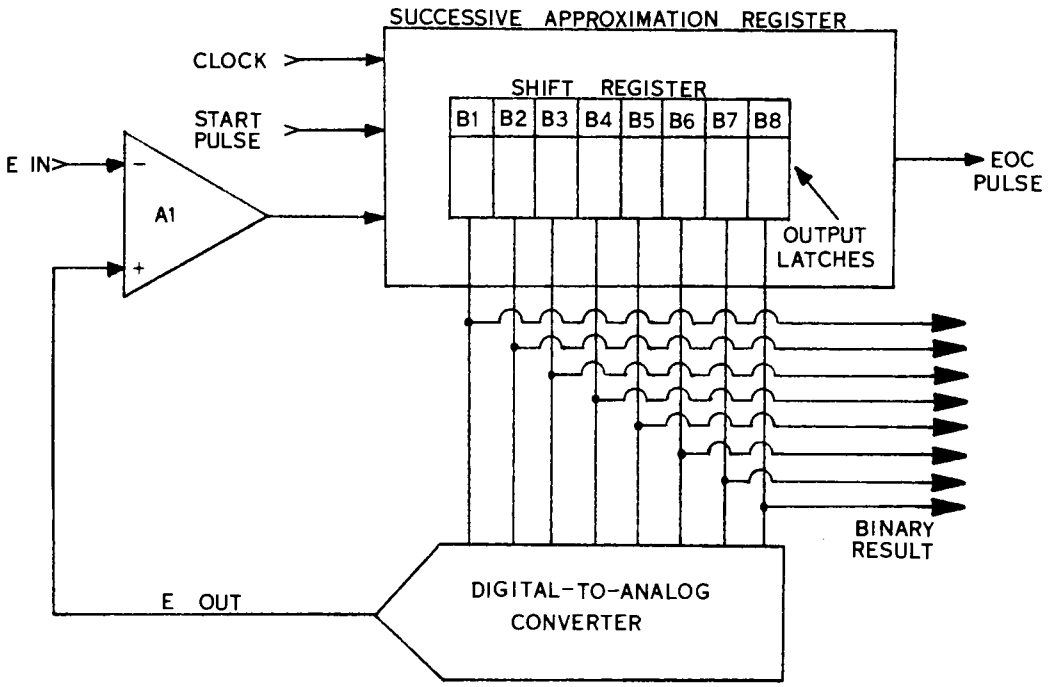


Figure 18. Block diagram of SA ADC; A1 = comparator, EOC = end-of-conversion pulse.

E OUT $\leq$ E IN, B1 is latched at '1'; otherwise, it is latched at '0' for all subsequent tests. On the next clock pulse, B2 is set to '1' and the testing continues on all bits until the LSB(B8) is tested. When testing is complete, an end-of-conversion(EOC) pulse is issued to the CPU. The SA ADC converges on the binary value in a variable time period, but completes conversion in constant time, so that,

$$T_c = [(\text{number of bits} + 1)/\text{clock speed}] + \text{software overhead} \quad (4)$$

While this converter has a quick T_c, it does not reject common mode noise like integrating circuits. Unlike the parallel converter, its binary output is directly compatible with the 6502's data bus.[1,4,145] A 16-channel, SA ADC(called the AI13) was purchased from Interactive Structures to interface the Apple II plus to the CV-27. The AI13 can be programmed to return a binary value between 0 and 4095 corresponding to 12-bit resolution. It can select one of 16 input channels, scale the input according to any of eight full-scale ranges, and return the result in less than 20 microseconds(Figure 19). Unipolar and bipolar input ranges of 0 to 5000, 0 to 1000, 0 to 500, 0 to 100, -5000 to 5000, -1000 to 1000, -500 to 500, and -100 to 100 millivolts can be selected. The input impedance of each channel is 10 megohms and crosstalk from unselected channels is -95db.[165]

When a conversion is requested by the program, the input channel in Figure 19 is selected by a 16-channel multiplexer. Multiplexing allows a single ADC to share several inputs. The result is applied to the range select circuitry which provides amplification based on the

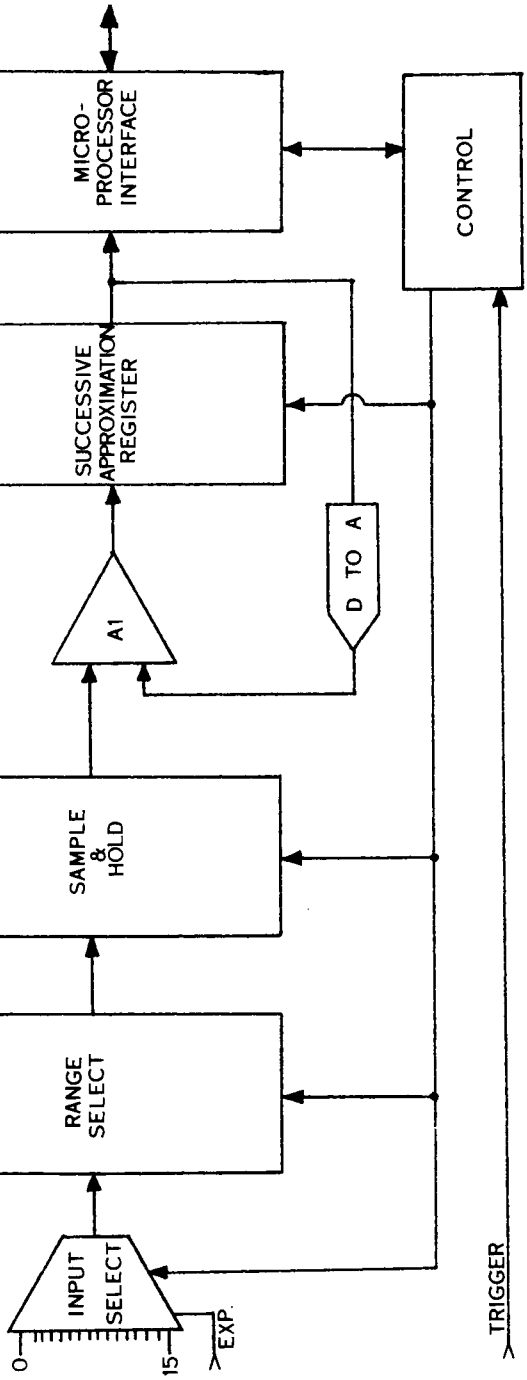


Figure 19. Block diagram of AI13 with SAR and associated control hardware.

selected input range. The analog input voltage is sampled for a six microsecond interval, then held constant by a sample and hold amplifier with an aperture time of 125 nanoseconds. The sample and hold hardware adds to the software overhead and is necessary to hold the analog input voltage constant over the 13 microsecond conversion period. The most significant four bits are completed after 11 microseconds and the entire result is ready after 19 microseconds.[165]

The input signal on any channel may be examined at various magnifications using 5:1 and 10:1 amplification of the 0 to 5.0V and -5.0V to 5.0V ranges. The 10:1 amplification function of the AI13 is designed for maximum stability at the expense of more software overhead. The bandwidth of the 10:1 amplifier is sufficient to handle a 20kHz signal. When channel and gain settings are changed dynamically however, this amplifier experiences voltage changes considerably faster than this. This poses no problem on the 5V and 1V ranges, or when repetitively reading the same channel at any range. The limitation appears when the gain is switched to a 0.5 or 0.1V range from any other range or when changing channels on a 0.5V and 0.1V range. In these cases, the software must first set the gain and channel, delay for at least 45 microseconds, then perform the conversion.[165]

The AI13 consists of two electronic modules; the first is an interface logic board, and the second is a yellow analog board which mounts outside the computer in a shielded cabinet. The two boards are connected by a 34-conductor ribbon cable. For both the AO-03 and AI13, a color coded ribbon cable was soldered to a set of female BNC connectors mounted in BUD boxes on the computer rack. Some additional

features of the AI13 are that it can be programmed to interrupt the 6502 when conversion is complete, externally triggered to start conversion, and customized by jumpers to change amplifier gains or voltage ranges.[165]

Word Processing, CP/M, and the StarCard. The peripherals described above present some interesting possibilities for word processing. The laboratory computer already had an 80-column monitor, printer and interface, disk drives and some aditional memory. Also, the 80-column and printer interface cards are CP/M compatible. WordStar is the best word processing software for pre-Apple IIe computers such as the II and II plus.[166] Exclusive of WordStar, there is already a large quantity of CP/M software available for the Apple which could further expand its use.[166] However, CP/M uses the 6MHz, Z-80B microprocessor, not the 6502, so a peripheral card, the StarCard, which contains the Z-80B chip and 64K of RAM was placed in slot 4.[167] The Z-80B on the StarCard and the Apple's 6502 run simultaneously at their full-rated speeds and communicate through a decoded parallel port. Further, the presence of the StarCard in slot 4 does not interfere with other operating systems or programs which use the 6502.

All of these discussions about peripherals have pointed out the necessity to consider operating system compatibility and application needs. Peripheral purchases were carefully scrutinized to maximize their versatility in Pascal, CP/M, and the laboratory environment.[14]

Thermistors and Temperature Measurement

Interaction parameter and thermodynamic calculations are all a sensitive function of temperature.[37,38] Preferably, a temperature measurement would be accurate to $\pm 0.5^{\circ}C$, recorded automatically by the computer immediately before and after the acquisition of a voltammogram, and averaged before placing it in a data file. These criteria led to an investigation of resistance thermometry, especially thermistors, for temperature measurement. The essential requirements for precise resistance thermometry are a resistor properly mounted and protected, a means of measuring its resistance, and a mathematical relation between resistance and temperature.[168]

Thermistors are electrical circuit elements formed of solid semiconducting materials which are characterized by a high negative coefficient of resistivity.[168] The two electrical temperature sensors most widely used today are the resistance thermometer and thermocouple. In comparison to them, thermistors have these advantages:a temperature coefficient of resistance about ten times that of metals, with a correspondingly greater sensitivity to temperature change; a much higher resistivity than the metals, so that small units may have high resistance, virtually eliminating the lead and contact resistance problems;and, no cold end or lead material compensation is necessary since the thermistor resistance is a function of its absolute temperature. For a limited temperature range, the thermistor combines all the best features of resistance thermometers and thermocouples and has greater sensitivity than either.[168]

There are however some limitations associated with thermistors. The R-T behavior of thermistors is highly non-linear; the two equations most commonly used to relate the two terms are only approximations. One equation that describes the R-T curve reasonbly well over narrow temperature ranges is,

$$R_{T2} = R_{T1}\ e^{\beta[(1/T2)-(1/T1)]} \qquad (5)$$

where T1 and T2 are temperatures in Kelvin, R_{T1} and R_{T2} are the thermistor's resistance at T1 and T2, and β is a constant determined by measuring the thermistor at two known temperatures.[63,66] To accommodate wider temperature ranges(ca. 100°C), the Steinhart and Hart equation is used,

$$T^{-1} = a + b\ln R + c(\ln R)^3 \qquad (6)$$

where T is in Kelvin, R is the thermistor's resistance, and a, b, and c are exponentially determined constants. For precision work, the thermistor has to be subjected to extensive point-by-point calibration to determine the best values for a, b, and c.[169] What's more is that typical thermistors will have tolerances of ±20% down to ±5%, so that thermistors of a given type cannot be used interchangeably. Ironically, the high sensitivity of thermistors limits their useful temperature range from -80° to +150°C. Imagine the difficulty in trying to design a circuit that measures a resistance change of 10,000 or 20,000 to 1 between -80° and +150°C. Stability considerations also

limit high temperature use. Thermistor structures tend to change if left at temperatures above 100°C for long periods of time and can cause temperature deviations of several degrees. Yet another possible limitation is a thermistor's dissipation constant(DC), expressed in milliwatts/°C. If DC = 1mW/°C, one milliwatt of internal power will raise the thermistor's internal temperature by 1°C. Keep in mind that as the thermistor's resistance changes with temperature so does its power dissipation. Like any thermometer, the mass of the thermistor must heat up or cool down to match its surroundings. Depending on a thermistor's size, surrounding medium, and construction, response time can vary from one second to one minute.[169]

It would seem that with all these limitations, it would be impossible to use the computer, ADC, and thermistor to measure solution temperature. Fortunately, technological strides have been made to overcome these limitations. Original circuit designs attempted to linearize the V_{OUT} versus T curve by placing the thermistor in one leg of a Wheatstone bridge.[169] This design sacrificed sensitivity for a fairly linear response over a 20°C range. Further linearization was accomplished over even wider ranges using parallel thermistor networks in the bridge, but one still had to go through complicated calculations with the Steinhart and Hart equation to come up with resistors and thermistors of the best values. Today's precision thermistor kits avoid this problem and permit accurate measurement down to $\pm$0.2°C over a 100°C range using parallel thermistor networks in a miniature bead construction.[169]

A thermistor composite bead(Omega, 44212) was used to measure the temperature.[170] The encircled resistors of the 44212, T1, T2, and T3, are the thermistors, and the entire network was enclosed in an epoxy coated bead 0.125" wide by 0.280" long. The Omega 44212 could measure the temperature with an accuracy of $\pm$0.15°C from -50.00 to +50.00°C. E IN of the power supply in Figure 20 was set to +3.500V with a digital voltmeter accurate to $\pm$0.0001V. In the voltage mode, two equations were available to convert E OUT to its temperature in degrees Celsius using the ADC and the Apple II plus,

$$E\ OUT1 = [(+0.00559149 * E\ IN) * T] + (0.40700 * E\ IN) \quad (7)$$

$$E\ OUT2 = [(-0.00559149 * E\ IN) * T] + (0.59300 * E\ IN) \quad (8)$$

Eq. (7) was selected, rearranged for T, and programmed as an equation for conversion in Pascal. To account for the 1-2 second response time and inherently noisy digitization process of the ADC, temperature was read in a Pascal loop until two values were obtained within $\pm$0.15°C of one another. The DC of this device, 8mW/°C, presented no problems. Obviously, one cannot immerse the thermistor bead directly into the solvent because of the risk of shorting leads. Before soldering the thermistor leads to the precision resistors R3 and R4 and the wires contained in a shielded coaxial cable, the leads were insulated from one another with heat shrink tubing and the entire assembly inserted into an NMR tube. The NMR tube was also sealed at the top with heat shrink tubing. The resulting probe assembly was easily cleaned, easily mounted in a Teflon cap with the electrodes, inexpensive, and very

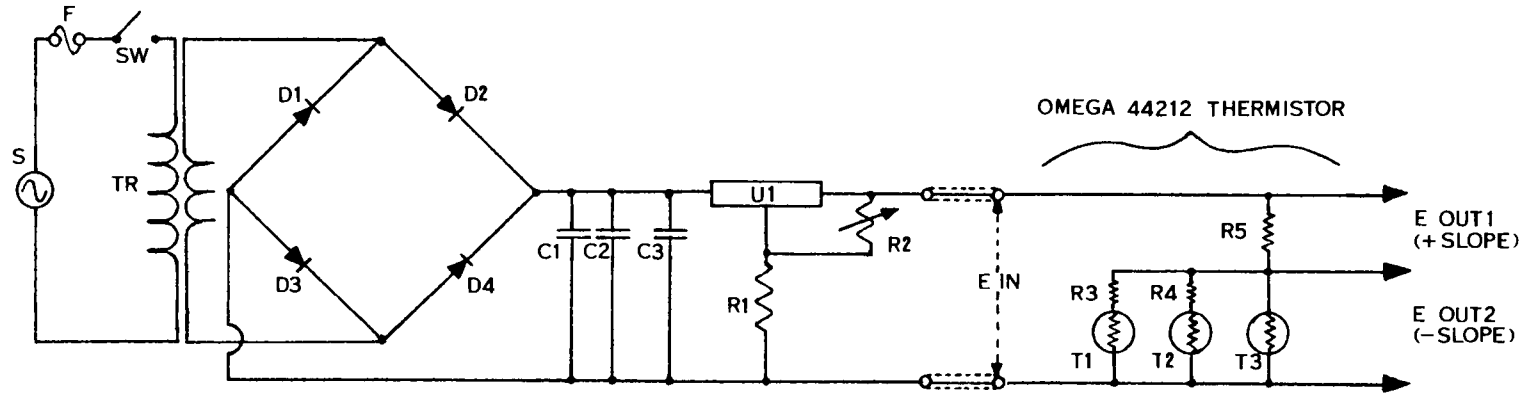


Figure 20. Schematic of power supply and thermistor probe assembly. S = 110VAC, 60Hz; F = 0.5A; SW = SPDT, 0.5A, 125V; TR = Thordarson, T-19F98 transformer, 6.3VAC, 6A secondary; D1-D4 = 1N4007 silicon diode; C1, C2 = 1000MFD; C3 = 10MFD; U1 = LM317T voltage regulator; R1 = 240 ohm(1%); R2 = 1K ohm; R3 = 88.2K ohm; R4 = 38K ohm; R5 = 23.1K ohm; T1 = 2K ohm @25°C; T2 = 15K ohm @25°C; T3 = 45K ohm @25°C; E IN = 3.500V.

accurate. The thin glass of the NMR tube did not interfere with the 1-2 second response time. The power supply in Figure 20, was mounted in a 4" x 6" BUD box and contained two female BNC connectors. One output was used to measure E OUT1 across precision resistor R5 and the other to feed E IN to the thermistor composite. Diodes D1-D4 rectified the 6.3VAC secondary output of the transformer to a constant direction current. Capacitors C1, C2, and C3 discharged very slowly due to their large time constants and removed nearly all ripple in the bridge output voltage.[145] Components U1, R1, and R2 formed a stable voltage regulator. E IN was set to +3.500V by adjusting R2.

CHAPTER IV

SOFTWARE DESCRIPTIONS AND ALGORITHMS

A. The Pascal Operating System

Introduction

In late 1974, Niklaus Wirth[171] designed Pascal to teach students the systematic or "structured" method of writing better programs. But, computer scientists at the University of California, San Diego(UCSD) soon realized Pascal could be used to create large, complex, system and application programs. To use Wirth's Pascal for these applications in practice required subtle changes and extensions to his teaching language. Thus was born the UCSD Pascal language.

One of the most important aims of the UCSD Pascal Project was to demonstrate that it is highly practical to implement a software system that permits large and complicated programs to run unaltered on a wide variety of dissimilar machines. Kenneth Bowles[171] and his students at UCSD accomplished this by writing a compiler that translated Pascal to pseudo code(p-code). The p-code was meant to execute on a hypothetical machine called the p-machine, and not their mainframe Burroughs 6700. The next step was to create a stand-alone program development system that contained a file handler, compiler and an operating system. In order to write such a system in Pascal, it was necessary to extend the language in the areas of machine-level I/O, dynamic string processing, random access to files, and a host of other areas.

Version 1.0 of the UCSD Pascal system debuted in 1976. By the summer of 1977, Version 1.3 appeared and was distributed to other educational institutions. Later, a p-code interpreter was written to run on the Intel 8080 microprocessor, and the entire UCSD Pascal system was transported to the 8080 without changing a single line! The 8080 version utilized the I/O system provided by the CP/M operating system running on the host 8080. This version was eventually streamlined and upgraded to a version now known as Turbo Pascal which would operate on any 8080-based microcomputer.

By the summer of 1978, the UCSD Pascal system was at Version 1.4 and being ported to the TI-9900, Motorola 6800, Commodore 6502, and GA-440. At the same time, Western Digital Corporation began development of their custom p-machine as a microcoded chipset called the MicroEngine. Since the interpreter was microcoded, the Western Digital processor executed UCSD Pascal four to five times faster than other processors. Meanwhile the UCSD project added assembly language and modular compilation facilities to obtain Version 1.5.

By early 1979, Apple Computer obtained a license to distribute the p-system as Apple Pascal 1.0.[150] This system fixed some bugs in Version 1.5 and included the development of Apple's Intrinsic UNIT scheme of modular compilation. Since then, the development and marketing of the UCSD Pascal system has been the responsibility of SofTech Microsystems. For the Apple II series, the p-system requires 64K of RAM and needs two disk drives to operate comfortably. Over time, Apple has released three versions of the p-system, 1.0, 1.1, and 1.2. Version 1.1 which was used here is the most widely used. Version 1.2 premiered

in the spring of 1984. It offers several advantages over 1.1, including the ability to use 128K of RAM when available. This allows longer programs to operate quickly by keeping more information in memory instead of reading it from disks. From this point on, all Pascal programming reserved words and file names are written in uppercase.

Command Level Structure

Apple Pascal 1.1 is based on a "tree-like" structure called the command level. At the time the system is booted, a prompt line appears which displays the command options available. In response to this prompt line, one can use the Editor, Run a program, operate the Filer, or choose any of several options by typing a single letter. The operating system itself is composed of the Filer, Editor, Compiler, Assembler, and Linker which are used to write, store and execute programs. In a dual drive system, the operating system is composed of a series of files on two disks labelled APPLE1: and APPLE2:. APPLE1: contains SYSTEM.APPLE, SYSTEM.PASCAL, SYSTEM.MISCINFO, SYSTEM.EDITOR, SYSTEM.FILER, SYSTEM.LIBRARY, SYSTEM.CHARSET, and SYSTEM.SYNTAX; APPLE2: contains SYSTEM.COMPILER, SYSTEM.LINKER, SYSTEM.ASSMBLER, 6500.OPCODES and 6500.ERRORS. These files perform various operating system tasks and are the skeleton upon which large application programs are built. Table III summarizes the contents and use of these files.[150] The reason for breaking the operating system into several smaller protions is that only a small part is in use at any given time and the operating system is much too large to be kept entirely in the Apple's memory.

Table III. Summary of the Apple Pascal operating system files and their uses.

Filename	Contents of File	Use of File	When File Needed
SYSTEM.APPLE	Interpreter, written in 65Ø2 machine language	Executes P-code on Apple's 65Ø2 processor	Power-on, H(alt
SYSTEM.PASCAL	Command level portion of operating system	Lets you pick E(dit, F(ile, R(un, etc.	Power-on, H(alt, RESET, I(nitialize, every return to Command level
SYSTEM.MISCINFO	Information about terminal in use	Tells system about terminal hardware	Power-on, H(alt, RESET, I(nitialize
SYSTEM.EDITOR	Text Editor	Lets you make & change text	E(dit, C(ompile R(un, A(ssemble
SYSTEM.FILER	Filer	Lets you store, delete & move disk files	F(ile
SYSTEM.LIBRARY	Routines for I/O, long integers, trig. functions, graphics, etc.	Many programs use these library routines	R(un, X(ecute, L(ink, C(ompile, if library routines are used
SYSTEM.CHARSET	Array providing upper & lower case graphic character set	Lets you put text on graphics screen	Used by WCHAR & WSTRING in TURTLEGRAPHICS
SYSTEM.SYNTAX	Compiler error messages	Provides message in E(ditor after Compiler finds an error	R(un, C(ompile followed by E(dit after an error
SYSTEM.COMPILER	Pascal Compiler	Converts Pascal program text to P-code	C(ompile, R(un
SYSTEM.LINKER	Linker	Puts library routines into your program	L(ink, R(un
SYSTEM.ASSMBLER	65Ø2 Assembler	Converts 65Ø2 assembly text into machine code	A(ssemble
65ØØ.OPCODES	Instruction set for Assembler	Used by the Assembler	A(ssemble
65ØØ.ERRORS	Assembler error messages	Provides message after Assembler finds an error	A(ssemble

Files and the Filer

A very useful file in the operating system is SYSTEM.FILER. The Filer contains further commands for saving, reading, transferring, and deleting other disk files. Other commands tell what peripheral devices and diskettes are currently available to the system, and what files are available on each diskette. The Filer can reveal a great deal of information about the external design of the operating system.[150]

Each of the sectors on a 35 track disk consists of an address and data field in Pascal. The address field tells the system exactly which sector of which track is about to be read from or written to by the disk's read/record head. The address fields are written on a diskette just once, when the disk is initially formatted. The data field is the portion of each sector used for storing data, code, or text information. Apple Pascal always stores information in two-sector units called "blocks", each containing 512 bytes of information. Each track on a disk can store eight blocks of information, for a total storage capacity of 280 blocks. Not all of the 280 blocks are available for storing programs or other files. Blocks 0 and 1 are reserved for the bootstrap program, and blocks 2 through 5 contain the diskette directory which is the only way to recover the other information on a disk. The directory may store information for up to 77 different files. Typing <E> at the Filer level gives an extended directory listing of files on the diskette.[150,162]

The Filer can also reveal how Pascal handles I/O devices. Pascal treats I/O devices as volumes, numbered 0...12. All I/O devices are either block or non-block structured. The only block structured de-

vices are the disk drives. Devices such as the printer and console are non-block structured, and simply produce or consume a stream of data. Peripherals can be referred to by their volume number or name. The volume name of a disk drive is the name of the disk currently in that disk drive. Table IV shows the volumes on line during execution of programs developed in this application. Volume #10: is called PSEUDO: and represents the installation of the 128K RAM card as an internal disk drive. The colon tells the system that the name or number preceding it is a volume specification, and not a diskfile's filename.[150,162]

A file is a collection of information which is stored on the disk and may be referred to by a filename. The directory on each disk contains the filenames and locations of each file and the Filer itself uses the information in the directory to manipulate files. The use of a file is determined by the file's type, which specifies the kind of information stored in the file. File types enable both system and user to determine the contents of a disk file, regardless of its file name. Text files and code files are file types used by the system. The file suffixes .TEXT and .CODE are added to the file name to distinguish them in some cases. Text files(programs) are entered through the Editor and compiled into a mixture of p-code and execution information located in a code file. A workfile called SYSTEM.WRK.TEXT is reserved for this purpose and is used by default in the Editor. The compiled code files are the files which one executes from the command level. Other files which are important are data files. These can be of any program or system designed structure and may not be executed; they contain infor-

Table IV. Volumes on-line when application code is executed.

Volume Number	Volume Name	Description of I/O Device
#0:		Not used
#1:	CONSOLE:	Screen and keyboard with echo
#2:	SYSTERM:	Reads keyboard without echoing it
#3:		Not used
#4:	<diskette name>:	Boot disk drive(slot 6, drive 1)
#5:	<diskette name>:	2nd disk drive(slot 6, drive 2)
#6:	PRINTER:	Printer(card in slot 1)
#7:	REMIN:	Remote input
#8:	REMOUT:	Remote output
#9:	<diskette name>:	5th disk drive(slot 4, drive 1)
#10:	PSEUDO:	6th disk drive(slot 4, drive 2)
#11:	<diskette name>:	3rd disk drive(slot 5, drive 1)
#12:	<diskette name>:	4th disk drive(slot 5, drive 2)

mation which can only be written over or read by another file during execution.[150,162,171]

This discussion enables the user to understand the file organization of the application software. Figure 21 is an extended directory listing of the files in the two drives when PVF data are collected and analyzed. Three files not previously mentioned form a device driver and provide an interface between the operating system and Titan Technologies 128K RAM card. SYSTEM.ATTACH uses numerical data in ATTACH.DATA and ATTACH.DRIVERS to attach the RAM card as a block structured device in the basic I/O system(BIOS). SYSTEM.ATTACH is automatically executed when Pascal is booted and prior to automatic execution of SYSTEM.STARTUP. Titan Technologies provided the Pascal software to attach their device as a disk drive emulator. NEWBOOT2.CODE designates the RAM card as the root volume, renames it PSEUDO:, automatically updates the system date, and informs the operating system of the location of SYSTEM.PASCAL, the Editor, Filer, and Compiler.[150,161,162,171] Volume PSEUDO:(Figure 21) contains SYSTEM.PASCAL and SYSTEM.FILER. The command level and Filer options operate faster when located in the RAM drive. While the programs were under development, slightly different versions of SYSTEM.STARTUP and NEWBOOT2.CODE(renamed NEWBOOT1.CODE) were used to automatically transfer SYSTEM.EDITOR to volume PSEUDO: and speed up the editing process.[162]

All of the files on volume PSEUDO: are copies of files on APPLE1: or APPLE2:. Some action must be performed to transfer these files when Pascal is booted. A logical approach would be to use the Filer's Transfer option and transfer them one by one. This requires many

```
APPLE1:
SYSTEM.APPLE        32  9-Nov-80     6 Data
SYSTEM.PASCAL       41 22-Sep-80    38 Code
SYSTEM.SYNTAX       14  1-Aug-80    79 Data
SYSTEM.EDITOR       47 24-Sep-80    93 Code
SYSTEM.MISCINFO      1  4-May-79   140 Data
SYSTEM.CHARSET       2 14-Jun-79   141 Data
SYSTEM.FILER        28 18-Sep-80   143 Code
ATTACH.DATA          1 14-Aug-80   171 Data
ATTACH.DRIVERS       4  6-Feb-86   172 Data      #4:
SYSTEM.ATTACH        7  6-Feb-81   176 Code
NEWBOOT2.CODE        5 18-Apr-86   183 Code
SYSTEM.LIBRARY      60 14-Jul-86   188 Data
SYSTEM.STARTUP       3  6-Oct-86   248 Code
< UNUSED >          27             253
14/14 files, 28 unused, 27 in largest

APPLE2:
SYSTEM.COMPILER     75 19-Sep-80     6 Code
SYSTEM.LINKER       24 16-Sep-80    81 Code
SYSTEM.ASSMBLER     54 19-Sep-80   105 Code
6500.OPCODES         2 20-Dec-78   159 Data
6500.ERRORS          7 28-Mar-79   161 Data
PEAKLOC.CODE        11 24-Jul-86   168 Code
INTERPAR.CODE        7 31-Jul-86   179 Code      #5:
CVOUTPUT.CODE       12 24-Aug-86   186 Code
PEAKFIT.CODE        12 16-Sep-86   198 Code
CVSMOOTH.CODE       16 15-Oct-86   210 Code
GRAPHCV.CODE         7  4-Nov-86   226 Code
DECONVOL.CODE       10  6-Nov-86   233 Code
CVDATA.CODE          9  5-Mar-86   243 Code
< UNUSED >          28             252
13/13 files, 28 unused, 28 in largest

PSEUDO:
SYSTEM.PASCAL       41 22-Sep-80     8 Code
SYSTEM.FILER        28 18-Sep-80    49 Code
SYSTEM.LIBRARY      60 14-Jul-86    77 Data
INTERPAR.CODE        7 31-Jul-86   137 Code
CVDATA.CODE          9  5-Mar-86   144 Code
CVSMOOTH.CODE       16 15-Oct-86   153 Code      #10:
GRAPHCV.CODE         7  4-Nov-86   169 Code
PEAKLOC.CODE        11 24-Jul-86   176 Code
PEAKFIT.CODE        12 16-Sep-86   187 Code
DECONVOL.CODE       10  6-Nov-86   199 Code
CVOUTPUT.CODE       12 24-Aug-86   209 Code
< UNUSED >           3             221
11/11 files, 3 unused, 3 in largest
```

Figure 21. Files located on volumes 4, 5, and 10 when cyclic voltammetry programs are booted.

keystrokes however and is more efficiently handled by a special file called an executive file. The most important aspect of executive files is that they can perform command level tasks such as file transfers from a text file created within an executable program.[150,162] Once the 128K RAM card has been installed in the BIOS, the eight .CODE programs which make up the PVF data acquisition and analysis software were copied from APPLE2: to PSEUDO: by SYSTEM.STARTUP. The text version of SYSTEM.STARTUP is shown in PROGRAM START2(Figure 22). These two simple programs demonstrate the basic structure of a Pascal program, which includes PROGRAM name, USES section, CONSTant section, VARiable section, PROCEDUREs and FUNCTIONs, and a MAIN level, BEGIN to END section.[172] A program may or may not contain PROCEDUREs or FUNCTIONs. Comments are written between "(*" and "*)", are ignored at compile time, and describe the overall purpose of the PROGRAM, PROCEDURE, or FUNCTION.[162,172] Each of these programs was entered via the Editor, compiled, and transferred to APPLE1: as SYSTEM.STARTUP. Each of the programs creates an executive text file of commands that instructs the Filer to copy the programs to volume PSEUDO:, execute NEWBOOT2.CODE to perform the tasks outlined earlier, and then remove itself once it is finished.

A minimal knowledge of the Filer, operating system, and executive files explains the operation of the software at the execution level. The basis of the file organization was to make the software as easy to use as possible and use the computer to perform mundane tasks. Placing APPLE1: and APPLE2: in the disk drives and typing <CTRL-RESET> prepares the software for immediate execution. After all of the above events

```
(*$S++*)
PROGRAM START;

(* This program sets up an EXECUTIVE file that is executed upon
   termination of the program.  It transfers the necessary files to
   volume 10 for rapid execution of the Pascal operating system and
   the interaction parameter programs which analyze PVF and TCNQ-TPE
   thin films. *)

USES CHAINSTUFF;  (* Chains to the EXECUTIVE text file for execution. *)

VAR T:TEXT;

(*** MAIN ***)

BEGIN
  REWRITE(T,'MOVEIT.TEXT');
  WRITELN(T,'%FTSYSTEM.PASCAL,#10:$');WRITELN(T,'TSYSTEM.FILER,#10:$');
  WRITELN(T,'TSYSTEM.LIBRARY,#10:$');WRITELN(T,'T#5:INTERPAR.CODE,#10:$');
  WRITELN(T,'T#5:CVDATA.CODE,#10:$');WRITELN(T,'T#5:CVSMOOTH.CODE,#10:$');
  WRITELN(T,'T#5:GRAPHCV.CODE,#10:$');WRITELN(T,'T#5:PEAKLOC.CODE,#10:$');
  WRITELN(T,'T#5:PEAKFIT.CODE,#10:$');WRITELN(T,'T#5:DECONVOL.CODE,#10:$');
  WRITELN(T,'T#5:CVOUTPUT.CODE,#10:$');WRITELN(T,'QXNEWBOOT2');
  WRITELN(T,'FR#4:MOVEIT.TEXT');WRITELN(T,'YQ%%%%');
  CLOSE(T,LOCK);SETCHAIN('EXEC/MOVEIT.TEXT')
END.
```

```
(*$S++*)
PROGRAM START;

(* This program sets up an EXECutive file that is executed upon
   termination of the program.  It transfers the necessary files to
   VOLUME 10 for rapid execution of the PASCAL operating system and
   the Chronoamperometric programs. *)

USES CHAINSTUFF;

VAR T:TEXT;

(*** MAIN ***)

BEGIN
  REWRITE(T,'MOVEIT.TEXT');
  WRITELN(T,'%FTSYSTEM.PASCAL,#10:$');WRITELN(T,'TSYSTEM.FILER,#10:$');
  WRITELN(T,'TSYSTEM.LIBRARY,#10:$');WRITELN(T,'T#5:CHRONOAMP.CODE,#10:$');
  WRITELN(T,'T#5:CADATA.CODE,#10:$');WRITELN(T,'T#5:SUBTRACT.CODE,#10:$');
  WRITELN(T,'T#5:DISPLAY.CODE,#10:$');WRITELN(T,'T#5:CAOUTPUT.CODE,#10:$');
  WRITELN(T,'QXNEWBOOT2');WRITELN(T,'FR#4:MOVEIT.TEXT');WRITELN(T,'YQ%%%%');
  CLOSE(T,LOCK);SETCHAIN('EXEC/MOVEIT.TEXT')
END.
```

Figure 22. Text file listings of the two versions of SYSTEM.STARTUP.

have occurred, one executes INTERPAR.CODE or CHRONOAMP.CODE at the command level to view a menu of program options. In Figure 23, the chronoamperometry versions of APPLE1: and APPLE2: are shown.

The Editor and Compiler

SYSTEM.EDITOR is the file invoked by typing <E> at the command level. Pascal, assembly language, and library UNIT programs were entered as text files with the Editor. Within the Editor various commands are available to move through files, change text, format, set margins and markers, and save the file. Volition Systems now markets a text editor, ASE(Advanced Screen Editor) Version 0.8a, which can be used in place of SYSTEM.EDITOR to improve the efficiency of the editing process.[150,162]

It was mentioned earlier that the Compiler converts the text of a Pascal program to p-code and compiles SYSTEM.WRK.TEXT by default if it exists on the root volume. After typing <C> at the command level, the Compiler is invoked. If the program USES Intrinsic UNITs, SYSTEM.LIBRARY must be on the disk of the root volume so that the library code can be linked into the program at compile time. SYSTEM.SYNTAX and SYSTEM.EDITOR are used to display syntactical error messages and debug during compilation. Also of concern to the Compiler are Compiler Options. Options are embedded between "(*$" and "*)". The NOLOAD option, (*$N+*), causes the code of a library UNIT used by a program to be resident in memory only when active. Used effectively, it reduces the greedy memory demands of some of the programs. The RESIDENT option, (*$R <name>*), forced the code of a <name>d UNIT or

```
APPLE1:
SYSTEM.APPLE        32  9-Nov-80     6 Data
SYSTEM.PASCAL       41 22-Sep-80    38 Code
SYSTEM.SYNTAX       14  1-Aug-80    79 Data
SYSTEM.EDITOR       47 24-Sep-80    93 Code
SYSTEM.MISCINFO      1  4-May-79   140 Data
SYSTEM.CHARSET       2 14-Jun-79   141 Data
SYSTEM.FILER        28 18-Sep-80   143 Code
ATTACH.DATA          1 14-Aug-80   171 Data
ATTACH.DRIVERS       4  6-Feb-86   172 Data
SYSTEM.ATTACH        7  6-Feb-81   176 Code
SYSTEM.LIBRARY      59 25-Feb-87   183 Data
NEWBOOT2.CODE        5 18-Apr-86   242 Code
SYSTEM.STARTUP       2 13-Mar-87   247 Code
< UNUSED >          30             250
14/14 files, 30 unused, 30 in largest
```

#4:

```
APPLE2:
SYSTEM.COMPILER     75 19-Sep-80     6 Code
SYSTEM.LINKER       24 16-Sep-80    81 Code
SYSTEM.ASSMBLER     54 19-Sep-80   105 Code
6500.OPCODES         2 20-Dec-78   159 Data
6500.ERRORS          7 28-Mar-79   161 Data
CADATA.CODE         13 14-Mar-87   168 Code
CAOUTPUT.CODE       12 24-Mar-87   181 Code
DISPLAY.CODE         7 14-May-87   193 Code
SUBTRACT.CODE        6 14-May-87   200 Code
CHRONOAMP.CODE       7 15-May-87   206 Code
< UNUSED >          67             213
```

#5:

```
PSEUDO:
SYSTEM.PASCAL       41 22-Sep-80     8 Code
SYSTEM.FILER        28 18-Sep-80    49 Code
SYSTEM.LIBRARY      59 25-Feb-87    77 Data
CHRONOAMP.CODE       7 15-May-87   136 Code
CADATA.CODE         13 14-Mar-87   143 Code
SUBTRACT.CODE        6 14-May-87   156 Code
DISPLAY.CODE         7 14-May-87   162 Code
CAOUTPUT.CODE       12 24-Mar-87   169 Code
< UNUSED >          43             181
```

#10:

Figure 23. Files located on volumes 4, 5, and 10 when chronoamperometry programs are booted.

SEGMENT to be kept in memory as long as the PROCEDURE or FUNCTION that contained the option was active. The last option, (*$S++*), causes the Compiler to slow down but compact the code into a smaller code space. This reduced the overall byte length of code files.[150,162,172]

The Assembler and Linker

Acquisition of cyclic voltammetric and chronoamperometric data in real-time required small assembly language routines to be written. These routines were entered via the Editor and assembled. The file, 6500.ERRORS, displayed error messages which could be fixed with the Editor during assembly. When a PROCEDURE or FUNCTION is written in assembly language, it is declared so in the Pascal host program with the reserved word EXTERNAL. Once the host program is compiled and the assembly language routine assembled, the two are linked by typing <L> at the command level and invoking SYSTEM.LINKER. Intrinsic UNITs in SYSTEM.LIBRARY are automatically linked at compile-time with a USES declaration.[150,172]

B. The Library

Introduction

Traversing the operating system was the first step in learning how to develop Pascal programs. Each SYSTEM task was broken down into an individual code file which was brought into memory when invoked at the command level. Pascal programming follows the same modular philosophy

of separating large programming problems into smaller ones with PROCEDUREs, FUNCTIONs, and UNITs. The modular concept allows the programmer to reduce the size of code files, reduce the memory space used by the program, and use a set of PROCEDUREs and FUNCTIONs in more than one program.[150,162]

Separation can be achieved both at the p-code level and at the source-language level. At the p-code level, any PROCEDURE or FUNCTION can be designated as a SEGMENT. The result is that its code is not loaded into memory until it is called by some other part of the program. As soon as the SEGMENT PROCEDURE or FUNCTION is no longer active, it is swapped out of memory. The Compiler's NOLOAD and RESIDENT options allow further control over the handling of SEGMENT PROCEDUREs and FUNCTIONs as outlined earlier for UNITs. The linking information of a SEGMENTed PROCEDURE or FUNCTION is maintained in SYSTEM.LIBRARY. As a result, the number of PROCEDUREs and FUNCTIONs that can be SEGMENTed in a program is determined by available space in the library.[150,172]

At the source-language level, a group of one or more PROCEDUREs or FUNCTIONs can be compiled separately as an Intrinsic UNIT and incorporated into SYSTEM.LIBRARY. When a host program USES an Intrinsic UNIT, the UNIT code remains in SYSTEM.LIBRARY and is automatically loaded into memory when the host program is executed. This keeps the size of the host program's code file down, which is particularly important if many programs use the UNIT. The UNIT can be modified and recompiled without relinking.[150]

Intrinsic UNITs

A Pascal Intrinsic UNIT is a collection of PROCEDUREs and FUNCTIONs. The main advantage of using UNITs is the reuse of code from one program to another. This allows the programmer to use code that has already been developed and debugged and avoids the problem of reinventing the wheel each time an application program is written. UNITs also facilitate the writing of large programs since they are swapped into memory only when needed. This maximizes the room left in memory for data and other portions of the host program.[150,162]

A UNIT's heading starts with the word UNIT, followed by the UNIT name. Following that is the INTERFACE part of the UNIT. Contained in this part are any VARiables, PROCEDUREs, and FUNCTIONs(including the parameters) that are available to the host program. This also means that any VARiables, TYPEs, or CONSTants can be considered global in the host program and do not to have to be redefined. The next section is the IMPLEMENTATION part, where the actual code for the PROCEDUREs and FUNCTIONs is located. Finally, there is the INITIALIZATION part, delineated by BEGIN and END(including a period). Any code in this section is executed before the host program is run. The purpose of the INITIALIZATION part, for example, could be to set up any tables that PROCEDUREs in the UNIT might need. Since an Intrinsic UNIT is prelinked, there must also be information in the UNIT to tell the host program where it is located. This is the purpose of the statement, INTRINSIC CODE 24 DATA 25(Figure A-1). These are the code and data segment locations in SYSTEM.LIBRARY. Segments 16...31 are for use with UNITs.[150,162]

Three UNITs which were used extensively in this work were supplied with the original operating system. The UNIT TURTLEGRAPHICS was used to develop all of the interactive graphics programs. The graphics package is called TURTLEGRAPHICS since it is based on the "turtles" devised by Papert and coworkers[150] at the Massachusetts Institute of Technology. The turtle can be programmed to draw very complex graphics images. To use any of the graphics routines, the statement, USES TURTLEGRAPHICS, must follow the program heading. The UNIT must also be on-line in SYSTEM.LIBRARY when the calling program is run or executed.

Some of the mathematical FUNCTIONs commonly used in scientific calculations are located in UNIT TRANSCEND. They are SIN, COS, EXP, ATAN, LN, LOG, and SQRT. The FUNCTIONs SIN and COS accept angle arguments in radians and all of the FUNCTIONs return real values. These FUNCTIONs can be accessed with the declaration USES TRANSCEND.

An often overlooked UNIT that is supplied with Version 1.1 is the CHAINSTUFF UNIT. A USES CHAINSTUFF declaration and a call to SETCHAIN('EXEC/MOVEIT.TEXT') appear in Figure 22. The effect of these calls is to cause the executive text file created by each version of SYSTEM.STARTUP to be executed as soon as SYSTEM.STARTUP terminates its execution. This UNIT allows one program to "chain" to another program and had a very powerful impact on the design of the application software.[150]

SYSTEM.LIBRARY

Installing UNITS. The Pascal operating system also includes a diskette of utility programs called APPLE3:. From the command level, one executes these utilities to format a disk, examine the library's contents, and install UNITs in SYSTEM.LIBRARY. LIBRARY.CODE is a utility program on APPLE3: that installs separately compiled Pascal UNITs and separately assembled routines into a library file. Intrinsic UNITs are automatically linked from SYSTEM.LIBRARY at compile time. During execution of the library utility, the screen displays the slot number, segment number(in parentheses), name, and byte length of each UNIT currently in the file. There are a maximum of 16 slots, each containing one code or data segment. An Intrinsic UNIT may occupy two slots, one for the code segment and one for the data segment.[150] The Intrinsic UNITs developed in this application were called PRNTRTNS, ADCRTNS, TIMETEMP, INPTRTNS, and FILERTNS, and their text file listings are given in APPENDIX A.

UNIT PRNTRTNS. Within any program, the programmer will usually want to produce hard copy results of a calculation. UNIT PRNTRTNS consists of five PROCEDUREs for printing single CHARacters, STRINGs of characters, INTEGERs, REALs, and changing formats. The firmware on the PKASO EP12-80/100 EPROM dictated the structure of the PROCEDUREs.[157] The global VARiable, T, was used as a text file to hold characters for printing. Note the overall structure of the UNIT in Figure A-1; it starts with a swapping option and general UNIT description. The INTER-

FACE declares all of the PROCEDUREs and input parameters along with a comment for what each PROCEDURE does. In the IMPLEMENTATION section, the parameter lists are included with the PROCEDUREs as comments to make the text easier to comprehend. Although the INITIALIZATION section contains no code, it must have at least a BEGIN and END. Standard calls to REWRITE open the text file and route the output to the PRINTER: volume.[172,173] The CHR(25) is a Pascal equivalent to ASCII CTRL-Y which activates the PKASO EPROM for printing. The CONTROL parameter sets the size of the characters in the first four PROCEDUREs. In PROCEDURE FORMAT, three letter sequences of control characters were used to send a carriage return/linefeed or high resolution screen image to the PRINTER:.[157]

UNIT ADCRTNS. This collection of FUNCTIONs and PROCEDUREs(Figure A-2) was designed for the conversion of binary to real values. Two of the routines were written for non-time-critical analog-to-digital conversion and are labelled PROCEDURE POKE and FUNCTION PEEK. POKE and PEEK are common routines and some languages even have them built into their programming environment.[14] Matuszek[173] has described a complete UCSD Pascal equivalent for the two routines based on variant records, but assembly language routines were written to achieve faster execution speeds.

PROCEDURE POKE performs a conversion on the AI13 by loading a channel address into the accumulator. Decoding circuitry on the AI13 analog board causes a conversion on a specified channel when it is accessed with a load instruction.[165] PROCEDURE POKE also stores the

binary value between 0 and 4095 in a specified memory location. The high order four bits are stored at MEMLOC and the low order byte is stored at MEMLOC+1. Each address location of the 6502 memory space can hold one 8-bit value.[34,149] Since the AI13 has 12-bit resolution(4096 states), it requires two bytes to store one result. FUNCTION PEEK(Figure A-4) returns the byte value at a specific memory location.

Both of these routines are declared as EXTERNAL in the UNIT and had to be manually linked to the UNIT. When the Pascal host program calls an EXTERNAL assembly language routine, passed parameters are pushed on the evaluation stack as they are encountered in the host program's calling statement. The first parameter is pushed on the stack, highest byte first, then the second parameter, and so on. A word indicating the length in words is pushed last. When all the parameters have been passed, the host program's return address(the address to which the program must return on completing the EXTERNAL routine) is pushed on the stack, high byte first, then low byte. The first thing the assembly language routine must do is save the return address. All comments and code explanations are written to the right of semicolons. Page zero locations in memory are reserved with .EQU statements to hold the passed parameter values. The .MACRO is a facility directive that allows a named section of text to be defined once and repeated in other sections simply by stating its name. Here, the MACRO facility pops passed parameters from the stack and places them in page zero locations. For EXTERNAL assembly language FUNCTIONs(.FUNCs) only, two additional conventions must be recognized. At the FUNCTION's entry time, the Pascal host program pushes four bytes

of zeroes on the evaluation stack after any passed parameters are put on the stack and before the return address is pushed on the stack. At the FUNCTION's exit time, the .FUNC must push the FUNCTION result, high byte first, just before pushing the return address on the stack.[31] To accomplish this, the 6502 instructions PLA and PHA were used to push and pop values to and from the top of the stack.[34,149,150]

FUNCTION GAINCODE(Figure A-2) returns an integer between 0 and 7 corresponding to one of the eight possible voltage ranges of the AI13. For maximum sensitivity, the routine chooses the narrowest voltage range which includes the potential limits input by the user during CV. An amplified conversion is performed by the statement,

LDA [CHANNEL+(GAINCODE*16)] (9)

Two other routines needed for conversion are FUNCTIONs FRACTION and LOWEND. One needs to know the fractional voltage value of one step and the lowest voltage value for each gaincode setting as well as the sensitivity. The voltage value of one step for each range was computed from,

ONE STEP = [VOLTAGE RANGE - (VOLTAGE RANGE/4096)]/4095 (10)

The reason for this offset is that all ranges are calibrated so that a zero result corresponds to the low end of the stated range, and a code of 2048 corresponds to the middle. This means, however, that the top of the range would be 4096 which is outside the 12-bit range. There-

fore, the high end of the range is calibrated to be one step below the stated upper limit. The actual reading can then be converted by,

BINARY RESULT = [PEEK(MEMLOC + 1) * 256] + PEEK(MEMLOC) (11)

REAL VALUE = SENSITIVITY * [(BINARY RESULT * FRACTION) + LOWEND] (12)

These routines and equations converted binary results to real values in the application code.[165]

UNIT TIMETEMP. This Intrinsic UNIT introduces a few new programming concepts which are worth discussing. Three of the PROCEDUREs work with the Thunderclock to set and retrieve the time from assembly language and are declared by convention to be EXTERNAL(Figure A-5). All of these assembly routines were supplied by Thunderware Inc. on a Pascal formatted diskette and linked to the UNIT. Each routine followed the entry and exit conventions described for assembly routines in UNIT ADCRTNS. The parameters passed to TIMEASCII and TIMEVALUES were preceded by the reserved word, VAR, to indicate transmission by reference.[150,152,173]

The two modes of parameter transmission available in Pascal are by value and reference. Call by value allows information to be passed to the routine, but does not allow it to be returned. Thus, if SETTIME(6,3,87,6,13,50) were called, all of these values on the Thunderclock would be set. The passed parameters are all constant values and there is no need to pass them back to the calling routine. For PROCEDURE TIMEASCII though, we want the TIME to be returned based

on either the 12 or 24 hour format. The calling routine will then declare another VARiable of TYPE STRING to match TIME and call TIMEASCII. The call, TIMEASCII(T,12), would return the time in 12 hour format in VARiable T, provided "VAR T:STRING;" was declared in the calling routine. FUNCTIONs can return values also, but only for simple TYPEs such as INTEGER, REAL, CHAR, or BOOLEAN. If one had a complex data structure that a routine would change, a FUNCTION could not be written to return its changed value. This is why one uses PROCEDUREs and the word VAR to pass parameters by reference; it allows the programmer to pass parameters of any predefined TYPE or complexity and keep track of their changed value from one routine to another.[172,173] FUNCTION CFIND is the fourth routine in the UNIT. Its parameters are all passed by value; it returns a simple TYPE, REAL, and can thus be written as a FUNCTION. CFIND calls CCALC which is not defined in the INTERFACE section. CCALC is therefore not accessible by any program that USES TIMETEMP. Since CCALC calls FUNCTIONs in UNIT ADCRTNS to compute the temperature, ADCRTNS is nested in UNIT TIMETEMP. PEEK, POKE, FRACTION, and LOWEND were called within CCALC to compute the temperature.[162,173]

UNIT INPTRTNS. Nothing is more disquieting to a user than to be inputting numerical data, press a character key, and experience a fatal run-time error. To validate input and control screen output, UNIT INPTRTNS was written(Figure A-6).

PROCEDUREs CLRSCRN and PROMPTAT controlled screen output and GETACHAR returns a valid character in VARiable CH from a predefined set

of characters. This routine was called to process menu selections. FUNCTIONs GETINT and GETREAL return a valid INTEGER and REAL number, respectively, while performing input type checking character-by-character. No single character on the keyboard can crash either of these FUNCTIONs. Both routines use ASCII backspace and the bell to correct and signal illegal entries.[171,174]

UNIT TRANSCEND is nested in INPTRTNS because the LOG FUNCTION is used in GETREAL. GETREAL calls GETINT to validate the exponent if <E> or <e> is typed. Willner and Demchak[171] provided more elaborate routines to validate numerical input. These routines are a stricter and shorter version of theirs.

UNIT FILERTNS. The previous four UNITs were designed for general program development. UNITs PRNTRTNS, ADCRTNS, and TIMETEMP facilitated control of peripheral devices and INPTRTNS validated input. UNIT FILERTNS on the other hand was designed specifically to facilitate file I/O within the 12 programs which acquired and analyzed cyclic voltammetric and chronoamperometric data. Programs that use this UNIT are committed to a predefined data structure and file organization implemented in the TYPE section of the UNIT. The FILERTNS UNIT had to be developed at this point in software development for two reasons. The UNIT USES TRANSCEND and INPTRTNS in SYSTEM.LIBRARY; and, any application program that USES FILERTNS requires that the UNIT be compiled, debugged, and on-line in SYSTEM.LIBRARY when the program is compiled.

One of the salient features of Pascal that distinguishes it from high-level languages like Fortran, Basic, and Algol is the variety of

data TYPEs the language supports. One of these TYPEs is the "pointer". In conjunction with the RECORD, pointers make it easy for the programmer to implement dynamically linked lists. The linked allocation method of storage can result in both the efficient use of computer storage and time.[162,175]

There are three general reasons for using linked lists, the importance of which increases with the size and complexity of the application program. It was not known at compile time for the UNIT how many entries would be in the data structure and an ARRAY might have necessitated recompilation to accommodate an unexpectedly large number of entries. The second reason is that the use of linked lists introduces the possibility of dynamic memory management using PROCEDUREs MARK and RELEASE. These built-in PROCEDUREs can be used to perform the inverse of NEW, namely to deallocate memory so that lists larger than the RAM storage available can be processed. The third reason for using dynamic lists has to do with organization and sorting. In some applications, it is desirable to insert and delete entries while maintaining the order of the list. Such exercises are cumbersome with an ARRAY. Deleting an entry leaves a "hole" that must be dealt with, and inserting an entry means that the surrounding entries have to be pushed around to make room. With pointers the job is simpler; one merely unlinks the entry to be deleted, or forms a new link for an entry to be added.[162,172,173]

In the TYPE section of Figure A-7, a singly linked list is implemented. Each of the NODEs contains a pair of VALUES and a pointer to a connecting NODE indicated by a circumflex(↑). Figure 24 is a diagram

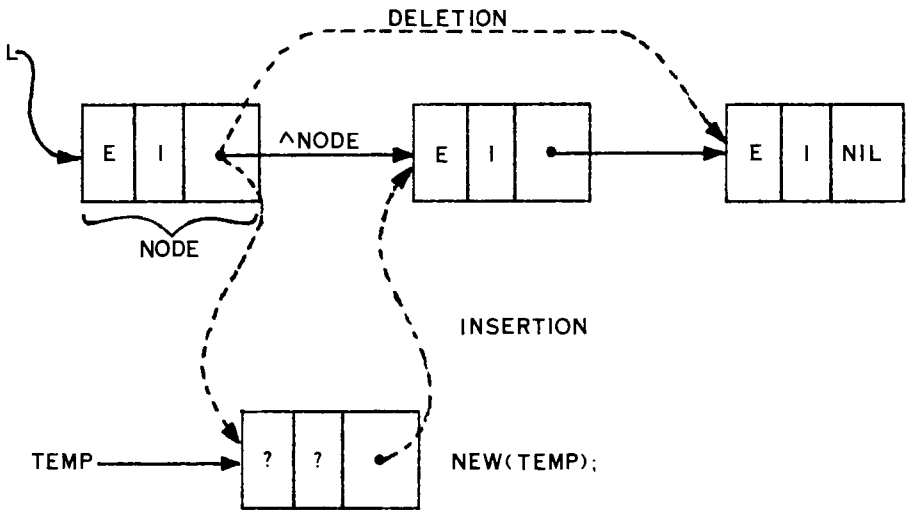


Figure 24. Block diagram of a singly linked list showing insertion and deletion.

of the linked list.[173] Potential and current data pairs were contained in individual NODEs and linked by pointers. Potential and current VALUES were denoted VAL[0] and VAL[1], respectively. A call to NEW(TEMP) creates an empty NODE pointed to by TEMP. The pointers can then be easily moved to incorporate the new NODE into the list. The end of the list is denoted by the Pascal reserved word, NIL. When initializing a list to empty, the statement "LIST:=NIL;" is made. To delete a VALUE, one simply moves the pointer ahead one cell. The singly linked list is the cornerstone for many other types of lists, such as stacks, queues, and circularly linked lists. In a stack, insertions and deletions take place at the top of the list.[172,173,175]

PROCEDURE CV2FILE performs the inverse operation of CV2LIST. An application program for example might load data from a disk file into a list with CV2LIST, smooth the data, then transfer the data back to the disk with CV2FILE. The list is traversed by moving the pointer each time a loop structure is executed. PROCEDUREs FILE2VAR and VAR2FILE perform inverse operations also. Associated with any voltammetric or chronoamperometric experiment are experimental parameters. These parameters may be REAL numbers, INTEGERs, or STRINGs of characters. Ideally, they should be grouped together in a parameter file of some sort. The variant record data structure of Figure A-7 permits objects of different TYPEs to be grouped together.[173] The CASE statement of RECORD PARAM allows a FILE OF PARAMeters to be of any combination of STRINGs, INTEGERs, or REAL values. The price is paid in wasted memory however, because the compiler alots space as if the file were made up of only the largest variant. The structures of typical

cyclic voltammetric and chronoamperometric PARAMeter files are shown in Tables V and VI. When the programmer relaxes type-checking restrictions with a variant record, Tables such as this become very important. This is because one gains access to a VARiable in the file by passing its RECORD number to a random access PROCEDURE called SEEK. Thus, it becomes the programmer's responsibility to keep track of the VARiable locations. Random access to files was one of the aims of the UCSD project.

PROCEDURE's CHAN2FILE and FILE2CHAN take advantage of the variant RECORD PARAM to store the channel numbers in a temporary file on volume PSEUDO: called CHANNELS. All of the CHANNELS are INTEGERs in the range 0...15. In combination with FUNCTION OPENCHAN, the CHANNELs file can be opened with RESET to retrieve the channel numbers or opened with REWRITE to store them. FUNCTIONs OPENCHAN and FILENAME are similar in that they use a compiler option, (*$I-*), to disable I/O checking. A call to REWRITE opens a file that does not yet exist. If one tries to open an existing file with RESET and the file is not on the disk, the system crashes. Trying to REWRITE a file that is already on the disk crashes the system also. Pascal monitors its IORESULT FUNCTION whenever I/O is performed to determine this. A common trick is to disable any I/O checking and sample the BOOLEAN value of IORESULT to open a file. Then, one circumvents the problem of determining whether the file already exists or not.[172]

Table V. Organization of a parameter file for a cyclic voltammogram.

RECORD	VARIABLE	TYPE	EXAMPLE & DESCRIPTION
0	PAIRS	INTEGER	935;the number of points in the CV
1	HIE	INTEGER	1;point position of highest potential
2	LOE	INTEGER	500;point position of lowest potential
3	ICODE	INTEGER	5;current channel gaincode
4	ECODE	INTEGER	2;potential channel gaincode
5	HII	INTEGER	250;point position of highest current
6	LOI	INTEGER	750;point position of lowest current
7	SENS	REAL	1E-3;sensitivity in Amps/V
8	DEGREE	REAL	2.28479E1;temperature in degrees Celsius at time of experiment
9	SCRATE	REAL	2.00000E-2;scan rate in V/s
10	IPC	REAL	-1.12050E-5;fitted cathodic peak current in amps
11	EPC	REAL	4.24488E-1;fitted cathodic peak potential in volts
12	IPA	REAL	1.16006E-5;fitted anodic peak current in amps
13	EPA	REAL	4.39882E-1;fitted anodic peak potential in volts
14	CCH	REAL	-8.56236E-5;cathodic peak charge in coulombs
15	ACH	REAL	7.84515E-5;anodic peak charge in coulombs
16	EPTWO	REAL	3.98436E-1;potential of half peak current in volts
17	INITE	REAL	8.06396E-1;initial potential in volts
18	START	STRING	'TUE SEP 24 4:33:11 PM'
19	TITLE	STRING	'PVF thin film/0.1M TEAP/CH_3CN vs. Ag/AgCl'
20	DIREC	STRING	'Negative scan'
21 -	SMOOTH	STRING	'Smoothing method:2 * five point moving average' OR 'Smoothing method:Fourier transformation, n=?'
22	PFLSLD	STRING	'FFF'...'FFT'...'FTF'...etc.

Table VI. Organization of a parameter file for a chronoamperometry experiment.

RECORD	VARIABLE	TYPE	EXAMPLE & DESCRIPTION
0	PAIRS	INTEGER	300;the number of points in the chronoamperometry experiment
1	ICODE	INTEGER	0;current channel gaincode
2	HII	INTEGER	20;point position of highest current
3	LOI	INTEGER	300;point position of lowest current
4	N	INTEGER	1;number of electrons
5	DEGREE	REAL	2.28479E1;temperature in degrees Celsius at time of experiment
6	SENS	REAL	5.00000E-5;sensitivity in Amps/V
7	AREA	REAL	2.12345E-1;area of the electrode in cm^2
8	DIFF	REAL	6.50000E-6;diffusion coefficient of the solute in cm^2/s
9	IPOT	REAL	1.00000E-1;initial potential in volts
10	FPOT	REAL	7.00000E-1;final potential in volts
11	DUR	REAL	3.05000E1;duration of the experiment in seconds
12	CONC	REAL	4.70000E-3;concentration of the solute, M
13	START	STRING	'TUE SEP 24 4:33:11 PM'
14	TITLE	STRING	'Chronoamp. of 20.0mg of $K_4Fe(CN)_6$/50ml of 0.1M KCl vs. Ag/AgCl
15	SMOOTH	STRING	'Smoothing method:Background subtracted, 2 * five point moving average'
16	DETER	STRING	'Electrode area' or 'Diffusion coefficient

FUNCTION FILENAME is more elaborate than OPENCHAN. Aside from opening an old or new file, it displays the screen output:

```
Enter the CV number <1..99>:1
  File CV1ORIGREAL does not exist!

Press <SPACE> to continue!


Enter the CV number <1..99>:1
  File CV1ORIGREAL already exists!

Press <SPACE> to continue!
```

This FUNCTION processes an INTEGER between 1 and 99, then constructs a STRING which embeds the INTEGER in the filename. The fileneme is returned by passing SUFFIX by reference. If the routine is instructed to open an OLD file that isn't on the disk, the message, "File CV1ORIGREAL does not exist!", appears and the file is not opened. If the routine is instructed to open a new file that already exists, the message, "File CV1ORIGREAL already exists!", appears and the file is not opened. When the routine cannot successfully open a file, the FUNCTION returns the BOOLEAN value FALSE.

FUNCTION FILENAME was called to construct and open all of the files in Figure 25. This Figure is an extended directory listing of original and analyzed cyclic voltammetric data. After the files were transferred to volume PSEUDO:(the internal RAM drive), a blank disk labelled B123: was placed in volume #5:. The "B" in the volume name is an acronym for Backup. All of the ORIGinal REAL potential, current data and input parameters were stored on a Backup disk; hence, they are

```
INTPAR3:
CV10ORIGREAL       15 22-Aug-86    6 Data
CV10PARAM           4 20-Sep-86   21 Data
CV11ORIGREAL       15 22-Aug-86   25 Data
CV11PARAM           4 20-Sep-86   40 Data
CV12ORIGREAL       15 22-Aug-86   44 Data
CV12PARAM           4 20-Sep-86   59 Data
CV13ORIGREAL       15 22-Aug-86   63 Data
CV13PARAM           4 20-Sep-86   78 Data
CV10SMTH           15 20-Sep-86   82 Data
CV10DECONVOL       15 20-Sep-86   97 Data
CV10SUB            15 20-Sep-86  112 Data
CV11SMTH           15 20-Sep-86  127 Data
CV12SMTH           15 20-Sep-86  142 Data
CV11DECONVOL       15 20-Sep-86  157 Data
CV11SUB            15 20-Sep-86  172 Data
CV12DECONVOL       15 20-Sep-86  187 Data
CV12SUB            15 20-Sep-86  202 Data
CV13SMTH           15 20-Sep-86  217 Data
CV13DECONVOL       15 20-Sep-86  232 Data
CV13SUB            15 20-Sep-86  247 Data
< UNUSED >         18            262
20/20 files, 18 unused, 18 in largest


B123:
CV1ORIGREAL        15 21-Aug-86    6 Data
CV1PARAM            4 21-Aug-86   21 Data
CV2ORIGREAL        15 21-Aug-86   25 Data
CV2PARAM            4 21-Aug-86   40 Data
CV3ORIGREAL        15 21-Aug-86   44 Data
CV3PARAM            4 21-Aug-86   59 Data
CV4ORIGREAL        15 21-Aug-86   63 Data
CV4PARAM            4 21-Aug-86   78 Data
CV5ORIGREAL        15 21-Aug-86   82 Data
CV5PARAM            4 21-Aug-86   97 Data
CV6ORIGREAL        15 21-Aug-86  101 Data
CV6PARAM            4 21-Aug-86  116 Data
CV7ORIGREAL        15 21-Aug-86  120 Data
CV7PARAM            4 21-Aug-86  135 Data
CV8ORIGREAL        15 22-Aug-86  139 Data
CV8PARAM            4 22-Aug-86  154 Data
CV9ORIGREAL        15 22-Aug-86  158 Data
CV9PARAM            4 22-Aug-86  173 Data
CV10ORIGREAL       15 22-Aug-86  177 Data
CV10PARAM           4 22-Aug-86  192 Data
CV11ORIGREAL       15 22-Aug-86  196 Data
CV11PARAM           4 22-Aug-86  211 Data
CV12ORIGREAL       15 22-Aug-86  215 Data
CV12PARAM           4 22-Aug-86  230 Data
CV13ORIGREAL       15 22-Aug-86  234 Data
CV13PARAM           4 22-Aug-86  249 Data
< UNUSED >         27            253
26/26 files, 27 unused, 27 in largest
```

Figure 25. Directory listing of the data files on a completely analyzed and backup copy of an experimental set of CV data.

called CV?ORIGREAL and CV?PARAM, respectively. There is just enough space on one disk to store 13 original CVs and their PARAMeter files. The integers 1, 2, and 3 in B123: signify that analysis of the original data is on disks INTPAR1:, INTPAR2:, and INTPAR3:. INTPAR3: contains the complete analysis of four of the original files on B123:. There is just enough room for the analysis of four original files as well as copies of the original data on B123:. Thus, the analysis of voltammetric data in one solvent/electrolyte system required four diskettes, whose Backup disks are labelled sequentially B123:, B456:, B789:, B101112:, etc. The PARAMeter files required four blocks(2K) of space which seems large considering there are only 23 PARAMeters in the file. But, the files were defined by a variant record whose largest member was a STRING of 80 bytes. To contain a file of 23, 80 byte STRINGs, at least four blocks are needed. Backup copies of all the code and data prevented catastrophic loss of any of the software and data.

For chronoamperometry, the PROCEDURE names CV2LIST and CV2FILE were changed to CA2LIST and CA2FILE, respectively. The acronym, "CV" , was changed to "CA" in FUNCTION FILENAME also to reflect chronoamperometry experiments. The names of the files were correspondingly CA1ORIGREAL, CA1PARAM, CA2ORIGREAL, CA2PARAM, etc. None of the internal workings of the UNIT were changed, only file and PROCEDURE names.

C. Application Code

Introduction

All of the text file versions of the application code for analysis and collection of PVF data are given in APPENDIX B. Code descriptions will focus on programming techniques and concepts since a discussion of the intricacies of implementation is beyond the scope of this manuscript. The reader is directed to the references of Scanlon,[149] Nealon et al.,[163] Demchak,[171] Matuszek,[173] Swan,[174] Tremblay,[175] Collins,[176] and the two Pascal programming manuals[150,172] for this purpose. The text file versions of the code used for analysis and collection of terpolymer data are not included because of their similarity to the PVF code. Instead, only the differences between the two sets of code will be discussed. Finally, a discussion of chronoamperometry code will be given; selected programs in this collection of code are given in APPENDIX C. Since the library UNITs discussed earlier are called within these programs, linked lists, memory management, chaining, program layout, and parameter transmission apply to these program descriptions. All of the .TEXT and .CODE files were stored on appropriately labelled disks with backup copies.

PVF Interaction Parameter Programs

PROGRAM INTERPAR. To start collection and analysis of PVF data, PROGRAM INTERPAR(Figure B-1) is Executed from the Command Level by

typing <INTERPAR>. When this PROGRAM is running, it displays the following,

```
Menu selections:
  C>hange input channels
  T>ime and temperature
  N>ew set of CV data
  G>raph a set of CV data
  S>mooth an original set of CV data
  L>ocate peaks in a CV
  F>it peaks in a CV
  D>econvolute a CV
  O>utput analysis of a CV
  Q>uit

Please type:Q
```

The purpose of this PROGRAM is to chain to one of the other PROGRAMs by keying on a selected character. A legal SET OF CHARacters is defined in FUNCTION MENU and passed to PROCEDURE GETACHAR in UNIT INPTRTNS for validation. Upper or lower case character selections are allowed. PROCEDURE OPTION processes the validated selection from the menu and causes execution to be transferred to the next PROGRAM.[171]

Just as it is logical in Pascal to break large programming problems into smaller ones with PROCEDUREs and FUNCTIONs, it is logical to break large application problems into a sets of PROGRAMs. Chaining provides a communication link between those PROGRAMs. From the listing of B-1, even a novice programmer can readily discern the number of possibilities on the menu, the volume number, and the names of the application PROGRAMs. One can easily delete from or add to the menu without having to change or recompile one of the other PROGRAMs.

The other responsibilities of PROGRAM INTERPAR were designed to accommodate experimental considerations, namely to setup or change a

file containing the DAC and ADC CHANNELS and to monitor the time and temperature prior to the collection of data,

```
Select data channels <0..15>
  Temperature:2
  Trigger    :0
  Current    :1
  Potential  :0
Press <SPACE> to continue!
```

```
Temperature (C): 23.01065
Time           : WED FEB 11  1:29:09 PM
```

UNIT FILERTNS was required to process file I/O to CHANNELS and UNIT TIMETEMP to retrieve the time and temperature. These two options were processed with <C> and <T> on the main menu, respectively.

PROGRAM CVDATA and PROCEDURE TAKEDATA. This PROGRAM and EXTERNAL PROCEDURE acquire cyclic voltammetric data. PROGRAM CVDATA stores the data in the two files, CV?ORIGREAL and CV?PARAM, and establishes their existence for other PROGRAMs. This PROGRAM is executed when <N> or <n> is typed at the main menu. CVDATA chains back to INTERPAR when it is finished acquiring and storing a voltammogram(Figure B-2). This causes the main menu to be presented once again so that another option can be selected.

To collect the data, PROGRAM CVDATA opens the CHANNELs file and retrieves the channel numbers. PROCEDURE FILENAME of UNIT FILERTNS is invoked to obtain the name of a new file. Figure 26 is a printout of the screen display when CVDATA is executed. Once a new file is selected, the screen is cleared and the PROGRAM prompts for the title,

```
Enter the CV number <1..99>:1

Press <SPACE> to continue!

------------------------------------------------

Title of CV:Electroppt. PVF thin film 1/0.1M TBAF/CH3CN vs. Ag/AgCl
  Sensitivity      (Amps/V):5E-5
  Scan direction      (P/N):N
  High potential        (V):0.700
  Low potential         (V):0.100
  Scan rate           (V/s):0.02

          GAINCODE          BIPOLAR RANGE
         ----------------------------------
              4                -5 <--> +5V
              5                -1 <--> +1V
              6              -0.5 <--> +0.5V
              7              -0.1 <--> +0.1V

  Current gaincode <4..7>:4

Set function switch to SCAN and press <SPACE>!

Experiment complete!

CV1ORIGREAL is now on disk in drive 2!

Press <SPACE> to continue!
```

Figure 26. Screen display when acquiring a N>ew set of CV data.

sensitivity, scan direction, potential limits, and scan rate. Routines in UNIT INPTRTNS are used to validate input. Selection of the ADC GAINCODE should be chosen relative to the sensitivity. As the sensitivity is increased, the voltage input to the ADC is increased. The idea is to choose the narrowest bipolar range which still includes the largest, bipolar, experimental output values. This takes advantage of the full 12-bit resolution of the ADC.[165]

Other microprocessor controlled instruments exemplify the sensitivity, GAINCODE relationship. On the BAS-100, one acquires voltammetric data at a GAINCODE selected by the sensitivity. Occasionally, the current output rails on the screen because the amplifier range chosen is too narrow for the output of the experiment. When this happens, the user halts the experiment, decreases the sensitivity and reruns the experiment. Greater flexibility in controlling the overall sensitivity can be achieved by allowing selection of the ADC amplifier setting as well as the potentiostat's amplifier setting.[143,165]

The on-screen display of the BAS-100 is a decided advantage over the operation of the interfaced CV-27. PROGRAM CVDATA does not display its data during acquisition, making it necessary to graph the data with PROGRAM GRAPHCV to see if the output has railed. This problem was circumvented with the initial result of a thin film voltammogram from the BAS-100. From the peak current value of a BAS-100 voltammogram and the sensitivity setting of the CV-27, one can obtain the maximum voltage outputs for a particular film and select the GAINCODE accordingly. Electroprecipitation time was a fine enough control of film thickness and peak current that one scan on the BAS-100 could be used to set the

sensitivity and GAINCODE on an entire set of CVs in one solvent/electrolyte system. After the GAINCODE has been chosen, the PROGRAM prompts the user to press <SPACE> to start data acquisition.

Two extremely important PROCEDUREs in this PROGRAM are TAKEDATA and CALCDELAY. The computer screen lays idle for the duration of the scan while collecting data in assembly language. The essence of the two PROCEDUREs is a 6502, software delay timer.[149] Factors which influenced the duration of the software delay were the overhead for storage of the binary data, determination of when to quit collecting data, the experiment duration set by the potential limits and scan rate, and delays associated with changes in ADC amplifier GAINCODEs.

The simplest way to program a time delay is by using one or more No Operation(NOP) instructions. Since each NOP instruction requires two clock cycles to be executed, a series of NOP instructions can generate a brief time delay. Very few applications, however, are so time-critical that they deal in delays of only a few microseconds. Most applications require delays of at least 50 or 100 microseconds.[149]

Longer delays are more efficiently programmed as nested loops; their execution time can be determined by the fact that each 6502 instruction requires a known number of clock cycles to execute. To determine the time(seconds) required to execute an instruction, one divides the number of clock cycles by 1.023MHz. The triply nested TIME DELAY ROUTINE is shown in Figure B-3. Integers between 1 and 255 are loaded into index registers X and Y and the Accumulator(A) to generate a delay time between 2.34604E-5 and 82.257340 seconds. When X = Y = A = 1, the minimum delay is generated; when X = Y = A = 255,

the maximum delay is generated. It is essential to understand the TIME DELAY ROUTINE because PROCEDURE CALCDELAY calculates the X, Y, and A integers which are passed as parameters DELAY1, DELAY2, and DELAY3 to the assembly language routine.

Figure 27 summarizes the timing analysis of this section of code. When the 6502 executes the DEX instruction at WAIT, the contents of the X register are decremented by one. If this instruction produces a non-zero result, the BNE WAIT instruction causes a branch back to WAIT to decrement the X register again. When the X register has finally been decremented to zero, the 6502 executes the DEY instruction. The minimum number of clock cycles when X = Y = A = 1 is apparent if we simply add the clock cycles on the left. None of the branch instructions are executed; thus, the minimum delay time(TD_{MIN}) = 24/1.023MHz = 2.34604E-5 seconds. If the result of the DEY instruction is non-zero, the second BNE WAIT instruction branches back to WAIT, which initiates the first of 255 decrement-X-register operations. The X register contents go from 0 to -255, then back to 0 for each Y count. Thus, each Y count generates (5 * 255) clock cycles, plus 4 clock cycles for the -255 to 0 DEX operation, plus 5 clock cycles for the DEY operation which caused a branch to WAIT. This nesting concept can be taken one step further using the Accumulator and applying the same timing analysis. If BNE WAIT is executed after SBC #01, an additional 5 clock cycles, plus 1279 clock cycles for all of the DEX operations, plus (255 * 1284) clock cycles for all of the DEY operations, plus 4 clock cycles for the final BNE WAIT, are generated. Each count in A generates 328,708 clock cycles. The maximum time delay(TD_{MAX}) can be determined by multiplying

Clock Cycles	6502 Instruction	
3		LDA DELAY1
3		LDX DELAY2
3		LDY DELAY3
2	WAIT	DEX
2*		BNE WAIT
2		DEY
2*		BNE WAIT
2		SBC #01
2*		BNE WAIT
3		BEQ DATA (ALWAYS EXECUTED)
24		

* = 3 clock cycles when branch executed

Each count in X generates:5 clock cycles

Each count in Y generates:5 + 4 + (5 * 255) = 1284 clock cycles

Each count in A generates:5 + 4 + (5 * 255) + (1284 * 255) + 4 = 328,708 clock cycles

TD_{MAX} = (24 clock cycles + (255 * 5) clock cycles + (255 * 1284) clock cycles + (255 * 328,708) clock cycles)/ 1.023MHz
= 82.257340 seconds

TD_{MIN} = 24 clock cycles/1.023MHz = 2.34604E-5 seconds

Figure 27. Timing analysis of the TIME DELAY ROUTINE.

each clock cycle count by 255 and adding TD_{MIN}(Figure 27). The entire result is then divided by 1.023MHz to obtain TD_{MAX} = 82.257340 seconds. By varying the X, Y, and A values, delays ranging from 23 microseconds to 82 seconds with a resolution of ca. 5 microseconds can be generated. Scanlon[149] gives an excellent description of this timing analysis.

The CONSTant section of the PROCEDURE CALCDELAY contains the three times each delay count can generate. These CONSTants were obtained by dividing each clock cycle sum in Figure 27 by 1.023MHz. A REPEAT...UNTIL loop was written to accept the potential limits and scan rate because of the limits of the software delay timer. When possible, 935 potential, current points are acquired. In reality, the duration of an experiment is limited by the time required to acquire the data and execute a delay. The maximum experiment time that the software can accommodate can be computed from the equation,

$$MAXTIME = (MAXPTS * 212/1.023MHz) + (MAXPTS * TD_{MAX}) \qquad (13)$$

For wide potential limit, slow scan experiments, the duration of the experiment would have to be less than MAXTIME. None of the graphics PROCEDUREs of UNIT TURTLEGRAPHICS were used in this PROGRAM, yet the declaration appears. When TURTLEGRAPHICS is used, it protects Apple HiRes Page 1(Figure 16) by moving the top of the heap to the top of the HiRes Page.[177] At this point, 3,746 bytes below the HiRes page are wasted since they are not accessible to the PROGRAM. This unused memory starts at memory location 4446 decimal and extends upward for 3,746 bytes. This unused memory is an ideal place to store the binary

data; therefore, MEMLOC was set to 4446 and passed to PROCEDURE TAKEDATA. Due to the 12-bit resolution of the ADC, each potential, current point uses four bytes of memory. Dividing 3,746 by 4 bytes/point allows a maximum of ca. 935 points. None of the PROGRAM code or data space that it uses will overwrite this 3,746 byte area. Substituting MAXPTS = 935 and TD_{MAX} = 82.257340 into eq. 13 gives a MAXTIME = 7.69109E4 seconds.

MAXTIME was computed by taking into account the maximum number of points collected and the maximum possible delay time. The minimum time(MINTIME) can be computed from the minimum number of points necessary to adequately define a CV and the minimum delay time,

$$MINTIME = (MINPTS * 212/1.023MHz) + (MINPTS * TD_{MIN}) \qquad (14)$$

MINPTS was determined to be 558, twice the x-coordinate pixel resolution of the Apple Pascal screen. Even at this minimum number of points, the sampling interval is much shorter than the Nyquist frequency of most one-cycle voltammetric experiments.[178] Substituting MINPTS = 558 and TD_{MIN} = 2.34604E-5 seconds into eq. 14 yields a MINTIME of 0.128727 seconds. Therefore, the software can accommodate one-cycle experiments ranging from ca. one tenth to 75,000 seconds! The REPEAT...UNTIL loop prompts the user for potential limits and a scan rate that are within this range.

PROGRAM CVDATA attempts to collect as many of the 935 points possible for any experiment. Generating the minimum delay between points leaves more time for collecting data. But in some cases, col-

lecting 935 points with only a minimum delay time may exceed the duration of the scan. An algorithm to compute the number of points can be written,

$$[\text{SCAN TIME} - (\text{PTS} * 212/1.023\text{MHz}) - (\text{PTS} * TD_{MIN})] < 0 \qquad (15)$$

This algorithm was coded and placed in a WHILE loop starting with PTS = 935 and TD_{MIN} = 2.34604E-5 seconds. The loop continues decrementing PTS by 1 until the result is greater than zero. Once the number of points is determined, the total delay time between points can be calculated. In Figure B-3 it requires 2.07234E-4 seconds to acquire one potential, current point. The delay time per point is then,

$$\text{DELAY TIME PER POINT} = [(\text{SCAN TIME} - (2.07234\text{E-}4 * \text{PTS})]/\text{PTS} \qquad (16)$$

Now the delay counts for X, Y, and A can be calculated recalling that there must be a minimum count of 1 in X, Y, and A. As Scanlon[149] points out, one starts by dividing the delay time per point by the largest delay count generator, D3, truncating the result and adding 1 for the minimum delay. The remainder of this division is divided by the next largest delay count generator, D2, truncated, and 1 added to it for the minimum delay. The remainder of the D2 division is then used to calculate the count of the smallest delay generator, D1. TD_{MIN} is subtracted before dividing by D1 because 1 is added to each delay count to account for the minimum delay.

All of these parameters are passed to PROCEDURE TAKEDATA once they have been calculated. The values are popped from the stack and placed in page zero locations. The peripheral address locations of the ADC and DAC are also located in page zero(Figure B-3). A three instruction sequence with the DAC triggers the scan and data acquisition simultaneously. In LOOP1, the potential conversion is performed and stored at memory locations MEMLOC and MEMLOC + 1. Indirect indexed addressing[34,149] was used to store the binary values at the page zero address(4446 at the start) located in MEMLOC. The current is sampled immediately after the potential to obtain a value as close to the potential reading as possible. Because of the overhead involved in storing data, incrementing registers, and generating delays, there is a lag in potential associated with each current. At faster scan rates, this lag can introduce significant errors. The PROGRAM corrects the potential values to their correct values in time using the number of clock cycles(87) between sampling of the current and potential and the scan rate. Depending on the scan direction, the correction term can be calculated by,

$$\pm[(87/1.023\text{MHz}) * \text{SCAN RATE}] \qquad (17)$$

PROCEDURE CORRECT implements this calculation prior to storing the data in file CV?ORIGREAL. The 41 cycle delay contributes the most to this error problem. The user may select very narrow potential limits over which the software may sample with one of its sensitive ranges. The current GAINCODE may not be set to a sensitive range and so the ADC

would be dynamically switching amplifiers to acquire one point. The assembly routine must delay at both sample times to accommodate the most rigorous conditions possible.[165]

Additional overhead is incurred during data acquisition to increment MEMLOC by four for each potential, current point and to subtract one from PAIRS(the number of points) for testing the end of data acquisition. The addition and subtraction involve 16-bit manipulations since both PAIRS and MEMLOC are larger than 255. If there are more points to obtain after subtracting one from PAIRS, the time delay routine is executed and the code branches back to DATA to acquire another point. When the last point has been taken, the DAC terminates the experiment by opening the SCAN-to-ground contact of the remote timing input. When the routine returns to CVDATA, the binary data are retrieved, converted to their real values, corrected for errors in potential, and stored on disk with the appropriate file names. The temperature is recorded prior to and immediately after data acquisition, then averaged for a final value. The PROGRAM stores the temperature and other vital input information in CV?PARAM.

PROGRAMs GRAPHCV and PEAKLOC. Graphing the voltammetric data was accomplished by typing <G> or <g> at the main menu. Following the acquisition of a CV, it was a good idea to check the result before proceeding to a new temperature. When the PROGRAM is invoked, a submenu appears(Figure 28) which presents graphing options. A completely analyzed CV consists of CV?SMTH, CV?SUB, and CV?DECONVOL as well as CV?ORIGREAL(Figure 25). Each of these data files contains the

```
Graphing selections:
  O>riginal set of CV data
  S>moothed set of CV data
  D>econvoluted set of CV data
  C>ombination of deconvoluted and subtracted CV data
  R>eturn to main menu

Please type:C

-----------------------------------------------

Enter the CV number <1..99>:1

Press <SPACE> to continue!

Press <SPACE> to view the graph and
     <SPACE> when finished viewing!

Do you want the graph printed <Y,N>:Y
```

Figure 28. Screen display when G>raphing a set of CV data.

same potential values at each point and the same number of points but differs in its current values. CV?SMTH is created as a result of PROGRAM CVSMOOTH; CV?SUB and CV?DECONVOL are created as a result of PROGRAM DECONVOL. PROGRAM GRAPHCV plots these data files on the Apple HiRes screen, then prompts the user to dump the screen to the printer.

Ordinarily, graphics output would be associated with the output of experimental analysis. Because the TURTLEGRAPHICS UNIT is used, nearly half of the available user memory is consumed. Even with system swapping on and efficient use of the NOLOAD and RESIDENT options, TURTLEGRAPHICS reduces the 20,500 word, user memory to 10,600 words. The built-in MEMAVAIL FUNCTION was used to determine this. Ultimately, only 10,600 words are available for program and data space.[172,177] The data sets that the PROGRAM graphs are quite large. A REAL number requires four bytes and a pointer two bytes.[150] In a linked list of 935 points, approximately 4,700 words are needed. This leaves enough room for PROGRAM GRAPHCV to execute safely if only one list is involved. Inspection of the menu reveals that there is a C>ombination option which plots the subtracted and deconvoluted CVs. This would require 9,400 words of memory and crash the PROGRAM because of a lack of memory for PROGRAM code and data.

PROCEDUREs MARK and RELEASE were used to return dynamic memory to the system.[172] The forms are MARK(HEAP) and RELEASE(HEAP) where HEAP is of TYPE ↑INTEGER. VARiables created by calls to NEW are stored in a stack-like structure called the "heap". MARK sets HEAP to the value of the system's current top-of-heap pointer, and RELEASE sets the system's top-of-heap pointer to the value of HEAP. The linked lists in this

application were created by successive calls to NEW. To avoid heap overflow when graphing the subtracted and deconvoluted curves, the stack was MARKed prior to constructing and graphing the subtracted curve, then RELEASEd prior to doing the same with the deconvoluted curve. Because linked lists and the NEW PROCEDURE were used, the recovery of memory was possible. This would not have been possible in an ARRAY-based implementation.[162,177]

The code for graphing a CV(Figure B-4) is remarkably short. The PROGRAM locates the high and low, potential and current, then scales the points in the list to plot them on the Apple screen. The Apple screen is a rectangle, with the origin (X = 0, Y = 0) at the lower left corner. The upper right corner has the coordinates (X = 279, Y = 191). Scaling converts the points in the CV to $0 \leq X \leq 279$ and $0 \leq Y \leq 191$. PROCEDURE PLOTDOT does the scaling and plots the points on the screen. PROCEDURE FORMAT of UNIT PRNTRTNS dumps the HiRes screen image to the printer. Sample outputs of this PROGRAM are shown in Figure 29.

PROGRAM PEAKLOC is related to PROGRAM GRAPHCV. PEAKLOC is an interactive graphics PROGRAM which allows the user to select the points of cathodic and anodic peak current. The same strategy was used to graph a CV in PEAKLOC as was used in GRAPHCV(Figure B-5). Choosing the file for peak locating is slightly more complicated than selecting a file to graph. FUNCTION FILENAME of UNIT FILERTNS validates a CV number concatenated with a SUFFIX such as ORIGREAL, SMTH, SUB or DECONVOL. This made the graphing options easy to process because only one PROCEDURE was required. PEAKLOC, however, is a PROGRAM which can be executed on a smoothed(CV?SMTH) or deconvoluted(CV?DECONVOL) set of

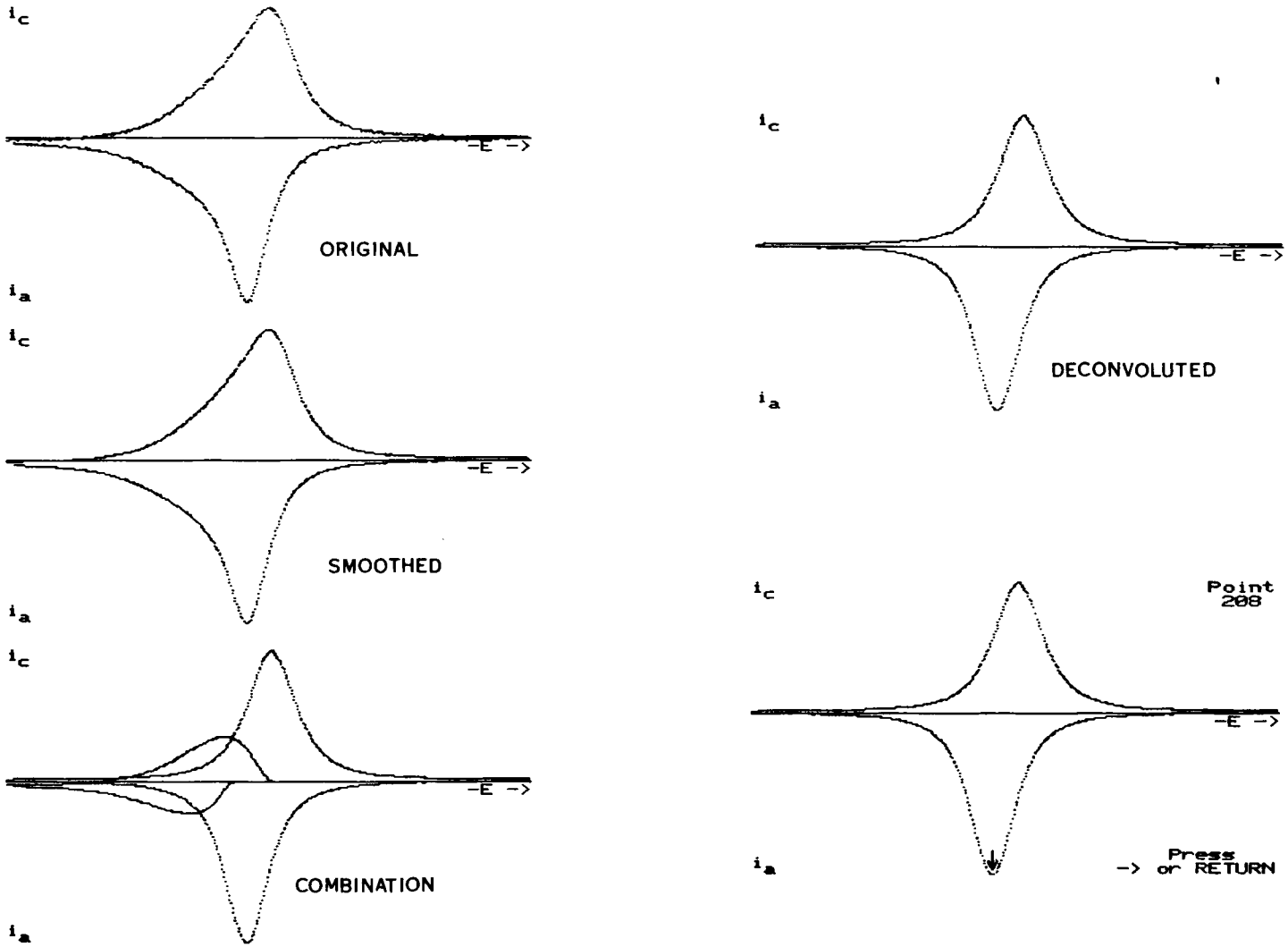


Figure 29. Graphics as they appear in PROGRAMs GRAPHCV and PEAKLOC.

data and must be executed prior to peak fitting smoothed or deconvoluted data. For that reason, record number 22 of Table V contains VARiable PFLSLD, a six letter acronym for Peak Fitting, Locating Smoothed, and Locating Deconvoluted data. When the data are first acquired, CV?PARAM contains PFLSLD = 'FFF' to signify False for each option. If at anytime an original set of data is resmoothed, PFLSLD is reset to 'FFF'.

PEAKLOC first attempts to find CV?SMTH with PROCEDURE FILENAME and aborts if the original data have not been smoothed. If CV?SMTH does exist, PEAKLOC takes the next step and tries to locate CV?DECONVOL. If CV?DECONVOL exists, it is placed in a list for peak locating. If CV?DECONVOL does not exist, CV?SMTH is processed for peak locating. Depending on which file is peak located, PFLSLD is altered in CV?PARAM; thus, PFLSLD = 'FTF' or 'FFT' at the end of the PROGRAM.

A graphical display of PEAKLOC is shown in the lower right corner of Figure 29. Before the graph appears, the user is prompted with the instructions in Figure 30A. The list is circularized near the peak position by changing one pointer(Figure 30B). By pressing the right arrow key, the arrow in Figure 29 can be moved around the circle until the best position is determined. The user may press the right arrow key indefinitely because of the circular nature of the list. The Point value changes each time the right arrow key is pressed. When <RETURN> is pressed, the stated Point value is stored in CV?PARAM as the integer position in the list of highest or lowest current. PEAKLOC also finds the integer positions of the highest and lowest potential. These values are retrieved for peak fitting calculations. In some cases, a

Enter the CV number <1..99>:1

Press <SPACE> to continue!

A graph of the CV data you have chosen will appear with an ARROW pointing near the highest cathodic point or anodic point of the curve. Press (->) to advance the ARROW to the center of the peak and <RETURN> when you are finished. Anytime peaks are relocated in a CV, the peaks must be refitted.

Press <SPACE> when you are ready.

A

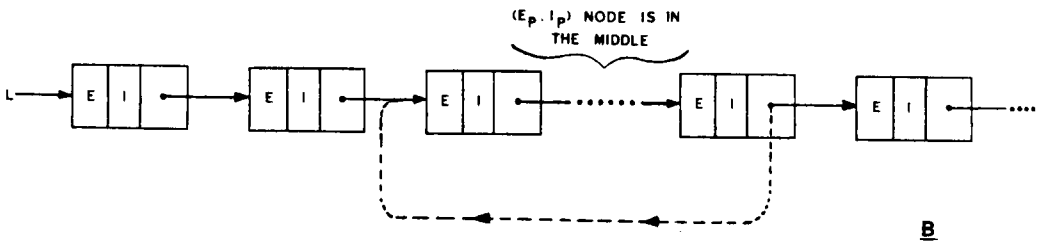


Figure 30. (A) Screen display when L>ocating peaks in a CV, and (B) construction of a circularly linked list.

flatter, more attenuated peak will appear in a CV. If sampling is fast enough, several points of equivalent value will be present at the peak locations. It is very difficult to write algorithms which take this into account and still give consistently accurate peak values.

PROGRAM CVSMOOTH. Every laboratory scientist has had to cope with noise and the resulting loss of valuable information from his or her experiment. The elimination of this noise is an experimental design consideration and becomes particularly important when one is working near the limit of detectability of the experimental apparatus. The process concerned with removing noise components from the signal is called filtering and can be implemented with hardware or software.[14]

Developing techniques to discriminate against noise requires knowledge of the sources and types of electrical noise. White noise is a mixture of all noise frequencies with random amplitudes and phase angles. Low frequency drifts in transducers and amplifiers give rise to pink noise. Interference noise often arises from 60Hz power lines, motors, and nearby oscillators, and has peaks at the fundamentals and harmonics of these frequencies. Interference noise can be effectively eliminated with integrating ADCs, twisted cabling, and/or by sampling at a frequency removed from the harmonics of the interfering frequency.[14,145] Frequency dependent noise can also be discriminated by hardware filtering. Filters are circuits containing reactive and components, such as capacitors, inductors, and resistors. Such circuits are classified as passive filters. This is in contrast to active filters, which also contain operational amplifiers as an integral part

of the circuit. Further classification occurs depending on the range of unattenuated signal frequencies passing the filter, low pass, high pass, bandpass, and notch filters. With these filters, it is simply necessary to choose the RC time constant, and hence the bandwidth and phase shift characteristics, such that the signal frequencies are affected as little as desired.[178] On the CV-27, changing the RC time constant of the passive network was accomplished with the Filter switch.[144]

Some sources of noise are not frequency related and must be eliminated by other means such as software. Thermal noise is produced by the random motion of electrons in resistive elements. Shot noise results from the random movement of discrete charges across junctions. Any noise above and beyond thermal or shot noise is considered excess noise. The main non-frequency dependent type of excess noise is a result of the finite resolution of the SA ADC and is called quantizing noise.[145]

Numerical methods of digital filtering modify a given data point to some function of itself and its nearby points. Digital filtering has an advantage over the analog RC filter because it can use both past and future information stored in the data structure of the experiment. The digital filter thus provides a smoother operation, without the phase shift present with an analog filter. Although it may lead to peak broadening, it does not change the fundamental shape of the signal form.[14]

Filtering is basically a convolution operation since the results of the operation depend on the shape and width of the filter function

convolved with the data.[179] One of the simplest filter operations is based on boxcar averaging. A sequence of data in a box are multiplied by integers, summed, and divided by the number of points in the box. This gives a new center point that is the weighted average of a number of the original data points. When all of the convolution integers are set equal to 1, the raw data are said to be convoluted with a rectangular filter. This filter is called a boxcar filter if it is moved a full boxcar ahead for the calculation of each point. For this very crude filter, minimum distortion is evident as long as the sampling rate is high compared to the frequency components of the signal waveform.[14,179]

A more efficient use of the data is obtained if the box is moved each time by only one data point and a new average point for the cluster is calculated. It is then called a moving average filter and yields a smoothed data set containing almost the same number of points as the original.[14,179] The moving average method used in CVSMOOTH is based on this filter and is the method used by Faulkner and He[143] on the BAS-100,

$$Y_0^* = (Y_{-2} + Y_{-1} + Y_1 + Y_2)/4 \qquad (18)$$

All of the convolution integers are 1, and Y and Y^* are the raw and smoothed data, respectively. The box was moved one point each time and the filtering operation applied to each data set twice. To account for the first and last two points in the set which do not get replaced with a weighted average, the first two points were set equal to the third

and the last two points set equal to the third from last. This retains the same number of points in the set after each filtering operation. After a single pass of the filter, the amount of noise remaining is inversely proportional to the square root of the filter width.[178] The graphics results in Figure 29 support the chosen filter width and algorithm since neither peak broadening nor signal distortion are evident in the SMOOTHED curve.

An alternative filtering approach is based on the fact that cyclic voltammetry produces a continuous signal that can be represented mathematically by a Fourier series.[145,178,179] A Fourier series expresses the signal as a summation of sine and cosine terms. There will be a fundamental, plus some number of odd, even, and odd-even harmonics of the fundamental frequency. Coefficients of each term correspond to the amplitude. When the experiment duration is very short, PROGRAM CVDATA reduces the number of points acquired to reduce the overall time spent performing analog to digital conversions. While there may be enough points to reproduce the signal, the sampling rate will be much closer to the Nyquist frequency. In these cases, the moving average filter can broaden and attenuate the peaks severely. For that reason, a set of algorithms based on the fast Fourier transform(FFT) were implemented as an alternative smoothing method.

The FFT algorithm, developed by Cooley and Tukey[180] in 1965, made it possible to use the transform on small computers. Even with the use of the FFT algorithm, the overhead time for calculating an FFT on a microcomputer in a high-level language is still large. Other factors, however, have played a part in the reluctance of electrochemists to

practice Fourier smoothing. One reason is that descriptions of Fourier transformation are often couched in the unfamiliar jargon of the communications engineer or applied spectroscopist. A second is the common misconception that Fourier transformation and inversion are massive number crunching operations which cannot be implemented on small microcomputers. Another reason is that the deserved success of the FFT has spawned the belief that this is the only practical algorithm for transformation and inversion. The paucity of information on Fourier algorithms in the chemical literature also makes software implementation difficult.[181]

For a comprehensive discussion of the FFT, one should consult some of the classic references dealing with its underlying theory and operation.[182,183] The forward Fourier transform of a function h(t) is defined by the integral,

$$H(v) = \int_{-\infty}^{+\infty} h(t)\ \exp(-i2\pi vt)\ dt \qquad (19)$$

where $i = \sqrt{-1}$. The functions h(t), H(v) are Fourier transform pairs. The inverse transform to go from frequency space back to time space is given by,

$$h(t) = \int_{-\infty}^{+\infty} H(v)\ \exp(i2\pi vt)\ dv \qquad (20)$$

The analytical solution of these Fourier transform equations cannot be directly carried out on a computer. Therefore, the range of the integral is made finite and the integral is approximated by evaluation at a

finite, discrete number of points. Imposing these restrictions leads to transform eq. (1) in Figure 31. The transformation operates on a sequence $z_0, z_1, \ldots, z_{N-1}$ of complex-valued numbers to produce another set, also with N members, of complex-valued numbers, $Z_0, Z_1, \ldots, Z_{N-1}$ known as the transforms. Fourier inversion, represented by eq. (2) in Figure 31, regenerates the original numbers from the transforms.

In most electrochemical applications, the original numbers consist of real data, gathered at equally-spaced times(Δt), so that no imaginary components exist and $z_j = x_j + iy_j$. The transforms remain complex, $Z_k = R_k + iI_k$, but the set of transforms may be treated as a double set of real numbers calculable from x_j by eqs. (3) and (4) in Figure 31. The time sequence data are regenerable via eq. (5)(Figure 31). For the case of real original data(all $y_j = 0$), eqs. (3-5) in Figure 31 are strictly equivalent to eqs. (1) and (2) and there is no need to use complex numbers or represent sinusoids by exponential notation.[182,183]

The information content of the original data is transferred, on Fourier transformation, into about the first half of the R_k, I_k numbers. The second half merely duplicates the first in magnitude, $R_{N-k} = R_k$ and $I_{N-k} = -I_k$. For each R_k, I_k pair, $\sqrt{R_k^2 + I_k^2}$ gives the fractional content of the original data that can be said to have the frequency $k/(N\Delta t)$. To reduce the higher frequency content of the data we need only diminish the magnitude of R_k and I_k for k values close to the upper (N-1)/2 limit, leaving those with k values close to zero unmodified, and perform a Fourier inversion. This is the principle of Fourier smoothing.[180-183]

$$Z_k = \frac{1}{N} \sum_{j=0}^{N-1} z_j \exp\left(\frac{-2i\pi jk}{N}\right) \qquad k = 0,1,\ldots,N-1 \quad (1)$$

$$z_j = \sum_{k=0}^{N-1} Z_k \exp\left(\frac{2i\pi jk}{N}\right) \qquad j = 0,1,\ldots,N-1 \quad (2)$$

$$R_k = \frac{1}{N} \sum_{j=0}^{N-1} x_j \sin\left(\frac{2\pi jk}{N}\right) \quad (3)$$

$$I_k = \frac{-1}{N} \sum_{j=0}^{N-1} x_j \sin\left(\frac{2\pi jk}{N}\right) \quad (4)$$

$$x_j = \sum_{k=0}^{N-1} R_k \cos\left(\frac{2\pi jk}{N}\right) - I_k \sin\left(\frac{2\pi jk}{N}\right) \qquad j = 0,1,\ldots,N-1 \quad (5)$$

$$R_0 = \frac{1}{N} \sum_{j=0}^{N-1} x_j \quad (6)$$

$$R_k = \frac{X_0}{N} + \frac{1}{N} \sum_{j=1}^{N-1} x_j \cos\left(\frac{2\pi jk}{N}\right) \qquad k = 1,2,\ldots,n-1 \quad (7)$$

$$I_k = \frac{-1}{N} \sum_{j=1}^{N-1} x_j \sin\left(\frac{2\pi jk}{N}\right) \qquad k = 1,2,\ldots,n-1 \quad (8)$$

$$\tilde{x}_0 = R_0 + 2 \sum_{k=1}^{n-1} R_k \quad (9)$$

$$\tilde{x}_j = R_0 + 2 \sum_{k=1}^{n-1} R_k \cos\left(\frac{2\pi jk}{N}\right) - I_k \sin\left(\frac{2\pi jk}{N}\right) \qquad j = 1,2,\ldots,N-1 \quad (10)$$

Figure 31. FFT algorithms for smoothing electrochemical data.

If one permits the user to select an integer n, a simple means of selectively modifying the values of R_k and I_k by replacing those having $n \leq k \leq N/2$ with zero is established. When the number of transforms, n, is small, the more denuded of high frequencies will the subsequent invert be; the closer n is to N/2 the less affected will the regenerated signal be.[181]

Because there is no purpose in calculating those values of R_k and I_k that will be zero-filled, the algorithms for Fourier smoothing can be abbreviated to eqs. (6-10)(Figure 31), where $\tilde{x}_j$ is the high frequency, stripped analog of x_j. The factor of 2 in eqs. (9) and (10) in Figure 31 arises from eq. (5) by restricting $n < N/2$ and by taking advantage of the symmetries ($R_{N-k} = R_k$, $I_{N-k} = -I_k$) already noted.[181]

The method used is neither fast nor truly Fourier in nature. It still requires a 20Nn multiplications or divisions and 4Nn cosine or sine evaluations. For N = 935 and n = 40 in this application, it would take the better part of an hour to execute the Fourier program. Aubanel et al.[181] provided the Fourier algorithms for implementation in UCSD Pascal. It was therefore worth the effort to evaluate its application as an alternative smoothing method. While the algorithms are not an FFT they share the advantages of an FFT by reducing the number of needed sines and cosines and replacing redundant multiplications with summations. Within the code of Figure B-6, the non-faradaic background current is subtracted before Fourier smoothing. The PROGRAM computes two straight lines of background current from the initial, final and switching points. An option is provided to add the background current back in(Figure 32).

```
Enter the CV number <1..99>:1

Press <SPACE> to continue!

-----------------------------------------------

Anytime a data set is resmoothed, the peaks
  must be relocated and refitted for output!

Smoothing selections:
  M>oving average
  F>ourier transformation

Please type:F

Add non-faradaic current back in <Y,N>:N

Input number of Fourier transforms <1..50>:25

Computing real part of transforms.

Computing imaginary part of transforms.

CV1SMTH is now on disk in drive 2!

Press <SPACE> to continue!
```

```
Anytime a data set is resmoothed, the peaks
  must be relocated and refitted for output!

Smoothing selections:
  M>oving average
  F>ourier transformation

Please type:M

Computing five point moving average!

CV1SMTH is now on disk in drive 2!

Press <SPACE> to continue!
```

Figure 32. Screen displays possible when executing CVSMOOTH.

A comparison of the outputs from both methods is shown in Figure 33. The moving average and Fourier methods were executed on a single set of original real data. In all cases, N = 935 and it is obvious that the moving average filter is the most efficient. The original data sre sufficiently smooth that a very high number of transforms is needed to make Fourier smoothing effective. This increases the computing time but yields i_ps closer to that obtained by the moving average filter. To decrease the computing time, n can be lowered, but this oversmooths the data causing a portion of the desired signal to be rejected along with the high frequency noise. The wave shape is generally broader and lower in magnitude as shown by the values of i_p. Conversely, choosing too large a value of n causes much of the mid-frequency noise to be retained and appears as ripples on the smoothed data.[181]

PROGRAM CVSMOOTH implements the moving average method in less than 15 lines of code while the Fourier method requires ca. 200 lines! The Fourier method was very difficult to debug and required implementation as an ARRAY. To determine the size of the ARRAY, the maximum N value of 935 had to be known beforehand. PROCEDUREs CV2ARRAY and ARRAY2CV performed file I/O for the Fourier option. VARiable SMOOTH in Tables V and VI in each parameter file reflects the type of smooth and other information such as the number of transforms. The PROGRAM is also designed to accommodate resmoothing of original data at any time by resetting VARiable PFLSLD to 'FFF'. This is necessary in the Fourier option because the user may choose too large or small an n value and have to resmooth the data. In these cases, old CV?SMTH files are

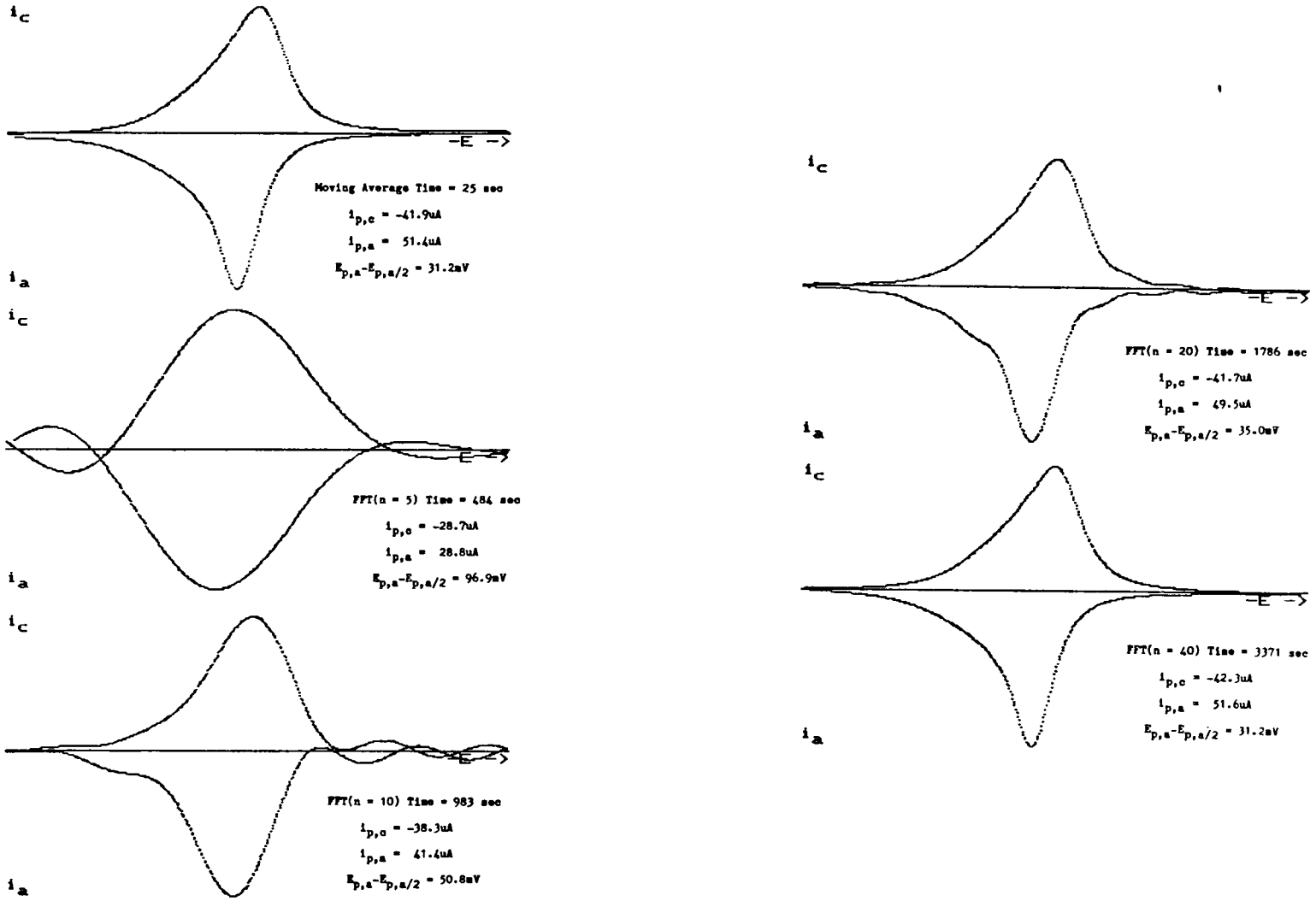


Figure 33. Comparison of the moving average and FFT filters.

simply overwritten by the newly smoothed values. Although the moving average method was used to smooth all of the data for the results in this study, the Fourier option is left as an option to handle other noisy electrochemical experiments.

An important topic in numerical analysis is the introduction of error into a calculation by representing it in discrete terms as a summation. An FFT may be thought of as an infinite sequence of similar discretely bounded functions. The boundaries between the functions represent sharp discontinuities that can lead to an enhancement of the high-frequency components in the transformed space. To avoid any error arising from the false accentuation of high frequencies in the vicinity of $n/(N\Delta t)$, the R_k and I_k transforms were multiplied by a quadratic filter to drop the functions to zero at both ends.[179,181] Error in approximation is also important when fractional calculus is used to integrate cyclic voltammograms.[184]

PROGRAM PEAKFIT. Once a certain degree of smoothness has been imparted to the digitally acquired data, the user may proceed with an option to perform numerical analysis of the CV. Within this PROGRAM, a series of points on either side of the peak points selected in PEAKLOC are fit by the method of cubic splines to determine optimum values of $E_{p,a}$, $E_{p,c}$, $i_{p,a}$, and $i_{p,c}$. The Newton-Cotes, trapezoid rule was implemented to determine the cathodic(Q_c) and anodic(Q_a) charges. From the screen output in Figure 34, the execution is straightforward. The user is required to have previously executed PEAKLOC, thereby setting VARiable PFLSLD = 'FTF' or 'FTT'. If the deconvoluted curve has had

```
Enter the CV number <1..99>:1

Press <SPACE> to continue!
You must locate peaks in the smoothed data!

------------------------------------------------

Enter the CV number <1..99>:1

Press <SPACE> to continue!
Computing Ep and Ip by spline fitting!
Computing cathodic and anodic charges!
Transferring parameters to CV1PARAM!
Press <SPACE> to continue!
```

Figure 34. Screen displays possible when executing PEAKFIT.

its peaks located(PFLSLD = 'FTT'), then PEAKFIT automatically retrieves and analyzes the curve, CV?DECONVOL; otherwise CV?SMTH is retrieved and analyzed. At the termination of the PROGRAM, all of these calculated values are placed in CV?PARAM and the first character in PFLSLD is set to 'T' to signify peak fitting has been performed. This procedure seems complicated at first, but recall that one original set of cyclic voltammetric data gives rise to two other related CVs, CV?SMTH and CV?DECONVOL. VARiable PFLSLD is used to track which file has been analyzed so that only one parameter file is needed to describe both CVs. PROCEDURE DATA2FILE in Figure B-7 demonstrates the Pascal method of checking the value of PFLSLD.

Two electrochemical parameters were calculated by numerical methods, the peak current and charge. Recently, errors arising from calculus operations on digitally acquired data have been evaluated for electrochemical methods. Although Oldham[184] has shed much light in this area, these error calculations are still approximate. Sources of error in numerical analysis can arise from measurements, round-off or truncation in the computer, approximations of numerical algorithms, and propagation of these errors by the numerical method.[185] Round-off errors are the result of the finite resources of the computer and can be diminished by performing calculations in double precision. Truncation errors can occur when using approximations in a calculation. These sources of error will often determine which numerical method is best suited for the particular calculus operation.

For numerical integration, there are hundreds of algorithms available, depending on the form of the data and the behavior of the

integrand.[185] The Newton-Cotes methods are adequate for most applications when the integrands are well behaved.[14] To evaluate the indefinite integral,

$$Q(t) = \int_0^t i(t)dt \tag{21}$$

one splits the integration interval into several subintervals, and approximates f(x) in each of these by a local polynomial of low degree to obtain the integral approximation.[186] In essence, we determine the area of each subinterval and add the subareas to obtain the total area. To evaluate the charge, the potential points were converted to time by dividing by the scan rate and the trapezoid rule applied to the i-t data. The area of each subinterval is then,

$$\int_{t_1}^{t_2} f(x)dx = 0.5(t_2-t_1)[f(t_1) + f(t_2)] + I_E, \tag{22}$$

where I_E is the integration error. The digitally acquired data were well suited to this formula which was implemented in FUNCTION CHARGE(Figure B-7). Composite formulae such as these introduce truncation error when the number of subintervals is small and round-off error when the number of subintervals is large.

Oldham[184] showed that the integration of i-t data by composite formulae such as the Simpson rule is not superior to the trapezoidal rule. In fact, for cyclic voltammetry, the trapezoid rule generates an absolute error of, $\pm Q = (\sigma)(\Delta)(N)^{1/2}$, where σ is ths absolute error in the current, Δ is the time in seconds between points, and N is the

number of intervals. The surprising implication of this equation is that the absolute error in the calculated charge steadily increases as the experiment proceeds due to inaccuracy in the current data. The error in the integral depends only slightly on the composite algorithm used for integration since Simpson's rule gives the absolute error, $\pm Q = 1.05(\sigma)(\Delta)(N)^{1/2}$. For a typical deconvoluted curve, where Q_a = 200.0uC, Δ = (30sec/467intervals) or 0.064sec, N = 467subintervals, and $\sigma = \pm 2$uA, then the absolute error in charge would be $\pm Q_a = \pm 2.8$uC or $\pm 1.4\%$. Thus, the trapezoid rule does not introduce significant approximational error.

So far, the existence of error has been regarded as a limitation upon what can be accomplished. Yet it is also possible to introduce error into a numerical process as a benefit, whereby computational speed can be increased at the expense of unnecessary precision.[186,187] The classical theory of approximation alludes to this approach when an interpolation method is used. Interpolation methods tabulate the value of a function at such close intervals that intermediate values can be readily inferred with sufficient accuracy. The functions used most frequently for interpolation are polynomials. In initial attempts to fit the peak data to a single third degree polynomial, one or two values in some of the data sets were inexact enough to lead to oscillations in the cubic equation. Since the cubic polynomial is not a constrained function, the function continues oscillating with an undiminished amplitude. Thus, one might input an E_p value and produce a completely erroneous i_p, even in double precision. If, however, we constrain the previously smoothed function at both ends and fit each

interval in a piecewise fashion with polynomials of lower degree, we can compute more accurate values of i_p.[185-188]

In cubic spline interpolation, the function and its first and second derivative are required to agree at the ends of the subintervals where one cubic polynomial meets the next. This means that the first and second derivatives of the splines are continuous across the knots and that one spline is a function of neighboring splines. The functions are called splines because they exhibit a maximum degree of smoothness and a minimal degree of oscillation and overshoot. An additional condition is required to constrain the values of the second derivatives at either end of the data set so that the spline equations can be solved.[187,188]

A derivation of the equations follows. Let the given set of points (x_1,y_1), (x_2,y_2),...(x_m,y_m), be arranged in order of increasing values of x. We accomplish the spline fit by connecting each pair of adjacent points with a section of a third degree polynomial, matching up the sections so that the first and second derivatives are continuous at each point. Let $Z_1,Z_2,\ldots,Z_m$ be the values of the second derivatives at the points. Then between points (x_k,y_k) and (x_{k+1},y_{k+1}), the second derivative has the value,

$$y'' = [Z_k(x_{k+1} - x)/d_k] + [Z_{k+1}(x - x_k)/d_k] \qquad (23)$$

where $d_k = x_{k+1} - x_k$. Integrating, we obtain for the first derivative,

$$y' = [-Z_k(x_{k+1} - x)^2/2d_k] + [Z_{k+1}(x - x_k)^2/2d_k] + c_1 \qquad (24)$$

where c_1 is a constant of integration. Integrating again, we obtain for the equation of the curve,

$$y = [Z_k(x_{k+1} - x)^3/6d_k] + [Z_{k+1}(x - x_k)^3/6d_k] + c_1x + c_2 \qquad (25)$$

where c_2 is a constant of integration. The constants c_1 and c_2 can be evaluated from the fact that the spline curve passes through (x_k,y_k) and (x_{k+1},y_{k+1}). We have,

$$y_k = (Z_k d_k^2/6) + c_1x_k + c_2 \qquad (26)$$

$$y_{k+1} = (Z_{k+1}d_k^2/6) + c_1x_{k+1} + c_2 \qquad (27)$$

from which,

$$c_1 = [(y_{k+1} - y_k)/d_k] - [(Z_{k+1} - Z_k)d_k/6] \qquad (28)$$

$$c_2 = [(y_kx_{k+1} - y_{k+1}x_k)/d_k] - [(Z_kx_{k+1} - Z_{k+1}x_k)d_k/6] \qquad (29)$$

Substituting these values into eq. (25), we have for the equation of the curve,

$$\begin{aligned} y = {} & [Z_k(x_{k+1} - x)^3/6d_k] + [Z_{k+1}(x - x_k)^3/6d_k] \\ & + [(x_{k+1} - x)(y_k/d_k - Z_kd_k/6)] \\ & + [(x - x_k)(y_{k+1}/d_k - Z_{k+1}d_k/6)] \end{aligned} \qquad (30)$$

In this equation, all quantities are known except Z_k and Z_{k+1}, the values of the second derivative at the end points of the interval. One

condition which will help determine these values is that the slope at (x_k,y_k) as determined from eq. (24) above must be the same as that determined by the corresponding formula for the interval (x_{k-1},y_{k-1}) to (x_k,y_k). When the value of c_1 from eq. (28) is used in eq. (24), the equation becomes,

$$y' = -[Z_k(x_{k+1} - x)^2/2d_k] + [Z_{k+1}(x - x_k)^2/2d_k] + [(y_{k+1} - y_k)/d_k] - [(Z_{k+1} - Z_k)d_k/6] \quad (31)$$

The corresponding relation for the preceding interval is,

$$y' = -[Z_{k-1}(x_k - x)^2/2d_{k-1}] + [Z_k(x - x_{k-1})^2/2d_{k-1}] + [(y_k - y_{k-1})/d_{k-1}] - [(Z_k - Z_{k-1})d_{k-1}/6] \quad (32)$$

At the point (x_k,y_k), these relations give,

$$y'_k = (-Z_k d_k/2) + [(y_{k+1} - y_k)/d_k] - [(Z_{k+1} - Z_k)d_k/6]$$
$$= (Z_k d_{k-1}/2) + [(y_k - y_{k-1})/d_{k-1}] - [(Z_k - Z_{k-1})d_{k-1}/6] \quad (33)$$

or collecting the unknowns Z_{k-1}, Z_k, Z_{k+1} on one side of the equation,

$$(Z_{k-1}d_{k-1}/6) + Z_k[(d_{k-1} + d_k)/3] + (Z_{k+1}d_k/6) = [(y_{k+1} - y_k)/d_k] - [(y_k - y_{k-1})/d_{k-1}] \quad (34)$$

We have an equation like this for each of the internal points, that is, $k = 2,3,\ldots,m-1$. There are $m-2$ equations in the m unknowns

$Z_1, Z_2, \ldots, Z_m$. To determine these quantities exactly, we must specify the values of the second derivatives, Z_1 and Z_m, at the end points. The choice will influence the shape of the fit, especially near the end points. PROGRAM PEAKFIT requires the third derivative to be continuous at (x_2, y_2) and (x_{m-1}, y_{m-1}). From eq. (23) the third derivative is,

$$y''' = -Z_k/d_k + Z_{k+1}/d_k \tag{35}$$

Equating values for k = 1 and k = 2,

$$-Z_1/d_1 + Z_2/d_2 = -Z_2/d_2 + Z_3/d_2 \tag{36}$$

or,

$$-Z_1/d_1 + Z_2(1/d_1 + 1/d_2) - Z_3/d_2 = 0 \tag{37}$$

Similarly, equating values for k = m - 2 and k = m - 1,

$$-Z_{m-2}/d_{m-2} + Z_{m-1}(1/d_{m-2} + 1/d_{m-1}) - Z_m/d_{m-1} = 0 \tag{38}$$

Eqs. (34), (37), and (38) constitute m equations in m unknowns for the quantities $Z_1, Z_2, \ldots, Z_m$. The system of equations is tridiagonal and strictly diagonally dominant. Therefore its solution is fast. Interpolation consists of finding which two points (x_k, y_k) and (x_{k+1}, y_{k+1}) the given value of x lies between and then finding y from the equation,

$$y = c_{1,k}(x_{k+1} - x)^3 + c_{2,k}(x - x_k)^3 + c_{3,k}(x_{k+1} - x) + c_{4,k}(x - x_k) \quad (39)$$

where the constants $c_{1,k}, c_{2,k}, c_{3,k}, c_{4,k}$ have been previously computed and stored.[187]

These equations were implemented to find the peak currents. The potential values between the end points were calculated by subtracting the end point potentials and dividing by the number of subintervals. Equal spacing of the x_ks prevented sudden jumps in the first derivative.[186] For the FFT(n=40) CV of Figure 33, the first and second derivatives of the cubic splines for the anodic data are shown in Figure 35.[14] The continuity constraints are readily apparent in the plots. The current data are further apart at either end of the data set where the highest and lowest values of the first derivative are anticipated. Near the true i_p, the current values are closer together, so the first derivative approaches zero. Thus, a sigmoidal shaped curve can be predicted for the first derivative. The second derivative curve peaks near 422mV where the first derivative curve exhibits its maximum slope. Cubic splines can give excellent results at the expense of unnecessary precision. The greatest source of error in peak analysis software arises from the user-interactive, peak locating software.

Once the charges and related peak potentials and currents have been calculated, $E_{p,a/2}$ on the more negative potential side of the anodic peak is determined. $E_{p,a/2}$ is the potential of half peak current and was determined by searching the list for the NODE with the current value closest to $i_{p,a/2}$. The error in $E_{p,a/2}$ should be greater than that for Q(t) or i_p since a point selection method is used to find

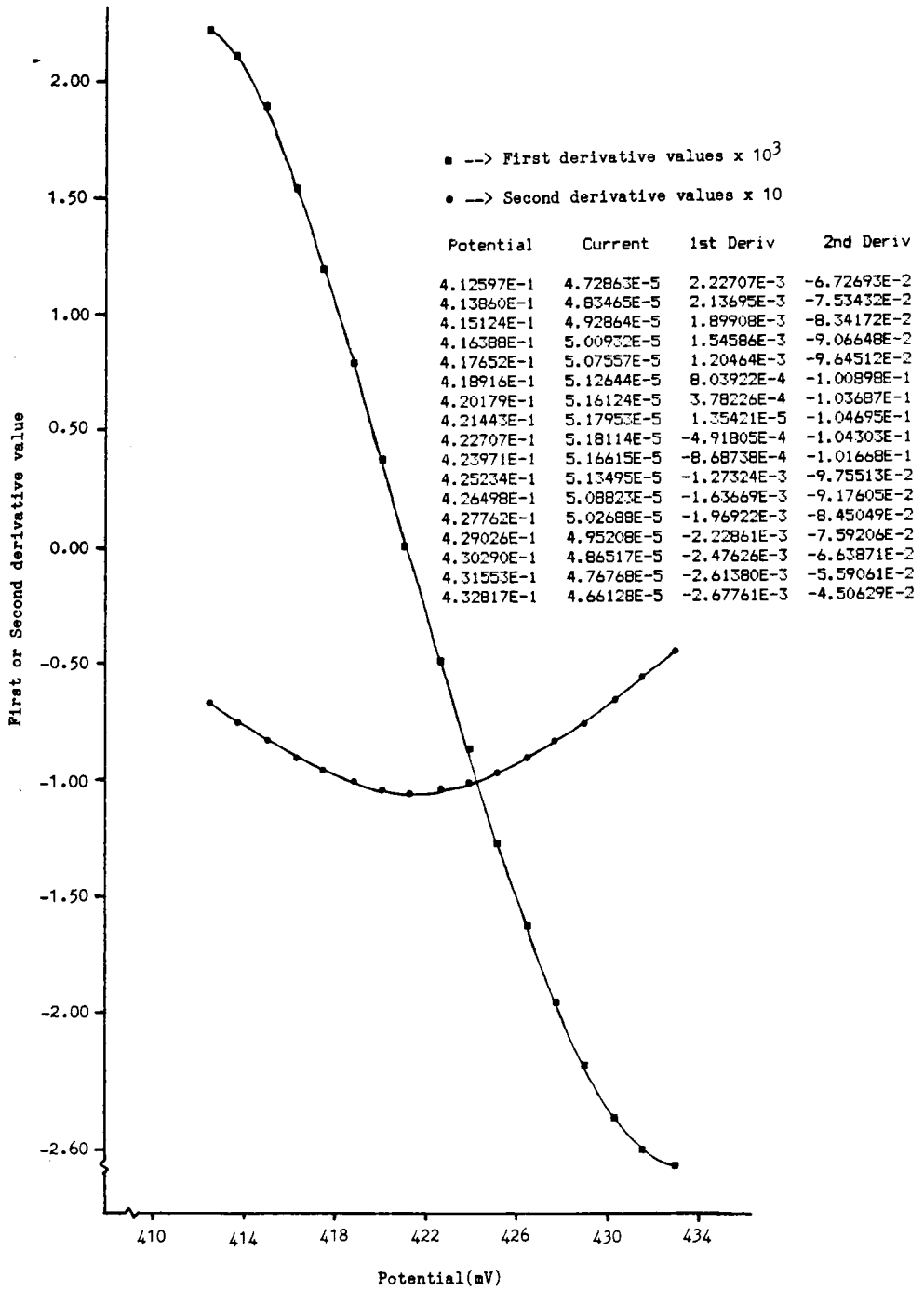


Figure 35. First and second derivative plots of the cubic splines determined for the FFT(n=40) CV of anodic peak data. Current is in amperes.

the value in a region of the CV where di/dt is of greatest magnitude. The error in this value can be reduced by sampling at a high frequency or using an inverse interpolation method.[189]

PROGRAM DECONVOL. In this application, deconvolution is defined as an inverse convolution operation. The effect of deconvolution is shown in Figure 29 where a component of the SMOOTHED curve is removed. We are left with a symmetric, DECONVOLUTED CV which appears to describe one and only one redox process and closely resembles a theoretical thin film voltammogram.[36,37] If the SMOOTHED curve is viewed as the overall time-varying signal, y(t), and the subtracted and deconvoluted curves viewed as two other time-varying signals, f(t) and g(t), respectively, the three functions can be related by the general convolution operation,[14,190]

$$y(t) = \int_{-\infty}^{+\infty} f(u) \cdot g(x-u)du = f(t) * g(t) \qquad (40)$$

where the convolution operator symbol * is a shorthand notation for the integral.

In the FFT method described earlier, the original, noisy data function, f(t), was convolved with a quadratic filter function g(t) to yield a smoothed function, y(t). In terms of eq. (40), this can be represented,

$$
\begin{array}{ccccccc}
f(t) & * & g(t) & = & y(t) & \longleftarrow \text{Convolution} & (41) \\
\downarrow \text{FFT} & & \downarrow \text{FFT} & & \uparrow \text{FFT}^{-1} & & \\
f(v) & x & g(v) & = & y(v) & \longleftarrow \text{Multiplication} & (42)
\end{array}
$$

where FFT^{-1} stands for the inverse Fourier transform. Changing the filter function g(t) changes the effect of the convolution operation on the original data. This points out the potential danger of carrying out real-time data filtering with hardware filters. Hardware filtering can unknowingly distort peak positions. It is far better in many instances to software filter the original data with a convolution operation to decide on the best filter function.[14]

PROGRAM CVSMOOTH allows the user to carry out this convolution investigation to filter the original data. This y(t) curve now needs to be separated into its two other functions, f(t) and g(t), to obtain the DECONVOLUTED curve. A completely non-numerical set of algorithms were implemented to perform deconvolution! Deconvolution is exact, and execution of the algorithms requires ca. 20 seconds. Moreover, the linked list data structure facilitated implementation.

PROGRAM DECONVOL(Figure B-8) first checks for a valid CV that has been smoothed. If there is a CV?SMTH file, the PROGRAM checks to see if the file, CV?DECONVOL, already exists for that CV number. If it does, the user is not allowed to continue(Figure 36). Otherwise, the ubiquitous VARiable , PFLSLD, is checked to see if the second character in the string is 'T' or 'F'. At this point, the possible values of PFLSLD are 'FFF', 'FTF', or 'TTF' and the third character will be set to 'T' once deconvolution is completed. This is how PEAKFIT knows

```
Enter the CV number <1..99>:1

Press <SPACE> to continue!
You cannot deconvolute twice!
Press <SPACE> to continue!
```

```
Enter the CV number <1..99>:1

Press <SPACE> to continue!
You must locate peaks in the CV!
Press <SPACE> to continue!
```

```
Enter the CV number <1..99>:1

Press <SPACE> to continue!

Computing deconvoluted curve!
CV1DECONVOL & SUB are now on disk in drive 2!
Press <SPACE> to continue!
```

Figure 36. Screen output possibilities for PROGRAM DECONVOL.

which curve, CV?SMTH or CV?DECONVOL, to load for calculation and also how CVOUTPUT knows which curve the parameter file, CV?PARAM, currently describes. For deconvolution to work, the PROGRAM requires the second character in PFLSLD to equal 'T'(Table VI). This signifies that the peak currents and switching potential have been located and stored in CV?PARAM.

Figure 37 is a schematic of the deconvolution operation on a SMOOTHED curve. There are two cases, one for a positive and one for a negative scan. The algorithms are restricted to one cycle CVs and initial potentials must equal one of the potential limits. PROCEDURE NEGCONST constructs a DECONVOLUTED curve from the SMOOTHED one for a negative scan. The smoothed data are retrieved from the file and placed in a linked list in the reverse order in which they were acquired. The potential values can be copied into the new, deconvoluted list verbatim as the list is traversed NODE-by-NODE. The current values in the new list were set to zero from START to HII. Once HII is reached, the deconvoluted curve is constructed identically of the current and potential values from HII to LOI. While traversing the smoothed list two stacks of current values are made for the current values between HII and LOE and LOE and LOI. The purpose of the stacks is to reverse the order of data as they are encountered in the list.[173,175] That is, points closer to LOE and LOI in Figure 37 are closer to the top of their respective stacks, S1 and S2. Between LOI and FINISH, the potentials are copied and the currents set to zero.

Now there are two sections of the CV which lack current values. Since the stacks have reversed the order of the currents, we can begin

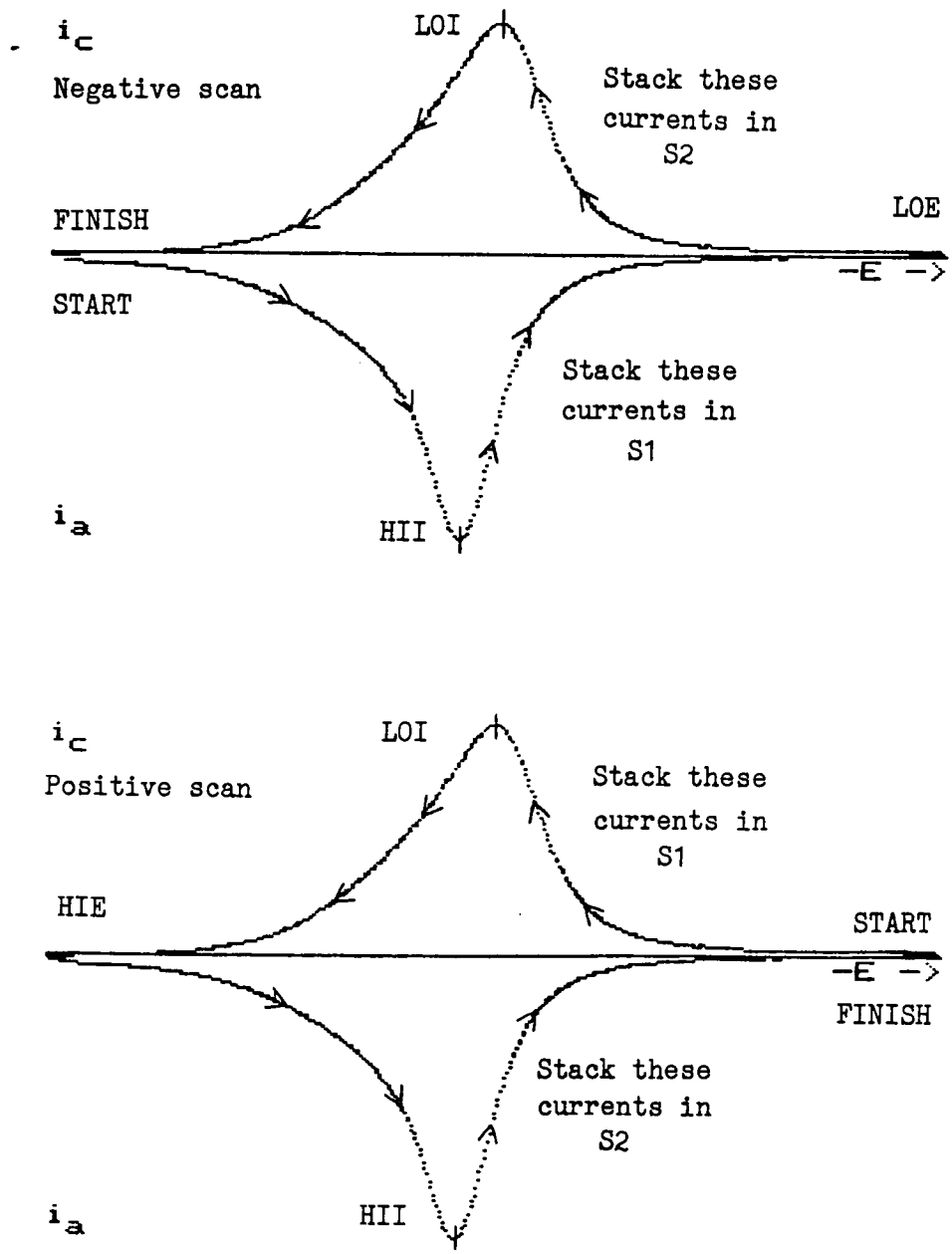


Figure 37. Schematic representation of deconvolution algorithms.

at START in the deconvoluted list and pop values of the stack one-by-one to copy them into the new list.[173,175,176] This reflects the current values between LOE and HII onto the portion between START and HII. The same algorithms were applied to copy the current values from LOI to LOE into the NODEs between LOI and FINISH.

For a positive scan, the first and last sections of the SMOOTHED curve are stacked and copied into the middle two sections. Overall, this approach becomes slightly complicated when accounting for integer peak positions not coinciding with the 1/4 and 3/4 positions in the data set. Seldom would one encounter peaks at the integer positions of (1/4) * 935 or (3/4) * 935. To account for this, the first or last point in the stacks was copied several times, or the last few points in the stacks were ignored. A point-by-point subtraction of the deconvoluted currents from the smoothed ones gives the subtracted curve. Both CVs were subsequently stored as files CV?DECONVOL and CV?SUB.

PROGRAM CVOUTPUT. After the PROGRAMs have been executed to collect, smooth, deconvolute, and calculate key parameters, the CV parameters can be printed. The PROGRAM consists mainly of print and formatting statements and a couple of PROCEDUREs to perform either interaction parameter or thermodynamic calculations. One version of CVOUTPUT, called CVBROWN, performs interaction parameter calculations and the other, called CVTHERM, performs thermodynamic calculations. One uses the Filer to transfer one of these files to APPLE2: as CVOUTPUT before booting the system. The programming strategy facilitated the changing of code. The thermodynamic theory appeared in the literature after all

of the data had been collected.[38-40] CVBROWN was modified for thermodynamic output by changing one PROCEDURE. This reiterates the advantage of modular programming and the way we can facilitate code changes by isolating certain PROGRAMs, PROCEDUREs, and FUNCTIONs. There are two curves which CV?PARAM may describe, the SMOOTHED and DECONVOLUTED CVs. They are distinguished by checking PFLSLD. The curve that CV?PARAM currently corresponds to is reflected in the "Data form" of Figure 38. The PROGRAMs will perform calculations on the SMOOTHED or DECONVOLUTED CVs. CVOUTPUT will not execute unless the peaks have been located and fitted. The scale of the CV is printed here rather than displaying it in the graphics output, since there was not enough code space available in GRAPHCV to implement a scaling calculation. Sample outputs which were obtained for all CVs are shown in Figure 38.

TCNQ-TPE Programs

Most of the PROGRAMs written for PVF calculations were used verbatim for the TCNQ-TPE calculations. PROGRAMs INTERPAR, CVDATA, GRAPHCV, PEAKLOC, DECONVOL did not change. PROGRAM CVSMOOTH was streamlined by removing the FFT option which we knew from previous work to be very inefficient. CVSMOOTH incorporates provisions for subtracting-then-adding two straight lines of charging current since evaporation-coated TCNQ-TPE films exhibit significant charging current in their voltammetry.

Additional minor changes were made to PEAKFIT and CVOUTPUT so that calculations could be performed on the cathodic peaks. The interaction

EXPERIMENT DESCRIPTION:
CV number :1
Title :Electroppt. PVF thin film/0.1M TBAP/C6H5CN vs. Ag/AgCl
Time :THU OCT 30 12:45:54 PM
Points in CV :935
Smooth type :2 x five point moving average
Data form :Simple smoothed

EXPERIMENTAL CONDITIONS:
Temperature (C)= 2.25984E1
High potential (V)= 6.87011E-1
Low potential (V)= 9.76581E-2
Scan rate (V/s)= 2.00000E-2
Initial potential (V)= 1.05955E-1
Sensitivity (A/V)= 1.00000E-5
Scan direction = Positive

A

SCALE:
Current (A/cm)= 1.05236E-5
Potential (V/cm)= 4.93678E-2

CATHODIC:
Peak potential (V)= 3.55837E-1
Peak current (A)= -3.71401E-5
Charge (Coul)= -2.02961E-4

ANODIC:
Peak potential (V)= 3.85741E-1
Peak current (A)= 3.49050E-5
Charge (Coul)= 2.28689E-4

USEFUL VALUES:
Gamma T * r = 7.66756E-1
Epa/2 (V)= 3.61327E-1
Epa/2 - Epa (mV)= -2.44141E1
1/T * 1000 (1/K)= 3.38125
(i/Qv) * (4 - (2 * Gamma T * r)) (V)= 1.88231E1

EXPERIMENT DESCRIPTION:
CV number :1
Title :Electroppt. PVF thin film/0.1M TBAP/C6H5CN vs. Ag/AgCl
Time :THU OCT 30 12:45:54 PM
Points in CV :935
Smooth type :2 x five point moving average
Data form :Deconvoluted

EXPERIMENTAL CONDITIONS:
Temperature (C)= 2.25984E1
High potential (V)= 6.87011E-1
Low potential (V)= 9.76581E-2
Scan rate (V/s)= 2.00000E-2
Initial potential (V)= 1.05955E-1
Sensitivity (A/V)= 1.00000E-5
Scan direction = Positive

B

SCALE:
Current (A/cm)= 1.04205E-5
Potential (V/cm)= 4.93678E-2

CATHODIC:
Peak potential (V)= 3.53152E-1
Peak current (A)= -3.64339E-5
Charge (Coul)= -1.22985E-4

ANODIC:
Peak potential (V)= 3.86840E-1
Peak current (A)= 3.48633E-5
Charge (Coul)= 1.06905E-4

THERMODYNAMIC DATA:
RT/nF (V)= 5.86782E-2

E-Epa(V)	E-E0(V)	f	E(V)	log(Fc+/Fc)
-2.44751E-1	-2.41723E-1	7.59430E-5	1.42088E-1	-4.11948
-2.27173E-1	-2.13263E-1	2.31978E-4	1.59667E-1	-3.63445
-2.08863E-1	-1.91593E-1	5.42763E-4	1.77977E-1	-3.26515
-1.91285E-1	-1.75856E-1	1.00602E-3	1.95555E-1	-2.99696
-1.72974E-1	-1.61310E-1	1.77898E-3	2.13866E-1	-2.74906
-1.55152E-1	-1.48421E-1	2.94655E-3	2.31688E-1	-2.52940
-1.37085E-1	-1.36276E-1	4.73712E-3	2.49755E-1	-2.32242
-1.19507E-1	-1.24404E-1	7.52671E-3	2.67333E-1	-2.12011
-1.01196E-1	-1.11596E-1	1.23812E-2	2.85643E-1	-1.90183
-8.31300E-2	-9.81190E-2	2.08306E-2	3.03710E-1	-1.67216
-6.50636E-2	-8.30800E-2	3.69643E-2	3.21776E-1	-1.41586
-4.69972E-2	-6.52831E-2	7.16397E-2	3.39843E-1	-1.11256
-2.91749E-2	-4.37663E-2	1.52202E-1	3.57665E-1	-7.45871E-1
-1.06201E-2	-1.63172E-2	3.45178E-1	3.76220E-1	-2.78080E-1
6.71384E-3	1.26482E-2	6.21596E-1	3.93554E-1	2.15553E-1
2.47803E-2	4.01788E-2	8.28730E-1	4.11620E-1	6.84732E-1

Figure 38. (A) Sample outputs from interaction parameter, and (B) thermodynamic versions of CVOUTPUT.

parameter model requires a value for $E_{p,c/2}$ in this case so the FUNCTION for calculating EPTWO in PEAKFIT had to be changed from an anodic peak to a cathodic peak. For the interaction parameter and thermodynamic versions of CVOUTPUT, changes were made to perform cathodic peak calculations also.

Chronoamperometry Programs

Library Code Changes. The library UNITs previously described were general enough that they could be used directly in chronoamperometry PROGRAMs. Only one UNIT, FILERTNS, was changed to reflect chronoamperometry files rather than CV files. PROCEDUREs CV2FILE and CV2LIST were changed to CA2FILE and CA2LIST, respectively, and FUNCTION FILENAME was changed to return names such as CA?ANLYTE instead of CV?ORIGREAL. The internal workings of the routines remained the same.

PROGRAM CHRONOAMP. This PROGRAM performs the same function as INTERPAR in that it presents the main menu options for chronoamperometry. It is executed by typing <CHRONOAMP> at the Command level. At that time, options are presented to change the ADC input and DAC output channels, display the time and temperature, collect a new set of data, subtract background and smooth the curve, display $it^{1/2}$ data and set time limits, and output the analysis. CHAINing directs execution to the PROGRAM which the option represents.

PROGRAM CADATA and PROCEDURE TAKEDATA. PROGRAM CVDATA and PROCEDURE TAKEDATA from CV experiments were modified to automate chronoamperometry(CA) experiments and are listed in Figures C-1 and C-2, respectively. The input parameters for CA experiments include the title, sensitivity, solute concentration, number of electrons per mole, the step potentials, and the electrode area if the diffusion coefficient is being determined or the diffusion coefficient if the electrode area is being determined. One thousand current points are collected over an experiment time between 25 and 35 seconds which the user also selects. Calculation of DELAY times for the software delay timer were calculated as described in CVDATA. The final potential, the DAC potential channel, and DAC channel for switching from STANDBY to cell are all passed by value to TAKEDATA. CADATA prompts the user to record a background curve first which is stored as CA?BCKGRND, then purge the system with inert gas, add the analyte and record the analyte curve, CA?ANLYTE. The experimental parameters are stored in CA?PARAM.

Chronoamperometry is one of the simplest voltammetric experiments. The applied potential is stepped from a value at which no electrochemical reaction occurs to a value which causes complete oxidative or reductive electrolysis at the electrode surface. The Cottrell[191] equation describes the current versus time relationship,

$$i = nFACD^{1/2}/(\pi t)^{1/2} \qquad (43)$$

where i = current(amps), n = number of electrons per mole, C = concentration(moles/cm^3), D = diffusion coefficient(cm^2/sec), A = electrode

area(cm^2), F = 96,487 coul/eq, and t = seconds. Cottrell's equation predicts $it^{1/2}$ will remain constant under conditions of semiinfinite linear diffusion. This is difficult to achieve in practice. The time requíred by the potentiostat to charge the electrode cause negative deviations in $it^{1/2}$ at short times and convection causes postitive deviations in $it^{1/2}$ at times greater than 40 to 45 seconds.[138]

True semiinfinite linear diffusion requires a shielded electrode as von Stackelberg et al.[141] have shown in their definitive work on diffusion coefficients. Otherwise, diffusion becomes somewhat spherical[192] and the Cottrell equation must be modified,

$$i = nFACD^{1/2}/(\pi t)^{1/2} \; [1 + a(Dt)^{1/2}/r] \qquad (44)$$

where r = electrode radius(cm) and a = 1.92.[142,193,194] Eq. (43) was rearranged and D or A solved for each i-t data pair. Time values were generated artificially since the rate of data acquisition in TAKEDATA was precisely known. Eq. (44) can also be used solve for A or D by linear regression of the $it^{1/2}$-$t^{1/2}$ data. From the slope and intercept of the line, A or D can be computed. A PROGRAM for linear regression analysis is given in Appendix D. Soos-Lingane analysis was not used to compute A or D because of the finite resources of the microcomputer. The $it^{1/2}$-$t^{1/2}$ line has nearly zero slope. Subtraction error and its propagation in the linear regression calclution prevented practical use of slope and intercepts for determination of A or D.[187] Soos-Lingane analysis should therefore be performed in double precision whenever it is used.

PROGRAMs SUBTRACT, DISPLAY, and CAOUTPUT. Before computing A or D for each i-t data pair, the user must subtract the background curve from the analyte curve and smooth with the five-point moving average. SUBTRACT sets VARiable SMOOTH in CA?PARAM to signify the option has been executed.

DISPLAY is executed to establish the time limits of the i-t data we want to include in the calculation of A or D. Short term and long term deviations in $it^{1/2}$ values from surface roughness, charging of the double layer and convection[138] are removed from the calculation with this PROGRAM. There are no graphics presentations in these PROGRAMs, only columns of i, t, $it^{1/2}$, and $t^{1/2}$ data.

Samples of printed output obtained from chronoamperometry are shown if Figures 39 and 40. The average area of diffusion coefficient, standard deviation, and error were computed from the equations,[189]

$$s^2 = \sum_{j=1}^{n} (x_j - \bar{x})^2/(n - 1) \qquad (45)$$

and

$$e = \pm s/n^{1/2} \qquad (46)$$

CAOUTPUT consists mainly of printing and formatting statements as in CVOUTPUT.

i(sub) A	t sec	i(sub)t1/2 A sec1/2	t1/2 sec1/2	Area cm 2
9.94659E-5	3.30000	1.80689E-4	1.81659	3.25485E-1
9.72912E-5	3.99300	1.94412E-4	1.99825	3.50206E-1
8.41600E-5	4.68600	1.82183E-4	2.16472	3.28176E-1
8.39166E-5	5.37900	1.94625E-4	2.31927	3.50590E-1
7.61854E-5	6.07200	1.87732E-4	2.46414	3.38172E-1
7.65613E-5	6.76500	1.99133E-4	2.60096	3.58710E-1
7.10226E-5	7.45800	1.93958E-4	2.73093	3.49388E-1
6.78999E-5	8.15100	1.93854E-4	2.85500	3.49200E-1
6.15748E-5	8.84400	1.83116E-4	2.97389	3.29858E-1
6.14399E-5	9.53700	1.89739E-4	3.08820	3.41788E-1
5.95620E-5	1.02300E1	1.90505E-4	3.19844	3.43168E-1
5.72820E-5	1.09230E1	1.89317E-4	3.30500	3.41027E-1
5.67847E-5	1.16160E1	1.93535E-4	3.40823	3.48626E-1
5.56132E-5	1.23090E1	1.95115E-4	3.50842	3.51471E-1
5.54129E-5	1.30020E1	1.99809E-4	3.60583	3.59928E-1
4.84540E-5	1.36950E1	1.79313E-4	3.70068	3.23006E-1
5.17217E-5	1.43880E1	1.96188E-4	3.79315	3.53405E-1
4.78847E-5	1.50810E1	1.85957E-4	3.88343	3.34975E-1
5.14415E-5	1.57740E1	2.04307E-4	3.97165	3.68031E-1
4.23739E-5	1.64670E1	1.71951E-4	4.05796	3.09746E-1
5.03093E-5	1.71600E1	2.08404E-4	4.14246	3.75411E-1
4.19415E-5	1.78530E1	1.77215E-4	4.22528	3.19227E-1
4.92772E-5	1.85460E1	2.12212E-4	4.30651	3.82271E-1
3.68986E-5	1.92390E1	1.61846E-4	4.38623	2.91542E-1
4.68964E-5	1.99320E1	2.09370E-4	4.46453	3.77151E-1
3.92253E-5	2.06250E1	1.78141E-4	4.54148	3.20895E-1
4.84417E-5	2.13180E1	2.23662E-4	4.61714	4.02896E-1
3.27159E-5	2.20110E1	1.53489E-4	4.69159	2.76489E-1
4.67354E-5	2.27040E1	2.22688E-4	4.76487	4.01141E-1
3.37182E-5	2.33970E1	1.63097E-4	4.83704	2.93796E-1
4.53781E-5	2.40900E1	2.22723E-4	4.90816	4.01204E-1
3.71356E-5	2.47830E1	1.84870E-4	4.97825	3.33018E-1
4.59781E-5	2.54760E1	2.32069E-4	5.04738	4.18039E-1
3.71353E-5	2.61690E1	1.89968E-4	5.11556	3.42201E-1
4.12036E-5	2.68620E1	2.13552E-4	5.18286	3.84684E-1
3.39562E-5	2.75550E1	1.78246E-4	5.24929	3.21085E-1
4.02669E-5	2.82480E1	2.14014E-4	5.31488	3.85516E-1
3.88737E-5	2.89410E1	2.09128E-4	5.37968	3.76714E-1
3.79608E-5	2.96340E1	2.06648E-4	5.44371	3.72246E-1
3.81644E-5	3.03270E1	2.10171E-4	5.50700	3.78594E-1

```
EXPERIMENT DESCRIPTION:
  Exp. number    :1
  Title          :Chronoamp. of 84.5mg of K4Fe(CN)6 3H2O/50ml of 0.1M KCl vs. Ag/AgCl
  Time           :MON MAR 16 10:24:25 AM
  Points         : 40
  Smooth type    :Background subtracted, 2 * five point moving average

EXPERIMENTAL CONDITIONS:
  Temperature          (C)=   2.36120E1
  Init. potential      (V)= -1.70000E-1
  Final potential      (V)=  4.80000E-1
  Duration           (sec)=   3.30000E1
  Number of electrons   =             1
  Sensitivity        (A/V)=  5.00000E-5
  Diff. coeff. (cm2/sec)=  6.50000E-6

DATA ANALYSIS:
  Average area         (cm2)=  3.41685E-1
  Standard deviation   (cm2)=  6.30701E-2
  Error        (+ or - cm2)=  9.97226E-3
```

Figure 39. Sample output for an electrode area determination.

i(sub) A	t sec	i(sub)t1/2 A sec1/2	t1/2 sec1/2	Diff. coeff. cm 2/sec
1.06304E-4	3.30000	1.93111E-4	1.81659	6.38433E-6
9.48402E-5	3.99300	1.89514E-4	1.99825	6.14870E-6
8.97867E-5	4.68600	1.94363E-4	2.16472	6.46734E-6
7.97244E-5	5.37900	1.84902E-4	2.31927	5.85306E-6
7.92476E-5	6.07200	1.95278E-4	2.46414	6.52835E-6
7.54169E-5	6.76500	1.96156E-4	2.60096	6.58725E-6
7.04591E-5	7.45800	1.92419E-4	2.73093	6.33864E-6
6.76049E-5	8.15100	1.93012E-4	2.85500	6.37774E-6
6.66698E-5	8.84400	1.98269E-4	2.97389	6.72987E-6
6.19857E-5	9.53700	1.91424E-4	3.08820	6.27326E-6
6.12220E-5	1.02300E1	1.95815E-4	3.19844	6.56432E-6
5.63186E-5	1.09230E1	1.86133E-4	3.30500	5.93123E-6
5.77042E-5	1.16160E1	1.96669E-4	3.40823	6.62173E-6
5.67960E-5	1.23090E1	1.99264E-4	3.50842	6.79762E-6
5.34617E-5	1.30020E1	1.92774E-4	3.60583	6.36203E-6
5.33468E-5	1.36950E1	1.97419E-4	3.70068	6.67233E-6
5.38550E-5	1.43880E1	2.04280E-4	3.79315	7.14416E-6
4.98684E-5	1.50810E1	1.93660E-4	3.88343	6.42066E-6
5.05110E-5	1.57740E1	2.00612E-4	3.97165	6.88990E-6
4.65575E-5	1.64670E1	1.88928E-4	4.05796	6.11073E-6
4.87042E-5	1.71600E1	2.01755E-4	4.14246	6.96867E-6
4.71822E-5	1.78530E1	1.99358E-4	4.22528	6.80405E-6
4.50721E-5	1.85460E1	1.94103E-4	4.30651	6.45009E-6
4.53797E-5	1.92390E1	1.99046E-4	4.38623	6.78274E-6
4.69262E-5	1.99320E1	2.09503E-4	4.46453	7.51418E-6
4.21143E-5	2.06250E1	1.91261E-4	4.54148	6.26256E-6
4.50270E-5	2.13180E1	2.07896E-4	4.61714	7.39932E-6
4.09018E-5	2.20110E1	1.91894E-4	4.69159	6.30411E-6
4.21362E-5	2.27040E1	2.00774E-4	4.76487	6.90101E-6
4.17480E-5	2.33970E1	2.01937E-4	4.83704	6.98119E-6
4.00956E-5	2.40900E1	1.96796E-4	4.90816	6.63026E-6
4.15208E-5	2.47830E1	2.06701E-4	4.97825	7.31450E-6
4.17703E-5	2.54760E1	2.10830E-4	5.04738	7.60965E-6
3.80075E-5	2.61690E1	1.94430E-4	5.11556	6.47180E-6
4.03003E-5	2.68620E1	2.08871E-4	5.18286	7.46886E-6
3.77303E-5	2.75550E1	1.98057E-4	5.24929	6.71554E-6
3.85295E-5	2.82480E1	2.04780E-4	5.31488	7.17918E-6
3.85231E-5	2.89410E1	2.07242E-4	5.37968	7.35284E-6
3.58101E-5	2.96340E1	1.94940E-4	5.44371	6.50579E-6
3.79098E-5	3.03270E1	2.08769E-4	5.50700	7.46160E-6

EXPERIMENT DESCRIPTION:
Exp. number :4
Title :Chronoamp. of 65.8mg of K3Fe(CN)6/50.0ml of 0.1M KCl vs. Ag/AgCl
Time :TUE MAR 24 1:43:53 PM
Points : 40
Smooth type :Background subtracted, 2 * five point moving average

EXPERIMENTAL CONDITIONS:
Temperature (C)= 2.64501E1
Init. potential (V)= 4.50000E-1
Final potential (V)= -2.00000E-1
Duration (sec)= 3.30000E1
Number of electrons = 1
Sensitivity (A/V)= 5.00000E-5
Area (cm2)= 3.51000E-1

DATA ANALYSIS:
Average diff. coeff. (cm2/sec)= 6.54344E-6
Standard deviation (cm2/sec)= 1.13817E-6
Error (+ or - cm2/sec)= 1.79960E-7

Figure 40. Sample output for a diffusion coefficient determination.

CHAPTER V

RESULTS AND DISCUSSION

A. Chronoamperometry

A summary of the three replicate determinations of the Pt-button, electrode area is shown in Table VII. Areas were computed for 40 $it^{1/2}$ points between 3 and 30 seconds at intervals of approximately 700 milliseconds.[138] BAS-100 voltammograms of these systems(Figure 41) provided sensitivity and step potential(vs. Ag/AgCl) information for final experiments on the CV-27. For the oxidation of 4.0mM $K_4Fe(CN)_6$ in 0.1M KCl/H_2O, a diffusion coefficient of 6.50×10^{-6} cm^2/sec was used.[141] The three area determinations were averaged to give $A = 0.351 \pm 0.005 cm^2$ and the error calculated from eqs. (45) and (46).[189] The measured geometrical area was $0.349 cm^2$. The value, $A = 0.351 \pm 0.005$ cm^2, was used to calculate Γ_T from the relation, $Q = nFA\Gamma_T$.[35,37] To check the diffusion coefficient option in the chronoamperometry software, determination of the diffusion coefficient for the reduction of $K_3Fe(CN)_6$ was performed.[141] The electrode area(0.351 cm^2) previously determined was used in these experiments. An average value, $D = (6.38 \pm 0.05) \times 10^{-6}$ cm^2/sec, was obtained. This value is in error $\pm 19\%$ from the expected value(Table VII).

B. Scan Rate and Film Thickness Effects on PME Voltammetry

Before final experiments were performed in the various supporting electrolyte/solvent systems, control studies of film thickness(Γ_T) and

Table VII. Chronoamperometry results for electrode area determination and check of a known diffusion coefficient in 0.1M KCl/H_2O at T = 25°C.

DETERMINATION	CONC $K_4Fe(CN)_6$	AREA(cm^2)
1	4.0mM	0.342 ± 0.010
2	4.0	0.368 ± 0.010
3	4.0	0.342 ± 0.010

AVERAGE AREA = 0.351 ± 0.005 cm^2

DETERMINATION	CONC $K_3Fe(CN)_6$	DIFF. COEFF.(cm^2/sec)
4	4.0mM	(6.54 ± 0.18) * 10^{-6}
5	4.0	(6.32 ± 0.18) * 10^{-6}
6	4.0	(6.28 ± 0.18) * 10^{-6}

AVERAGE DIFFUSION COEFFICIENT = (6.38 ± 0.08) * 10^{-6} cm^2/sec
EXPECTED DIFFUSION COEFFICIENT = 7.63 * 10^{-6} cm^2/sec

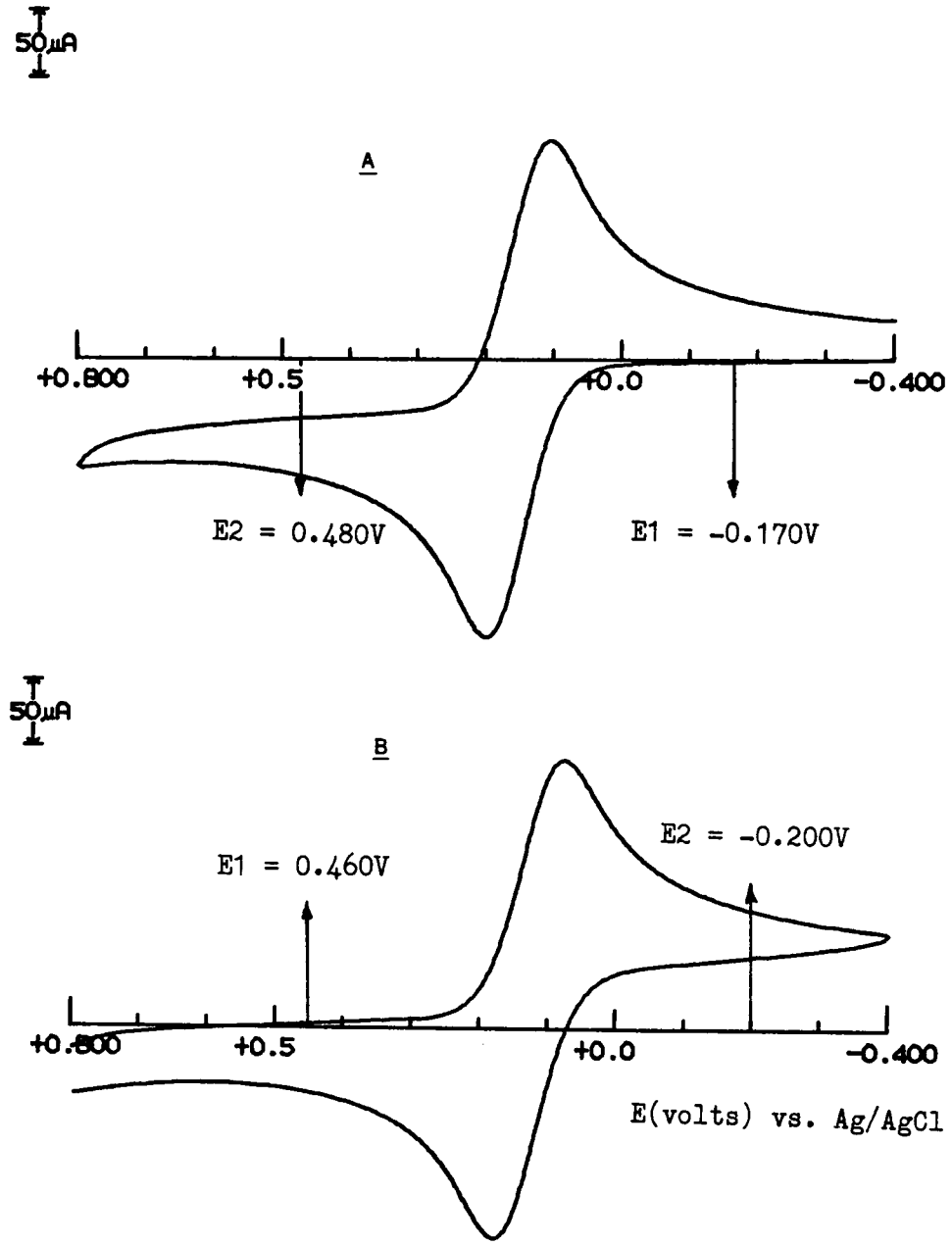


Figure 41. CVs of (A) $K_4Fe(CN)_6$ oxidation and (B) $K_3Fe(CN)_6$ reduction used to determine step potentials vs. Ag/AgCl in 0.1M KCl.

scan rate were performed. Scan rate results for a PVF thin film in 0.1M $TBABF_4/CH_3CN$ are shown in Figure 42. For these Cvs the ferrocene/ferrocenium ion in the polymer is the electroactive couple that gives rise to the peak current. As the scan rate was increased above 25-30mV/s, voltammetric waves exhibited diffusional tailing and $i_p \alpha v^{1/2}$ rather than v.[82,83,103] Bard et al.[56] have also observed this effect on PVF thin films. Thus application of the models requires scan rates less than ca. 25mV/s where thin film behavior is observed.[37,56] It seems ΔE_p is never zero in these films as Evans and coworkers have shown, even at scan rates less than 1mV/s.[40] Sweeping very slowly is impractical not only in the interest of time but also because charging current increases as peak currents attenuate.[35] Another significant feature of the one-cycle CVs in Figure 42 is that the data acquisition routine terminates prematurely at higher scan rates. This is a result of subtraction and round-off error in the software delay calculation. While, in principle, delays of 23 microseconds to 82 seconds can be generated, there are not enough significant figures in the computation to avoid these errors and their propagation. These errors plagued voltammetric peak analysis and Soos-Lingane analysis of chronoamperometric data.[162,185-189]

Film thickness is another experimental factor which influences charge transport in thin, polymer films.[65,73,83,103] Electroprecipitation times were used to control film thickness in a given solvent system.[69,139,140] Reproducibility of film thickness by dip and evaporation coating is more difficult, but can be controlled for the most part by varying the concentration of the casting

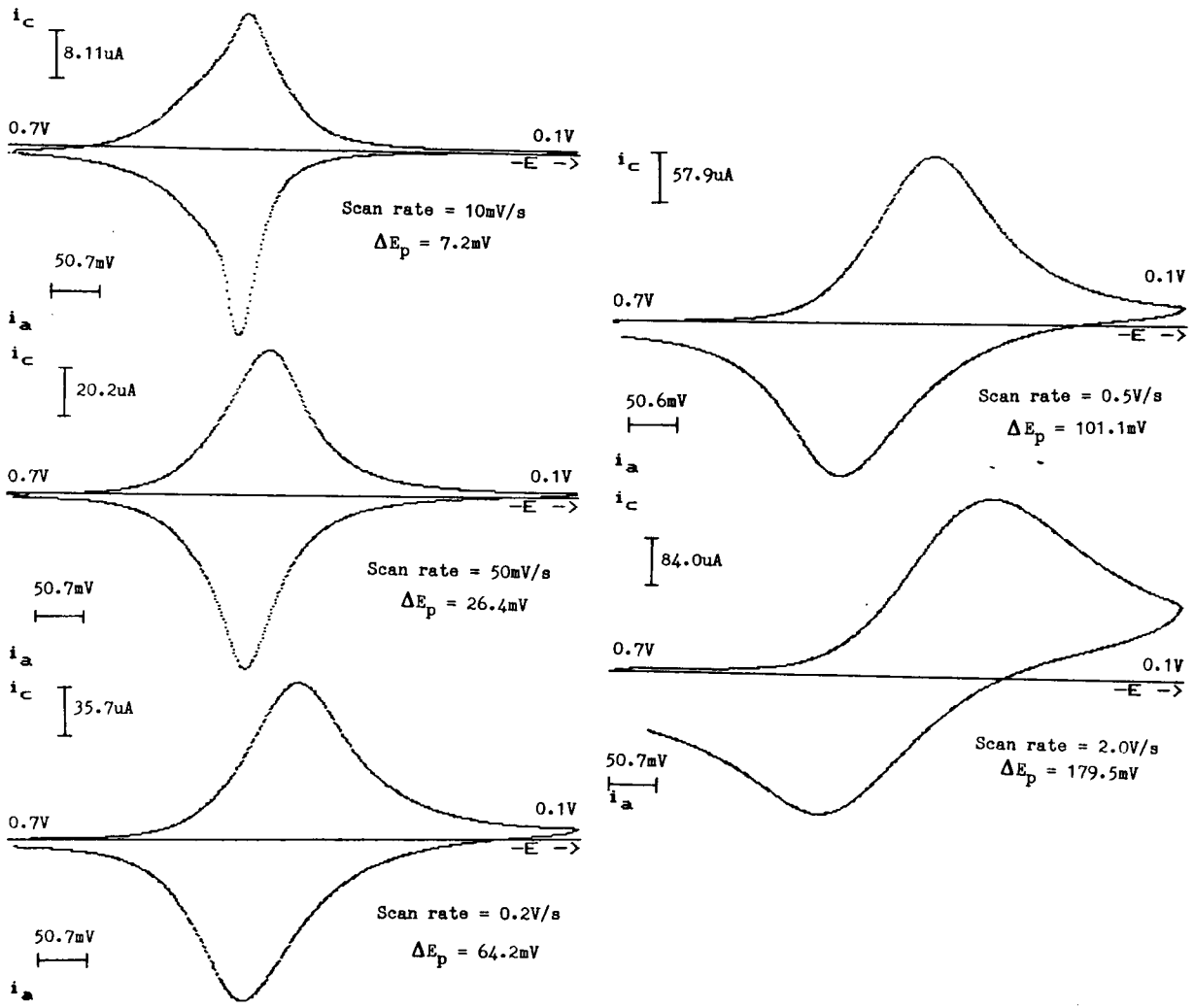


Figure 42. Scan rate study of an electroprecipitated(15s) PVF thin film in 0.1M $TBABF_4/CH_3CN$ vs. Ag/AgCl.

solution.[73] Film thickness results coupled with the scan rate study were used to optimize experimental conditions. For example, thicker films require slower scan rates than thinner films for the best thin film behavior. In thicker films, one must allow more time for propagation of charge by site-site hopping and sweep slowly to remain in the thin film domain.[56,81]

Murray[35] has suggested that at $\Gamma_T > 10^{-8}$mol/cm^2, charge transport complications prevent useful comparison to the theory of thin film electrochemistry. The advantage of preparing a thicker film, however, is realized in electrocatalysis applications where larger peak currents are desirable.[35,41,42,76] In this study, an optimum Γ_T near 10^{-9}mol/cm^2 was found. Thinner films($\Gamma_T = 10^{-10}$mol/cm^2) experienced film dissolution after only a couple of cycles in solvents which "overswelled" the film.[37] In the "overswollen" condition, swelling overcomes the chemisorptive, covalent-like bond between the polymer film and electrode.[35] Diffusional tailing appears in the voltammetry because the redox sites are being diluted by the solvent to a point where electron self-exchange is impaired.[84-86] Eventually, complete delamination of the film occurs. Dilution of redox sites by the solvent is equivalent to diluting the redox sites in the polymer backbone as Karimi and Chambers have shown.[73]

In the interaction parameter model, plots of $\Gamma_T r$ versus T^{-1} are constructed(eq. (2), Figure 6).[36,37] Each $\Gamma_T r$ value in the plots of Figure 43 was computed from a thin film voltammogram at that temperature. Film thicknesses were computed from integrated anodic peaks between 20 and 25°C.[37] Open circles represent virgin film scans and

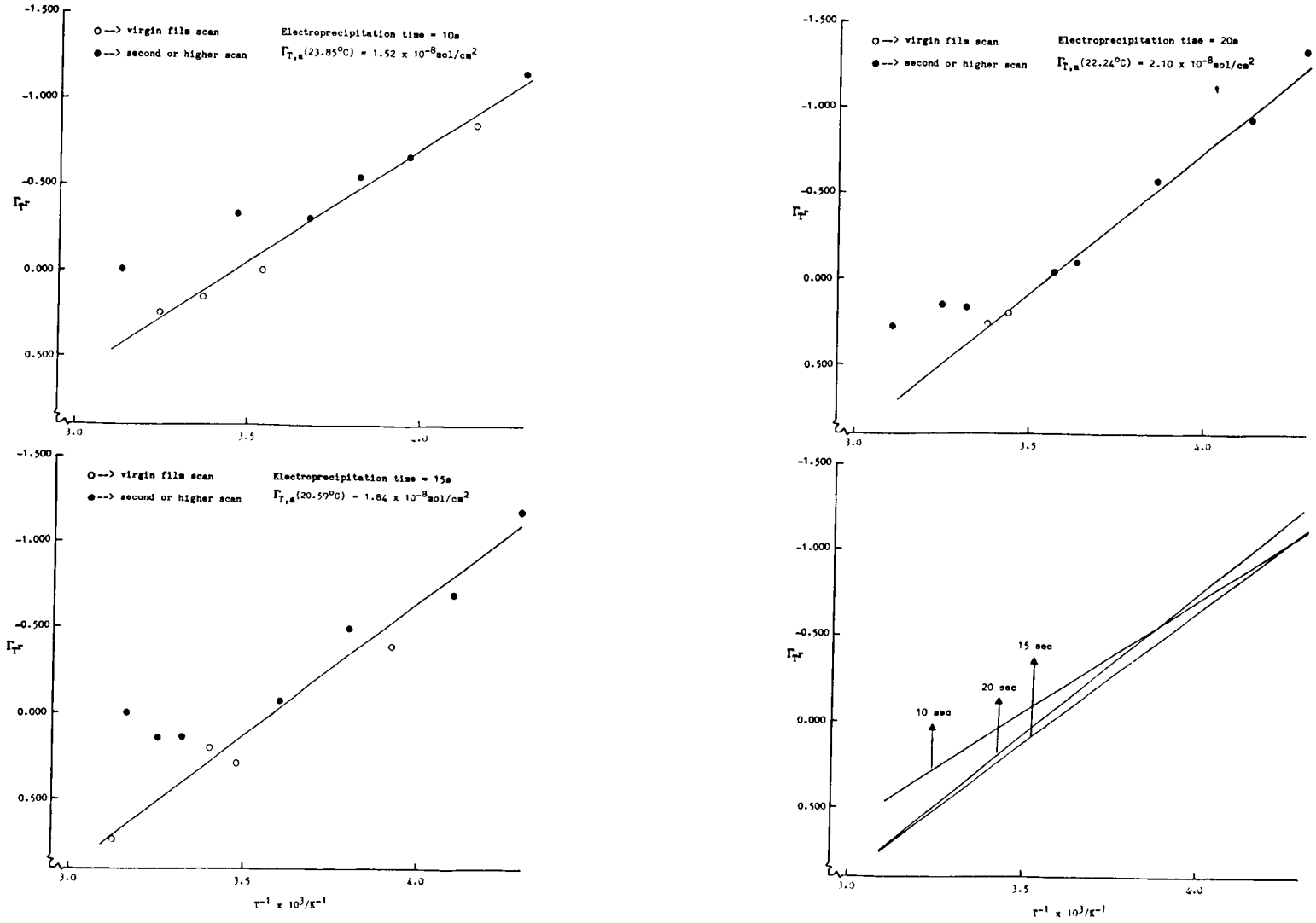


Figure 43. PVF thin film thickness effect on $\Gamma_T r$ results.

solid circles represent second or higher scans. As the electroprecipitation time was increased from 10 to 20s, Γ_T increased. In the lower right corner of Figure 43, the three plots are combined. The slopes are somewhat different, and this is attributed to charge transport complications in thick films($\Gamma_T = 10^{-8}$mol/cm^2).[65,83,103] Reduction of the interaction parameter plots in Figure 43 to straight lines was accomplished by linear regression of virgin film, Γ_Tr values for temperatures greater than 12.7°C and all of the data points below 12.7°C. Film thickness results indicated films with a $\Gamma_T < 10^{-8}$mol/cm^2 should be prepared in subsequent studies.

The mechanical/electrochemical(ME) model[38] provides an explanation for discriminating second or higher scan, Γ_Tr values in the linear regression analysis of interaction parameter plots. Temperature effects on the mechanical properties of polymers are well known.[130,131,195-197] The ME model discusses mechanical effects in PMEs in terms of stress-strain relationships. The result of this treatment is a Nernst equation, which is modified by a bulk modulus term, K(dyn/cm^2). K describes the polymer's elasticity in a particular supporting electrolyte/solvent system. Evans and coworkers[38] compared PME results to their equations at 25°C only. But modulus of elasticity versus temperature curves reveal that a polymer can undergo orders of magnitude change in elasticity over temperature ranges as narrow as 130°C.[195] Plasma-polymerized PVF(PPVF) would not experience as great a change in elasticity as linear PVF because of crosslinking. Lower molecular weight polymers such as linear PVF and TCNQ-TPE, however, were expected to significantly change their elasticity from glassy to

rubbery to flow behavior over the temperature range studied. An intimate relationship also exists between time and temperature. A PME subjected to a stress from counterion uptake, solvent swelling, and change in temperature relaxes that stress as time proceeds. Depending on the elapsed time between imposition of the stress and time of measurement, markedly different elasticities can be observed.[195-197] Evans and coworkers[38] used the classic spring or Hooke's law approach to account for stress-strain(i.e. mechanical) effects in their model of PME behavior. This told us that mechanical stress could be dissipated either elastically or plastically on the voltammetric time scale as one continues cycling on the same film. At higher temperatures, rubbery or flow behavior overstressed the films and caused delamination. Severe stress resulted in plastic rather than elastic deformation of the PMEs and explained why virgin film, $\Gamma_T r$ results fit straight lines at higher temperatures.[38] In virgin PVF thin films, insufficient time had elapsed to result in plastic deformation and the voltammetric responses indicated excellent thin film behavior.

Related to this temperature-time effect on mechanical stress in PMEs is the "break-in" wave of certain electroactive polymers.[38] This "break-in" wave has been observed for TCNQ-PMEs in aqueous solvent in our laboratory.[99,102] This is viewed as a solvent-time effect. Aqueous solvents swell TCNQ-polyester films very poorly and it requires a cycle or two to reach the elastic state. Differences in the "break-in" and subsequent voltammetric waves are attributed to slow, plastic deformation of the film by counterion uptake and solvent swelling.[38,195-197]

C. Implications of the Interaction Parameter Model

After film thickness and general temperature-time effects on the mechanical/electrochemical behavior of PVF thin films were elucidated, supporting electrolyte/solvent studies on PVF thin films were performed. Plastic deformation of the films on second or higher scans at $T^{-1} < 3.5 \times 10^{-3} K^{-1}$ was observed in all of these results also. Electroprecipitation times were varied in linear PVF films to obtain surface coverages between 1 and $9 \times 10^{-9} mol/cm^2$. In Figures 44 through 47, interaction parameter versus temperature results for the PVF and TCNQ-TPE thin films are shown. At lower temperatures in all solvents, $\Gamma_T r$ values from voltammetric surface waves were the most negative. This indicated repulsive interactions among neighboring redox sites and charge transport problems.[37,82,83,103,122] Peak separations($\Delta E_p s$) increased and peak currents attenuated, but practically symmetrical waveshapes were observed. PMEs behave like glasses at low temperatures; thus, segmental motions of the polymer chains are short ranged and solvent swelling of the film is hindered.[195-197] The films are less elastic, and distances between Fc^+A^- sites are shorter due to film compaction.[73,84-86,195-197] Repulsive interactions between similarly charged sites and thermal deactivation of the charge transport process dominated the voltammetric behavior at low temperatures.[37,73] Thus, PMEs are poorly swollen at low temperatures where the rate of electron self-exchange is slow and diffusion complicates thin film voltammetry, but plastic deformation of the films by excessive mechanical stress is not a problem.

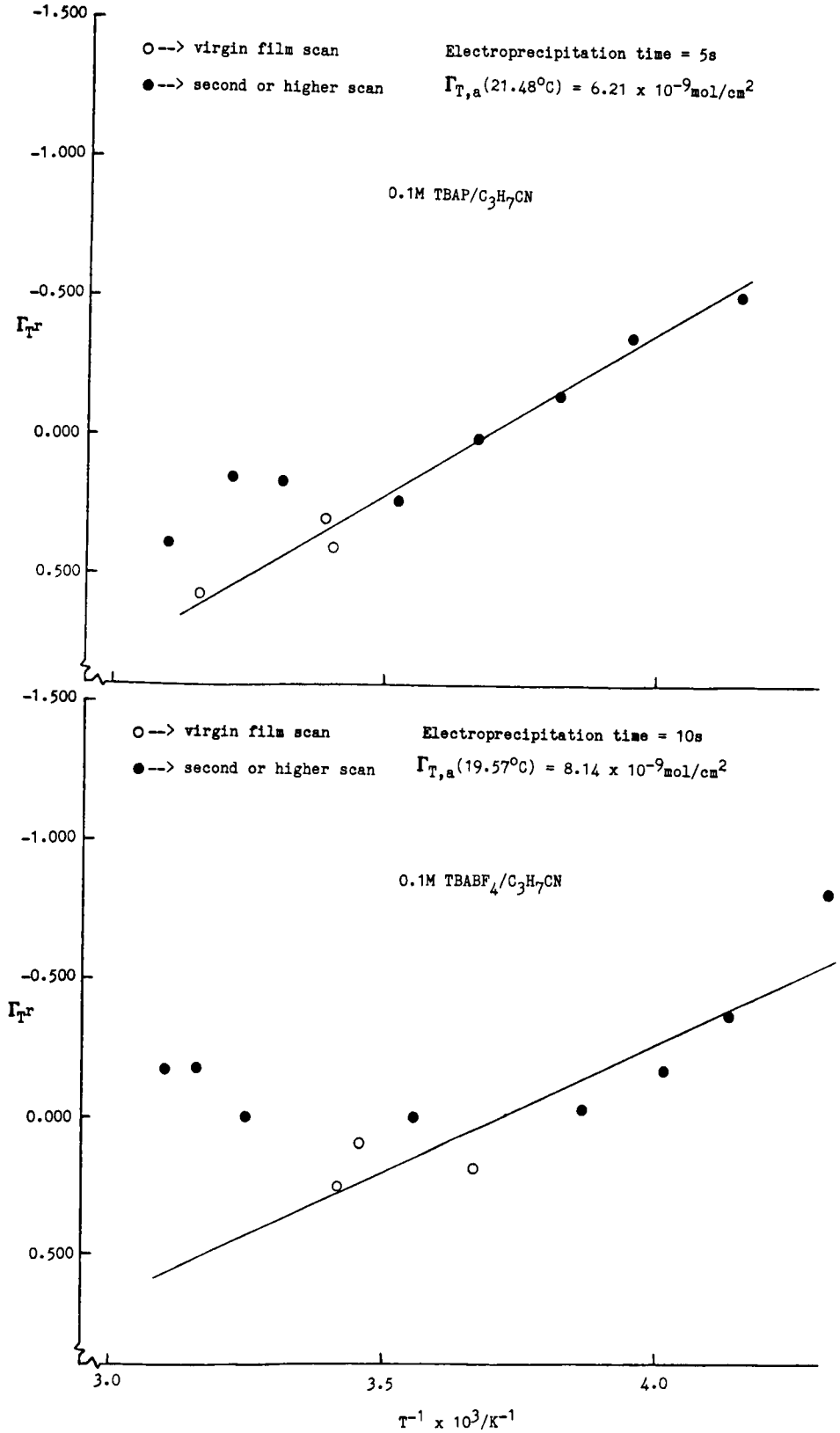


Figure 44. PVF interaction parameter results in C_3H_7CN.

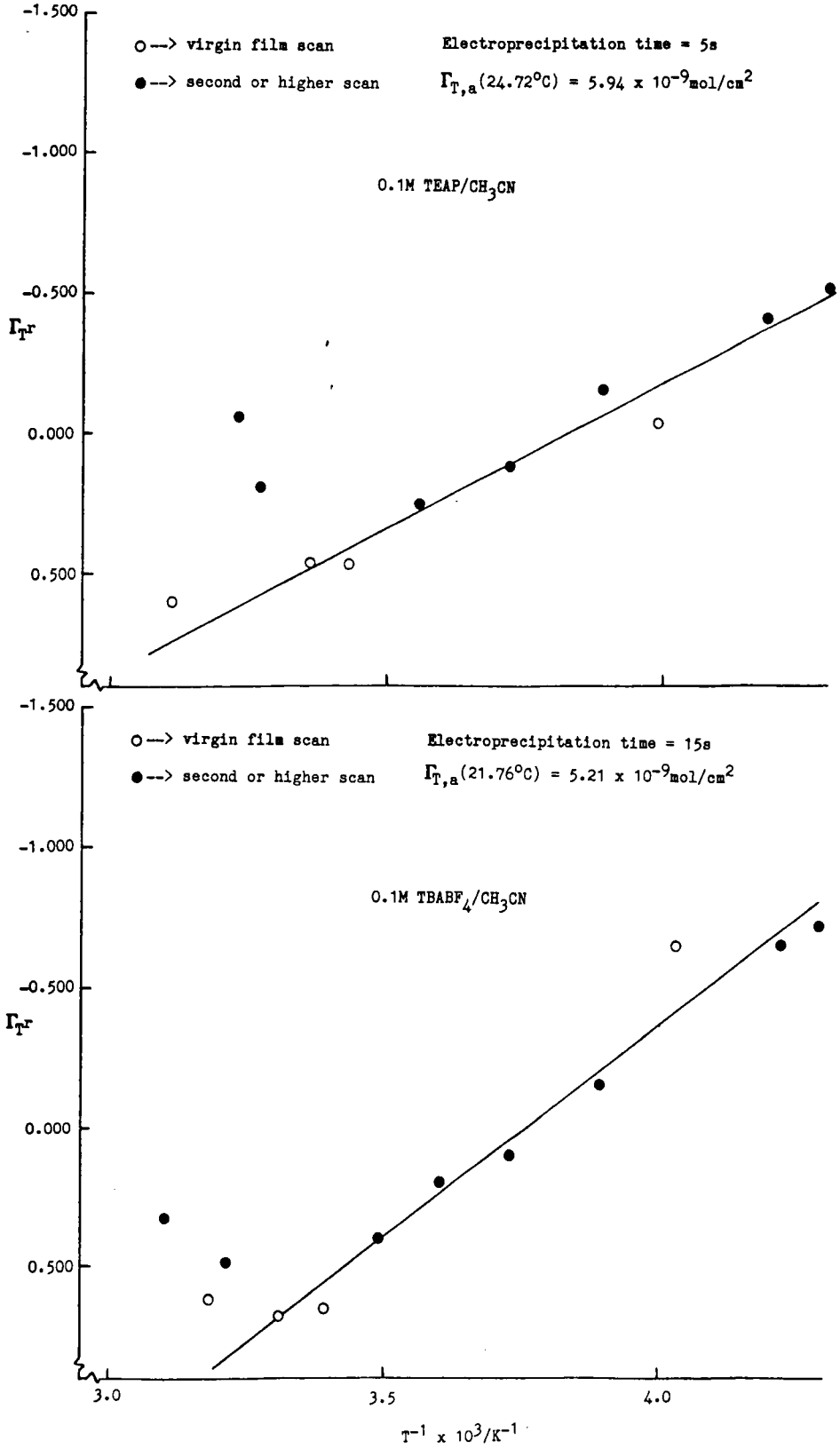


Figure 45. PVF interaction parameter results in CH_3CN.

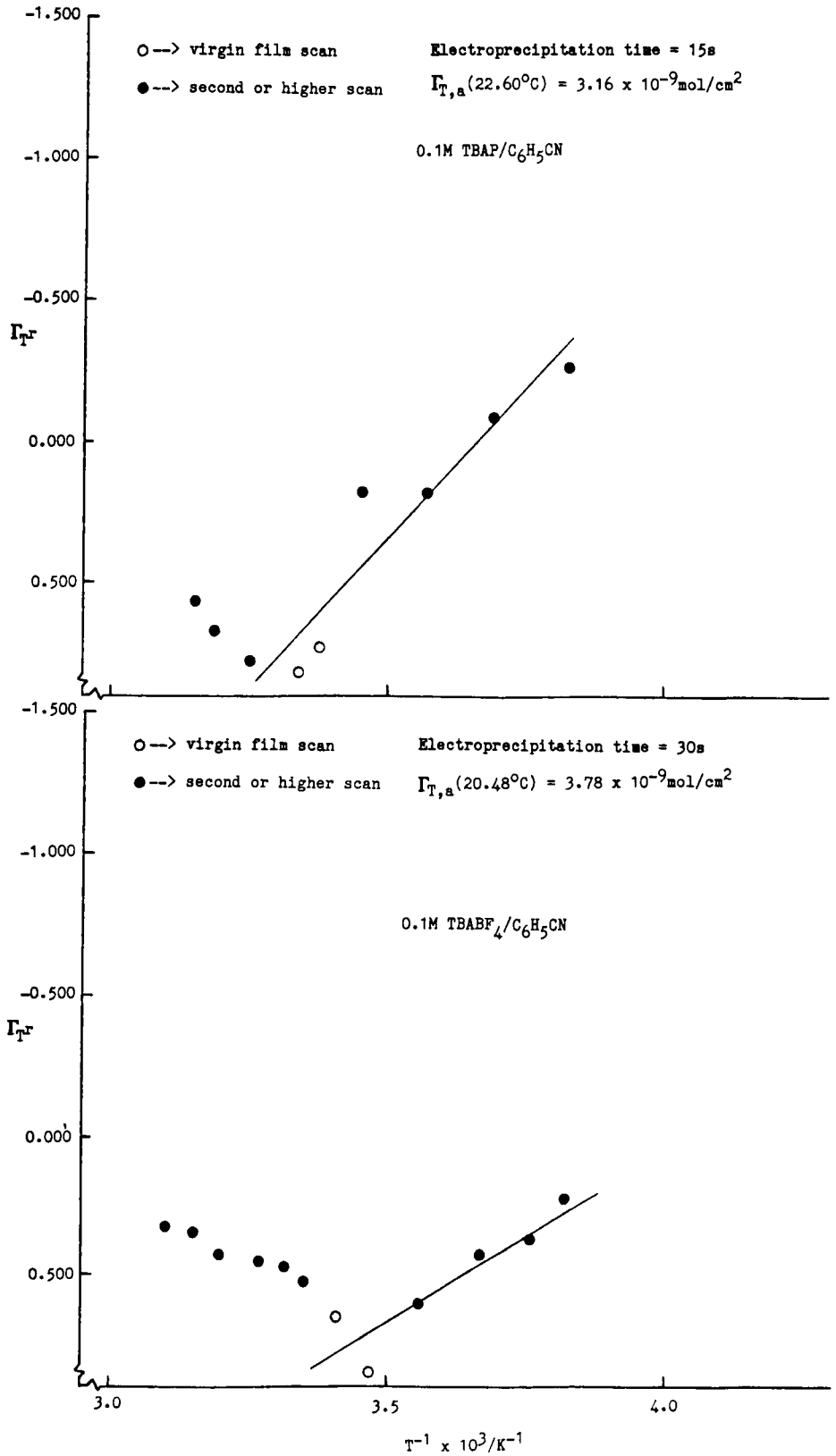


Figure 46. PVF interaction parameter results in C_6H_5CN.

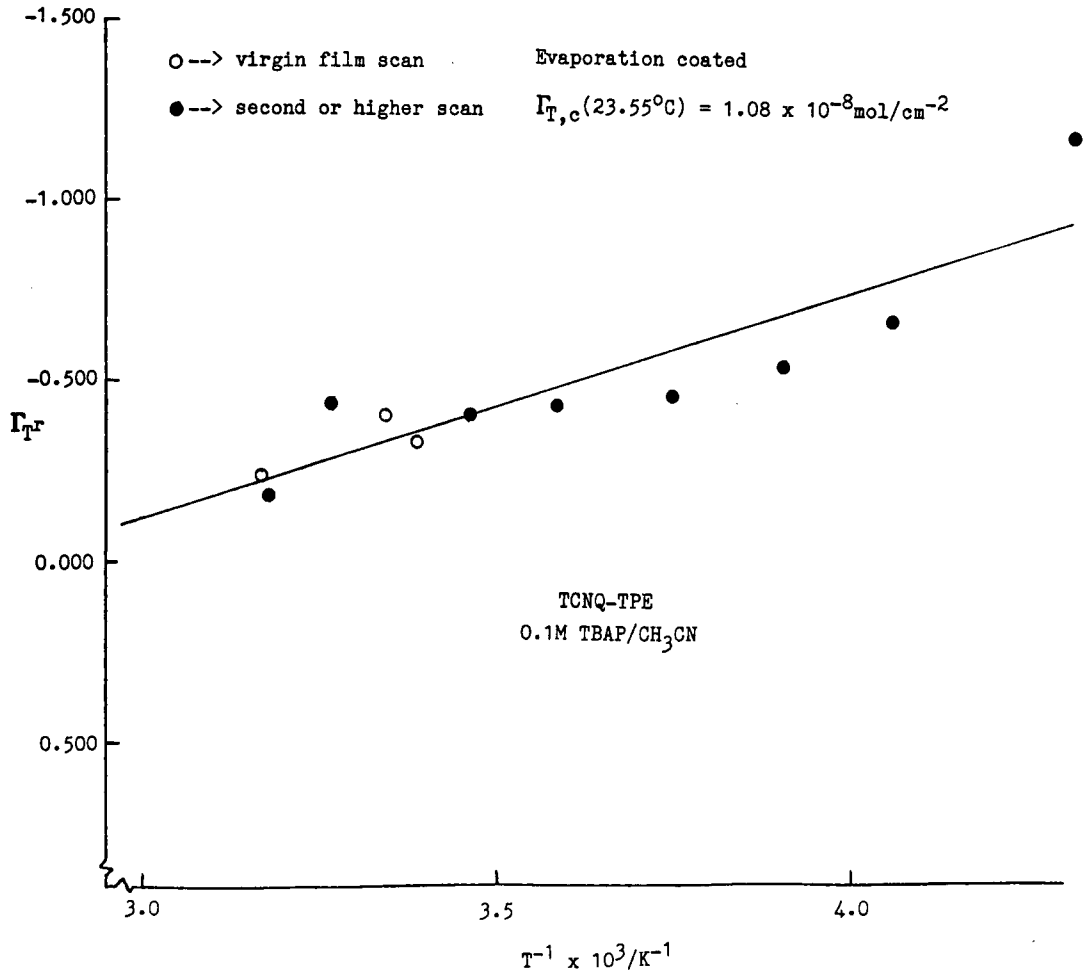


Figure 47. TCNQ-TPE interaction parameter results in 0.1M TBAP/CH_3CN.

Figures 48 through 51 are representative sequences of thin film voltammograms from which $\Gamma_T r$ values were extracted for interaction parameter plots. Attenuation and broadening of PVF anodic peaks and the TCNQ-TPE cathodic peak is evident at low temperatures. Peak separations increased as the temperature was lowered due to slow electron self-exchange.[81] This is summarized in Table VIII for all of the systems studied. Virgin films at high temperatures had the smallest ΔE_ps. Thicker films impeded charge transport even more at low temperatures. In addition to film thickness and temperature, it appeared that the supporting electrolyte and solvents affected waveshapes, peak separations, and slopes of interaction parameter plots.[82,98] From the linear regression and slope information in Table VIII, a combined $\Gamma_T r$ versus T^{-1} plot was constructed(Figure 52). Correlation coefficients for all of the regression analyses were close to -1.000 except for one benzonitrile system and the TCNQ-TPE results.

There are several significant features in the combined, interaction parameter plot. The TCNQ-TPE line reflects poor swelling($r << 0$), even at high temperatures. Plastic deformation of the TCNQ-TPE films after second or higher scans was not a problem at higher temperatures because poor swelling prevented even short range motions of polymer chains.[38,195-197] The lack of film elasticity in 0.1M TBAP/CH_3CN prevents film delamination(Figure 47), but magnifies iR drop and charge transport complications at low temperatures. The TCNQ-TPE results exhibited diffusional tailing at higher temperatures than the more elastic, PVF films.[37,73] One can therefore question the inclusion of all the data in Figure 47 in the regression analysis. Rejection of a

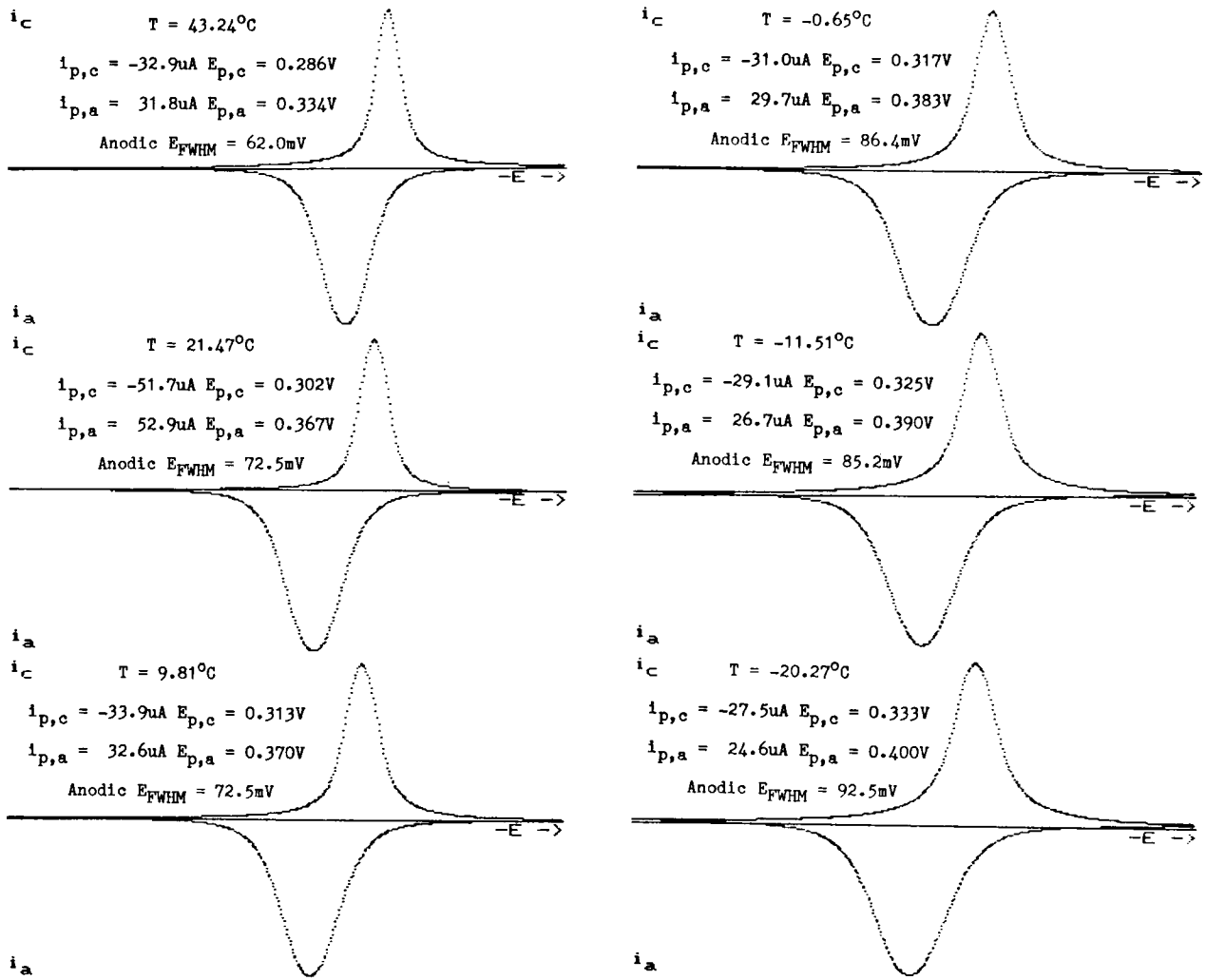


Figure 48. Representative PVF, CVs in 0.1M TBAP/C_3H_7CN vs. Ag/AgCl. Negative scans at 20mV/s.

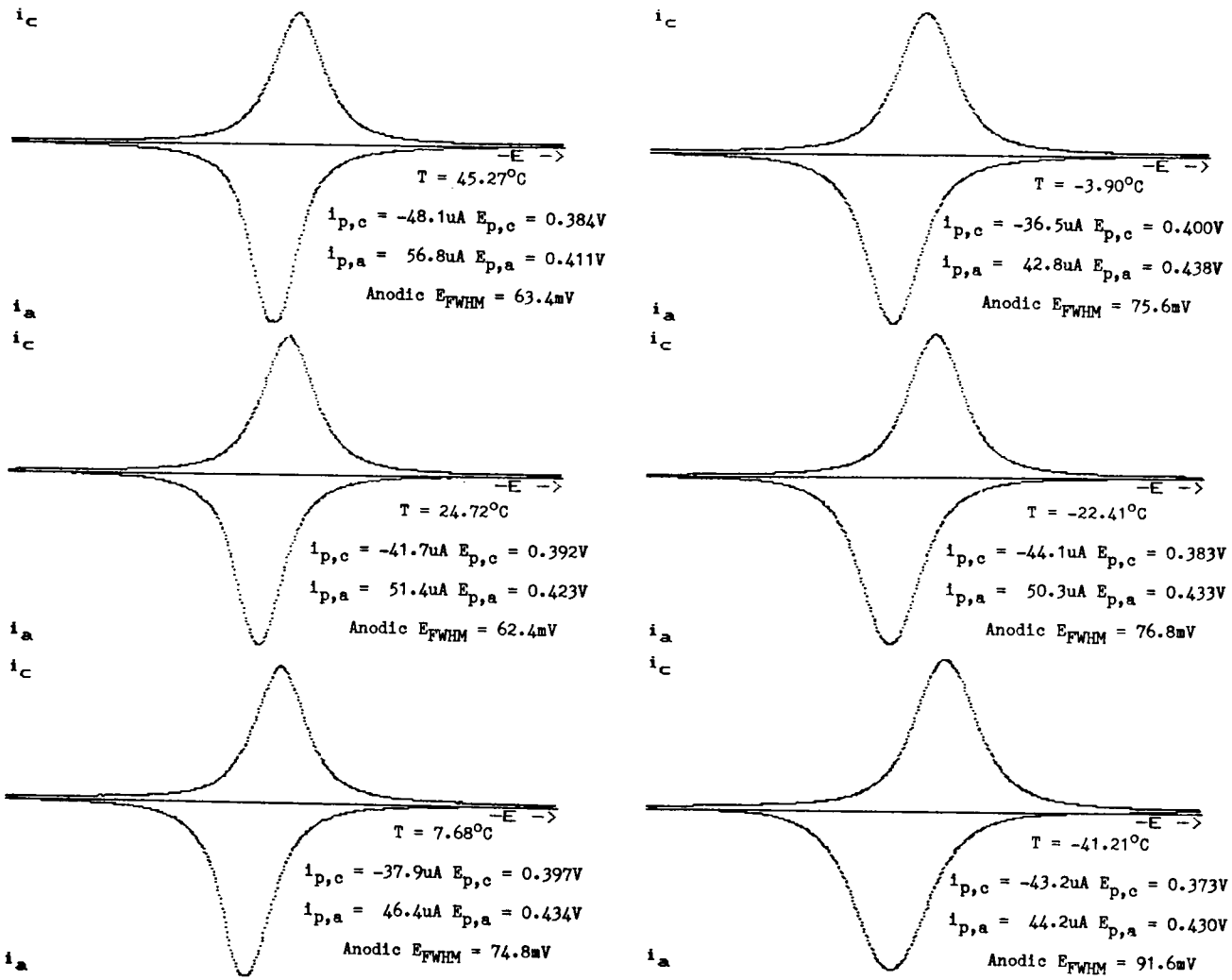


Figure 49. Representative PVF, CVs in 0.1M TEAP/CH_3CN vs. Ag/AgCl. Negative scans at 20mV/s.

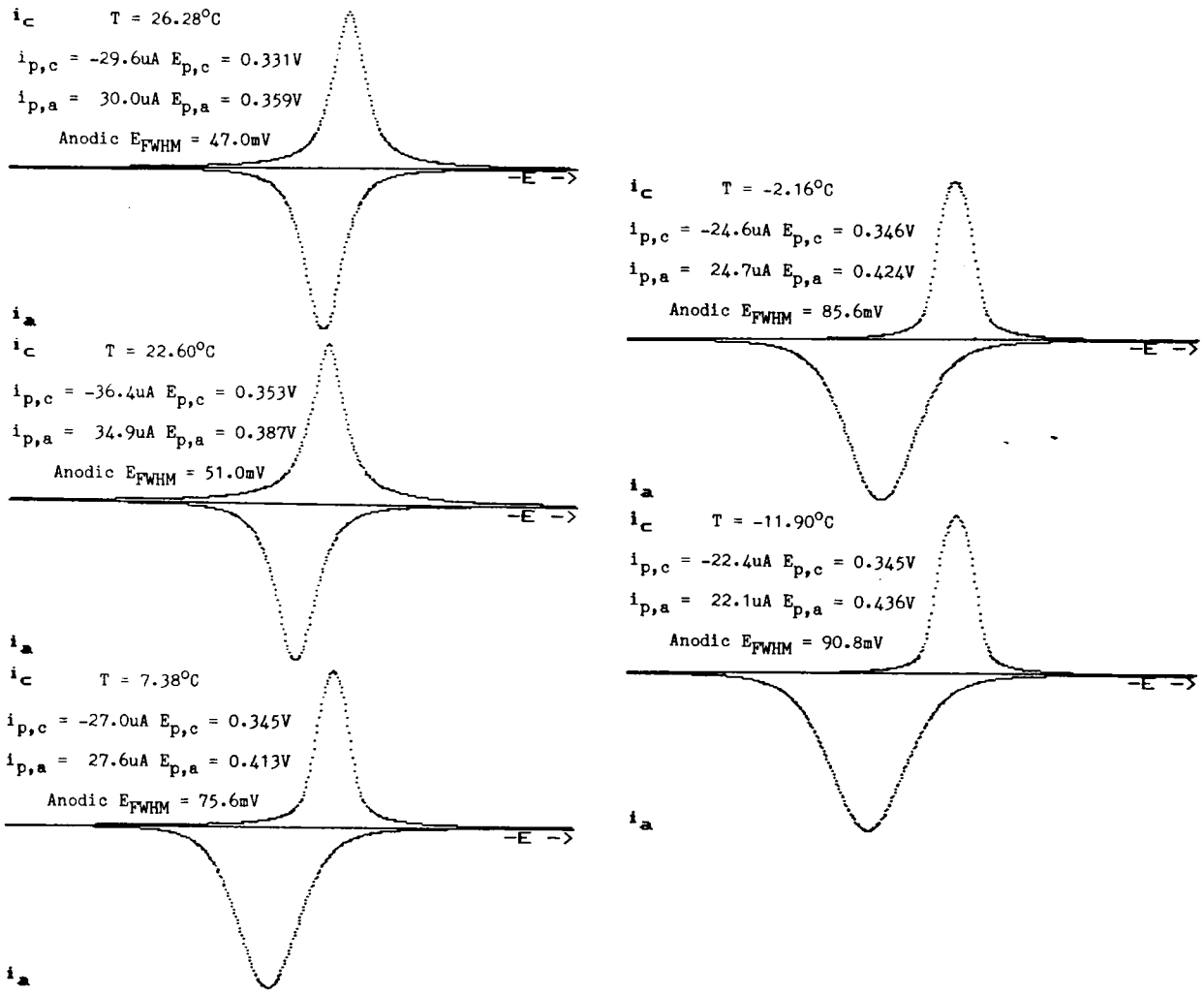


Figure 50. Representative PVF, CVs in 0.1M TBAP/C_6H_5CN vs. Ag/AgCl. Positive scans at 20mV/s.

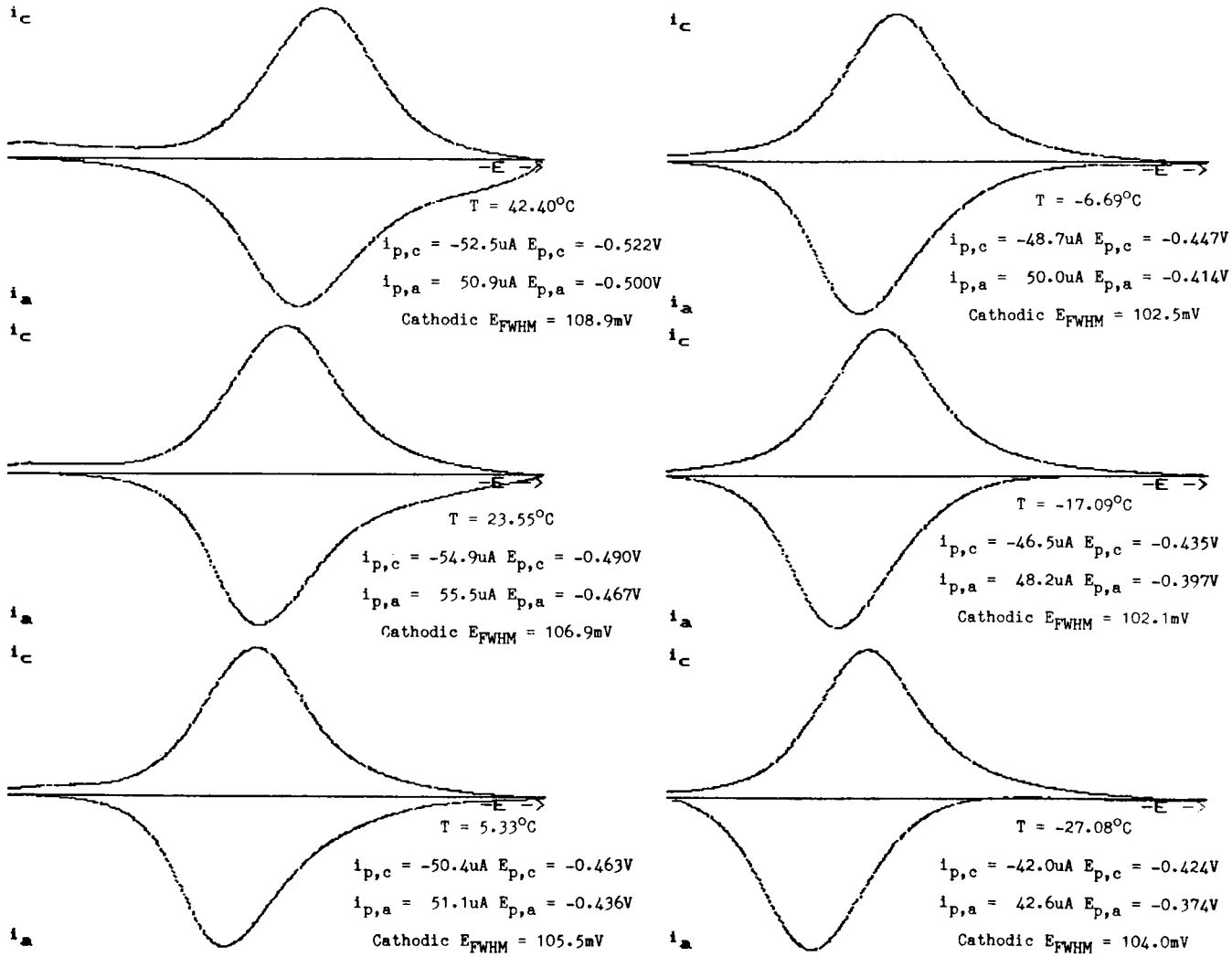


Figure 51. Representative TCNQ-TPE, CVs in 0.1M TBAP/CH_3CN vs. Ag/AgCl. Negative scans at 20mV/s.

Table VIII. Linear regression analysis of interaction parameter data and peak separation comparison.

CONC./SUPPORTING ELECTROLYTE/SOLVENT	LINEAR REGR. SLOPE	LINEAR REGR. INTERCEPT	CORRELATION COEFFICIENT	$\Delta E_{p,low}$(mV)	$\Delta E_{p,high}$(mV)
		(PVF FILM LOADING EXPERIMENTS)			
0.1M/TBAP/CH_3CN(10s)	-1.312	4.538	-0.990	47.5(35.40°C)	90.3(-40.10°C)
0.1M/TBAP/CH_3CN(15s)	-1.570	5.630	-0.985	32.0(46.85°C)	94.3(-40.00°C)
0.1M/TBAP/CH_3CN(20s)	-1.689	5.993	-0.991	55.5(22.27°C)	103.3(-39.89°C)
		(PVF SUPPORTING ELECTROLYTE/SOLVENT EXPERIMENTS)			
0.1M/TEAP/CH_3CN	-1.027	3.926	-0.984	26.7(45.32°C)	57.2(-41.19°C)
0.1M/$TBABF_4$/CH_3CN	-1.518	5.713	-0.985	17.9(29.51°C)	68.2(-39.94°C)
0.1M/TBAP/C_6H_5CN	-2.123	7.775	-0.932	28.4(26.25°C)	90.9(-11.92°C)
0.1M/$TBABF_4$/C_6H_5CN	-1.226	4.960	-0.900	46.6(20.45°C)	59.7(-11.58°C)
0.1M/TBAP/C_3H_7CN	-1.177	4.332	-0.987	48.2(43.21°C)	71.7(-30.31°C)
0.1M/$TBABF_4$/C_3H_7CN	-0.974	3.644	-0.957	64.3(24.65°C)	74.0(-31.49°C)
		(TCNQ-TPE EXPERIMENTS)			
0.1M/TBAP/CH_3CN	-0.609	1.705	-0.887	11.0(32.85°C)	73.0(-41.45°C)

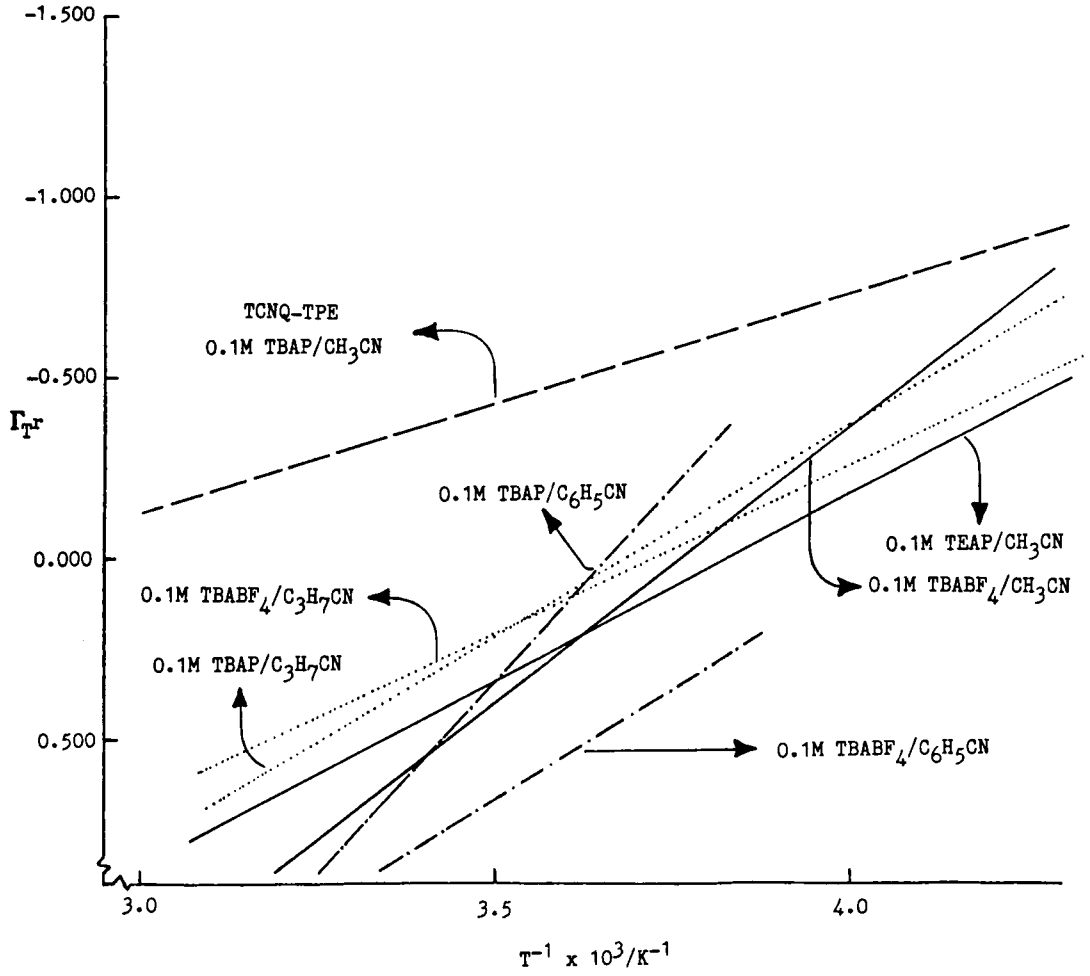


Figure 52. Combined interaction parameter plots for all systems studied.

point or two at low temperatures would probably give a better correlation coefficient.

Also, TCNQ-TPE films were evaporation-coated instead of electroprecipitated. Counterion uptake into an amorphous polymer structure[35] as well as solvent penetration can introduce random errors in $\Gamma_T r$ calculations considering the microampere sensitivities used. This was in fact observed for $\Gamma_T r$ results in 0.1M TEAP/CH_3CN for dip-coated PVF films. To reduce error in these highly sensitive films and control film thickness, PVF was deposited in its ionically conducting state. Similar, unsuccessful attempts were made to electroprecipitate TCNQ-TPE films in their ionically conducting state from CH_2Cl_2 solutions of the polymer.

Scrutiny of the $\Gamma_T r$ results for the PVF-PMEs(Figure 52) in different solvents is consistent with the previous discussions about the mechanical elasticity of polymers. Swelling of the films increased in the order, C_3H_7CN > CH_3CN > C_6H_5CN. The interaction parameter model predicts no deviation from ideal, thin film behavior when $\Gamma_T r$ = 0. Five of the six PVF solvent systems crossed the $\Gamma_T r$ = 0, ideal line in Figure 52. In benzonitrile, $\Gamma_T r$ values were the most positive. Solvent swelling increases to a point where little or no repulsive interactions exist between neighboring redox sites. In the case of 0.1M $TBABF_4/C_6H_5CN$, overswelling of the film caused severe plastic deformation of the films at higher temperatures(Figure 47). Another interesting feature of the results is that the slopes of the lines become a stronger function of the supporting electrolyte as solvent swelling increases. Mechanical effects on PMEs should differ in different

solvents since acetonitrile, butyronitrile, and benzonitrile have different thermal expansibilities, isothermal compressibilities, and coefficients of viscosity.[198] Development of a chemical potential equation for the solvent was not included in the ME model. This eliminated possible mechanical effects from solvent swelling.[38]

Additionally, thin film CVs in benzonitrile had to be obtained as positive instead of negative scans. In fact, PMEs in benzonitrile had to be poised at 0.1V vs. Ag/AgCl for 20-30s before reversible behavior could be obtained. In the reduced state, Fc^{o} sites in the film are not perturbed by counterion uptake and solvent swelling to the extent Fc^{+} sites are in the oxidized state. For negative scans in benzonitrile however, cycling from $Fc^{+} \longrightarrow Fc^{o} \longrightarrow Fc^{+}$ was too fast relative to the time needed for elastic shrinking and swelling of the film, and voltammetric results were not reproducible. Films which are severely stressed by counterion uptake and solvent swelling require more time to relax that stress.[195-197] This was observed in benzonitrile.

D. Implications of the Mechanical/Electrochemical Model

Previous discussions involving mechanical stress-strain effects on the interactions parameter model have been invoked by Evans and coworkers[38-40] to demonstrate that internal mechanical forces play a major role in the electrochemistry of PMEs. Morphological changes in PPVF films were observed in scanning electron micrographs at different degrees of film oxidation. Delamination of their films from the electrode surface was the result of stress-strain dissipation as counterions and solvent penetrated the film. In highly crosslinked, PPVF

films, the modulus of elasticity did not vary systematically with choice of solvent(water, acetonitrile, or methylene chloride) or counterion(p-toluene sulfonate(pTS), or perchlorate). Perchlorate is considerably larger than pTS(V_A of eq. (7), Figure 7) and should have changed the mechanical contribution to electrode potential in Region III.[39,56] But, the model neglects elasticity changes from solvent-polymer interactions.[38] Interaction paremeter plots displayed this effect clearly, although only qualitatively, because r is a composite of all charge transport phenomena. Most likely, $\Gamma_T r$ versus T^{-1} plots of PPVF would be close to the TCNQ-TPE results in Figure 52. Crosslinking impedes counterion uptake and solvent penetration; hence, molecular motions of polymer chains are short ranged and PPVF films would exhibit lower elasticities.[195-197] There appears to be insufficient elasticity from solvent penetration and counterion uptake in PPVF films to display different electrochemistry.

In linear PVF, crosslinks are not present and temperature, supporting electrolyte, and solvent effects on elasticity are greater than in PPVF films.[40] Region II of the ME model is equivalent to the interaction parameter results in Figure 52. Thermodynamic plots of two PVF systems appear in Figures 53 and 54. Integrated voltammetric data gave an equivalent presentation to coulometric data. It is possible that Eisenberg quartets[134,135] are favored at higher temperatures and/or as solvent swelling increases, but there is no real quantitative significance in fitting Region II slopes of these curves to the equations in Figure 8B-D.[40] The $E-E_{p,a}$ vs. $\log(Fc^+/Fc^o)$ plots are very insensitive in Region II. Thus, only the extreme cases for butyro-

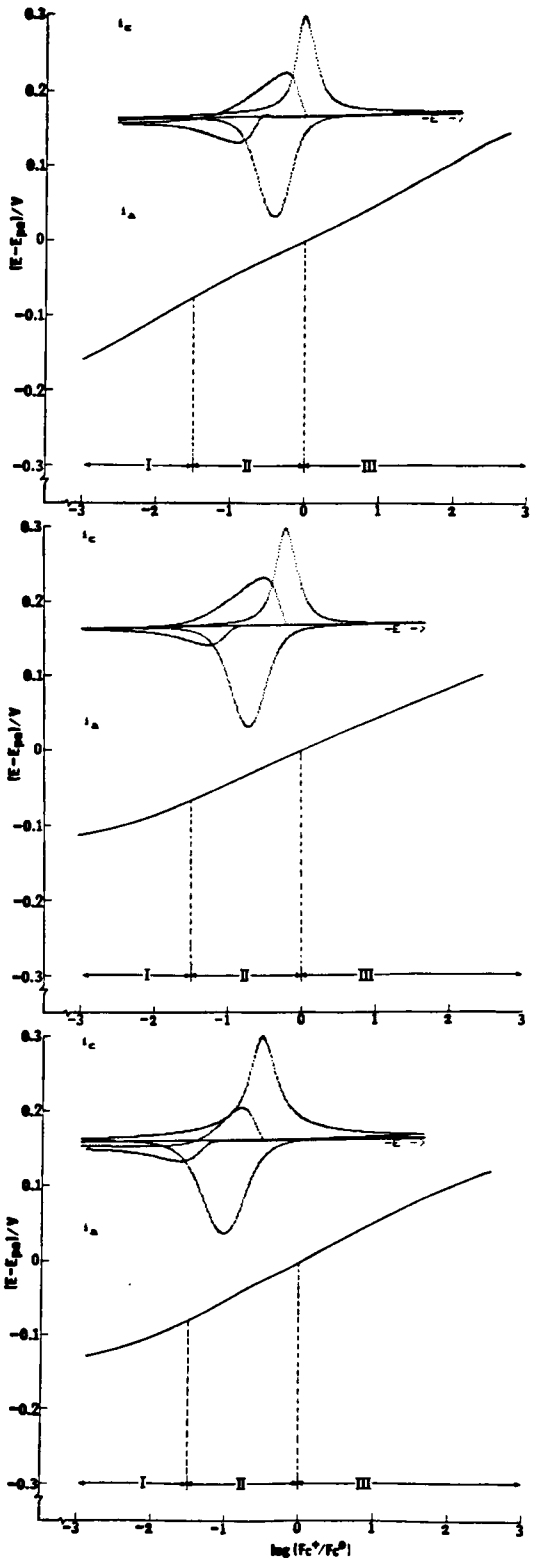
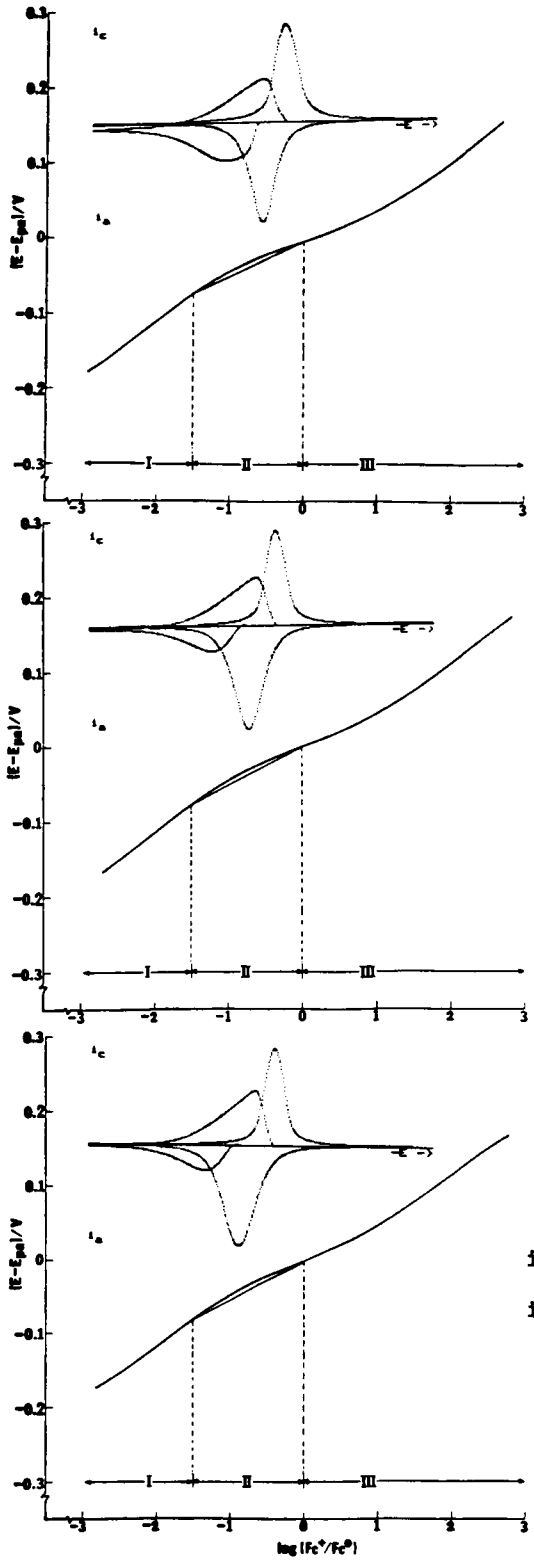


Figure 53. Correlation of Region II of the ME model and the percent mechanical contribution to anodic peak charge of a PVF thin film in 0.1M $TBABF_4/C_3H_7CN$.



T = 20.48°C

$i_{p,c}$ = -40.6uA $E_{p,c}$ = 0.352V

$i_{p,a}$ = 41.1uA $E_{p,a}$ = 0.399V

Anodic E_{FWHM} = 54.0mV

Region III, $\%Q_a$ = 47%

T = -0.85°C

$i_{p,c}$ = -24.0uA $E_{p,c}$ = 0.366V

$i_{p,a}$ = 26.2uA $E_{p,a}$ = 0.417V

Anodic E_{FWHM} = 60.0mV

Region III, $\%Q_a$ = 27%

T = -11.60°C

$i_{p,c}$ = -21.3uA $E_{p,c}$ = 0.369V

$i_{p,a}$ = 21.7uA $E_{p,a}$ = 0.429V

Anodic E_{FWHM} = 67.4mV

Region III, $\%Q_a$ = 24%

Figure 54. Correlation of Region II of the ME model and the percent mechanical contribution to anodic peak charge of a PVF thin film in 0.1M $TBABF_4/C_6H_5CN$.

nitrile and benzonitrile are shown in Figures 53 and 54. A greater arc is inscribed in Region II in benzonitrile, so greater mechanical contributions to electrode potential are expected in this solvent. The interaction parameter model is more sensitive to redox site interactions than the Region II presentation of the ME model, although the former considers only two-dimensional, lateral interactions and the latter considers three-dimensional bulk interactions.[40]

Evans and coworkers[40] did not observe mechanical damage to PVF thin films in 0.1M TBAP/CH_3CN at 25°C. Our results are consistent with theirs but further tested applicability of the ME model to PVF thin films in different solvents. The BF_4^- and ClO_4^- anions are nearly the same size. This rules out the possibility that changes in the molar volume of the counterion alone, $V_A(cm^3/mol)$, altered the mechanical effect in Region III. Instead of comparing super-Nernstian slopes in Region III to the modulus of elasticity(K) term(eq. (7), Figure 7), the mechanical contribution to the voltammetric response was compared by deconvolution, subtraction, and integration of the CVs(Figures 53 and 54). The $\%Q_a$ was easily computed because integration of voltammetric peaks was performed on both the SMOOTHED and DECONVOLUTED forms of each CV. Four or five of the points closest to the regression line in the interactions parameter plots(Figure 52) were used to construct the Region III $\%Q_a$ vs. T^{-1} plots in Figure 55. The top set of curves shows that mechanical effects are not drastically altered by film thickness; the bottom set of curves show that solvent and temperature exert the greatest influence on film elasticity. The supporting electrolyte only slightly displaced each curve in a particular solvent.

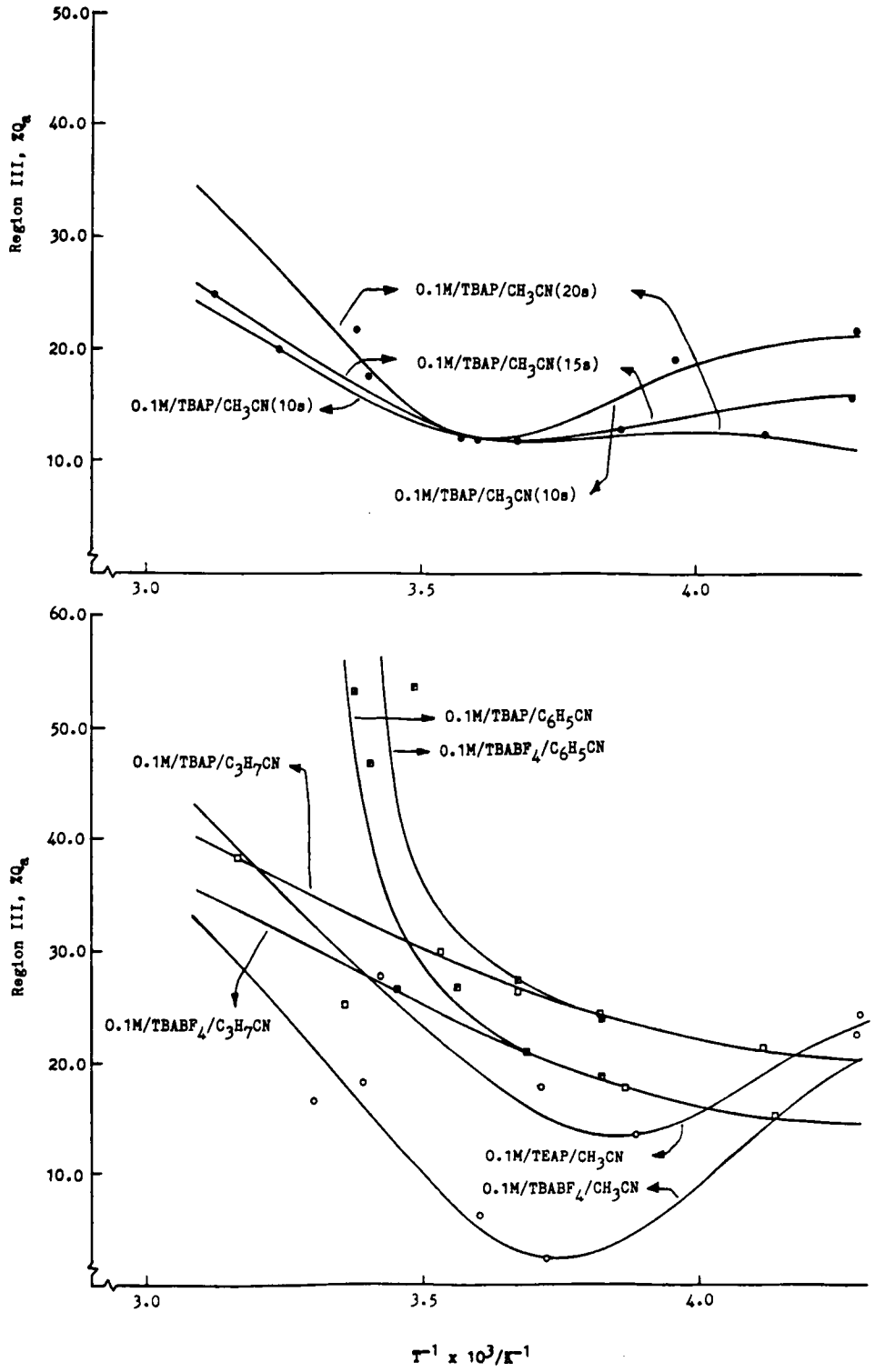


Figure 55. Mechanical contribution plots for (A) film thickness results in 0.1M TBAP/CH_3CN and (B) all the supporting electrolyte/solvent systems studied.

Near the freezing point of the solvents, the PMEs behaved like glasses,[195-197] solvent and counterion intrusion were hindered, and all of the curves impinged on a $\%Q_a$ of 20-25%. The anomalous feature of the $\%Q_a$ vs. T^{-1} data is that the butyronitrile curves are not below the acetonitrile curves. Interaction parameter results in Figure 52 indicate that acetonitrile swells the films more than butyronitrile. Others interpret this as an indication that the counterion has a smaller solvation sheath in acetonitrile than in butyronitrile.[38,198] For more positive $\Gamma_T r$ values, greater ion pairing, and redox site aggregation, one expects more swelling, elasticity, and mechanical effects in acetonitrile than in butyronitrile.

Correction of the data for ferricenium or TCNQ decay was not performed in these charge calculations. This might explain the anomaly in Figure 55. In general, as voltammetric waves attenuate and broaden, mechanical effects are more difficult to observe and extract numerically from digitally acquired data. The above interpretation does however correlate superbly with the interaction parameter model for a rational explanation of temperature, supporting electrolyte, and solvent effects on PME electrochemistry.

CHAPTER VI

CONCLUSIONS

PMEs are strongly influenced by temperature,[37,195-197] solvent,[82,98] supporting electrolyte,[56,82,98] film thickness,[35,73,81] elasticity,[38-40] the nature of the redox moiety[35,41] and its concentration,[73,83-86] and the experimental measurement time[196] when subjected to the stress and strain of redox conversion. It seems neither the interaction parameter nor mechanical/electrochemical model alone can explain all of these complicating phenomena. A combination of the two, when applied to voltammetric data not only rationalized these results but helped to evaluate the recently proposed ME model. Existing studies of charge transport in PMEs attribute negative $\Gamma_T r$ values to low temperature, film thickness, redox site concentration, and fast potential scan rates.[35] The ME model suggests that polymer elasticity is greatly affected by the particular solvent and supporting electrolyte and should be included in discussions of charge transport.[35,38] Without consideration of elasticity, i.e. the second term in eq. (7) of Figure 7, and the interaction parameter term, plastic deformation of PVF thin films from overswelling and overstress at high temperatures cannot be explained. While interaction parameter results support such a conclusion, extraction of quantitative information remains difficult. The interaction parameter model is nevertheless simple to apply[37] and useful for comparative studies.

The ME model is somewhat deficient in both its derivation and evaluation as these results indicate. Changes in osmotic pressure from solvent swelling as well as counterion intrusion should be included in the thermodynamic treatment. The derivation would then include the electrochemical potential equation for the solvent,

$$\bar{\mu}_S = \mu_S^o + RT \ln (\gamma_S c_S) + z_S F\phi + PV_S \qquad (47)$$

where the PV_S term includes osmotic pressure(i.e. mechanical) changes imparted by the solvent. The magnitude of this term is affected by polymer-solvent interactions,[195-197] the size of the solvation sheath around the counterion,[198] and the electrostatic attraction of the charged redox site for the solvated counterion.[38] When Evans and coworkers[38,40] evaluated their model for PVF-PMEs, they investigated only one film in 0.1M TBAP/CH_3CN at 25°C. Hence, they never observed osmotic pressure or solvent elasticity changes in their results. PPVF was a poor choice for studying solvent and counterion effects on elasticity because crosslinking inhibits intrusion of solvents and ions into the film.[39,195,196] Modification of the ME model to account for the mechanical contribution to electrode potential from the forced intrusion of the solvent is certainly warranted. It is also worth noting that cyclic voltammetry provides useful information for comparison to both models, whereas controlled potential coulometry does not. This is due to the microcomputer which was used to numerically convert digital, i-t data to Q-t data.[184-186]

Also, successful application of the UCSD Pascal programming language to two electrochemical methods has been demonstrated. Modular programming, data structures, and library UNITs allowed us to solve a large and complex interfacing problem. Undoubtedly, much of the application research in this area goes unreported. We have suggested some programming and interfacing techniques to assist other analytical chemists in their interfacing problems. It is also worth noting that the system described is inexpensive and readily available. The most difficult task for us was choosing the microcomputer, peripheral hardware, and programming language best suited for the job in hand. Only recently has information appeared to assist the analytical chemist in interfacing decisions.[3-5,14]

Our discussion here would be incomplete without some comment on further interfacing possibilities. For example, one might ask if later Apple II series microcomputers and higher Versions of Pascal or similar languages would provide more flexibility. Modula-2 and C are among the possibilities.[14] Or, one might ask if some of the more complex square wave voltammetric methods[199] could be automated on the present system. These questions are now under consideration for future electrochemical research.

Data acquisition peripherals, including the ones we used on the Apple II+, are described by Annino and Driver.[14] They present many vendors from which these peripherals can be purchased and a specific interface to the Apple IIe with the ADALAB card. Interactive Microware[200] now markets a complete II+ or IIe system with software similar to ours for cyclic voltammetric, chronocoulometric, and differential

pulse voltammetric analysis. An upgrade to Version 1.2 Pascal and an Apple IIe with the same peripherals would easily rival their commercial software. Version 1.2 would be able to address 128K of memory[162] and remove the memory problems we encountered; most of the Pascal code written in Version 1.1 is portable to an upgraded system also.

Earlier this year, the Apple IIGS was introduced.[201] Apple has not developed a Pascal operating system for the IIGS, but TML offers a version that could be used to automate chromatographic and electrochemical analysis.[202] The IIGS represents the high end of the price range for most laboratories, but the possibilities for a Pascal interface are unlimited. The peripherals of the II+ and IIe for data acquisition and control are directly compatible with the IIGS, memory-mapped architecture.[203] The IIGS contains a 65816, 16-bit version of the 8-bit 6502 which can operate in native or emulation mode. In the native mode, the Pascal programmer can take advantage of a 2.5MHz clock speed to execute software, perform I/O, and graphics; in the emulation mode, we might collect electrochemical data based on the 1MHz clock speed of the original 6502 so that Scanlon's[149] software delay timer can be applied. The emulation mode was designed to take advantage of software developed for the Apple II, II+, and IIe.[203]

UCSD Pascal is not the only structured language available for the Apple II series. Modula-2 and C are worth considering for interfacing on the Apple.[14] Both languages can be extended with a library and linked to assembly languages for real-time data acquisition. Both languages support the record and pointer data types for advanced data structure applications.[14] Modula-2 could be used to improve the speed

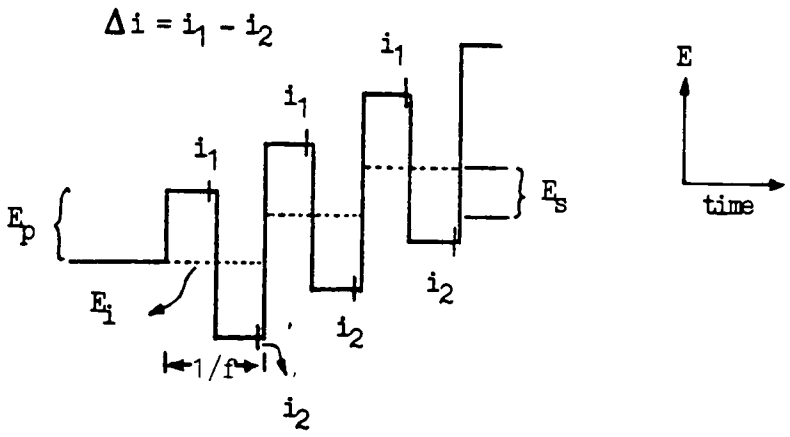
of PROGRAM execution by using its IMPORT and EXPORT facilities.[15,204] We noticed in our software design that a FILE of data had to be transferred into a linked list each time one of the PROGRAMs for smoothing, peak fitting, etc. were executed. This slowed execution considerably and could not be prevented. Values of VARiables are lost when a Pascal program terminates execution.[150,162,172] In Modula-2, programs are written as MODULEs;[15,204] VARiables such as the FILEs of CV parameters and data could be EXPORTed to or IMPORTed by other MODULEs which perform smoothing or peak fitting. Then, we would have to perform I/O to floppy disk one time only to execute all of the MODULEs. Improvement in the efficiency of our software could also be achieved in the C programming environment.[14] It is more assembly-like in compiled form and executes more quickly than Pascal or Modula-2. Annino and Driver[14] have shown the advantages of C for double precision numerical analysis with listings of a fast Fourier transform and cubic spline interpolation, although C would not be particularly useful for the file I/O problems. Even without the improvements of Modula-2 or C in this application, an increase in speed could be achieved in the native mode of the IIGS using TML Pascal.[201,203]

A good test of the versatility of the system described is whether or not it could be used to apply a more complex digital waveform to the cell and collect i-E data. The CV-27 is incapable of generating the more complex voltammetric square waves,[144] so a DAC must be used.[163] If we consider the inherently sensitive and fast square wave voltammetry described by Osteryoung,[199] the experimental parameters the Pascal PROGRAM would request are the initial E(V), final E(V), the

magnitude of each pulse with respect to the staircase base($10mV \leq E_p \leq 100mV$), the potential step height($5mV \leq E_s \leq 35mV$), and the pulse frequency, f, as shown in the potential versus time waveform of Figure 56. The first problem of our system for generating this waveform is the lack of 12-bit resolution in the DAC. A 12-bit DAC similar to the one on the ADALAB card would be appropriate.[14] On a $\pm$2.0V output range, the 12-bit, ADALAB DAC would have ca. 0.5mV resolution. The idea is to pulse very quickly(i.e. a high f), but not so quickly that the time required for adding $2E_p + E_s$, sampling i_1, i_2, incrementing memory and storing binary data, subtracting $2E_p$, latching the DAC, executing a minimum delay time, and checking to see if done exceeds 1/f. If it required 300 microseconds for example to do this, f could not exceed 3,333 Hz. We would write the code and determine this overhead time from an assembled listing, then use it to calculate the DELAY values just as we did for CV.[149] The number of i_1, i_2, values taken is $(E_f - E_i)/E_s$. The Pascal calling statement for acquisition of square wave voltammetric data would look something like,

```
TAKEDATA(PAIRS,EP,ES,EI,TCHAN,ICHAN,ECHAN,DELAY1,
DELAY2,DELAY3,MEMLOC);
```

where PAIRS is the number of i_1, i_2 data pairs, EP, ES, and EI are the binary numbers representing E_p, E_s, and E_i in Figure 56, respectively, TCHAN is the DAC trigger channel, ICHAN is the ADC channel for sampling the current, ECHAN is the DAC channel issuing the digital waveform to the CV-27, the DELAYs are the same as described for CV, and MEMLOC is



1. Pull passed parameters and Pascal return address from evaluation stack and store in page 0 memory.
2. Latch DAC ECHAN to EI.
3. Send 5.0V to DAC TCHAN and switch from STANDBY to CELL.
4. Add EP to EI and latch DAC output on ECHAN.
5. Jump to current conversion SUBROUTINE to collect i_1.
6. Subtract 2 * EP from DAC ECHAN and latch voltage at output.
7. Jump to current conversion SUBROUTINE to collect i_2.
8. Subtract 1 from PAIRS and BRANCH to step 11 if 0, otherwise go step 9.
9. Add (2 * EP) and ES to DAC ECHAN and latch voltage at output.
10. BRANCH unconditionally to step 5.
11. Send 0.0V to DAC TCHAN and switch from CELL to STANDBY.
12. Restore Pascal return address and return to Pascal caller.

SUBROUTINE

1. Execute software delay timer.
2. Convert on ICHAN and store in MEMLOC and MEMLOC +1.
3. Increment MEMLOC variable by 2.

Figure 56. Demonstration of the use of a DAC and a microprocessor assembly language to perform square wave voltammetry.

the RAM location where we start storing binary data. We assume that the potential values for each stair step are accurate enough that they do not have to be sampled. A summary of the execution steps in a 6502 data acquisition routine for square wave voltammetry is shown in Figure 56. This is the type of outline an analytical chemist would develop to write his/her own data acquisition software. We hope that the topics and ideas discussed here encourage other chemists to adopt similar approaches to solving their interfacing problems.

REFERENCES

REFERENCES

1. Ridgway, T. H.; Mark, H. B., Jr., in "Comprehensive Treatise of Electrochemistry," White, R.E.; Bockris, J. O'M; Conway, B. E.; Yeager, E.(Eds.), Vol. 8, Plenum Press, New York, 1984, Ch. 2.

2. Macero, B. J.; Burgess, L. W., Jr.; Banks, T. M.; Chapman, D., in "Personal Computers in Chemistry," Lykos, P.(Ed.), John Wiley & Sons, New York, 1981, Ch. 17.

3. Dessy, R. E. Anal. Chem. **1986**, 58, 678A.

4. Dessy, R. E. Anal. Chem. **1986**, 58, 793A.

5. Dessy, R. E. Anal. Chem. **1986**, 58, 919A.

6. Wikmark, G. Anal. Instr. **1984**, 13(2), 145.

7. Gray, E. T., Jr.; Workman, H. J., in "Personal Computers in Chemistry," Lykos, P.(Ed.), John Wiley & Sons, New York, 1981, Ch. 5.

8. Rasmussen, M. D.; Akinc, M. Rev. Sci. Instr. **1983**, 4, 1558.

9. Glass, A. S.; Low, M. J. D. Anal. Instr. **1985**, 14(1), 97.

10. Moore, J. W.(Ed.) J. Chem. Ed., Computer Series 62 **1985**, 62(8), 687-692.

11. Pyun, C.; Park, S. Anal. Instr. **1984**, 13(2), 159.

12. Patonay, G.; Warner, I. M. Anal. Instr. **1985**, 14(2), 155.

13. Goliber, P. A.; Michel, R. G. Anal. Instr. **1986**, 15(1), 77.

14. Annino, R.; Driver, R. D. "Scientific and Engineering Applications with Personal Computers," John Wiley & Sons, New York, 1986, pp. 10, 66-72, 118-120, 134-165, 289-298, 341-364, 495-502.

15. Gleaves, R. "Modula-2 for Pascal Programmers," Springer-Verlag, New York, 1984.

16. Britz, D. Anal. Chim. Acta **1982**, 143, 95.

17. Fanelli, N.; Fucco, R.; Guidarini, D.; Papoff, P. Anal. Chim. Acta **1986**, 185, 33.

18. Grande, D. G.; Paul, D. W. Anal. Instr. **1986**, 15(3), 227.

19. von Wandruszka, R. M.; Maraschin, M. Anal. Lett. **1981**, 14, 463.

20. A^+ **1986,** 4(11), 150.

21. Lindenbaum, S. J. Ann. Rev. Nucl. Sci. **1966,** 16, 619.

22. Lauer, G.; Abel, R.; Anson, F. C. Anal. Chem. **1967,** 39, 765.

23. VandenBorn, H. W.; Whitson, P. E.; Evans, D. H. Anal. Chem. **1973,** 45, 1298.

24. Perone, S. P.; Frazer, J. W.; Kray, A. Anal. Chem. **1971,** 43, 1485.

25. Keller, H. E.; Osteryoung, R. A. Anal. Chem. **1971,** 43, 342.

26. Gutknecht, W. F.; Perone, S. P. Anal. Chem. **1970,** 42, 906.

27. Gaarenstroom, P. D.; English, J. C.; Perone, S. P. Anal. Chem. **1978,** 50, 811.

28. Leggett, D. J. Anal. Chem. **1978,** 50, 718.

29. Johnson, D. E; Enke, C. G. Anal. Chem. **1970,** 42, 329.

30. Ramaley, L. Anal. Instr. **1986,** 15(2), 101.

31. Li, C.; Barrett, T. H.; Lunney, D.; Salt, A. Anal. Chim. Acta. **1982,** 134, 167.

32. Baecklund, P.; Danielson, R. Anal. Chim. Acta **1983,** 154, 61.

33. Mann, C. K. Anal. Chem. **1961,** 33, 1484.

34. Leventhal, L. A., "6502 Assembly Language Programming," Osborne/ McGraw-Hill, Berkeley, CA, 1979, Ch 3.

35. Murray, R. W., in "Electroanalytical Chemistry," Bard, A. J.(Ed.), Marcel Dekker, New York, Vol. 13, 1984, p.191.

36. Brown, A. P.; Anson, F. C. Anal. Chem. **1977,** 49, 1589.

37. Chambers, J. Q.; Inzelt, G. Anal. Chem. **1985,** 57, 1117.

38. Bowden, E. F.; Dautartas, M. F.; Evans, J. F. J. Electroanal. Chem. **1987,** 219, 49.

39. Bowden, E. F.; Dautartas, M. F.; Evans, J. F. J. Electroanal. Chem. **1987,** 219, 71.

40. Bowden, E. F.; Dautartas, M. F.; Evans, J. F. J. Electroanal. Chem. **1987,** 219, 91.

41. Murray, R. W. Acc. Chem. Res. **1980**, 13, 135.

42. Faulkner, L. R. C & E News **Feb. 27, 1984**, 28-45.

43. Laviron, E., in "Electroanalytical Chemistry," Bard, A. J.(Ed.), Marcel Dekker, New York, Vol. 12, 1982, p. 53.

44. Murray, R. W. Ann. Rev. Mat. Sci. **1984**, 14, 145.

45. Wrighton, M. S. Science **1986**, 251, 32.

46. Bolts, J. M.; Bocarsly, A. B.; Palazzotto, M. C.; Walton, E. G.; Lewis, N. S.; Wrighton, M. S. J. Am. Chem. Soc. **1979**, 101, 1378.

47. Chao, S.; Robbins, J. L.; Wrighton, M. S. J. Am. Chem. Soc. **1983**, 181, 105.

48. Durand, R. R.; Bencosme, C. S.; Collman, J. P.; Anson, F. C. J. Am. Chem. Soc. **1983**, 105, 2710.

49. Lau, A. N. K.; Miller, L. L. J. Am. Chem. Soc. **1983**, 105, 527.

50. Bookbinder, C. S.; Wrighton, M. S. J. Electrochem. Soc. **1983**, 130, 1080.

51. Price, J. F.; Baldwin, R. P. Anal. Chem. **1980**, 52, 1940.

52. Lane, R. F.; Hubbard, A. T. J. Phys. Chem. **1973**, 77, 1401.

53. Lane, R. F.; Hubbard, A. T. J. Phys. Chem. **1973**, 77, 1411.

54. Conway, B. E.; Gileadi, E.; Dzieciuch, M. Electrochim. Acta **1963**, 8, 143.

55. Laviron, E. J. Electroanal. Chem. **1981**, 122, 37.

56. Peerce, P. J.; Bard, A. J. J. Electroanal. Chem. **1980**, 114, 89.

57. Laviron, E.; Roullier, L. J. Electroanal. Chem. **1980**, 115, 65.

58. Moses, P. R.; Wier, L.; Murray, R. W. Anal. Chem. **1975**, 47, 1882.

59. Kuo, K.; Moses, P. R.; Lenhard, J. R.; Green, D. C.; Murray, R. W. Anal. Chem. **1979**, 51, 745.

60. Finklea, H. O.; Murray, R. W. J. Phys. Chem. **1979**, 83, 353.

61. Abruna, H.; Meyer, T. J.; Murray, R. W. Inorg. Chem. **1979**, 11, 3233.

62. Wrighton, M. S.; Austin, R. G.; Bocarsly, A. B.; Bolts, J. M.; Legg, K. D.; Nadjo, L; Palazzotto, M. C. J. Am. Chem. Soc. **1978**, 100, 1602.

63. Bolts, J. M.; Wrighton, M. S. J. Am. Chem. Soc. **1978**, 100, 5257.

64. Kaufman, F. B.; Engler, E. M. J. Am. Chem. Soc. **1979**, 101, 547.

65. Kaufman, F. B.; Schroeder, A. H.; Engler, E. M.; Kramer, S. R.; Chambers, J. Q. J. Am. Chem. Soc. **1980**, 102, 483.

66. Miller, L. L.; Van de Mark, M. R. J. Am. Chem. Soc. **1978**, 100, 639.

67. Daum, P.; Murray, R. W. J. Phys. Chem. **1981**, 85, 389.

68. Schneider, J.; Murray, R. W. Anal. Chem. **1982**, 54, 1508.

69. Merz, A.; Bard, A. J. J. Am. Chem. Soc. **1978**, 100, 3222.

70. Kanazawa, K. K.; Diaz, A. F.; Geiss, R. H.; Gill, W. D.; Kwak, J. F.; Logan, J. A.; Rabolt, J. F.; Street, G. B. J. Chem. Soc. Commun. **1979**, 854.

71. Rubinstein, I.; Bard, A. J. J. Am. Chem. Soc. **1980**, 102, 6641.

72. Peerce, P. J.; Bard, A. J. J. Electroanal. Chem. **1980**, 112, 97.

73. Karimi, H.; Chambers, J. Q. J. Electroanal. Chem. **1987**, 217, 313.

74. Itaya, K.; Bard, A. J. Anal. Chem. **1978**, 50, 1487.

75. Evans, J. F.; Kuwana, T.; Henne, M. T.; Royer, G. P. J. Electroanal. Chem. **1977**, 80, 409.

76. Andrieux, C. P.; Saveant, J. M. J. Electroanal. Chem. **1980**, 111, 377.

77. Diaz, A.; Vallejo, J. M. V.; Duran, A. M. IBM J. Res. Dev., **1979**, 23, 316.

78. Anson, F. C. J. Phys. Chem. **1980**, 84, 3336.

79. Bouchart, J. M. D.; Saveant, J. M. J. Electroanal. Chem. **1980**, 114, 159.

80. Miller, L. L.; Van de Mark, M. R. J. Am. Chem. Soc. **1978**, 100, 3223.

81. Chambers, J. Q. J. Electroanal. Chem. **1981**, 130, 381.

82. Daum, P.; Murray, R. W. J. Electroanal. Chem. **1979**, 103, 289.

83. Daum, P.; Lenhard, J. R.; Rolison, D. R.; Murray, R. W. J. Am. Chem. Soc. **1980**, 102, 4649.

84. Dahms, H. J. Phys. Chem. **1968**, 72, 362.

85. Lengyel, S. Magy. Kem. Foly. **1974**, 80, 187.

86. Ruff, I. Electrochim. Acta **1970**, 15, 1059.

87. Facci, J. S.; Schmel, R. H.; Murray, R. W. J. Am. Chem. Soc. **1982**, 104, 4959.

88. Wrighton, M. S.; Austin, R. G.; Bocarsly, A. B.; Bolts, J. M.; Haas, O.; Legg, K. D.; Nadjo, L.; Palazzotto, M. C. J. Electroanal. Chem. **1978**, 87, 429.

89. Willman, K. W.; Rocklin, R. D.; Nowak, R.; Kuo, K.; Schultz, F. A.; Murray, R. W. J. Am. Chem. Soc. **1980**, 102, 7629.

90. Oyama, N.; Anson, F. C. J. Electrochem. Soc. **1980**, 127, 247.

91. Faulkner, L. R.; Majda, M. J. Electroanal. Chem. **1982**, 137, 149.

92. Mortimer, R. J.; Anson, F. C. J. Electroanal. Chem. **1982**, 138, 325.

93. Abruna, H. D.; Bard, A. J. Anal. Chem. **1981**, 103, 6898.

94. Schroeder, A. H.; Kaufman, F. B.; Patel, V.; Engler, E. M. J. Electroanal. Chem. **1980**, 113, 193.

95. Day, R. W.; Inzelt, G.; Kinstle, J. F.; Chambers, J. Q. J. Am. Chem. Soc. **1982**, 104, 6804.

96. Martin, C. R.; Rhodes, T. A.; Ferguson, J. A. Anal. Chem. **1982**, 54, 1639.

97. Oyama, N.; Anson, F. C. J. Am. Chem. Soc. **1979**, 101, 739.

98. Joo, P.; Chambers, J. Q. J. Electrochem. Soc. **1985**, 132, 1345.

99. Inzelt, G.; Chambers, J. Q.; Kinstle, J. F.; Day, R. W. J. Am. Chem. Soc. **1984**, 106, 3396.

100. Inzelt, G.; Chambers, J. Q.; Kaufman, F. B. J. Electroanal. Chem. **1983**, 159, 443; **1985**, 185, 208.

101. Inzelt, G.; Day, R. W.; Kinstle, J. F.; Chambers, J. Q. J. Physical. Chem. **1983**, 87, 4592.

102. Inzelt, G.; Day, R. W.; Kinstle, J. F.; Chambers, J. Q. J. Electroanal. Chem. **1984**, 161, 147.

103. Nowak, R. J.; Schultz, F. A.; Umana, M.; Lam, R.; Murray, R. W. Anal. Chem. **1980**, 52, 315.

104. Nakahama, S.; Murray, R. W. J. Electroanal. Chem. **1983**, 158, 303.

105. Inzelt, G.; Backsai, J.; Chambers, J. Q.; Day, R. W. J. Electroanal. Chem. **1986**, 201, 301.

106. Day, R. W. Ph. D. Thesis, University of Tennessee, Knoxville, 1984.

107. Francis, C. V.; Chambers, J. Q. J. Phys. Chem. **1987**, 91, 6315.

108. Laviron, E. Bull. Soc. Chim. Fr. **1967**, 3717.

109. Laitinen, H. A.; Vincent, C. A.; Bednarski, J. J. J. Electrochem. Soc. **1968**, 115, 1024.

110. Bard, A. J.; Faulkner, L. R. "Electrochemical Methods," John Wiley & Sons, New York, 1980, a)p. 516, b)p.522, c)p. 583.

111. Laviron, E. J. Electroanal. Chem. **1979**, 100, 263.

112. Angerstein-Kozlowska, H.; Klinger, J.; Conway, B. E. J. Electroanal. Chem. **1977**, 75, 45.

113. Angerstein-Kozlowska, H.; Klinger, J.; Conway, B. E. J. Electroanal. Chem. **1977**, 75, 61.

114. Angerstein-Kozlowska, H.; Klinger, J.; Conway, B. E. J. Electroanal. Chem. **1978**, 87, 301.

115. Angerstein-Kozlowska, H.; Klinger, J.; Conway, B. E. J. Electroanal. Chem. **1978**, 87, 321.

116. Srinivasan, S.; Gileadi, E. Electrochim. Acta **1966**, 11, 143.

117. Gileadi, E.; Conway, B. E. J. Chem. Phys. **1963**, 39, 3420.

118. Conway, B. E. J. Electroanal. Chem. **1964**, 8, 486.

119. Laviron, E. J. Electroanal. Chem. **1974**, 52, 355.

120. Laviron, E. J. Electroanal. Chem. **1974**, 52, 395.

121. Laviron, E. J. Electroanal. Chem. **1979**, 101, 1979.

122. Smith, D. F.; Willman, K.; Kuo, K.; Murray, R. J. Electroanal. Chem. **1979**, 95, 217.

123. Lenhard, J. R.; Murray, R. W. J. Am. Chem. Soc. **1978**, 100, 7870.

124. Oyama, N.; Anson, F. C. J. Electrochem. Soc. **1980**, 127, 640.

125. Ikeda, T.; Leidner, C. R.; Murray, R. W. J. Electroanal. Chem. **1982**, 138, 343.

126. Daifuku, H.; Aoki, K.; Tokuda, K.; Matsuda, H. J. Electroanal. Chem. **1985**, 183, 1.

127. Buck, R. P. Crit. Rev. Anal. Chem. **1975**, 5, 323.

128. Helfferich, F. "Ion Exchange," McGraw-Hill, New York, 1962, Ch.5.

129. Braun, H.; Storck, W.; Dolbhofer, K. J. Electrochem. Soc. **1983**, 130, 807.

130. Kuhn, W.; Ramel, A.; Walters, D. H., in "Size and Shape Changes of Contractile Polymers," Wassermann, A.(Ed.), Pergamon, Oxford, 1960, p. 41.

131. Van Krevelen, D. W. "Properties of Polymers:Correlations with Chemical Structure," Elsevier, Amsterdam, 1972, pp. 50, 150.

132. Huntington, H. B. "The Elastic Constants of Crystals," Academic Press, New York, London, 1958, p. 64.

133. Dautartas, M. F.; Evans, J. F. J. Electroanal. Chem. **1980**, 109, 301.

134. Eisenberg, A. Macromolecules **1970**, 3, 147.

135. MacKnight, W. J.; Earnest, T. R. J. Polym. Sci., Macromol. Rev. **1981**, 16, 41.

136. Smith, T. W.; Kuder, J. E.; Wychick, D. J. Polym. Sci., Polym. Chem. Ed. **1976**, 14, 2433.

137. Day, R. W.; Karimi, H.; Francis, C. V.; Kinstle, J. F.; Chambers, J. Q. J. Polym. Sci., Polym. Chem. Ed. **1986**, 24, 645.

138. Greef, R.; Peat, R.; Peter, L. M.; Pletcher, D.; Robinson, J., in "Instrumental Methods in Electrochemistry," Kemp, T. J.(Ed.), John Wiley & Sons, New York, 1985, pp. 49-55, 178-206.

139. Leddy, J.; Bard, A. J. J. Electroanal. Chem. **1985**, 189, 203.

140. Leddy, J.; Bard, A. J. J. Electroanal. Chem. **1983**, 153, 223.

141. von Stackelberg, M.; Pilgram, M; Toome, V. Zeit. fur Elektro. **1953**, 57(5), 342.

142. Lingane, P. Anal. Chem. **1964**, 36(9), 1723.

143. BAS-100 Electrochemical Analyzer/Operation, Instruction, & Maintenance Manual, Bioanalytical Systems, Inc., 1983.

144. Operation, Instruction Manual for the CV-27 Voltammograph, Bioanalytical System, Inc., 1984.

145. Carr, J. J. "Microcomputer Interfacing Handbook:A/D & D/A," Tab Books, Blue Ridge Summit, Pennsylvania, 1980, pp. 33-61, 98-101, 117-161.

146. Titus, J. A.; Larsen, D. G.; Titus, C. A. "Apple Interfacing," Howard W. Sams & Co., Indianapolis, 1981, pp. 9-14.

147. Apple II Reference Manual, Apple Computer Inc., Cupertino, California, 1981.

148. DeJong, M. L. "Programming and Interfacing the 6502, with Experiments," Howard W. Sams & Co., Indianapolis, 1980, pp. 20-25.

149. Scanlon, L. J. "6502 Software Design," Howard W. Sams & Co., Indianapolis, 1980, pp. 13-91.

150. Apple Pascal, Operating System Reference Manual, Apple Computer Inc., Cupertino, California, 1980.

151. Viewmaster-80 User Manual, Applied Engineering, Carrollton, Texas, 1983.

152. Thunderclock Plus Installation and Operating Manual, Thunderware Inc., Oakland, California, 1982.

153. Pascal Software Manual for the Thunderclock Plus, Thunderware Inc., Oakland, California, 1980.

154. Townsley, D.; Townsley, F. A^+ **1985**, 3(10), 54.

155. Rosenthal, S. A^+ **1986**, 4(7), 22.

156. Lipson, N. E. Apple Orchard **1982**, 3(4), 24.

157. PKASO EP12-80/100 Printer Interface User's Manual, Interactive Structures Inc., Bala Cynwyd, Pennsylvania, 1982.

158. Raleigh, L. A^+ **1987** 5(1), 39.

159. Apple II Disk Operating System Manual, Apple Computer Inc., Cupertino, California, 1981.

160. Little, G. B. A^+ **1987,** 5(1), 28.

161. 128K RAM Board Operations Manual, Titan Technologies, 1982.

162. Nealon, K.; Woteki, T.; Stokes, J.; Knepley, E.; Hartman, M.; and Dr. Wo, in "Perfect Pascal Programs," Platt, R.(Ed.), Washington Apple Pi & Tab Books, Blue Ridge Summit, Pennsylvania, 1985, pp. v-vii, 3-16, 93-107, 189-206, 245-6.

163. Clayton, G. B. "Data Converters," John Wiley & Sons, New York, 1982, Chs. 2-5.

164. AO-03 User's Manual, Interactive Structures Inc., Bala Cynwyd, Pennsylvania, 1981.

165. AI13 User's Manual, Interactive Structures Inc., Bala Cynwyd, Pennsylvania, 1981.

166. Niemac, A. P. Apple Orchard **1984,** 5(1), 24.

167. StarCard Installation and User's Manual, MicroPro Int., San Rafael, California, 1984.

168. Benedict, R. P. "Fundamentals of Temperature, Pressure, and Flow Measurements," John Wiley & Sons, New York, 1977, Ch. 6.

169. Trietley, H. L. Radio Elec. **January 1985,** 47; **February 1985,** 73; **March 1985,** 67.

170. Omega Temperature Measurement Handbook, 1986, p. F-9.

171. Willner, E.; Demchak, B. "Advanced UCSD Pascal Programming Techniques," Prentice-Hall, Englewood Cliffs, New Jersey, 1985, pp. 1-4.

172. Apple Pascal, Language Reference Manual, Apple Computer Inc., Cupertino, California, 1980.

173. Matuszek, D. L. "Quick Pascal," John Wiley & Sons, New York, 1982, Chs. 8,12.

174. Swan, T. "Pascal Programs For Games and Graphics," Hayden Book Co., Rochelle Park, New Jersey, 1983, pp. 144-214.

175. Tremblay, J.-P.; Sorenson, Paul G. "An Introduction to Data Structures With Applications," 2nd ed., McGraw-Hill, New York, 1984, Ch. 4.

176. Collins, W. J. "Intermediate Pascal Programming," McGraw-Hill, New York, 1986, Chs. 1,3,4.

177. Munro, A.; Johnson, M. C. Apple Orchard **1983**, 4(3), 18.

178. Currell, G. "Instrumentation," John Wiley & Sons, New York, 1987, pp. 124-190.

179. Hamming, R. W. "Digital Filters," 2nd ed., Prentice-Hall, Englewood Cliffs, New Jersey, 1983, Chs. 1,4.

180. Cooley, J. W.; Tukey, J. W. Math. Comp. **1965**, 19, 297.

181. Aubanel, E. E.; Myland, J. C.; Oldham, K. B.; Zoski, C. G. J. Electroanal. Chem. **1985**, 184, 239.

182. Brigham, E. O. "The Fast Fourier Transform," Prentice-Hall, Englewood Cliffs, New Jersey, 1974.

183. Bracewell, R. N. "The Fourier Transform and Its Applications," McGraw-Hill, New York, 1978.

184. Oldham, K. B. J. Electroanal. Chem. **1986**, 208, 1.

185. Norris, A. C. "Computational Chemistry, An Introduction to Numerical Analysis," John Wiley & Sons, New York, 1981, Chs. 2,6,8.

186. Nonweiler, T. R. F. "Computational Mathematics, An Introduction to Numerical Approximation," John Wiley & Sons, 1984, Chs. 1-3,5.

187. Pennington, R. H. "Introductory Computer Methods and Numerical Analysis," Macmillan Co., New York, 1965, Chs. 1,13.

188. Hamming, R. W. "Numerical Methods for Scientists and Engineers," 2nd ed., McGraw-Hill, New York, 1973, pp. 349-355.

189. Pachner, J. "Handbook of Numerical Analysis Applications," McGraw Hill, New York, 1984, Ch. 2.

190. Oldham, K. B. Anal. Chem. **1986**, 58, 2296.

191. Cottrell, F. G. Z. Phys. Chem. **1903**, 41, 354.

192. Bard, A. J. Anal. Chem. **1961**, 33, 11.

193. Soos, Z. G.; Lingane, P. J. J. Phys. Chem. **1964**, 68, 3821.

194. Flanagan, J. B.; Marcoux, L. J. Phys. Chem **1973**, 77, 1051.

195. Aklonis, J. J. J. Chem. Ed. **1981**, 58(11), 892.

196. Mark, J. E. J. Chem. Ed. **1981**, 58(11), 898.

197. Anderson, J. C.; Leaver, K. D.; Rawlings, R. D.; Alexander, J. M. "Materials Science," 3rd ed., Van Nostrand Reinhold, Berkshire, England, 1985, Chs. 9,12.

198. Marcus, Y. "Ion Solvation," John Wiley & Sons, New York, 1985, Chs. 1-3,6.

199. Osteryoung, J. G. Anal. Chem. **1985**, 57, 101A.

200. Interactive Microware, Inc. A^+ **1986**, 4(11), 150.

201. Little, G. B. A^+ **1986**, 4(11), 45.

202. TML Systems A^+ **1987**, 5(9), 82.

203. Scanlon, L. "Apple IIGS Assembly Language Programming," Bantam Books, New York, 1987, Chs. 1,7.

204. Ogilvie, J. W. L. "Modula-2 Programming," McGraw-Hill, New York, 1985, pp. 1-42, 262-287.

APPENDIXES

APPENDIX A

TEXT FILE LISTINGS OF LIBRARY UNITS

```
(*$S++*)

(* These routines are designed specifically for printing with a
   PKASO parallel interface. *)

UNIT PRNTRTNS; INTRINSIC CODE 24 DATA 25;

  INTERFACE

    VAR T:TEXT;

    PROCEDURE PRSTR (WORDS,CONTROL:STRING);
      (* This routine prints strings up to 80 characters long. *)
    PROCEDURE PRCHR (CH:CHAR;CONTROL:STRING);
      (* This routine prints single characters. *)
    PROCEDURE PRINT (NUMBER,PLACE:INTEGER;CONTROL:STRING);
      (* This routine prints integers to a specified number of places. *)
    PROCEDURE PRREAL (NUMBER:REAL;CONTROL:STRING);
      (* This routine prints real numbers to 12 places. *)
    PROCEDURE FORMAT (CONTROL:STRING);
      (* This routine is used for CR/LF, screen dumping, etc. *)


  IMPLEMENTATION

    PROCEDURE PRSTR(* (WORDS,CONTROL:STRING) *);
      BEGIN
        REWRITE(T,'PRINTER:');
        WRITE(T,CHR(25),CONTROL,WORDS);
        CLOSE(T)
      END; (* PRSTR *)

    PROCEDURE PRCHR(* (CH:CHAR;CONTROL:STRING) *);
      BEGIN
        REWRITE(T,'PRINTER:');
        WRITE(T,CHR(25),CONTROL,CH);
        CLOSE(T)
      END; (* PRCHR *)

    PROCEDURE PRINT(* (NUMBER,PLACE:INTEGER;CONTROL:STRING) *);
      BEGIN
        REWRITE(T,'PRINTER:');
        WRITE(T,CHR(25),CONTROL,NUMBER:PLACE);
        CLOSE(T)
      END; (* PRINT *)

    PROCEDURE PRREAL(* (NUMBER:REAL;CONTROL:STRING) *);
      BEGIN
        REWRITE(T,'PRINTER:');
        WRITE(T,CHR(25),CONTROL,NUMBER:12);
        CLOSE(T)
      END; (* PRREAL *)

    PROCEDURE FORMAT(* (CONTROL:STRING) *);
      BEGIN
        REWRITE(T,'PRINTER:');
        WRITELN(T,CHR(25),CONTROL);
        CLOSE(T)
      END; (* FORMAT *)


  (*** INITIALIZATION ***)

  BEGIN
  END.
```

Figure A-1. Library UNIT of printing routines.

```
(*$S++*)

(* These Assembler and Pascal routines are for use with the
   Interactive Structures AI13 analog-to-digital converter. *)

UNIT ADCRTNS; INTRINSIC CODE 16;

  INTERFACE

    PROCEDURE POKE (CHANNEL,MEMLOC:INTEGER);
      (* This routine retrieves a value from a specified CHANNEL and
         places it in a specified memory location. *)
    FUNCTION PEEK (MEMLOC:INTEGER):INTEGER;
      (* This routine returns a value from a specified memory location. *)
    FUNCTION GAINCODE (HIGH,LOW:REAL):INTEGER;
      (* This routine selects the smallest voltage range over which
         to sample. *)
    FUNCTION FRACTION (CODE:INTEGER):REAL;
      (* This routine returns,the fractional value of one voltage step of a
         specified voltage range. *)
    FUNCTION LOWEND (CODE:INTEGER):REAL;
      (* This routine returns the lowest voltage of a specified voltage
         range. *)

  IMPLEMENTATION

    PROCEDURE POKE(* (CHANNEL,MEMLOC:INTEGER) *);
      EXTERNAL;

    FUNCTION PEEK(* (MEMLOC:INTEGER):INTEGER *);
      EXTERNAL;

    FUNCTION GAINCODE(* (HIGH,LOW:REAL):INTEGER *);
      BEGIN
        IF LOW < 0 THEN
          BEGIN
            IF (HIGH <= 0.1) AND (LOW >= -0.1) THEN
              GAINCODE:=7
            ELSE
              BEGIN
                IF (HIGH <= 0.5) AND (LOW >= -0.5) THEN
                  GAINCODE:=6
                ELSE
                  BEGIN
                    IF (HIGH <= 1.0) AND (LOW >= -1.0) THEN
                      GAINCODE:=5
                    ELSE
                      GAINCODE:=4
                  END (* ELSE *)
              END (* ELSE *)
          END (* IF *)
        ELSE
          BEGIN
            IF HIGH <= 0.1 THEN
              GAINCODE:=3
            ELSE
              BEGIN
                IF HIGH <= 0.5 THEN
                  GAINCODE:=2
                ELSE
                  BEGIN
                    IF HIGH <= 1.0 THEN
                      GAINCODE:=1
                    ELSE
                      GAINCODE:=0
                  END (* ELSE *)
              END (* ELSE *)
          END (* ELSE *)
      END; (* GAINCODE *)
```

Figure A-2. Library UNIT written for Interactive Structures AI13, ADC.

```
  FUNCTION FRACTION(* (CODE:INTEGER):REAL *);
    BEGIN
      IF CODE=0 THEN
        FRACTION:=1.22070E-3
      ELSE IF CODE IN [1,6] THEN
        FRACTION:=2.44141E-4
      ELSE IF CODE=2 THEN
        FRACTION:=1.22070E-4
      ELSE IF CODE=3 THEN
        FRACTION:=2.44141E-5
      ELSE IF CODE=4 THEN
        FRACTION:=2.44141E-3
      ELSE IF CODE=5 THEN
        FRACTION:=4.88281E-4
      ELSE IF CODE=7 THEN
        FRACTION:=4.88281E-5
    END; (* FRACTION *)

  FUNCTION LOWEND(* (CODE:INTEGER):REAL *);
    BEGIN
      IF CODE IN [0..3] THEN
        LOWEND:=0
      ELSE IF CODE=4 THEN
        LOWEND:=-5
      ELSE IF CODE=5 THEN
        LOWEND:=-1
      ELSE IF CODE=6 THEN
        LOWEND:=-0.5
      ELSE IF CODE=7 THEN
        LOWEND:=-0.1
    END; (* LOWEND *)


(*** INITIALIZATION ***)

BEGIN
END.
```

Figure A-2. (continued).

```
;-----------------------------------------------------------------
;pops 16 bit argument
;
          .NOMACROLIST
          .MACRO POP16
          PLA
          STA %1
          PLA
          STA %1+1
          .ENDM

;pops 8 bit agrument & discards MSB
;
          .MACRO POP8
          PLA
          STA %1
          PLA
          .ENDM

          .PROC POKE,2     ;two words of parameters
;-----------------------------------------------------------------
;PROCEDURE POKE (CHANNEL,MEMLOC:INTEGER);
;
;-----------------------------------------------------------------
RETURN    .EQU 0           ;temporary return address
MEMLOC    .EQU 2           ;temporary MEMLOC address
CHANNEL   .EQU 4           ;temporary CHANNEL address
AI13      .EQU -16224.     ;starting address of AI13

                           ;place paremeters in PAGE 0 locations
          POP16 RETURN     ;save Pascal return address at 0,1
          POP16 MEMLOC     ;places MEMLOC at address 2,3
          POP8  CHANNEL    ;LSB of CHANNEL at 4
          LDY #0
          LDA CHANNEL      ;start conversion
          STA AI13         ;store LSB at -16224 HEX
          PHA              ;delay 7 usec to complete 4 MSB
          PLA
          LDA AI13+1       ;get MSB
          AND #0F          ;mask off high bits
          STA @MEMLOC,Y    ;store MSB at MEMLOC+1
          INY
          LDA AI13         ;get LSB
          STA @MEMLOC,Y
          LDA RETURN+1     ;restore Pascal return address
          PHA
          LDA RETURN
          PHA
          RTS              ;return to Pascal caller
;-----------------------------------------------------------------
          .END             ;end of assembly
```

Figure A-3. PROCEDURE POKE, which is linked to UNIT ADCRTNS.

```
;-------------------------------------------------------------
;pops 16 bit argument
;
              .NOMACROLIST
              .MACRO POP16
              PLA
              STA %1
              PLA
              STA %1+1
              .ENDM

              .FUNC PEEK,1  ;one word of parameters
;-------------------------------------------------------------
;FUNCTION PEEK (MEMLOC:INTEGER):INTEGER;
;
;-------------------------------------------------------------
RETURN        .EQU 0        ;temporary return address
MEMLOC        .EQU 2        ;temporary memory location

               POP16 RETURN ;save Pascal return address
               PLA          ;discard four byte stack bias
               PLA          ;only in case of FUNCTION
               PLA
               PLA
               POP16 MEMLOC ;get MEMLOC
               LDA #0       ;initialize A-index to zero
               PHA          ;MSB of result
               TAY          ;for indirect addressing
               LDA @MEMLOC,Y;retrieve result
               PHA          ;LSB of result
               LDA RETURN+1 ;restore Pascal return address
               PHA
               LDA RETURN
               PHA
               RTS
;-------------------------------------------------------------
              .END          ;end of assembly
```

Figure A-4. FUNCTION PEEK, which is linked to UNIT ADCRTNS.

```
(*$S++*)

(* These assembler routines use the THUNDERCLOCK to retrieve the time in
   an ASCII string or integer variables; a routine is also included for
   setting the time.  There is also a routine for calculating the temp-
   erature in degrees Celsius using an OMEGA thermistor composite probe
   and homemade voltage supply.  The thermistor probe can measure the
   temperature to within plus or minus 0.15C from -50.0C to +50.0C. *)

UNIT TIMETEMP; INTRINSIC CODE 17 DATA 18;

  INTERFACE

    USES ADCRTNS;

    VAR MO,DT,DY,HR,MI,SE:INTEGER;
        TIME:STRING;

    PROCEDURE SETTIME (MONTH,DAY,DATE,HOUR,MINUTE,SECOND:INTEGER);
      (* This routine sets the THUNDERCLOCK time variables to their
         respective values; the write-protect switch must be enabled. *)
    PROCEDURE TIMEASCII (VAR TIME:STRING;FMT:INTEGER);
      (* This routine returns the time in an ASCII string; 12 OR 24 HOUR
         format may be selected. *)
    PROCEDURE TIMEVALUES (VAR MONTH,DATE,DAY,HOUR,MINUTE,SECOND:INTEGER);
      (* This routine returns the time as integer values in the six
         respective arguments. *)
    FUNCTION CFIND (CODE,CHANNEL,MEMLOC:INTEGER):REAL;
      (* This routine calls CCALC and computes the temperatures in degrees
         Celsius until it is within plus or minus 0.15C. *)

  IMPLEMENTATION

    PROCEDURE SETTIME(* (MONTH,DAY,DATE,HOUR,MINUTE,SECOND:INTEGER) *);
    EXTERNAL;

    PROCEDURE TIMEASCII(* (VAR TIME:STRING;FMT:INTEGER) *);
    EXTERNAL;

    PROCEDURE TIMEVALUES(* (VAR MONTH,DATE,DAY,HOUR,MINUTE,SECOND:INTEGER) *);
    EXTERNAL;

    FUNCTION CCALC (CODE,CHANNEL,MEMLOC:INTEGER):REAL;
      (* This routine calculates the temperature in degrees Celsius. *)
      VAR VOUT:REAL;
      BEGIN
        POKE(CHANNEL+(CODE*16),MEMLOC);
        VOUT:=(FRACTION(CODE)*((PEEK(MEMLOC)*256)+PEEK(MEMLOC+1)))+
          LOWEND(CODE);
        CCALC:=(VOUT-1.4245)/1.95702E-2
      END; (* CCALC *)

    FUNCTION CFIND(* (CODE,CHANNEL,MEMLOC:INTEGER):REAL *);
      VAR DEG1,DEG2:REAL;
      BEGIN
        (*$R ADCRTNS *)
        REPEAT
          DEG1:=CCALC(CODE,CHANNEL,MEMLOC);
          DEG2:=CCALC(CODE,CHANNEL,MEMLOC);
        UNTIL ABS(DEG1-DEG2) <= 0.15;
        CFIND:=(DEG1+DEG2)/2
      END; (* CFIND *)


  (*** INITIALIZATION ***)

  BEGIN
    (*$N+*)
  END.
```

Figure A-5. Library UNIT of routines for reading time and temperature.

```
(*$S++*)
(* This intrinsic unit is a composite of routines from "Pascal Programs
   For Games and Graphics", by Tom Swan(1983)188-214 and, "Advanced UCSD
   Pascal Programming Techniques",by Eliakim Willner and Barry Demchak
   (1985)203-210. *)

UNIT INPTRTNS; INTRINSIC CODE 19;

  INTERFACE

    USES TRANSCEND;

    TYPE CHARSET = SET OF CHAR;

    PROCEDURE CLRSCRN;
      (* Clears the screen and places the cursor at 0,0. *)
    PROCEDURE PROMPTAT (X,Y:INTEGER; MESSAGE:STRING);
      (* Displays a message at screen position X,Y. *)
    PROCEDURE GETACHAR (VAR CH:CHAR; VALID:CHARSET);
      (* Assigns a character from the valid set of char-
         acters to CH; lowercase characters are automatically
         shifted up. *)
    FUNCTION GETINT (LOVAL,HIVAL:INTEGER):INTEGER;
      (* Returns an integer between LOVAL and HIVAL;
         validates input once <RETURN> is pressed. *)
    FUNCTION GETREAL:REAL;
      (* Returns a real number by validating input on a character
         to character basis; long integers beyond seven digits
         and exponents less than -37 or greater than 37 are not
         accepted. *)


  IMPLEMENTATION

    PROCEDURE BELL;
      BEGIN
        WRITE(CHR(7))
      END; (* BELL *)

    PROCEDURE CLRSCRN;
      BEGIN
        WRITE(CHR(12))
      END;

    PROCEDURE PROMPTAT(* (X,Y:INTEGER; MESSAGE:STRING) *);
      BEGIN
        GOTOXY(X,Y);
        WRITE(MESSAGE)
      END;

    PROCEDURE BACKSPACE (COUNT:INTEGER);
      VAR I:INTEGER;
      BEGIN
        FOR I:= 1 TO COUNT DO
          WRITE(CHR(8));
        FOR I:=1 TO COUNT DO
          WRITE(' ');
        FOR I:=1 TO COUNT DO
          WRITE(CHR(8))
      END; (* BACKSPACE *)

    PROCEDURE SHIFTUP (VAR CH:CHAR);
      BEGIN
        IF CH IN ['a'..'z'] THEN
          CH:=CHR(ORD(CH)-32)
      END; (* SHIFTUP *)

    PROCEDURE GETACHAR(* (VAR CH:CHAR; VALID:CHARSET) *);
      VAR OK:BOOLEAN;
      BEGIN
        REPEAT
          READ(KEYBOARD,CH);
          SHIFTUP(CH);
          IF EOLN(KEYBOARD) THEN
            CH:=CHR(13);
          OK:=(CH IN VALID);
          IF NOT OK THEN
            BELL;
        UNTIL OK
      END; (* GETACHAR *)
```

Figure A-6. Library UNIT of input validation and console routines.

```
FUNCTION CH2DIGIT (CH:CHAR):INTEGER;
  BEGIN
    CH2DIGIT:=ORD(CH)-ORD('0')
  END; (* CH2DIGIT *)

FUNCTION GETINT(* (LOVAL,HIVAL:INTEGER):INTEGER *);
  VAR GOTSIGN,MINUS,DOOVER:BOOLEAN;
      VALUE,NUMCHARS:INTEGER;
      CH:CHAR;
  BEGIN
    REPEAT
      NUMCHARS:=0;
      VALUE:=0;
      MINUS:=FALSE;
      GOTSIGN:=FALSE;
      REPEAT
        GETACHAR(CH,['0'..'9','-',CHR(8),CHR(13)]);
        IF CH IN ['0'..'9'] THEN
          BEGIN
            CASE MINUS OF
              TRUE :VALUE:=(VALUE*10)-CH2DIGIT(CH);
              FALSE:VALUE:=(VALUE*10)+CH2DIGIT(CH)
            END;
            GOTSIGN:=TRUE;
            WRITE(CH);
            NUMCHARS:=SUCC(NUMCHARS)
          END;
        IF CH = CHR(8) THEN
          BEGIN
            BACKSPACE(NUMCHARS);
            DOOVER:=TRUE
          END
        ELSE IF CH = '-' THEN
          BEGIN
            IF GOTSIGN THEN BELL
            ELSE
              BEGIN
                MINUS:=TRUE;
                GOTSIGN:=TRUE;
                WRITE(CH);
                NUMCHARS:=SUCC(NUMCHARS)
              END
          END
        ELSE IF CH = CHR(13) THEN
          BEGIN
            IF (VALUE > HIVAL) OR (VALUE < LOVAL) THEN
              BEGIN
                BELL;
                BACKSPACE(NUMCHARS);
                DOOVER:=TRUE
              END
            ELSE DOOVER:=FALSE
          END;
      UNTIL CH IN [CHR(8),CHR(13)];
    UNTIL NOT DOOVER;
    GETINT:=VALUE
  END; (* GETINT *)

FUNCTION GETREAL(* :REAL *);
  VAR GOTSIGN,GOTDEC,MINUS,DOOVER:BOOLEAN;
      EXPO,NUMCHARS,NUMRIGHT:INTEGER;
      VALUE:REAL;
      CH:CHAR;
  BEGIN
    (*$R TRANSCEND*)
    REPEAT
      NUMCHARS:=0;
      NUMRIGHT:=0;
      VALUE:=0.0;
      MINUS:=FALSE;
      GOTSIGN:=FALSE;
      GOTDEC:=FALSE;
      REPEAT
        GETACHAR(CH,['0'..'9','-','.','E','e',CHR(8),CHR(13)]);
        IF NUMCHARS = 8 THEN
          BEGIN
            BELL;
            CH:=CHR(8)
          END;
```

Figure A-6. (continued).

```
        IF CH IN ['E','e'] THEN
          BEGIN
            IF VALUE = 0.0 THEN
              BEGIN
                BELL;
                CH:=CHR(8)
              END
            ELSE
              BEGIN
                WRITE(CH);
                IF VALUE >= 1.0 THEN
                  EXPO:=GETINT(-37,37-TRUNC(LOG(VALUE)))
                ELSE
                  EXPO:=GETINT(1-TRUNC(LOG(VALUE))-37,37);
                IF EXPO < 0 THEN
                  VALUE:=VALUE*(1.0/PWROFTEN(-EXPO))
                ELSE
                  VALUE:=VALUE*PWROFTEN(EXPO);
                CH:=CHR(13)
              END
          END;
        IF CH = CHR(13) THEN
          DOOVER:=FALSE
        ELSE IF CH = CHR(8) THEN
          BEGIN
            BACKSPACE(NUMCHARS);
            DOOVER:=TRUE
          END
        ELSE IF CH IN ['0'..'9'] THEN
          BEGIN
            IF NOT GOTDEC THEN
              VALUE:=(VALUE*10.0)+CH2DIGIT(CH)
            ELSE
              BEGIN
                NUMRIGHT:=SUCC(NUMRIGHT);
                VALUE:=VALUE+((1.0/PWROFTEN(NUMRIGHT))*CH2DIGIT(CH))
              END;
            GOTSIGN:=TRUE;
            WRITE(CH);
            NUMCHARS:=SUCC(NUMCHARS)
          END
        ELSE IF CH = '-' THEN
          BEGIN
            IF GOTSIGN THEN BELL
            ELSE
              BEGIN
                MINUS:=TRUE;
                GOTSIGN:=TRUE;
                WRITE(CH);
                NUMCHARS:=SUCC(NUMCHARS)
              END
          END
        ELSE IF CH = '.' THEN
          BEGIN
            GOTSIGN:=TRUE;
            IF GOTDEC THEN BELL
            ELSE
              BEGIN
                GOTDEC:=TRUE;
                WRITE(CH);
                NUMCHARS:=SUCC(NUMCHARS)
              END
          END;
      UNTIL CH IN [CHR(8),CHR(13)];
    UNTIL NOT DOOVER;
    IF MINUS THEN
      GETREAL:=-VALUE
    ELSE
      GETREAL:=VALUE
  END; (* GETREAL *)

(*** INITIALIZATION ***)

BEGIN
  (*$N+*)
END.
```

Figure A-6. (continued).

```
(*$S++*)

(* These routines move parameter and cyclic voltammetric data
   to and from diskfile.  Since this UNIT uses the SEEK intrin-
   sic; it must appear in SYSTEM.LIBRARY. *)

UNIT FILERTNS; INTRINSIC CODE 26 DATA 27;

  INTERFACE

    USES TRANSCEND,INPTRTNS;

    TYPE VALUES = ARRAY [0..1] OF REAL;

         PARAM = RECORD                (* This is a variant record. *)
                   CASE INTEGER OF
                     0:(LTTRS:STRING);
                     1:(INTGR:INTEGER);
                     2:(DCML:REAL)
                   END; (* PARAM *)

         NODE = RECORD
                  VAL:VALUES;
                  NEXT:^NODE
                END; (* NODE *)

         LIST = ^NODE;

    VAR FOO:FILE OF PARAM;
        PTS:FILE OF VALUES;

    PROCEDURE PUSH (E,I:REAL; VAR L:LIST);
      (* This routine PUSHes a data pair on the top of the List. *)
    PROCEDURE CV2LIST (VAR L:LIST);
      (* This routine retrieves CV data from a diskfile and places it
         in a linked list. *)
    PROCEDURE CV2FILE (L:LIST);
      (* This routine transfers CV data to diskfile. *)
    PROCEDURE FILE2VAR (RECNUM:INTEGER; VAR P:PARAM);
      (* This routine transfers random file records to variables;
         the parameter file must be open. *)
    PROCEDURE VAR2FILE (RECNUM:INTEGER;P:PARAM);
      (* This routine transfers variables to random records in a file;
         the parameter file must be open. *)
    PROCEDURE CHAN2FILE (RECNUM,I:INTEGER);
      (* This routine places the channel numbers in the file. *)
    PROCEDURE FILE2CHAN (RECNUM:INTEGER; VAR CHAN:INTEGER);
      (* This routine retrieves a channel number and places it in CHAN. *)
    FUNCTION FILENAME (OLD:BOOLEAN; VAR SUFFIX:STRING):BOOLEAN;
      (* Prompt for an old or new file and try to open it, then return
         true if successfully opened. *)
    FUNCTION OPENCHAN:BOOLEAN;
      (* This routine opens the file which contains the channel numbers
         and returns true if the file already exists. *)
```

Figure A-7. Library UNIT of file routines for cyclic voltammetry.

```
IMPLEMENTATION

  PROCEDURE PUSH (* (E,I:REAL; VAR L:LIST) *);
  (* This routine pushes a data pair on top of the list. *)
    VAR TEMP:LIST;
    BEGIN
      NEW(TEMP);
      WITH TEMP^ DO BEGIN
        VAL[0]:=E;
        VAL[1]:=I;
        NEXT:=L
      END;
      L:=TEMP;GET(PTS)
    END;

  PROCEDURE CV2LIST (* (VAR L:LIST) *);
    VAR FRONT,TEMP:LIST;
    BEGIN
      IF L = NIL THEN BEGIN
        PUSH(PTS^[0],PTS^[1],L);FRONT:=L
      END;
      WHILE NOT EOF(PTS) DO BEGIN
        WHILE L^.NEXT <> NIL DO L:=L^.NEXT;
        NEW(TEMP);
        WITH TEMP^ DO BEGIN
          VAL[0]:=PTS^[0];
          VAL[1]:=PTS^[1];
          NEXT:=NIL
        END;
        L^.NEXT:=TEMP;GET(PTS)
      END;
      L:=FRONT
    END; (* CV2LIST *)

  PROCEDURE CV2FILE (* (L:LIST) *);
    BEGIN
      WHILE L <> NIL DO BEGIN
        PTS^[0]:=L^.VAL[0];
        PTS^[1]:=L^.VAL[1];
        PUT(PTS);
        L:=L^.NEXT
      END
    END; (* CV2FILE *)

  PROCEDURE FILE2VAR (* (RECNUM:INTEGER; VAR P:PARAM) *);
    BEGIN
      SEEK(FOO,RECNUM);GET(FOO);
      IF RECNUM IN [0..6] THEN
        P.INTGR:=FOO^.INTGR
      ELSE IF RECNUM IN [7..17] THEN
        P.DCML:=FOO^.DCML
      ELSE IF RECNUM IN [18..22] THEN
        P.LTTRS:=FOO^.LTTRS
    END; (* FILE2VAR *)

  PROCEDURE VAR2FILE (* (RECNUM:INTEGER;P:PARAM) *);
    BEGIN
      SEEK(FOO,RECNUM);
      IF RECNUM IN [0..6] THEN
        FOO^.INTGR:=P.INTGR
      ELSE IF RECNUM IN [7..17] THEN
        FOO^.DCML:=P.DCML
      ELSE IF RECNUM IN [18..22] THEN
        FOO^.LTTRS:=P.LTTRS;
      PUT(FOO)
    END; (* VAR2FILE *)
```

Figure A-7. (continued).

```
  PROCEDURE CHAN2FILE (* (RECNUM,I:INTEGER) *);
    BEGIN
      SEEK(FOO,RECNUM);
      FOO^.INTGR:=I;
      PUT(FOO)
    END; (* CHAN2FILE *)

  PROCEDURE FILE2CHAN (* (RECNUM:INTEGER;VAR CHAN:INTEGER) *);
    BEGIN
      SEEK(FOO,RECNUM);
      GET(FOO);
      CHAN:=FOO^.INTGR
    END; (* FILE2CHAN *)

  FUNCTION FILENAME (* (OLD:BOOLEAN;VAR SUFFIX:STRING):BOOLEAN *);
    VAR NAME:STRING;
        NUM:INTEGER;
        CH:CHAR;
    BEGIN
      (*$R INPTRTNS*)
      CLRSCRN;
      (*$I-*)                    (* Disable I/O checking *)
      PROMPTAT(0,2,'Enter the CV number <1..99>:');
      STR(GETINT(1,99),NAME);WRITELN;
      NAME:=CONCAT('#5:CV',NAME,SUFFIX);
      RESET(PTS,NAME);
      IF OLD AND (IORESULT <> 0) THEN BEGIN
        PROMPTAT(2,3,'File ');
        WRITELN(NAME,' does not exist!');
        FILENAME:=FALSE
      END
      ELSE IF OLD AND (IORESULT = 0) THEN
        FILENAME:=TRUE
      ELSE IF NOT OLD AND (IORESULT = 0) THEN BEGIN
        PROMPTAT(2,3,'File ');
        WRITELN(NAME,' already exists!');
        FILENAME:=FALSE;
        CLOSE(PTS)
      END
      ELSE IF NOT OLD AND (IORESULT <> 0) THEN BEGIN
        REWRITE(PTS,NAME);
        FILENAME:=TRUE
      END;
      (*$I+*)                    (* Enable I/O checking *)
      PROMPTAT(0,6,'Press <SPACE> to continue!');
      GETACHAR(CH,[' ']);WRITELN;
      SUFFIX:=NAME
    END; (* FILENAME *)

  FUNCTION OPENCHAN (* :BOOLEAN *);
    BEGIN
      (*$I-*)
      RESET(FOO,'#10:CHANNELS');
      IF IORESULT <> 0 THEN
        BEGIN
          REWRITE(FOO,'#10:CHANNELS');
          OPENCHAN:=FALSE
        END
      ELSE
        OPENCHAN:=TRUE;
      (*$I+*)
    END; (* OPENCHAN *)


(*** INITIALIZATION ***)

BEGIN
  (*$N+*)
END.
```

Figure A-7. (continued).

APPENDIX B

TEXT FILE LISTINGS OF CV PROGRAMS

```
(*$S++*)
PROGRAM INTERPAR;

(* This program presents the main menu and allows the user to select
   from its options.  The appropriate option is executed by CHAINing
   to its program code.  Once the option has finished executing, the
   program CHAINs back to the main menu. *)

USES CHAINSTUFF,ADCRTNS,TIMETEMP,TRANSCEND,INPTRTNS,FILERTNS;

SEGMENT FUNCTION MENU:CHAR;
(* This routine presents a menu of options to the user. *)
  VAR CH:CHAR;
  BEGIN
    (*$R INPTRTNS*)
    CLRSCRN;
    PROMPTAT(0,5,'Menu selections:');WRITELN;
    PROMPTAT(2,6,'C>hange input channels');WRITELN;
    PROMPTAT(2,7,'T>ime and temperature');WRITELN;
    PROMPTAT(2,8,'N>ew set of CV data');WRITELN;
    PROMPTAT(2,9,'G>raph a set of CV data');WRITELN;
    PROMPTAT(2,10,'S>mooth an original set of CV data');WRITELN;
    PROMPTAT(2,11,'L>ocate peaks in a CV');WRITELN;
    PROMPTAT(2,12,'F>it peaks in a CV');WRITELN;
    PROMPTAT(2,13,'D>econvolute a CV');WRITELN;
    PROMPTAT(2,14,'O>utput analysis of a CV');WRITELN;
    PROMPTAT(2,15,'Q>uit');WRITELN;
    PROMPTAT(0,17,'Please type:');
    GETACHAR(CH,['C','c','T','t','N','n','G','g','S','s','L','l',
      'F','f','D','d','O','o','Q','q']);
    WRITELN(CH);MENU:=CH
  END; (* MENU *)

SEGMENT PROCEDURE TANDT;
(* This routine displays the current time and temperature. *)
  VAR CCHAN:INTEGER;
      CH:CHAR;
  BEGIN
    (*$R INPTRTNS*)
    (*$R TIMETEMP*)
    CLRSCRN;
    PROMPTAT(0,5,'Temperature (C):');
    RESET(FOO,'#10:CHANNELS');
    FILE2CHAN(0,CCHAN);
    CLOSE(FOO);
    WRITELN(CFIND(0,CCHAN,4446));
    PROMPTAT(0,6,'Time              : ');
    TIMEASCII(TIME,12);WRITELN(TIME);
    PROMPTAT(0,11,'Press <SPACE> to continue!');
    GETACHAR(CH,[' ']);WRITELN
  END; (* TANDT *)

SEGMENT PROCEDURE SETCHAN;
(* This routine prompts for channel numbers and calls CHAN2FILE to place
   them in a file. *)
  VAR CH:CHAR;
  BEGIN
    (*$R INPTRTNS*)
    CLRSCRN;
    PROMPTAT(0,5,'Select data channels <0..15>');WRITELN;
    PROMPTAT(2,6,'Temperature:');CHAN2FILE(0,GETINT(0,15));WRITELN;
    PROMPTAT(2,7,'Trigger     :');CHAN2FILE(1,GETINT(0,15));WRITELN;
    PROMPTAT(2,8,'Current     :');CHAN2FILE(2,GETINT(0,15));WRITELN;
    PROMPTAT(2,9,'Potential   :');CHAN2FILE(3,GETINT(0,15));WRITELN;
    PROMPTAT(0,11,'Press <SPACE> to continue!');
    GETACHAR(CH,[' ']);WRITELN
  END; (* SETCHAN *)
```

Figure B-1. The main menu program for cyclic voltammetry.

```
SEGMENT PROCEDURE CHECK;
(* This routine checks to see if CHANNEL file exists and takes the
   appropriate action if it doesn't. *)
  BEGIN
    IF NOT OPENCHAN THEN
      BEGIN
        SETCHAN;
        CLOSE(FOO,LOCK)
      END
    ELSE CLOSE(FOO)
  END; (* CHECK *)

SEGMENT PROCEDURE OPTION;
(* This routine processes a selection from the MENU. *)
  VAR CH:CHAR;
  BEGIN
    REPEAT
      CH:=MENU;
      IF CH = 'C' THEN BEGIN
        IF NOT OPENCHAN THEN BEGIN
            SETCHAN;CLOSE(FOO,LOCK)
          END
        ELSE BEGIN
            SETCHAN;CLOSE(FOO)
          END
      END
      ELSE IF CH = 'T' THEN BEGIN
        CHECK;TANDT
      END
      ELSE IF CH = 'N' THEN BEGIN
        CHECK;SETCHAIN('#10:CVDATA');EXIT(OPTION)
      END
      ELSE IF CH = 'G' THEN BEGIN
        SETCHAIN('#10:GRAPHCV');EXIT(OPTION)
      END
      ELSE IF CH = 'S' THEN BEGIN
        SETCHAIN('#10:CVSMOOTH');EXIT(OPTION)
      END
      ELSE IF CH = 'L' THEN BEGIN
        SETCHAIN('#10:PEAKLOC');EXIT(OPTION)
      END
      ELSE IF CH = 'F' THEN BEGIN
        SETCHAIN('#10:PEAKFIT');EXIT(OPTION)
      END
      ELSE IF CH = 'D' THEN BEGIN
        SETCHAIN('#10:DECONVOL');EXIT(OPTION)
      END
      ELSE IF CH = 'O' THEN BEGIN
        SETCHAIN('#10:CVOUTPUT');EXIT(OPTION)
      END;
    UNTIL CH = '0'
  END; (* OPTION *)


(*** MAIN ***)

BEGIN
  (*$N+*)
  OPTION
END.
```

Figure B-1. (continued).

```
(*$S++*)
PROGRAM CVDATA;

(* In this program, cyclic voltammetric data are acquired at
   a precalculated, fixed delay rate.  The data are placed in a
   3,746 byte unused area of memory at location 4446.  The binary
   data are retrieved, converted, corrected, and placed on diskfile
   for subsequent calculations.  Some of the parameter data are
   placed on diskfile also. *)

USES CHAINSTUFF,TURTLEGRAPHICS,ADCRTNS,
     TIMETEMP,TRANSCEND,INPTRTNS,FILERTNS;

(* None of the code in TURTLEGRAPHICS is actually called by CVDATA,
   but it frees the 3,746 bytes of user RAM below Hi-Res Page 1 where
   TAKEDATA places the binary data for a cyclic voltammogram. *)

VAR NAME:STRING;
    TCHAN,CCHAN,ICHAN,ECHAN:INTEGER;
    START,TITLE,PAIRS,DEGREE,ECODE,ICODE,SCRATE,SENS,DIREC:PARAM;

                                    (* PARAM is declared in FILERTNS *)

  PROCEDURE TAKEDATA (PAIRS,DELAY1,DELAY2,DELAY3,TCHAN,ICHAN,
                      ECHAN,MEMLOC:INTEGER);
    EXTERNAL;

  SEGMENT FUNCTION SELCODE:INTEGER;
  (* This routine prompts for the voltage range on the current channel;
     the most sensitive range should be selected. *)
    BEGIN
      PROMPTAT(10,8,' GAINCODE       BIPOLAR RANGE');WRITELN;
      PROMPTAT(10,9,'--------------------------------');WRITELN;
      PROMPTAT(10,10,'     4             -5 <--> +5V');WRITELN;
      PROMPTAT(10,11,'     5             -1 <--> +1V');WRITELN;
      PROMPTAT(10,12,'     6           -0.5 <--> +0.5V');WRITELN;
      PROMPTAT(10,13,'     7           -0.1 <--> +0.1V');WRITELN;
      PROMPTAT(2,15,'Current gaincode <4..7>:');SELCODE:=GETINT(4,7);
      WRITELN
    END; (* SELCODE *)
```

Figure B-2. The Pascal calling program for acquiring cyclic voltammetric data.

```
SEGMENT PROCEDURE CALCDELAY;
(* This routine checks the input and determines the highest number of
   data pairs that can be collected and still be within limits of time
   delay routine. *)
  CONST  D3=3.21318E-1; (* Time(sec) each delay count can generate *)
         D2=1.25513E-3;
         D1=4.88758E-6;
  VAR PLIMIT,NLIMIT,ONTIME,OFFTIME:REAL;
      I,DELAY3,DELAY2,DELAY1:INTEGER;
      CH:CHAR;
  BEGIN
    (*$R ADCRTNS*)
    (*$R INPTRTNS*)
    PROMPTAT(0,1,'Title of CV:');READLN(TITLE.LTTRS);
    PROMPTAT(2,2,'Sensitivity     (Amps/V):');SENS.DCML:=GETREAL;WRITELN;
    PROMPTAT(2,3,'Scan direction     (P/N):');
    GETACHAR(CH,['P','p','N','n']);WRITELN(CH);
    IF CH = 'P' THEN DIREC.LTTRS:='Positive'
    ELSE DIREC.LTTRS:='Negative';
    REPEAT
      FOR I:=4 TO 6 DO
        BEGIN
          PROMPTAT(26,I,'             ');WRITELN
        END;
      PROMPTAT(2,4,'High potential       (V):');PLIMIT:=GETREAL;WRITELN;
      PROMPTAT(2,5,'Low  potential       (V):');NLIMIT:=GETREAL;WRITELN;
      PROMPTAT(2,6,'Scan rate          (V/s):');SCRATE.DCML:=GETREAL;WRITELN;
      ONTIME:=2*(PLIMIT-NLIMIT)/SCRATE.DCML;    (* Duration of scan in sec *)
    UNTIL (ONTIME > 1.28727E-1) AND (ONTIME < 7.69109E4);
    PAIRS.INTGR:=935;
    WHILE (ONTIME-(PAIRS.INTGR*2.30694E-4)) < 0 DO
      PAIRS.INTGR:=PAIRS.INTGR-1;
    OFFTIME:=(ONTIME-(2.07234E-4*PAIRS.INTGR))/PAIRS.INTGR;
    DELAY3:=TRUNC(OFFTIME/D3)+1;
    DELAY2:=TRUNC(((OFFTIME)-((DELAY3-1)*D3))/D2)+1;
    DELAY1:=TRUNC(((OFFTIME)-(DELAY3*D3)-(DELAY2*D2)+3.22550E-1)/D1)+1;
    ICODE.INTGR:=SELCODE;
    ECODE.INTGR:=GAINCODE(PLIMIT,NLIMIT);
    ICHAN:=ICHAN+(ICODE.INTGR*16);
    ECHAN:=ECHAN+(ECODE.INTGR*16);
    DEGREE.DCML:=CFIND(0,CCHAN,4446);     (* First temperature reading *)
    PROMPTAT(0,17,'Set function switch to SCAN and press <SPACE>!');
    GETACHAR(CH,[' ']);WRITELN(CH);
    TAKEDATA(PAIRS.INTGR,DELAY1,DELAY2,DELAY3,TCHAN,ICHAN,ECHAN,4446);
    PROMPTAT(0,19,'Experiment complete!')
  END; (* CALCDELAY *)

SEGMENT PROCEDURE ENLIST (MEMLOC,POINTS:INTEGER; VAR L:LIST);
(* Transfers data to a linked list by inserting at the head of the list. *)
  VAR TEMP:LIST;
  BEGIN
    (*$R ADCRTNS*)
    WHILE POINTS <> 0 DO BEGIN
      NEW(TEMP);
      WITH TEMP^ DO BEGIN
        VAL[0]:=(PEEK(MEMLOC)*256)+PEEK(MEMLOC+1);
        VAL[1]:=(PEEK(MEMLOC+2)*256)+PEEK(MEMLOC+3);
        NEXT:=L
      END; (* WITH *)
      L:=TEMP;MEMLOC:=MEMLOC+4;POINTS:=POINTS-1
    END (* WHILE *)
  END; (* ENLIST *)
```

Figure B-2. (continued).

```
SEGMENT PROCEDURE CORRECT (S:REAL; VAR L:LIST);
(* This routine corrects the potential values based on the scan
   rate and the 85 usec delay between current and potential. *)
  CONST Q=8.50440E-5;
  VAR TEMP:LIST;
  BEGIN
    TEMP:=L;
    WHILE L <> NIL DO BEGIN
      IF L^.NEXT^.VAL[0] < L^.VAL[0] THEN
        L^.VAL[0]:=L^.VAL[0]-(Q*S)
      ELSE
        L^.VAL[0]:=L^.VAL[0]+(Q*S);
      L:=L^.NEXT
    END;
    L:=TEMP
  END; (* CORRECT *)

SEGMENT PROCEDURE BIN2REAL (CODE,I:INTEGER;FACTOR:REAL; VAR L:LIST);
(* Convert digits between 0 and 4095 to real numbers based on the voltage
   range and the sensitivity. *)
  VAR TEMP:LIST;
      F,LOW:REAL;
  BEGIN
    (*$R ADCRTNS*)
    F:=FRACTION(CODE);
    LOW:=LOWEND(CODE);
    TEMP:=L;
    WHILE L <> NIL DO BEGIN
      L^.VAL[I]:=(((L^.VAL[I])*F)+LOW)*FACTOR;
      L:=L^.NEXT
    END; (* WHILE *)
    L:=TEMP
  END; (* BIN2REAL *)

SEGMENT PROCEDURE DATA2FILE (S:STRING);
(* Place the original real, binary CV data and parameters on file. *)
  VAR INITE,PFLSLD:PARAM;
      CH:CHAR;
      L:LIST;
  BEGIN
    (*$R FILERTNS*)
    L:=NIL;
    IF FILENAME(FALSE,S) THEN BEGIN
      CLRSCRN;PFLSLD.LTTRS:='FFF';
      TIMEASCII(START.LTTRS,12);CALCDELAY;
      ENLIST(4446,PAIRS.INTGR,L);
      BIN2REAL(ECODE.INTGR,0,1,L);
      BIN2REAL(ICODE.INTGR,1,SENS.DCML,L);
      CORRECT(SCRATE.DCML,L);INITE.DCML:=L^.VAL[0];
      DEGREE.DCML:=(DEGREE.DCML+CFIND(0,CCHAN,4446))/2;
      CV2FILE(L);CLOSE(PTS,LOCK);
      DELETE(S,POS('ORIGREAL',S),8);
      REWRITE(FOO,CONCAT(S,'PARAM'));VAR2FILE(17,INITE);
      VAR2FILE(18,START);VAR2FILE(19,TITLE);VAR2FILE(20,DIREC);
      VAR2FILE(0,PAIRS);VAR2FILE(3,ICODE);VAR2FILE(4,ECODE);
      VAR2FILE(7,SENS);VAR2FILE(8,DEGREE);VAR2FILE(9,SCRATE);
      VAR2FILE(22,PFLSLD);CLOSE(FOO,LOCK);
      DELETE(S,POS('PARAM',S),5);DELETE(S,POS('#5:',S),3);
      S:=CONCAT(S,' is now on disk in drive 2!');
      PROMPTAT(0,21,S);WRITELN;
      PROMPTAT(0,23,'Press <SPACE> to continue!');
      GETACHAR(CH,[' ']);WRITELN
    END
  END; (* DATA2FILE *)


(*** MAIN ***)

BEGIN
  (*$N+*)
  SETCHAIN('#10:INTERPAR');
  RESET(FOO,'#10:CHANNELS');
  FILE2CHAN(0,CCHAN);FILE2CHAN(1,TCHAN);
  FILE2CHAN(2,ICHAN);FILE2CHAN(3,ECHAN);CLOSE(FOO);
  NAME:='ORIGREAL';
  DATA2FILE(NAME)
END.
```

Figure B-2. (continued).

```
;-------------------------------------------------------------------------
; This routine is the heart of the program for acquisition of cyclic
; voltammetric data.   The data are acquired at a precalculated, fixed
; delay rate using a triply nested software delay timer.  From an assembled
; listing, the acquisition time for a (current,potential) pair can be calc-
; ulated.  The digital data are placed in a prespecified location using in-
; direct addressing.  Since one knows the exact time between current and
; potential points, it is a simple matter to use this time and scan rate to
; correct the potentials to their true values.  In some cases the correc-
; tion may not be necessary if the correction is less than the value mul-
; tiplied by 1E-5.
;-------------------------------------------------------------------------
;POPS 16 BIT ARGUMENT
;
         .NOMACROLIST
         .MACRO POP16
         PLA
         STA %1
         PLA
         STA %1+1
         .ENDM

;POPS 8 BIT ARGUMENT & DISCARDS MSB
;
         .MACRO POP8
         PLA
         STA %1
         PLA
         .ENDM

     .PROC TAKEDATA,8 ;eight words of parameters
;-------------------------------------------------------------------------
;  PROCEDURE TAKEDATA (PAIRS,DELAY1,DELAY2,DELAY3,TCHAN,ICHAN,ECHAN,
;                      MEMLOC:INTEGER);
;-------------------------------------------------------------------------
RETURN   .EQU 0        ;temporary return address
MEMLOC   .EQU 2        ;temporary MEMLOC address
ECHAN    .EQU 4        ;temporary ECHANNEL address
ICHAN    .EQU 5        ;temporary ICHANNEL address
TCHAN    .EQU 6        ;temporary TCHANNEL address
DELAY3   .EQU 7        ;temporary delay time addresses
DELAY2   .EQU 8
DELAY1   .EQU 9
PAIRS    .EQU 0A       ;number of data pairs address
AI13     .EQU -16224.  ;starting address of AI13 for slot 2
AO03     .EQU -14592.  ;starting address of AO03 for slot 7

;-------------------------------------------------------------------------
;  PLACE PARAMETERS IN PAGE 0 LOCATIONS
;-------------------------------------------------------------------------
         POP16 RETURN ;save Pascal return address at 0,1
         POP16 MEMLOC ;place MEMLOC at 2,3
         POP8  ECHAN  ;place LSB of ECHAN at 4
         POP8  ICHAN  ;place LSB of ICHAN at 5
         POP8  TCHAN  ;place LSB of TCHAN at 6
         POP8  DELAY3 ;place LSB of DELAY3 at 7
         POP8  DELAY2 ;place LSB of DELAY2 at 8
         POP8  DELAY1 ;place LSB of DELAY1 at 9
         POP16 PAIRS  ;place number of points at 0A,0B

;-------------------------------------------------------------------------
;  SCAN INITIATOR (TRIGGER)
;-------------------------------------------------------------------------
         LDY TCHAN    ;place trigger channel in Y
         LDA #255.    ;place 5.0V binary equivalent in accumulator
         STA AO03,Y   ;send 5.0V to DAC output channel and initiate scan
```

Figure B-3. Assembly code, which is linked to PROGRAM CVDATA, and performs data acquisition.

```
;-------------------------------------------------------------------------
;  DATA ACQUISITION    ;CLOCK CYCLES/COMMENT
;-------------------------------------------------------------------------
;     (One CLOCK CYCLE=1.023 MHz)
;     (It takes 212 CLOCK CYCLES to acquire one pair of points)
;     (There are 87 CLOCK CYCLES between current and potential points)
;-------------------------------------------------------------------------
DATA      LDA ECHAN     ;3/start conversion on ECHANNEL
          STA AI13      ;4/
          LDX #08       ;41/generates 41 cycle delay for a sensitive range
LOOP1     DEX
          BNE LOOP1
          STA AI13      ;4/now perform conversion
          PHA           ;3/delay 7 cycles to complete 4 MSB
          PLA           ;4/
          LDA AI13+1    ;4/get MSB
          AND #0F       ;2/mask off high BITS
          LDY #0        ;2/initialize Y index to 0 for indirect addressing
          STA @MEMLOC,Y;6/place at specified address
          INY           ;2/add 1 to Y
          LDA AI13      ;4/get LSB
          STA @MEMLOC,Y;6/place at next address
          INY           ;2/add 1 to Y
          LDA ICHAN     ;3/start conversion on ICHANNEL
          STA AI13      ;4/
          LDX #08       ;41/generates 41 cycle delay for a sensitive range
LOOP2     DEX
          BNE LOOP2
          STA AI13      ;4/now perform conversion
          PHA           ;3/delay 7 cycle to complete 4 MSB
          PLA           ;4/
          LDA AI13+1    ;4/get MSB
          AND #0F       ;2/mask off high BITS
          STA @MEMLOC,Y;6/place at specified address
          INY           ;2/add 1 to Y
          LDA AI13      ;4/get LSB
          STA @MEMLOC,Y;6/place at next address
          CLC           ;2/clear CARRY to zero
          LDA 02        ;3/add 4 to MEMLOC
          ADC #04       ;2/this is a 16 bit addition
          STA 02        ;3/
          LDA 03        ;3/
          ADC #0        ;2/
          STA 03        ;3/
          SEC           ;2/set CARRY to 1
          LDA 0A        ;3/subtract 1 from PAIRS
          SBC #01       ;2/this is a 16 bit subtraction
          STA 0A        ;3/
          LDA 0B        ;3/
          SBC #0        ;2/
          STA 0B        ;3/
          ORA 0A        ;3/
          BEQ OUT       ;2,3/branch to out when PAIRS=0
```

Figure B-3. (continued).

```
;-------------------------------------------------------------------------
;  TIME DELAY ROUTINE
;-------------------------------------------------------------------------
;     (This routine was taken in part from, "6502 Software Design",
;      by Leo J. Scanlon(1980)83-91 )
;-------------------------------------------------------------------------
          LDX DELAY1    ;X contains 5-clock-cycle usec count
          LDY DELAY2    ;Y contains 1284-clock-cycle count
          LDA DELAY3    ;A contains 328,708-clock-cycle count
WAIT      DEX           ;minimum delay is 23.4604 usec, (X=Y=A=1)
          BNE WAIT      ;loop until X=0
          DEY
          BNE WAIT      ;loop until X=Y=0
          SBC #01       ;maximum delay is 82.257340 sec, (X=Y=A=255)
          BNE WAIT      ;loop until X=Y=A=0
          BEQ DATA      ;end of TIME DELAY ROUTINE

;-------------------------------------------------------------------------
;  SCAN TERMINATOR
;-------------------------------------------------------------------------
OUT       LDY TCHAN     ;place trigger channel in Y
          LDA #128.     ;place 0.0V binary equivalent in accumulator
          STA A003,Y    ;send 0.0V to DAC output channel and terminate scan

;-------------------------------------------------------------------------
          LDA RETURN+1 ;restore Pascal return address
          PHA
          LDA RETURN
          PHA
          RTS           ;return to Pascal caller
         .END           ;end of assembly
```

Figure B-3. (continued).

```
(*$S++*)
PROGRAM GRAPHCV;

(* This program presents options to graph an original, smoothed,
   deconvoluted, or combined set of CV data.   An option is also
   included for dumping the graph on the screen to the printer. *)

USES CHAINSTUFF,ADCRTNS,TRANSCEND,INPTRTNS,FILERTNS,TURTLEGRAPHICS,
     PRNTRTNS;

VAR HII,LOI,HIE,LOE:REAL;
    HEAP:^INTEGER;
    M:CHAR;

SEGMENT FUNCTION MENU:CHAR;
(* This routine presents a menu of graphing options. *)
  VAR CH:CHAR;
  BEGIN
    (*$R INPTRTNS*)
    CLRSCRN;
    PROMPTAT(0,5,'Graphing selections:');WRITELN;
    PROMPTAT(2,6,'O>riginal set of CV data');WRITELN;
    PROMPTAT(2,7,'S>moothed set of CV data');WRITELN;
    PROMPTAT(2,8,'D>econvoluted set of CV data');WRITELN;
    PROMPTAT(2,9,'C>ombination of deconvoluted and subtracted CV data');
    WRITELN;PROMPTAT(2,10,'R>eturn to main menu');WRITELN;
    PROMPTAT(0,12,'Please type:');
    GETACHAR(CH,['O','o','S','s','D','d','C','c','R','r']);WRITELN(CH);
    MENU:=CH
  END; (* MENU *)

SEGMENT FUNCTION GETVALUE (I,HIORLO:INTEGER;L:LIST):REAL;
(* This routine saves the high(1) or low(0) value from a list of
   data; I=0 refers to the potential field and I=1 refers to the
   current field. *)
  VAR TEMP1,TEMP2:LIST;
  BEGIN
    TEMP1:=L;
    TEMP2:=L;
    WHILE TEMP2 <> NIL DO BEGIN
      CASE HIORLO OF
        0:IF TEMP1^.VAL[I] >= TEMP2^.VAL[I] THEN TEMP1:=TEMP2;
        1:IF TEMP1^.VAL[I] <= TEMP2^.VAL[I] THEN TEMP1:=TEMP2
      END; (* CASE *)
      TEMP2:=TEMP2^.NEXT
    END;
    GETVALUE:=TEMP1^.VAL[I]
  END; (* GETVALUE *)

SEGMENT PROCEDURE PLOTDOT (X,Y:REAL);
(* This routine places a dot at the X,Y coordinates of L. *)
  VAR Q,R:INTEGER;
      DOT:BOOLEAN;
  BEGIN
    Q:=ROUND((ABS(X-HIE)/(HIE-LOE))*278);
    R:=ROUND((ABS(Y-HII)/(HII-LOI))*190);
    DRAWBLOCK(DOT,1,Q,Q,1,1,Q,R,15)
  END; (* PLOTDOT *)

SEGMENT PROCEDURE CV2SCREEN (L:LIST);
(* This routine plots a set of CV of data on the screen. *)
  VAR YCOORD:INTEGER;
  BEGIN
    HII:=GETVALUE(1,1,L);LOI:=GETVALUE(1,0,L);
    HIE:=GETVALUE(0,1,L);LOE:=GETVALUE(0,0,L);
    YCOORD:=PEEK(-16206);      (* Initialize Hi-Resolution graphics *)
    INITTURTLE;
    YCOORD:=ROUND((HII*191)/(HII-LOI));
    MOVETO(246,YCOORD-9);WSTRING('-E ->');
    MOVETO(0,183);WCHAR(CHR(105));MOVETO(7,179);WCHAR(CHR(99));
    MOVETO(0,4);WCHAR(CHR(105));MOVETO(7,0);WCHAR(CHR(97));
    MOVETO(279,YCOORD);PENCOLOR(WHITE);MOVETO(0,YCOORD);
    WHILE L <> NIL DO BEGIN
      PLOTDOT(L^.VAL[0],L^.VAL[1]);
      L:=L^.NEXT
    END
  END; (* CV2SCREEN *)
```

Figure B-4. Graphics program used to print and display voltammograms.

```
SEGMENT PROCEDURE DATAPLOT (S:STRING; SUB:BOOLEAN);
(* This routine retrieves a set of CV data and plots it. *)
  VAR CH:CHAR;
      L:LIST;
  BEGIN
    (*$R TURTLEGRAPHICS*)
    L:=NIL;
    IF FILENAME(TRUE,S) THEN BEGIN
      CV2LIST(L);CLOSE(PTS);
      PROMPTAT(0,8,'Press <SPACE> to view the graph and');WRITELN;
      PROMPTAT(5,9,'<SPACE> when finished viewing!');
      GETACHAR(CH,[' ']);WRITE(CH);
      CV2SCREEN(L);
      IF SUB THEN BEGIN
        DELETE(S,POS('DECONVOL',S),8);
        RELEASE(HEAP);L:=NIL;
        RESET(PTS,CONCAT(S,'SUB'));CV2LIST(L);CLOSE(PTS);
        WHILE L <> NIL DO BEGIN
          PLOTDOT(L^.VAL[0],L^.VAL[1]);L:=L^.NEXT
        END
      END;
      GETACHAR(CH,[' ']);WRITE(CH);
      PROMPTAT(0,11,'Do you want the graph printed <Y,N>:');
      GETACHAR(CH,['Y','y','N','n']);WRITELN(CH);
      IF CH = 'Y' THEN FORMAT('H')
    END
  END; (* DATAPLOT *)


(*** MAIN ***)

BEGIN
  (*$N+*)
  SETCHAIN('#10:INTERPAR');
  M:=MENU;MARK(HEAP);
  CASE M OF
    'O':DATAPLOT('ORIGREAL',FALSE);
    'S':DATAPLOT('SMTH',FALSE);
    'D':DATAPLOT('DECONVOL',FALSE);
    'C':DATAPLOT('DECONVOL',TRUE);
    'R':EXIT(PROGRAM)
  END
END.
```

Figure B-4. (continued).

```
(*$S++*)
PROGRAM PEAKLOC;

(* This program directs the user to select peak locations in a CV.
   An arrow appears on the graph of smoothed or deconvoluted data
   near the point of highest cathodic or anodic current.  The user
   is directed to press the < -> > key until the arrow is as close
   to the center of the peak as possible.  When <RETURN> is pressed
   the arrow's position is set as the point of highest or lowest
   current in the list.  If a DECONvoluted curve exists, then it is
   chosen; otherwise, the SMOOTHed curve is chosen for PEAK LOCating. *)

USES CHAINSTUFF,ADCRTNS,TRANSCEND,INPTRTNS,FILERTNS,TURTLEGRAPHICS;

VAR HII,LOI,HIE,LOE:REAL;

SEGMENT PROCEDURE INSTRUCT;
(* This routine presents instructions for locating the center of
   a peak. *)
  VAR CH:CHAR;
  BEGIN
    (*$R INPTRTNS*)
    CLRSCRN;
    PROMPTAT(0,2,'A graph of the CV data you have chosen will ap-');WRITELN;
    PROMPTAT(2,3,'pear with an ARROW pointing near the highest');WRITELN;
    PROMPTAT(2,4,'cathodic point or anodic point of the curve.');WRITELN;
    PROMPTAT(2,5,'Press ( -> ) to advance the ARROW to the center');WRITELN;
    PROMPTAT(2,6,'of the peak and <RETURN> when you are finished.');WRITELN;
    PROMPTAT(2,7,'Anytime peaks are relocated in a CV, the peaks ');WRITELN;
    PROMPTAT(2,8,'must be refitted.');WRITELN;
    PROMPTAT(0,10,'Press <SPACE> when you are ready.');
    GETACHAR(CH,[' ']);WRITELN
  END; (* INSTRUCT *)

SEGMENT FUNCTION INTPOS (I:INTEGER;NUM:REAL;L:LIST):INTEGER;
(* This routine returns the integer position of a current(1) or
   or potential(0) value in a list of CV data.  In this program
   it works in conjunction with GETVALUE. *)
  VAR P:INTEGER;
  BEGIN
    P:=0;
    WHILE L <> NIL DO BEGIN
      IF L^.VAL[I] = NUM THEN BEGIN
        INTPOS:=P;EXIT(INTPOS)
      END
      ELSE BEGIN
        P:=P+1;
        L:=L^.NEXT
      END
    END;
    INTPOS:=P
  END; (* INTPOS *)

SEGMENT FUNCTION GETVALUE (I,HIORLO:INTEGER;L:LIST):REAL;
(* This routine saves the high(1) or low(0) value from a list of
   data; I=0 refers to the potential field and I=1 refers to the
   current field. *)
  VAR TEMP1,TEMP2:LIST;
  BEGIN
    TEMP1:=L;
    TEMP2:=L;
    WHILE TEMP2 <> NIL DO BEGIN
      CASE HIORLO OF
        0:IF TEMP1^.VAL[I] >= TEMP2^.VAL[I] THEN TEMP1:=TEMP2;
        1:IF TEMP1^.VAL[I] <= TEMP2^.VAL[I] THEN TEMP1:=TEMP2
      END; (* CASE *)
      TEMP2:=TEMP2^.NEXT
    END;
    GETVALUE:=TEMP1^.VAL[I]
  END; (* GETVALUE *)

SEGMENT PROCEDURE PLOTDOT (X,Y,BORW:INTEGER);
(* This routine places a dot at coordinates X,Y in black or white. *)
  VAR DOT:BOOLEAN;
  BEGIN
    DRAWBLOCK(DOT,1,0,0,1,1,X,Y,BORW)
  END; (* PLOTDOT *)
```

Figure B-5. User-interactive, peak locating program.

```
SEGMENT PROCEDURE ARROW (REMOVE:BOOLEAN;L:LIST);
(* This routine draws an arrow to point to the coordinate pairs
   pointed to by L; if REMOVE is TRUE, then the arrow at that
   point is removed from the graph. *)
  VAR XCOORD,YCOORD,COLOR,I,J,K:INTEGER;
  BEGIN
    (*$R PLOTDOT*)
    XCOORD:=ROUND((ABS(L^.VAL[0]-HIE)/(HIE-LOE))*278);
    YCOORD:=ROUND((ABS(L^.VAL[1]-HII)/(HII-LOI))*190);
    IF REMOVE THEN COLOR:=0
    ELSE COLOR:=15;
    J:=0;
    IF L^.VAL[1] < 0 THEN BEGIN
      FOR I:=YCOORD-4 DOWNTO YCOORD-11 DO
        PLOTDOT(XCOORD,I,COLOR);
      FOR I:=YCOORD-5 DOWNTO YCOORD-6 DO BEGIN
        FOR K:=XCOORD-J TO XCOORD+J DO
          PLOTDOT(K,I,COLOR);
        J:=J+1
      END
    END
    ELSE BEGIN
      FOR I:=YCOORD+4 TO YCOORD+11 DO
        PLOTDOT(XCOORD,I,COLOR);
      FOR I:=YCOORD+5 TO YCOORD+6 DO BEGIN
        FOR K:=XCOORD-J TO XCOORD+J DO
          PLOTDOT(K,I,COLOR);
        J:=J+1
      END
    END
  END; (* ARROW *)

SEGMENT PROCEDURE CV2SCREEN (L:LIST);
(* This routine plots the CV on the screen. *)
  VAR XCOORD,YCOORD:INTEGER;
  BEGIN
    (*$R TURTLEGRAPHICS*)
    (*$R PLOTDOT*)
    YCOORD:=PEEK(-16206);      (* Initialize Hi-Resolution graphics *)
    INITTURTLE;
    YCOORD:=ROUND((HII*191)/(HII-LOI));
    MOVETO(246,YCOORD-9);WSTRING('-E ->');
    MOVETO(222,9);WSTRING('Press');MOVETO(194,0);
    WSTRING('-> or RETURN');MOVETO(243,183);WSTRING('Point');
    MOVETO(0,183);WCHAR(CHR(105));MOVETO(7,179);WCHAR(CHR(99));
    MOVETO(0,4);WCHAR(CHR(105));MOVETO(7,0);WCHAR(CHR(97));
    MOVETO(279,YCOORD);PENCOLOR(WHITE);MOVETO(0,YCOORD);
    WHILE L <> NIL DO BEGIN
      XCOORD:=ROUND((ABS(L^.VAL[0]-HIE)/(HIE-LOE))*278);
      YCOORD:=ROUND((ABS(L^.VAL[1]-HII)/(HII-LOI))*190);
      PLOTDOT(XCOORD,YCOORD,15);
      L:=L^.NEXT
    END
  END; (* CV2SCREEN *)
```

Figure B-5. (continued).

```
SEGMENT FUNCTION PKPOS (POS,N:INTEGER;L:LIST):INTEGER;
(* This routine constructs a circularly linked list of N/50 points
   about the peak POSition specified in order to determine the
   pointer position(integer) of a peak. *)
  VAR I,DUMMY,NUM,POINT:INTEGER;
      START:LIST;
      S:STRING;
      CH:CHAR;
  BEGIN
    (*$R TURTLEGRAPHICS*)
    (*$R ADCRTNS*)
    (*$R INPTRTNS*)
    NUM:=TRUNC(N/50);
    FOR I:=1 TO (POS-NUM) DO L:=L^.NEXT;START:=L;
    FOR I:=1 TO (2*NUM) DO L:=L^.NEXT;L^.NEXT:=START;
    PROMPTAT(0,11,'Press ( -> ) to get back to graph.');
    WRITELN;I:=0;PENCOLOR(NONE);
    REPEAT
      DUMMY:=PEEK(-16206);GRAFMODE;
      MOVETO(250,174);WSTRING('    ');
      MOVETO(250,174);ARROW(FALSE,START);
      POINT:=POS+I-NUM;STR(POINT,S);WSTRING(S);
      GETACHAR(CH,[CHR(21),CHR(13)]);
      IF CH = CHR(21) THEN BEGIN
        ARROW(TRUE,START);
        IF I = (2*NUM) THEN I:=0
        ELSE I:=I+1;
        START:=START^.NEXT
      END;
    UNTIL CH = CHR(13);
    ARROW(TRUE,START);
    PKPOS:=POINT
  END; (* PKPOS *)
```

Figure B-5. (continued).

```
SEGMENT PROCEDURE PKSELECT;
(* This routine sets the anodic and cathodic peak locations and
   sends their integer locations to the PARAMeter file of the CV.  *)
  VAR INTHIE,INTLOE,INTHII,INTLOI,PFLSLD,PAIRS:PARAM;
      O,S:STRING;
      L:LIST;
  BEGIN
    S:='SMTH';L:=NIL;
    IF FILENAME(TRUE,S) THEN BEGIN
      CLOSE(PTS);DELETE(S,POS('SMTH',S),4);
      (*$I-*)
      RESET(PTS,CONCAT(S,'DECONVOL'));
      IF IORESULT = 0 THEN BEGIN
        CV2LIST(L);O:='DECONVOL'
      END
      ELSE BEGIN
        REWRITE(PTS,CONCAT(S,'DECONVOL'));CLOSE(PTS);
        RESET(PTS,CONCAT(S,'SMTH'));CV2LIST(L);O:='SMTH'
      END;
      CLOSE(PTS);
      (*$I+*)
      INSTRUCT;HII:=GETVALUE(1,1,L);LOI:=GETVALUE(1,0,L);
      HIE:=GETVALUE(0,1,L);LOE:=GETVALUE(0,0,L);CV2SCREEN(L);
      INTHII.INTGR:=INTPOS(1,HII,L);INTLOI.INTGR:=INTPOS(1,LOI,L);
      INTHIE.INTGR:=INTPOS(0,HIE,L);INTLOE.INTGR:=INTPOS(0,LOE,L);
      RESET(FOO,CONCAT(S,'PARAM'));FILE2VAR(0,PAIRS);
      FILE2VAR(22,PFLSLD);CLOSE(FOO);
      IF INTLOI.INTGR > INTHII.INTGR THEN BEGIN
        INTLOI.INTGR:=PKPOS(INTLOI.INTGR,PAIRS.INTGR,L);
        INTHII.INTGR:=PKPOS(INTHII.INTGR,PAIRS.INTGR,L)
      END
      ELSE BEGIN
        INTHII.INTGR:=PKPOS(INTHII.INTGR,PAIRS.INTGR,L);
        INTLOI.INTGR:=PKPOS(INTLOI.INTGR,PAIRS.INTGR,L)
      END;
      IF O = 'SMTH' THEN PFLSLD.LTTRS[2]:='T'
      ELSE PFLSLD.LTTRS[3]:='T';PFLSLD.LTTRS[1]:='F';
      RESET(FOO,CONCAT(S,'PARAM'));VAR2FILE(1,INTHIE);
      VAR2FILE(2,INTLOE);VAR2FILE(5,INTHII);
      VAR2FILE(6,INTLOI);VAR2FILE(22,PFLSLD);CLOSE(FOO)
    END
  END; (* PKSELECT *)


(*** MAIN ***)

BEGIN
  (*$N+*)
  SETCHAIN('#10:INTERPAR');
  PKSELECT
END.
```

Figure B-5. (continued).

```
(*$S++*)
PROGRAM CVSMOOTH;

(* This program smoothes a set of original, real cyclic voltammetric data.
   Fourier smoothing is done without the fast Fourier transform.  The al-
   gorithm was taken from J. Electroanal. Chem., 184(1985)239-255 and in-
   corporates provisions for automatically subtracting-then-adding two
   straight lines of non-faradaic background current.  In some cases, the
   original data are very smooth and Fourier smoothing is not desired.
   The five point moving average is an option for this purpose and saves
   time.  The Fourier option should only be used for noisy data since it
   consumes 30 minutes of time for n=25 and N=935. *)

USES CHAINSTUFF,TRANSCEND,INPTRTNS,FILERTNS;

VAR FSLOPE,RSLOPE,FINT,RINT:REAL;

    X,Y:ARRAY [0..934] OF REAL;
    U,V:ARRAY [1..468] OF REAL;
    R:ARRAY [0..50] OF REAL;       (* Transform arrays *)
    I:ARRAY [1..50] OF REAL;

SEGMENT FUNCTION MENU:CHAR;
(* This routine presents a menu of smoothing options. *)
  VAR CH:CHAR;
  BEGIN
    (*$R INPTRTNS*)
    CLRSCRN;
    PROMPTAT(0,1,'Anytime a data set is resmoothed, the peaks');WRITELN;
    PROMPTAT(2,2,'must be relocated and refitted for output!');WRITELN;
    PROMPTAT(0,4,'Smoothing selections:');WRITELN;
    PROMPTAT(2,5,'M>oving average');WRITELN;
    PROMPTAT(2,6,'F>ourier transformation');WRITELN;
    PROMPTAT(0,7,'Please type:');
    GETACHAR(CH,['M','m','F','f']);WRITELN(CH);
    MENU:=CH
  END; (* MENU *)

SEGMENT PROCEDURE MOVAVGSM (VAR L:LIST);
(* This is the BAS-100, five point moving average. *)
  VAR TEMP:LIST;
  BEGIN
    TEMP:=L;
    WHILE L^.NEXT^.NEXT^.NEXT^.NEXT <> NIL DO BEGIN
      L^.NEXT^.NEXT^.VAL[1]:=(L^.VAL[1]+L^.NEXT^.VAL[1]+
        L^.NEXT^.NEXT^.NEXT^.VAL[1]+L^.NEXT^.NEXT^.NEXT^.NEXT^.VAL[1])/4;
      L:=L^.NEXT
    END;
    L^.NEXT^.NEXT^.VAL[1]:=L^.NEXT^.VAL[1];
    L^.NEXT^.NEXT^.NEXT^.VAL[1]:=L^.NEXT^.VAL[1];L:=TEMP;
    L^.NEXT^.VAL[1]:=L^.VAL[1]
  END; (* MOVAVGSM *)

SEGMENT FUNCTION ARRAYPOS (N,HIORLO:INTEGER):INTEGER;
(* This routine saves the integer position of the high(1) or low(0)
   potential value in an array of data. *)
  VAR L,M:INTEGER;
      OLDVAL:REAL;
  BEGIN
    L:=0;OLDVAL:=X[L];
    WHILE L <> N-1 DO BEGIN
      IF HIORLO = 0 THEN BEGIN
        IF X[L+1] < OLDVAL THEN BEGIN
          OLDVAL:=X[L+1];M:=L+1
        END
      END
      ELSE BEGIN
        IF X[L+1] > OLDVAL THEN BEGIN
          OLDVAL:=X[L+1];M:=L+1
        END
      END;
      L:=L+1
    END;
    ARRAYPOS:=M
  END; (* ARRAYPOS *)
```

Figure B-6. Data smoothing program with moving average and FFT filters.

```
SEGMENT PROCEDURE SUBORADD (N,WHICH,REVPT:INTEGER);
(* This routine subtracts the non-faradaic current before Fourier
   transforming if WHICH is 0 or adds it back in if WHICH is 1. *)
  VAR J:INTEGER;
  BEGIN
    CASE WHICH OF
      0:FOR J:=0 TO REVPT DO Y[J]:=Y[J]-(FSLOPE*X[J])-FINT;
      1:FOR J:=0 TO REVPT DO Y[J]:=Y[J]+(FSLOPE*X[J])+FINT
    END;
    CASE WHICH OF
      0:FOR J:=REVPT+1 TO N-1 DO Y[J]:=Y[J]-(RSLOPE*X[J])-RINT;
      1:FOR J:=REVPT+1 TO N-1 DO Y[J]:=Y[J]+(RSLOPE*X[J])+RINT
    END
  END; (* SUBORADD *)

SEGMENT PROCEDURE UVCALC (A,B:INTEGER; VAR S,C:REAL);
(* This routine calculates new values for arrays U and V, starting
   from original sine and cosine values, S and C; A represents P
   and B represents J2 or K. *)
  VAR L,O,T:INTEGER;
      M:REAL;
  BEGIN
    FOR O:=1 TO A DO BEGIN
      U[B+1]:=0;V[B+1]:=0;
      B:=TRUNC((B+1)/2);
      FOR T:=1 TO B DO BEGIN
        L:=(2*T)-1;
        M:=(C*U[L])-(S*V[L])+U[L+1];
        V[T]:=(S*U[L])+(C*V[L])+V[L+1];
        U[T]:=M
      END;
      S:=2*S*C;C:=(2*C*C)-1
    END
  END; (* UVCALC *)

SEGMENT PROCEDURE TRANSFORM (NT,N:INTEGER);
(* This routine calculates the real and imaginary Fourier transforms. *)
  CONST PI=3.14159;
  VAR C,G,S:REAL;
      J,J2,K,NO,P:INTEGER;
  BEGIN
    {$E TRANSCEND}
    G:=0;
    FOR J:=1 TO N-1 DO
      G:=G+Y[J];
    R[0]:=G/N;
    J2:=TRUNC((N-1)/2);P:=TRUNC(LOG((2*J2)-1)/LOG(2));
    PROMPTAT(0,14,'Computing real part of transforms.');WRITELN;
    FOR K:=1 TO NT-1 DO BEGIN
      S:=SIN(2*PI*K/N);C:=COS(2*PI*K/N);
      FOR J:=1 TO J2 DO BEGIN
        U[J]:=(C*Y[(2*J)-1])+Y[2*J];
        V[J]:=S*Y[(2*J)-1]
      END;
      S:=2*S*C;C:=(2*C*C)-1;
      UVCALC(P,J2,S,C);
      R[K]:=((C*U[1])+(S*V[1]))/N
    END;
    PROMPTAT(0,16,'Computing imaginary part of transforms.');WRITELN;
    FOR K:=1 TO NT-1 DO BEGIN
      S:=SIN(2*PI*K/N);C:=COS(2*PI*K/N);
      FOR J:=1 TO J2 DO BEGIN
        U[J]:=-S*Y[(2*J)-1];
        V[J]:=(C*Y[(2*J)-1])+Y[2*J]
      END;
      S:=2*S*C;C:=(2*C*C)-1;
      UVCALC(P,J2,S,C);
      I[K]:=-((C*U[1])+(S*V[1]))/N
    END
  END; (* TRANSFORM *)
```

Figure B-6. (continued).

```
SEGMENT PROCEDURE FILTER (NT,N:INTEGER);
(* This routine applies a quadratic filter to the data. *)
  CONST PI=3.14159;
  VAR F,F1,F2,S,C:REAL;
      J,K,P:INTEGER;
  BEGIN
    F1:=0;F2:=0;
    FOR K:=1 TO NT-1 DO BEGIN
      F1:=F1+R[K];
      F2:=F2+(K*K*R[K])
    END;
    Y[0]:=Y[0]+R[0]+(2*(F1-(F2/NT/NT)));
    P:=TRUNC(LOG((2*NT)-3)/LOG(2));
    FOR J:=1 TO N-1 DO BEGIN
      FOR K:=1 TO NT-1 DO BEGIN
        F:=1-SQR(K/N);
        U[K]:=F*R[K];V[K]:=-F*I[K]
      END;
      S:=SIN(2*PI*J/N);C:=COS(2*PI*J/N);
      K:=NT-1;
      UVCALC(P,K,S,C);
      Y[J]:=R[0]+(2*C*U[1])+(2*S*V[1])
    END
  END; (* FILTER *)

SEGMENT PROCEDURE CV2ARRAY;
(* This routine transfers the current and potential values from
   file to arrays X & Y. *)
  VAR I:INTEGER;
  BEGIN
    I:=0;
    WHILE NOT EOF(PTS) DO BEGIN
      Y[I]:=PTS^[1];
      X[I]:=PTS^[0];
      I:=I+1;
      GET(PTS)
    END
  END; (* CV2ARRAY *)

SEGMENT PROCEDURE ARRAY2CV (N:INTEGER);
(* This routine transfers the smoothed current values and unsmoothed
   potential values to file CV#SMTH. *)
  VAR I:INTEGER;
  BEGIN
    I:=0;
    FOR I:=0 TO N DO BEGIN
      PTS^[1]:=Y[I];
      PTS^[0]:=X[I];
      PUT(PTS)
    END
  END; (* ARRAY2CV *)

SEGMENT PROCEDURE OPENSMTH (S:STRING);
(* This routine opens a file for receiving smoothed data. *)
  BEGIN
    (*$I-*)
    RESET(PTS,CONCAT(S,'SMTH'));
    IF IORESULT <> 0 THEN REWRITE(PTS,CONCAT(S,'SMTH'));
    (*$I+*)
  END; (* OPENSMTH *)

SEGMENT FUNCTION LINES (HIE,LOE,PTS:INTEGER):INTEGER;
(* This routine calculates the equations of two lines of non-faradaic
   current and returns the array position of the reversal point. *)
  VAR REVPT:INTEGER;
  BEGIN
    IF ABS(HIE-TRUNC(PTS/2)) < ABS(LOE-TRUNC(PTS/2)) THEN
      REVPT:=HIE
    ELSE
      REVPT:=LOE;
    FSLOPE:=(Y[REVPT]-Y[0])/(X[REVPT]-X[0]);
    FINT:=Y[REVPT]-(FSLOPE*X[REVPT]);
    RSLOPE:=(Y[PTS-1]-Y[REVPT])/(X[PTS-1]-X[REVPT]);
    RINT:=Y[REVPT]-(RSLOPE*X[REVPT]);
    LINES:=REVPT
  END; (* LINES *)
```

Figure B-6. (continued).

```
SEGMENT PROCEDURE DATA2FILE;
(* This routine calls all of the above to smooth the data and transfer
   them to CV#SMTH. *)
  VAR NT,N,PFLSLD,SMTH:PARAM;     (* N = number of data pairs *)
      HIE,LOE,REVPT:INTEGER;     (* NT = number of transforms *)
      NUM,S:STRING;
      DATA:LIST;
      CH:CHAR;
  BEGIN
    (*$R FILERTNS*)
    S:='ORIGREAL';
    IF NOT FILENAME(TRUE,S) THEN
      EXIT(DATA2FILE);
    DELETE(S,POS('ORIGREAL',S),8);CH:=MENU;
    IF CH = 'M' THEN BEGIN
      DATA:=NIL;CV2LIST(DATA);CLOSE(PTS);
      PROMPTAT(0,12,'Computing five point moving average');WRITELN;
      MOVAVGSM(DATA);MOVAVGSM(DATA);OPENSMTH(S);CV2FILE(DATA);
      CLOSE(PTS,LOCK);SMTH.LTTRS:='2 x five point moving average'
    END
    ELSE IF CH = 'F' THEN BEGIN
      CV2ARRAY;CLOSE(PTS);
      RESET(FOO,CONCAT(S,'PARAM'));FILE2VAR(0,N);CLOSE(FOO);
      PROMPTAT(0,10,'Add non-faradaic current back in <Y,N>:');
      GETACHAR(CH,['Y','y','N','n']);WRITELN(CH);
      PROMPTAT(0,12,'Input number of Fourier transforms <1..50>:');
      NT.INTGR:=GETINT(1,50);WRITELN;HIE:=ARRAYPOS(N.INTGR,1);
      LOE:=ARRAYPOS(N.INTGR,0);REVPT:=LINES(HIE,LOE,N.INTGR);
      SUBORADD(N.INTGR,0,REVPT);TRANSFORM(NT.INTGR,N.INTGR);
      FILTER(NT.INTGR,N.INTGR);
      IF CH = 'Y' THEN BEGIN
        SUBORADD(N.INTGR,1,REVPT);
        SMTH.LTTRS:='non-faradaic current added back'
      END
      ELSE SMTH.LTTRS:='non-faradaic current subtracted';
      OPENSMTH(S);ARRAY2CV(N.INTGR-1);CLOSE(PTS,LOCK);STR(NT.INTGR,NUM);
      SMTH.LTTRS:=CONCAT('Fourier (n=',NUM,'), ',SMTH.LTTRS)
    END;
    PFLSLD.LTTRS:='FFF';RESET(FOO,CONCAT(S,'PARAM'));
    VAR2FILE(21,SMTH);VAR2FILE(22,PFLSLD);CLOSE(FOO);DELETE(S,POS('#5:',S),3);
    PROMPTAT(0,18,CONCAT(S,'SMTH is now on disk in drive 2!'));
    WRITELN;PROMPTAT(0,20,'Press <SPACE> to continue!');
    GETACHAR(CH,[' ']);WRITELN
  END; (* DATA2FILE *)


(*** MAIN ***)

BEGIN
  (*$N+*)
  SETCHAIN('#10:INTERPAR');
  DATA2FILE
END.
```

Figure B-6. (continued).

```
(*$S++*)
PROGRAM PEAKFIT;

(* The method of splines is used to find Ip; Ep is chosen in the follow-
   ing manner.  In PEAKLOC, the approximate Ep is selected and its inte-
   ger position in the list is determined.  This program starts its X,Y
   data set at that position minus the (number of data pairs/100).  Ep is
   chosen as the midpoint of the middle interval of the data set.  The
   numerical methods for spline fitting were taken from, Pennington, Ralph
   H., "Introductory Computer Methods and Numerical Analysis", Macmillan
   Co., New York, New York, 1965, pp. 365-371,404-411; Pizer, Stephen M.,
   "Numerical Computing and Mathematical Analysis", Prentice-Hall, Inc.,
   Englewood Cliffs, New Jersey, 1975, pp. 307-312; Norris, A. C., "Comp-
   utational Chemistry, an Introduction to Numerical Analysis", John Wiley
   & Sons, New York, New York, 1981, pp.25-37,153-208,261-323.  By checking
   VARiable PFLSLD, the program knows whether to fit the deconvoluted or
   smoothed curve.  There is also a routine for computing the charge of
   the CV by adding up the area of successive trapezoids.  Accurate values
   for Ep/a and Ip/a are needed to determine an accurate Ep/2 value. *)

USES CHAINSTUFF,TRANSCEND,INPTRTNS,FILERTNS;

TYPE POINTS = ARRAY [1..50] OF REAL;

VAR X,Y,D,F,E,B,Z:POINTS;     (* These arrays are for spline solving. *)
    A:ARRAY [1..50,1..3] OF REAL;
    C:ARRAY [1..4,1..50] OF REAL;

    IPC,EPC,IPA,EPA,EPTWO:PARAM;

SEGMENT FUNCTION CHARGE (HIORLO:INTEGER;SCRATE:REAL;L:LIST):REAL;
(* This routine computes the cathodic (HIORLO=0) or anodic (HIORLO=1)
   charge of a cyclic voltammogram. *)
  VAR SUM:REAL;
  BEGIN
    SUM:=0;
    WHILE L^.NEXT <> NIL DO
      BEGIN
        CASE HIORLO OF
          0:IF L^.VAL[1] < 0 THEN
              SUM:=SUM+(0.5*(L^.VAL[1]+L^.NEXT^.VAL[1])*
                ABS(L^.NEXT^.VAL[0]-L^.VAL[0])/SCRATE);
          1:IF L^.VAL[1] > 0 THEN
              SUM:=SUM+(0.5*(L^.VAL[1]+L^.NEXT^.VAL[1])*
                ABS(L^.NEXT^.VAL[0]-L^.VAL[0])/SCRATE)
        END; (* CASE *)
        L:=L^.NEXT
      END;  (* WHILE *)
    CHARGE:=SUM
  END;  (* CHARGE *)

SEGMENT FUNCTION EP2VAL (EP,IP:REAL;L:LIST):REAL;
(* This routine selects Ep/2 on the more negative potential side of Ep/a. *)
  VAR EP21,EP22:REAL;
      I:INTEGER;
      CH:CHAR;
  BEGIN
    WHILE L^.VAL[1] < IP/2 DO
      L:=L^.NEXT;
    EP21:=L^.VAL[0];
    FOR I:=1 TO 10 DO
      L:=L^.NEXT;
    WHILE L^.VAL[1] > IP/2 DO
      L:=L^.NEXT;
    EP22:=L^.VAL[0];
    IF EP22 < EP21 THEN EP2VAL:=EP22
    ELSE EP2VAL:=EP21
  END; (* EP2VAL *)
```

Figure B-7. Cubic spline fitting program used to determine E_ps and i_ps.

```
SEGMENT PROCEDURE IPCALC (M:INTEGER);
(* This routine uses the method of splines to calculate Ip. *)
  VAR EP,IP,Q,R:REAL;
      K,N:INTEGER;
  BEGIN
    Q:=D[M-2]/D[M-1];
    A[M,1]:=1+Q+A[M-2,3];A[M,2]:=-Q-(A[M,1]*A[M-1,3]);
    B[M]:=B[M-2]-(A[M,1]*B[M-1]);Z[M]:=B[M]/A[M,2];
    FOR K:=M-1 DOWNTO 2 DO Z[K]:=B[K]-(A[K,3]*Z[K+1]);
    Z[1]:=-(A[1,2]*Z[2])-(A[1,3]*Z[3]);
    FOR K:=1 TO M-1 DO BEGIN
      Q:=1/(6*D[K]);C[1,K]:=Z[K]*Q;C[2,K]:=Z[K+1]*Q;
      C[3,K]:=(6*Y[K]*Q)-(Z[K]*P[K]);C[4,K]:=(6*Y[K+1]*Q)-(Z[K+1]*P[K])
    END;
    N:=TRUNC(M/2);
    EP:=(X[N+1]+X[N])/2;Q:=X[N+1]-EP;R:=EP-X[N];
    IP:=(C[1,N]*Q*Q*Q)+(C[2,N]*R*R*R)+(C[3,N]*Q)+(C[4,N]*R);
    IF IP < 0 THEN BEGIN
      IPC.DCML:=IP;EPC.DCML:=EP
    END
    ELSE BEGIN
      IPA.DCML:=IP;EPA.DCML:=EP
    END
  END; (* IPCALC *)

SEGMENT PROCEDURE SPLINE (M:INTEGER);
(* This routine starts the spline fit calculation. *)
  VAR K:INTEGER;
  BEGIN
    FOR K:=1 TO M-1 DO BEGIN
      D[K]:=X[K+1]-X[K];P[K]:=D[K]/6;E[K]:=(Y[K+1]-Y[K])/D[K]
    END;
    FOR K:=2 TO M-1 DO B[K]:=E[K]-E[K-1];
    A[1,2]:=-1-(D[1]/D[2]);A[1,3]:=D[1]/D[2];
    A[2,3]:=P[2]-(P[1]*A[1,3]);
    A[2,2]:=2*(P[1]+P[2])-(P[1]*A[1,2]);
    A[2,3]:=A[2,3]/A[2,2];B[2]:=B[2]/A[2,2];
    FOR K:=3 TO M-1 DO BEGIN
      A[K,2]:=2*(P[K-1]+P[K])-(P[K-1]*A[K-1,3]);
      B[K]:=(B[K]-(P[K-1]*B[K-1]))/A[K,2];A[K,3]:=P[K]/A[K,2]
    END;
    IPCALC(M)
  END; (* SPLINE *)

SEGMENT PROCEDURE EPIPCALC (POS,NUM:INTEGER;L:LIST);
(* This routine orders the array of data values around a peak and
   calls the others to find Ep,Ip. *)
  VAR XTEMP,YTEMP:POINTS;
      FIRST,LAST,INC:REAL;
      J,K,M:INTEGER;
  BEGIN
    FOR J:=1 TO POS-NUM DO L:=L^.NEXT;
    FIRST:=L^.VAL[0];
    M:=2*NUM;
    FOR J:=1 TO M DO BEGIN
      LAST:=L^.VAL[0];YTEMP[J]:=L^.VAL[1];L:=L^.NEXT
    END;
    INC:=(LAST-FIRST)/(M-1);
    FOR J:=0 TO M-1 DO XTEMP[J+1]:=FIRST+(J*INC);
    IF XTEMP[1] < XTEMP[M] THEN BEGIN
      FOR K:=1 TO M DO BEGIN
        X[K]:=XTEMP[K];Y[K]:=YTEMP[K]
      END
    END
    ELSE BEGIN
      FOR K:=1 TO M DO BEGIN
        X[K]:=XTEMP[M-K+1];Y[K]:=YTEMP[M-K+1]
      END
    END;
    SPLINE(M)
  END; (* EPIPCALC *)
```

Figure B-7. (continued).

```
SEGMENT PROCEDURE MESSAGE (X:INTEGER);
(* This routine writes a message on the screen at line X. *)
  VAR CH:CHAR;
  BEGIN
    (*$R INPTRTNS*)
    PROMPTAT(0,X,'Press <SPACE> to continue!');
    GETACHAR(CH,[' ']);WRITELN(CH)
  END;

SEGMENT PROCEDURE DATA2FILE;
(* This routine calls EPCALC to calculate Ip, Ep, and Ep/2.  *)
  VAR PAIRS,HII,LOI,SCRATE,CCH,ACH,INITE,PFLSLD:PARAM;
      NUM:INTEGER;
      S:STRING;
      TEMP,L:LIST;
  BEGIN
    (*$R INPTRTNS*)
    (*$R FILERTNS*)
    L:=NIL;S:='SMTH';
    IF FILENAME(TRUE,S) THEN BEGIN
      CLOSE(PTS);DELETE(S,POS('SMTH',S),4);
      RESET(FOO,CONCAT(S,'PARAM'));
      FILE2VAR(0,PAIRS);FILE2VAR(5,HII);FILE2VAR(6,LOI);
      FILE2VAR(9,SCRATE);FILE2VAR(22,PFLSLD);CLOSE(FOO);
      DELETE(S,POS('PARAM',S),5);
      IF (PFLSLD.LTTRS[2] = 'F') AND (PFLSLD.LTTRS[3] = 'F') THEN BEGIN
        PROMPTAT(0,8,'You must locate peaks in the smoothed data !');
        WRITELN;MESSAGE(10);EXIT(DATA2FILE)
      END
      ELSE IF (PFLSLD.LTTRS[2] = 'T') AND (PFLSLD.LTTRS[3] = 'F') THEN BEGIN
        RESET(PTS,CONCAT(S,'SMTH'));CV2LIST(L);CLOSE(PTS)
      END
      ELSE IF (PFLSLD.LTTRS[2] = 'T') AND (PFLSLD.LTTRS[3] = 'T') THEN BEGIN
        RESET(PTS,CONCAT(S,'DECONVOL'));CV2LIST(L);CLOSE(PTS)
      END;
      TEMP:=L;
      WHILE TEMP^.NEXT <> NIL DO TEMP:=TEMP^.NEXT;
      INITE.DCML:=TEMP^.VAL[0];NUM:=TRUNC(PAIRS.INTGR/100);
      PROMPTAT(0,8,'Computing Ep and Ip by spline fitting!');
      WRITELN;EPIPCALC(HII.INTGR,NUM,L);EPIPCALC(LOI.INTGR,NUM,L);
      PROMPTAT(0,10,'Computing cathodic and anodic charges!');
      WRITELN;CCH.DCML:=CHARGE(0,SCRATE.DCML,L);
      ACH.DCML:=CHARGE(1,SCRATE.DCML,L);
      EPTWO.DCML:=EP2VAL(EPA.DCML,IPA.DCML,L);
      PFLSLD.LTTRS[1]:='T';DELETE(S,POS('#5:',S),3);
      PROMPTAT(0,12,CONCAT('Transferring parameters to ',S,'PARAM!'));
      WRITELN;RESET(FOO,CONCAT('#5:',S,'PARAM'));
      VAR2FILE(10,IPC);VAR2FILE(11,EPC);
      VAR2FILE(12,IPA);VAR2FILE(13,EPA);
      VAR2FILE(14,CCH);VAR2FILE(15,ACH);
      VAR2FILE(16,EPTWO);VAR2FILE(17,INITE);
      VAR2FILE(22,PFLSLD);CLOSE(FOO);MESSAGE(14)
    END
  END; (* DATA2FILE *)

(*** MAIN ***)

BEGIN
  (*$N+*)
  SETCHAIN('#10:INTERPAR');
  DATA2FILE;
  EXIT(PROGRAM)
END.
```

Figure B-7. (continued).

```
(*$S++*)
PROGRAM DECONVOL;

(* This program deconvolutes a list of smoothed, CV data. The basis of
   the deconvolution is a reflection of the right cathodic and anodic
   halves of the curve's current values onto the left cathodic and anodic
   halves of the curve.  Point-by-point subtraction of the deconvoluted
   curve from the smoothed curve then gives a CV describing the hidden
   redox process.  Accommodations are made for both a positive and nega-
   tive going sweep. *)

USES CHAINSTUFF,TRANSCEND,INPTRTNS,FILERTNS;

VAR FRONT:LIST;

SEGMENT PROCEDURE STACK (E,I:REAL; VAR L:LIST);
(* This routine inserts a data pair at the front of the list; insertion
   is like that of a stack. *)
  VAR TEMP:LIST;
  BEGIN
    NEW(TEMP);
    WITH TEMP^ DO BEGIN
      VAL[0]:=E;
      VAL[1]:=I;
      NEXT:=L
    END;
    L:=TEMP
  END; (* STACK *)

SEGMENT PROCEDURE COPY (E,I:REAL; VAR L:LIST);
(* This routine copies a data pair into a list by inserting at the rear. *)
  VAR TEMP:LIST;
  BEGIN
    IF L = NIL THEN BEGIN
      STACK(E,I,L);
      FRONT:=L
    END
    ELSE BEGIN
      WHILE L^.NEXT <> NIL DO L:=L^.NEXT;
      NEW(TEMP);
      WITH TEMP^ DO BEGIN
        VAL[0]:=E;
        VAL[1]:=I;
        NEXT:=NIL
      END;
      L^.NEXT:=TEMP
    END
  END; (* COPY *)
```

Figure B-8. Positive and negative scan, deconvolution program.

```
SEGMENT PROCEDURE NEGCONST (HI,LI,LE:INTEGER; OLD:LIST; VAR N:LIST);
(* This routine constructs a deconvoluted list of data from an old list
   of data for a negative going sweep. *)
  VAR S1,S2:LIST;
      COUNT,I:INTEGER;
  BEGIN
    S1:=NIL;S2:=NIL;COUNT:=1;      (* Initialize stacks to empty *)
    WHILE COUNT <> HI DO BEGIN
      COPY(OLD^.VAL[0],0,N);
      OLD:=OLD^.NEXT;COUNT:=COUNT+1
    END;
    WHILE COUNT <> LE DO BEGIN
      COPY(OLD^.VAL[0],OLD^.VAL[1],N);
      STACK(0,OLD^.VAL[1],S1);
      OLD:=OLD^.NEXT;COUNT:=COUNT+1
    END;
    WHILE COUNT <> LI DO BEGIN
      COPY(OLD^.VAL[0],OLD^.VAL[1],N);
      STACK(0,OLD^.VAL[1],S2);
      OLD:=OLD^.NEXT;COUNT:=COUNT+1
    END;
    WHILE OLD <> NIL DO BEGIN
      COPY(OLD^.VAL[0],0,N);
      OLD:=OLD^.NEXT
    END;
    N:=FRONT;COUNT:=LE-HI+1;
    WHILE HI < COUNT DO BEGIN
      S1:=S1^.NEXT;COUNT:=COUNT-1
    END;
    WHILE HI >= COUNT DO BEGIN
      N^.VAL[1]:=S1^.VAL[1];
      N:=N^.NEXT;COUNT:=COUNT+1
    END;
    WHILE S1 <> NIL DO BEGIN
      N^.VAL[1]:=S1^.VAL[1];
      N:=N^.NEXT;S1:=S1^.NEXT
    END;
    FOR I:=1 TO LI-HI-2 DO N:=N^.NEXT;
    WHILE N <> NIL DO BEGIN
      N^.VAL[1]:=S2^.VAL[1];
      IF S2^.NEXT <> NIL THEN S2:=S2^.NEXT;
      N:=N^.NEXT
    END;
    N:=FRONT
  END; (* NEGCONST *)
SEGMENT PROCEDURE POSCONST (HI,LI,HE,PTS:INTEGER; OLD:LIST; VAR N:LIST);
(* This routine constructs a deconvoluted list of data from an old list
   of data for a positive going sweep. *)
  VAR S1,S2:LIST;
      COUNT,I:INTEGER;
  BEGIN
    S1:=NIL;S2:=NIL;COUNT:=1;
    WHILE COUNT <> LI DO BEGIN
      COPY(OLD^.VAL[0],OLD^.VAL[1],N);
      STACK(0,OLD^.VAL[1],S1);OLD:=OLD^.NEXT;COUNT:=COUNT+1
    END;
    WHILE COUNT <> HI DO BEGIN
      COPY(OLD^.VAL[0],0,N);OLD:=OLD^.NEXT;COUNT:=COUNT+1
    END;
    WHILE OLD <> NIL DO BEGIN
      COPY(OLD^.VAL[0],OLD^.VAL[1],N);
      STACK(0,OLD^.VAL[1],S2);OLD:=OLD^.NEXT
    END;
    N:=FRONT;COUNT:=1;
    WHILE COUNT < LI DO BEGIN
      N:=N^.NEXT;COUNT:=COUNT+1
    END;
    WHILE COUNT < HE DO BEGIN
      N^.VAL[1]:=S1^.VAL[1];
      IF S1^.NEXT <> NIL THEN S1:=S1^.NEXT;
      N:=N^.NEXT;COUNT:=COUNT+1
    END;
    WHILE HI-COUNT > PTS-HI DO BEGIN
      N^.VAL[1]:=S2^.VAL[1];N:=N^.NEXT;COUNT:=COUNT+1
    END;
    IF HI-COUNT < PTS-HI THEN BEGIN
      FOR I:=1 TO (PTS-(2*HI)+COUNT) DO S2:=S2^.NEXT
    END;
    WHILE S2 <> NIL DO BEGIN
      N^.VAL[1]:=S2^.VAL[1];N:=N^.NEXT;S2:=S2^.NEXT
    END;
    N:=FRONT
  END; (* POSCONST *)
```

Figure B-8. (continued).

```
SEGMENT PROCEDURE SUBTRACT (L1:LIST; VAR L2:LIST);
(* This routine subtracts L1 from L2 to give the hidden redox process. *)
  VAR START:LIST;
  BEGIN
    START:=L2;
    WHILE L1 <> NIL DO BEGIN
      L2^.VAL[1]:=L2^.VAL[1]-L1^.VAL[1];
      L2:=L2^.NEXT;L1:=L1^.NEXT
    END;
    L2:=START
  END; (* SUBTRACT *)

SEGMENT PROCEDURE MESSAGE (S:STRING);
(* This routine displays a message if trying to deconvolute twice. *)
  VAR CH:CHAR;
  BEGIN
    PROMPTAT(0,8,S);WRITELN;
    PROMPTAT(0,10,'Press <SPACE> to continue.');
    GETACHAR(CH,[' ']);WRITELN(CH);EXIT(PROGRAM)
  END; (* MESSAGE *)

SEGMENT PROCEDURE GETVAR;
(* This routine opens the PARAMeter file and retrieves the list
   positions for the high current, low current, and low potential
   used in constructing the deconvoluted CV. *)
  VAR PAIRS,HII,LOI,HIE,LOE,PFLSLD:PARAM;
      L,DECON:LIST;
      NUM:STRING;
      CH:CHAR;
  BEGIN
    (*$R FILERTNS*)
    NUM:='SMTH';L:=NIL;
    IF FILENAME(TRUE,NUM) THEN BEGIN
      CV2LIST(L);CLOSE(PTS);
      DELETE(NUM,POS('SMTH',NUM),4);
      (*$I-*)
      RESET(PTS,CONCAT(NUM,'DECONVOL'));
      IF IORESULT = 0 THEN
        MESSAGE('You cannot deconvolute twice!')
      ELSE
        REWRITE(PTS,CONCAT(NUM,'DECONVOL'));
      CLOSE(PTS);
      (*$I+*)
      RESET(FOO,CONCAT(NUM,'PARAM'));
      FILE2VAR(22,PFLSLD);CLOSE(FOO);
      IF PFLSLD.LTTRS[2] = 'F' THEN
        MESSAGE('You must locate peaks in the CV!');
      RESET(FOO,CONCAT(NUM,'PARAM'));FILE2VAR(0,PAIRS);
      FILE2VAR(1,HIE);FILE2VAR(2,LOE);FILE2VAR(5,HII);
      FILE2VAR(6,LOI);CLOSE(FOO);DECON:=NIL;
      PROMPTAT(0,10,'Computing deconvoluted curve!');WRITELN;
      IF HII.INTGR < LOI.INTGR THEN
        NEGCONST(HII.INTGR,LOI.INTGR,LOE.INTGR,L,DECON)
      ELSE
        POSCONST(HII.INTGR,LOI.INTGR,HIE.INTGR,PAIRS.INTGR,L,DECON);
      SUBTRACT(DECON,L);
      REWRITE(PTS,CONCAT(NUM,'DECONVOL'));CV2FILE(DECON);CLOSE(PTS,LOCK);
      REWRITE(PTS,CONCAT(NUM,'SUB'));CV2FILE(L);CLOSE(PTS,LOCK);
      DELETE(NUM,POS('#5:',NUM),3);
      NUM:=CONCAT(NUM,'DECONVOL & SUB are now on disk in drive 2!');
      PROMPTAT(0,12,NUM);WRITELN;
      PROMPTAT(0,14,'Press <SPACE> to continue!');
      GETACHAR(CH,[' ']);WRITELN
    END
    ELSE EXIT(PROGRAM)
  END; (* GETVAR *)


(*** MAIN ***)

BEGIN
  (*$N+*)
  SETCHAIN('#10:INTERPAR');
  GETVAR
END.
```

Figure B-8. (continued).

APPENDIX C

TEXT FILE LISTINGS OF CHRONOAMPEROMETRY PROGRAMS

```
(*$S++*)
PROGRAM CADATA;

(* In this program, chronoamperometric data are acquired at a
   precalculated, fixed delay rate.  The data are placed in a
   3,746 byte unused area of memory at location 4446.  The binary
   data are retrieved, converted, and placed on diskfile for sub-
   sequent calculations.  A background curve is collected then the
   analyte is added and its response recorded.  Most of the param-
   eter data are placed on diskfile also. *)

USES CHAINSTUFF,TURTLEGRAPHICS,ADCRTNS,
     TIMETEMP,TRANSCEND,INPTRTNS,FILERTNS;

(* None of the code in TURTLEGRAPHICS is actually called by CADATA,
   but it frees the 3,746 bytes of user RAM below Hi-Res Page 1 where
   TAKEDATA places the binary data for chronoamperometry. *)

CONST  ONTIME=7.91789E-5; (* Time(sec) to acquire one current point=
                             81 clock cycles x 1/1.023MHz x 1E-6 *)

VAR CH:CHAR;
    NAME:STRING;
    OFFTIME:REAL;
    DELAY3,DELAY2,DELAY1,E1,E2,CELL,CCHAN,ICHAN,ECHAN:INTEGER;
    PAIRS,ICODE,N,DEGREE,SENS,AREA,DIFF,IPOT,FPOT,DUR,CONC,START,
      TITLE,DETER,SMOOTH:PARAM;

                                    (* PARAM is declared in FILERTNS *)

  PROCEDURE TAKEDATA (E2,PAIRS,DELAY1,DELAY2,DELAY3,CELL,ICHAN,
                      ECHAN,MEMLOC:INTEGER);
    EXTERNAL;

  SEGMENT FUNCTION SELCODE:INTEGER;
  (* This routine prompts for the voltage range on the current channel;
     the most sensitive range should be selected. *)
    BEGIN
      (*$R INPTRTNS*)
      PROMPTAT(10,11,' GAINCODE        BIPOLAR RANGE');WRITELN;
      PROMPTAT(10,12,'-------------------------------');WRITELN;
      PROMPTAT(10,13,'     0              0 <--> +5V');WRITELN;
      PROMPTAT(10,14,'     1              0 <--> +1V');WRITELN;
      PROMPTAT(10,15,'     2              0 <--> +0.5V');WRITELN;
      PROMPTAT(10,16,'     3              0 <--> +0.1V');WRITELN;
      PROMPTAT(2,18,'Current gaincode (0..3):');SELCODE:=GETINT(0,3);
      WRITELN
    END; (* SELCODE *)

  SEGMENT FUNCTION BVALUE (VOLTAGE:REAL):INTEGER;
  (* This function returns a binary value between 0 and 255 equivalent
     to a voltage between -4.190V and 4.160V on the CV-27's potentiostat. *)
    BEGIN
      BVALUE:=ROUND(((VOLTAGE+4.190)*255)/8.350)
    END; (* BVALUE *)

  SEGMENT PROCEDURE BVAL2DAC (VAL:INTEGER);
  (* This routine sends a value between 0 and 255 to a DAC output channel. *)
    TYPE  POINTER = RECORD
            CASE BOOLEAN OF
              TRUE:(ADDR:INTEGER);
              FALSE:(PTR:^INTEGER)
            END;
    VAR MEM:POINTER;
    BEGIN
      MEM.ADDR:=-14592+ECHAN;MEM.PTR^:=VAL
    END; (* BVAL2DAC *)
```

Figure C-1. Pascal calling program for acquisition of chronoamperometric data.

```
SEGMENT PROCEDURE PARAMETERS;
(* Obtain user input for chronoamperometry. *)
  BEGIN
    (*$R INPTRTNS*)
    CLRSCRN;
    PROMPTAT(0,1,'Title:');READLN(TITLE.LTTRS);
    PROMPTAT(2,2,'Sensitivity    (Amps/V):');SENS.DCML:=GETREAL;WRITELN;
    PROMPTAT(2,3,'Analyte conc.        (M):');CONC.DCML:=GETREAL;WRITELN;
    PROMPTAT(2,4,'Number of electrons    :');N.INTGR:=GETINT(1,10);WRITELN;
    PROMPTAT(2,5,'Determination of  <A,D>:');GETACHAR(CH,['a','A','d','D']);
    WRITELN(CH);
    IF CH = 'A' THEN
      BEGIN
        PROMPTAT(2,6,'Value of D   (cm^2/sec):');DIFF.DCML:=GETREAL;WRITELN;
        DETER.LTTRS:='Electrode area';AREA.DCML:=0.0
      END
    ELSE
      BEGIN
        PROMPTAT(2,6,'Value of A        (cm^2):');AREA.DCML:=GETREAL;WRITELN;
        DETER.LTTRS:='Diffusion coefficient';DIFF.DCML:=0.0
      END;
    PROMPTAT(2,7,'Initial potential    (V):');IPOT.DCML:=GETREAL;WRITELN;
    PROMPTAT(2,8,'Final potential      (V):');FPOT.DCML:=GETREAL;WRITELN;
    (* Send E1 to the cell as soon as it is known. *)
    E1:=BVALUE(IPOT.DCML);BVAL2DAC(E1);
    (* Adjust E2 by subtracting E1's voltage; this is done because
       the CV-27 act as an adder potentiostat when switched to CELL
       in TAKEDATA. *)
    E2:=BVALUE(FPOT.DCML-IPOT.DCML);
    REPEAT
      PROMPTAT(26,9,'             ');
      PROMPTAT(2,9,'Experiment time    (sec):');
      DUR.DCML:=GETREAL;WRITELN;
    UNTIL (DUR.DCML > 25.0) AND (DUR.DCML < 35.0);
    PAIRS.INTGR:=1000;WRITELN;ICODE.INTGR:=SELCODE;SMOOTH.LTTRS:='None'
  END; (* PARAMETERS *)

SEGMENT PROCEDURE CALCDELAY (X:INTEGER; B:BOOLEAN);
(* This routine collects the current data based on user input. *)
  CONST  D3=3.21318E-1;      (* Time(sec) each delay count can generate *)
         D2=1.25513E-3;
         D1=4.88758E-6;
  BEGIN
    (*$R ADCRTNS*)
    IF B THEN BEGIN
      PARAMETERS;
      OFFTIME:=(DUR.DCML-(ONTIME*PAIRS.INTGR))/PAIRS.INTGR;
      DELAY3:=TRUNC(OFFTIME/D3)+1;
      DELAY2:=TRUNC(((OFFTIME)-((DELAY3-1)*D3))/D2)+1;
      DELAY1:=TRUNC(((OFFTIME)-(DELAY3*D3)-(DELAY2*D2)+3.22550E-1)/D1)+1;
      ICHAN:=ICHAN+(ICODE.INTGR*16);
      DEGREE.DCML:=CFIND(0,CCHAN,4446)     (* First temperature reading *)
    END;
    PROMPTAT(0,X,'Set function switch to CELL and press <SPACE>!');
    GETACHAR(CH,[' ']);WRITELN(CH);
    TAKEDATA(E2,PAIRS.INTGR,DELAY1,DELAY2,DELAY3,CELL,ICHAN,ECHAN,4446);
    BVAL2DAC(E1);
    PROMPTAT(0,X+2,'Experiment complete!')
  END; (* CALCDELAY *)
```

Figure C-1. (continued).

```
SEGMENT PROCEDURE ENLIST (MEMLOC,POINTS:INTEGER; VAR L:LIST);
(* Transfers data to a linked list by inserting at the rear of the list;
   the time points are produced artificially using the OFFTIME and ONTIME
   per current point and delay time to charge the electrode. *)
  VAR FRONT,TEMP:LIST;
      I:INTEGER;
  BEGIN
    (*$R ADCRTNS*)
    I:=0;
    WHILE POINTS <> 0 DO BEGIN
      NEW(TEMP);
      WITH TEMP^ DO BEGIN
        VAL[0]:=I * (OFFTIME + ONTIME);
        VAL[1]:=(PEEK(MEMLOC)*256)+PEEK(MEMLOC+1);
        IF L = NIL THEN BEGIN
          NEXT:=L;L:=TEMP;FRONT:=L
        END
        ELSE BEGIN
          WHILE L^.NEXT <> NIL DO L:=L^.NEXT;
          NEXT:=NIL;L^.NEXT:=TEMP
        END
      END; (* WITH *)
      MEMLOC:=MEMLOC+2;I:=I+1;POINTS:=POINTS-1
    END; (* WHILE *)
    L:=FRONT
  END; (* ENLIST *)

SEGMENT PROCEDURE BIN2REAL (CODE,I:INTEGER;FACTOR:REAL; VAR L:LIST);
(* Convert digits between 0 and 4095 to real numbers based on the voltage
   range and the sensitivity. *)
  VAR TEMP:LIST;
      F,LOW:REAL;
  BEGIN
    (*$R ADCRTNS*)
    F:=FRACTION(CODE);
    LOW:=LOWEND(CODE);
    TEMP:=L;
    WHILE L <> NIL DO BEGIN
      L^.VAL[I]:=(((L^.VAL[I])*F)+LOW)*FACTOR;
      L:=L^.NEXT
    END; (* WHILE *)
    L:=TEMP
  END; (* BIN2REAL *)

SEGMENT PROCEDURE MESSAGE (X:INTEGER);
(* This routine displays a message at line X on the screen. *)
  BEGIN
    PROMPTAT(0,X,'Press <SPACE> to continue!');GETACHAR(CH,[' ']);
    WRITELN
  END; (* MESSAGE *)

SEGMENT PROCEDURE PARAMOUT (S:STRING);
(* This routine transfers parameters to a chronoamperometric file. *)
  BEGIN
    REWRITE(FOO,CONCAT(S,'PARAM'));
    VAR2FILE(0,PAIRS);VAR2FILE(1,ICODE);VAR2FILE(4,N);VAR2FILE(5,DEGREE);
    VAR2FILE(6,SENS);VAR2FILE(7,AREA);VAR2FILE(8,DIFF);VAR2FILE(9,IPOT);
    VAR2FILE(10,FPOT);VAR2FILE(11,DUR);VAR2FILE(12,CONC);
    VAR2FILE(13,START);VAR2FILE(14,TITLE);VAR2FILE(15,SMOOTH);
    VAR2FILE(16,DETER);CLOSE(FOO,LOCK)
  END; (* PARAMOUT *)
```

Figure C-1. (continued).

```
  PROCEDURE DATA2FILE (S:STRING);
  (* Place the analyte, background and parameter data on file. *)
    VAR L1,L2:LIST;
        I:INTEGER;
    BEGIN
      (*$R FILERTNS*)
      (*$R ADCRTNS*)
      L1:=NIL;
      TIMEASCII(START.LTTRS,12);
      IF FILENAME(FALSE,S) THEN BEGIN
        PROMPTAT(0,8,'Collect background data first.');WRITELN;
        MESSAGE(10);CALCDELAY(20,TRUE);
        ENLIST(4446,PAIRS.INTGR,L1);
        BIN2REAL(ICODE.INTGR,1,SENS.DCML,L1);
        CA2FILE(L1);CLOSE(PTS,LOCK);CLRSCRN;
        DELETE(S,POS('BCKGRND',S),7);REWRITE(PTS,CONCAT(S,'ANLYTE'));
        PROMPTAT(0,1,'Add analyte, stir, and purge with inert gas.');
        WRITELN;CALCDELAY(3,FALSE);L2:=L1;
        I:=4446;
        WHILE L1 <> NIL DO BEGIN
          L1^.VAL[1]:=(PEEK(I)*256)+PEEK(I+1);I:=I+2;L1:=L1^.NEXT
        END;
        L1:=L2;BIN2REAL(ICODE.INTGR,1,SENS.DCML,L1);
        CA2FILE(L1);CLOSE(PTS,LOCK);
        DEGREE.DCML:=(DEGREE.DCML+CFIND(0,CCHAN,4446))/2;
        DELETE(S,POS('ANLYTE',S),6);PARAMOUT(S);DELETE(S,POS('#5:',S),3);
        S:=CONCAT(S,'ANLYTE, BCKGRND, & PARAM are now on disk in drive 2!');
        PROMPTAT(0,7,S);WRITELN;MESSAGE(9)
      END
    END; (* DATA2FILE *)

(*** MAIN ***)

BEGIN
  (*$N+*)
  SETCHAIN('#10:CHRONOAMP');
  RESET(FOO,'#10:CHANNELS');
  FILE2CHAN(0,CCHAN);FILE2CHAN(1,CELL);
  FILE2CHAN(2,ICHAN);FILE2CHAN(3,ECHAN);CLOSE(FOO);
  NAME:='BCKGRND';
  DATA2FILE(NAME)
END.
```

Figure C-1. (continued).

```
;-------------------------------------------------------------------------
; This routine is the heart of the program for acquisition of chronoampero-
; metric data.   The current data are acquired at a precalculated, fixed
; delay rate using a triply nested software delay timer.  From an assembled
; listing, the acquisition time for a single current value can be calc-
; ulated.  The digital data are placed in a prespecified location using in-
; direct addressing.  E1 is applied in the calling program, then the CV-27
; is switched from STANDBY to CELL in assembler; E2, the final potential,
; is then applied and the current data recorded.  The timing points are
; generated artificially in CADATA, the program which calls this routine.
;-------------------------------------------------------------------------
;POPS 16 BIT ARGUMENT
;
         .NOMACROLIST
         .MACRO POP16
         PLA
         STA %1
         PLA
         STA %1+1
         .ENDM

;POPS 8 BIT ARGUMENT & DISCARDS MSB
;
         .MACRO POP8
         PLA
         STA %1
         PLA
         .ENDM

     .PROC TAKEDATA,9 ;nine words of parameters
;-------------------------------------------------------------------------
;  PROCEDURE TAKEDATA (E2,PAIRS,DELAY1,DELAY2,DELAY3,CELL,ICHAN,ECHAN,
;                      MEMLOC:INTEGER);
;-------------------------------------------------------------------------
RETURN   .EQU 0        ;temporary return address
MEMLOC   .EQU 2        ;temporary MEMLOC address
ECHAN    .EQU 4        ;temporary ECHANNEL address
ICHAN    .EQU 5        ;temporary ICHANNEL address
CELL     .EQU 6        ;temporary CELL address
DELAY3   .EQU 7        ;temporary delay time addresses
DELAY2   .EQU 8
DELAY1   .EQU 9
PAIRS    .EQU 0A       ;number of data pairs address
E2       .EQU 0C       ;temporary E2 address
AI13     .EQU -16224.  ;starting address of AI13 for slot 2
A003     .EQU -14592.  ;starting address of A003 for slot 7

;-------------------------------------------------------------------------
;  PLACE PARAMETERS IN PAGE 0 LOCATIONS
;-------------------------------------------------------------------------
         POP16 RETURN ;save Pascal return address at 0,1
         POP16 MEMLOC ;place MEMLOC at 2,3
         POP8  ECHAN  ;place LSB of ECHAN at 4
         POP8  ICHAN  ;place LSB of ICHAN at 5
         POP8  CELL   ;place LSB of CELL  at 6
         POP8  DELAY3 ;place LSB of DELAY3 at 7
         POP8  DELAY2 ;place LSB of DELAY2 at 8
         POP8  DELAY1 ;place LSB of DELAY1 at 9
         POP16 PAIRS  ;place number of points at 0A,0B
         POP8  E2     ;place LSB of E2 at 0C

;-------------------------------------------------------------------------
;  SWITCH THE CV-27 FROM STANDBY TO CELL
;-------------------------------------------------------------------------
         LDY CELL     ;place remote timing input channel in Y
         LDA #255.    ;place 5.0V binary equivalent in accumulator
         STA A003,Y   ;send 5.0V to DAC output channel, switching to CELL

;-------------------------------------------------------------------------
;  APPLY E2 TO THE EXTERNAL INPUT
;-------------------------------------------------------------------------
         LDY ECHAN    ;place the DAC potential channel in Y
         LDA E2       ;place E2, the final voltage, in accumulator
         STA A003,Y   ;send voltage to DAC output
```

Figure C-2. Assembly code, which is linked to PROGRAM CADATA, and acquires chronoamperometric data.

```
;-------------------------------------------------------------------------
;  DATA ACQUISITION   ;CLOCK CYCLES/COMMENT
;-------------------------------------------------------------------------
;     (One CLOCK CYCLE=1.023 MHz)
;     (It takes 81 CLOCK CYCLES to acquire one current point)
;-------------------------------------------------------------------------
DATA      LDA ICHAN     ;3/start conversion on ICHANNEL
          STA AI13      ;4/
          PHA           ;3/delay 7 clock cycles to complete 4 MSB
          PLA           ;4/
          LDA AI13+1    ;4/get MSB
          AND #0F       ;2/mask off high BITS
          LDY #0        ;2/initialize Y index to 0 for indirect addressing
          STA @MEMLOC,Y;6/place at specified address
          INY           ;2/add 1 to Y
          LDA AI13      ;4/get LSB
          STA @MEMLOC,Y;6/place at next address
          CLC           ;2/clear CARRY to zero
          LDA 02        ;3/add 2 to MEMLOC
          ADC #02       ;2/this is a 16 bit addition
          STA 02        ;3/
          LDA 03        ;3/
          ADC #0        ;2/
          STA 03        ;3/
          SEC           ;2/set CARRY to 1
          LDA 0A        ;3/subtract 1 from PAIRS
          SBC #01       ;2/this is a 16 bit subtraction
          STA 0A        ;3/
          LDA 0B        ;3/
          SBC #0        ;2/
          STA 0B        ;3/
          ORA 0A        ;3/
          BEQ OUT       ;2,3/branch to out when PAIRS=0

;-------------------------------------------------------------------------
;  TIME DELAY ROUTINE
;-------------------------------------------------------------------------
;     (This routine was taken in part from, "6502 Software Design",
;      by Leo J. Scanlon(1980)83-91 )
;-------------------------------------------------------------------------
          LDX DELAY1    ;X contains 5-clock-cycle count
          LDY DELAY2    ;Y contains 1284-clock-cycle count
          LDA DELAY3    ;A contains 328,708-clock-cycle count
WAIT      DEX           ;minimum delay is 23.4604 usec, (X=Y=A=1)
          BNE WAIT      ;loop until X=0
          DEY
          BNE WAIT      ;loop until X=Y=0
          SBC #01       ;maximum delay is 84.149259 sec, (X=Y=A=255)
          BNE WAIT      ;loop until X=Y=A=0
          BEQ DATA      ;end of TIME DELAY ROUTINE

;-------------------------------------------------------------------------
;  SWITCH THE CV-27 FROM CELL TO STANDBY
;-------------------------------------------------------------------------
OUT       LDY CELL      ;place remote timing input channel in Y
          LDA #128.     ;place 0.0V binary equivalent in accumulator
          STA A003,Y    ;send 0.0V to DAC output channel, switch to STANDBY


;-------------------------------------------------------------------------
          LDA RETURN+1 ;restore Pascal return address
          PHA
          LDA RETURN
          PHA
          RTS           ;return to Pascal caller
         .END           ;end of assembly
```

Figure C-2. (continued).

APPENDIX D

LINEAR REGRESSION PROGRAM

```
(*$S++*)
PROGRAM LINREG;

(* This program does linear regression on any size data set.  It allows
   for changes to be made in incorrectly input data and removal of data
   the user subsequently does not want to include.  The algorithms were
   taken from: Caulcutt, R.; Boddy, R. "Statistics for Analytical Chem-
   ists," Chapman and Hall, New York, 1983, Ch. 5. *)

USES PRNTRTNS,TRANSCEND,INPTRTNS;

CONST Z = '96F';

TYPE  VALUES = ARRAY [0..1] OF REAL;

      NODE = RECORD
               VAL:VALUES;
               NEXT:^NODE
             END;

      LIST = ^NODE;

VAR SXX,SXY,SYY:REAL;   (* These are globals *)
    TITLE:STRING;
    DATA:LIST;
    KEY:CHAR;


PROCEDURE CR;
(* This routine prints a CR/LF. *)
  BEGIN
    FORMAT(Z)
  END; (* CR *)

FUNCTION FINDXA (L:LIST):REAL;
(* This routine computes the average X-value.  *)
  VAR X:REAL;
      COUNT:INTEGER;
  BEGIN
    X:=0.0;COUNT:=0;
    WHILE L <> NIL DO BEGIN
      COUNT:=COUNT + 1;
      X:=X + L^.VAL[0];
      L:=L^.NEXT
    END;
    FINDXA:=X/COUNT
  END; (* FINDXA *)

FUNCTION FINDYA (L:LIST):REAL;
(* This routine comutes the average Y-value.  *)
  VAR Y:REAL;
      COUNT:INTEGER;
  BEGIN
    Y:=0.0;COUNT:=0;
    WHILE L <> NIL DO BEGIN
      COUNT:=COUNT + 1;
      Y:=Y + L^.VAL[1];
      L:=L^.NEXT
    END;
    FINDYA:=Y/COUNT
  END; (* FINDYA *)

FUNCTION B (L:LIST; XA, YA:REAL):REAL;
(* This routine computes the slope, Sxx, Sxy, and Syy. *)
  VAR NUM,DEN:REAL;
      COUNT:INTEGER;
  BEGIN
    NUM:=0.0;DEN:=0.0;
    COUNT:=0;SYY:=0.0;
    WHILE L <> NIL DO BEGIN
      NUM:=NUM + (L^.VAL[0] * L^.VAL[1]);
      DEN:=DEN + (SQR(L^.VAL[0]));
      SYY:=SYY + (SQR(L^.VAL[1]));
      COUNT:=COUNT + 1;L:=L^.NEXT
    END;
    SXX:=DEN - (COUNT * SQR(XA));
    SXY:=NUM - (COUNT * XA * YA);
    SYY:=SYY - (COUNT * SQR(YA));
    B:=SXY/SXX
  END; (* B *)
```

Figure D-1. Linear regression analysis program.

```
FUNCTION MENU:CHAR;
(* This routine presents a menu of options for linear regression.*)
  VAR CH:CHAR;
  BEGIN
    CLRSCRN;
    PROMPTAT(0,6,'Menu selections:');WRITELN;
    PROMPTAT(2,7,'E>nter X,Y data');WRITELN;
    PROMPTAT(2,8,'D>isplay data and analysis');WRITELN;
    PROMPTAT(2,9,'R>emove a data point from the set');WRITELN;
    PROMPTAT(2,10,'G>o to a new data set');WRITELN;
    PROMPTAT(2,11,'C>hange a data point');WRITELN;
    PROMPTAT(2,12,'O>utput linear regression');WRITELN;
    PROMPTAT(2,13,'Q>uit');WRITELN;
    PROMPTAT(0,15,'Please type:');
    GETACHAR(CH,['E','e','D','d','C','c','G','g','O','o','Q','q','R','r']);
    WRITELN(CH);MENU:=CH
  END; (* MENU *)

PROCEDURE ENTER (VAR L:LIST);
(* This routine pushes a data pair on top of the list. *)
  VAR TEMP:LIST;
      X,Y:REAL;
  BEGIN
    CLRSCRN;
    NEW(TEMP);
    WITH TEMP^ DO BEGIN
      PROMPTAT(0,2,'Enter X value:');VAL[0]:=GETREAL;WRITELN;
      PROMPTAT(0,4,'Enter Y value:');VAL[1]:=GETREAL;WRITELN;
      NEXT:=L
    END;
    L:=TEMP
  END; (* ENTER *)

PROCEDURE DISPLAY (L:LIST);
(* This routine displays the data. *)
  VAR XAVG,YAVG,SLOPE,S,R:REAL;
      COUNT:INTEGER;
      TEMP:LIST;
      CH:CHAR;
  BEGIN
    CLRSCRN;
    IF L = NIL THEN EXIT(DISPLAY);
    PROMPTAT(3,0,'Point #       X-value       Y-value');
    PROMPTAT(0,1,'-----------------------------------------');WRITELN;
    COUNT:=1;TEMP:=L;
    WHILE L <> NIL DO BEGIN
      WRITELN('      ',COUNT,'       ',L^.VAL[0]:12,'  ',L^.VAL[1]:12);
      COUNT:=COUNT+1;L:=L^.NEXT
    END;
    L:=TEMP;
    IF L^.NEXT = NIL THEN BEGIN
      PROMPTAT(0,4,'Press <SPACE> to continue.');GETACHAR(CH,[' ']);
      EXIT(DISPLAY)
    END;
    XAVG:=FINDXA(L);YAVG:=FINDYA(L);SLOPE:=B(L,XAVG,YAVG);
    WRITELN;WRITELN('y = (',SLOPE,')x + ',YAVG - (SLOPE * XAVG));
    S:=SYY-(SQR(SLOPE) * SXX);
    R:=SXY/SQRT(SXX * SYY);WRITELN;
    WRITELN('Sum of residuals squared    =',S);
    WRITELN('Correlation coefficient (r) =',R);WRITELN;
    WRITELN('Press <SPACE> to continue.');GETACHAR(CH,[' '])
  END; (* DISPLAY *)
```

Figure D-1. (continued).

```
PROCEDURE OUTPUT (L:LIST);
(* This routine outputs the data. *)
  VAR XAVG,YAVG,SLOPE,S,R:REAL;
      COUNT:INTEGER;
      TEMP:LIST;
      CH:CHAR;
  BEGIN
    CLRSCRN;
    IF L = NIL THEN EXIT(OUTPUT);
    PRSTR(TITLE,Z);CR;CR;
    PRSTR('   Point #         X-value          Y-value',Z);CR;
    PRSTR('-------------------------------------------',Z);CR;
    COUNT:=1;TEMP:=L;
    WHILE L <> NIL DO BEGIN
      PRINT(COUNT,7,Z);PRSTR('         ',Z);PRREAL(L^.VAL[0],Z);
      PRSTR('     ',Z);PRREAL(L^.VAL[1],Z);CR;
      COUNT:=COUNT+1;L:=L^.NEXT
    END;
    L:=TEMP;CR;CR;
    IF L^.NEXT = NIL THEN EXIT(OUTPUT);
    XAVG:=FINDXA(L);YAVG:=FINDYA(L);SLOPE:=B(L,XAVG,YAVG);
    PRSTR('y = (',Z);PRREAL(SLOPE,Z);PRSTR(')x + ',Z);
    PRREAL((YAVG - (SLOPE * XAVG)),Z);CR;
    S:=SYY-(SQR(SLOPE) * SXX);
    R:=SXY/SQRT(SXX * SYY);CR;
    PRSTR('Sum of residuals squared     =',Z);PRREAL(S,Z);CR;
    PRSTR('Correlation coefficient (r) =',Z);PRREAL(R,Z);CR;CR;CR
  END; (* OUTPUT *)

PROCEDURE CHANGE (VAR L:LIST);
(* This routine allows the user to change the data values in the set. *)
  VAR COUNT:INTEGER;
      TEMP:LIST;
  BEGIN
    CLRSCRN;
    IF L = NIL THEN EXIT(CHANGE);
    COUNT:=0;
    TEMP:=L;
    WHILE L <> NIL DO BEGIN
      COUNT:=COUNT + 1;L:=L^.NEXT
    END;
    L:=TEMP;
    PROMPTAT(0,0,'Enter Point # to be changed (1..');WRITE(COUNT,'):');
    COUNT:=GETINT(1,COUNT);
    WHILE COUNT-1 <> 0 DO BEGIN
      L:=L^.NEXT;COUNT:=COUNT-1
    END;
    WITH L^ DO BEGIN
      PROMPTAT(0,2,'Enter new X-value:');VAL[0]:=GETREAL;WRITELN;
      PROMPTAT(0,4,'Enter new Y-value:');VAL[1]:=GETREAL;WRITELN;
    END;
    L:=TEMP
  END;  (* CHANGE *)
```

Figure D-1. (continued).

```
PROCEDURE REMOVE (VAR L:LIST);
(* This routine allows the user to remove data values in the set. *)
  VAR COUNT:INTEGER;
      PREV,TEMP:LIST;
  BEGIN
    CLRSCRN;
    IF L = NIL THEN EXIT(REMOVE);
    IF L^.NEXT = NIL THEN BEGIN
      L:=NIL;EXIT(REMOVE)
    END;
    COUNT:=0;TEMP:=L;
    WHILE L <> NIL DO BEGIN
      COUNT:=COUNT + 1;L:=L^.NEXT
    END;
    L:=TEMP;PREV:=L;
    PROMPTAT(0,0,'Enter Point # to remove <1..');WRITE(COUNT,'>:');
    COUNT:=GETINT(1,COUNT);
    WHILE COUNT-1 <> 0 DO BEGIN
      PREV:=L;L:=L^.NEXT;COUNT:=COUNT-1
    END;
    IF L^.NEXT = NIL THEN BEGIN
      PREV^.NEXT:=NIL;
      L:=TEMP
    END
    ELSE L:=L^.NEXT
  END;   (* REMOVE *)

PROCEDURE GO (VAR L:LIST);
(* This routine starts with a new list. *)
  BEGIN
    CLRSCRN;
    WRITE('Title of data set:');READLN(TITLE);
    L:=NIL
  END; (* GO *)


(*** MAIN ***)

BEGIN
  DATA:=NIL;GO(DATA);
  REPEAT
    KEY:=MENU;
    CASE KEY OF
      'E':ENTER(DATA);
      'D':DISPLAY(DATA);
      'C':CHANGE(DATA);
      'G':GO(DATA);
      'O':OUTPUT(DATA);
      'R':REMOVE(DATA);
      'Q':EXIT(PROGRAM)
    END
  UNTIL KEY = 'Q'
END.
```

Figure D-1. (continued).

VITA

Martin Todd Mansfield was born in San Diego, California, on December 2, 1958. After his sister was born in 1959, he moved with his family to Winter Park, Florida where he graduated from Winter Park High School in 1976. His sense of inspiration arose from his family and their achievements during these years. His Father is a talented mechanical/safety engineer in the Space Shuttle program. His Mother's compassion and loyalty are equally impressive. His eldest brother is a mechanical engineer and his elder brother recently completed his Ph.D. in Molecular Biology. His younger sister is a Financial Planner. All four children in his family have college degrees, and the three sons are Eagle Scouts. Both of his grandfathers and a great grandfather are graduates of the United States Naval Academy.

He began college at Rollins College in Winter Park, where he received elite instruction in Chemistry, Physics, and Biology and became interested in long distance running. In 1978, he left Rollins for a more versatile degree program at Texas A & M University, College Station. He completed his B.A., Cum Laude, in Biology and Chemistry in August 1981. He competed as a distance runner and pursued his interests in computers and mathematics during that time. He entered the graduate program in Chemistry at The University of Tennessee, Knoxville in August of 1981 where he pursued research in electrochemistry with Professor J. Q. Chambers. His research interests include process automation, chemical instrumentation, robotics, and microcomputer applications in Analytical Chemistry. Among his goals are a sub-2:20 marathon and a sub-1:10 half-marathon.