A STUDY OF CO OXIDATION OVER IR(110) WITH SURFACE SENSITIVE PROBES

Thesis by

James Long Taylor

In Partial Fulfillment of the Requirements

for the Degree of

Doctor of Philosophy

California Institute of Technology

Pasadena, California

1979

(Submitted May 26, 1978)

To my family and friends

Journalism largely consists in saying
"Lord Jones Dead"
to people who never knew Lord Jones
was alive.

- G. K. Chesterton

ACKNOWLEDGMENTS

I am most indebted to Blair Lewis, who taught me the principles of first-rate engineering and the sound approach to experimental research. Despite his many professional and personal commitments, he unselfishly gave me advice, particularly when I was incapable of taking action alone. Moreover, Blair introduced me to hiking, a form of escape that is most necessary for people at institutions like Caltech.

Dale Ibbotson also helped me a great deal. He assisted me in obtaining much of the data presented in this thesis. He was quite patient in our discussions of my unorthodox research endeavors.

I am grateful for the support of my research advisor, W. Henry Weinberg. The counsel of Sunney Chan, Gerhart Ertl, William P. Forniciari, Jr., William A. Goddard, III, Richard M. Lambert and William C. Taylor, III (my brother) is appreciated also.

Bill Bowser and Howie Evans did a damn good job in helping Blair Lewis and me so that we could get the PDP-11/10 computer up and running. The computer, my experimental apparatus, and most of my salary and tuition was paid for through the generous support of the National Science Foundation.

I thank the numerous people who perform crucial tasks in a research program and who, unlike engineers and scientists, neither demand nor expect recognition for their most necessary work. These people are the secretaries - Jennifer Burkhart, Kathy Lewis, and Susan Shaar - the machinists - George Griffith, Chic Nakawatase and Ray Reed - and the electronics whizzes - Mike Nagy and John Yehle.

Ted Watson aided me in my continuing crusade to keep the brewing industry solvent. Dan, the proprietor of the "Rendezvous", also

helped in this endeavor.

I am also indebted to my parents, Mary and William Taylor, and Louise and Paul Forniciari for their support and kindness.

ABSTRACT

The heterogenously catalyzed reaction of CO and O_2 to form CO_2 over (110), as well as the chemisorption of the individual reactants, has been studied at low pressures ($<10^{-5}$ torr). The experiments were performed with four surface sensitive probes - thermal desorption mass spectrometry (TDS), contact potential difference (CPD) measurements, Auger electron spectroscopy (AES) and LEED. Oxidation of the surface complicated several aspects of this study; that is, a second, tightly bound, dissociatively chemisorbed state of oxygen forms irreversibly on Ir(110) near 700 K. Hence, several of the experiments were performed for both clean and oxidized surfaces of Ir(110).

Chapter 2 presents a method for analyzing thermal desorption spectra. The method stresses the coverage dependence of the activation energy and pre-exponential factor in an Arrhenius expression. Techniques for assessing, avoiding and offsetting errors in TDS are presented also. Although this method was used for determing rate parameters throughout this study, the method is applied specifically to the desorption of CO in this chapter. As the coverage increases from 0 to 0.75 ML (1 ML = 10^{15} cm^{-2}), the activation energy decreases continuously from 35 to 20 kcal/mole whereas the pre-exponential factor decreases continuously from 10^{13} and 10^{9} sec^{-1}.

Chapter 3 considers the chemisorption of CO on Ir(110). The adsorption kinetics of CO are described by a second-order precursor model that accounts for ordering of the adlayer through intermolecular repulsion. The second-order model is apposite because two sites are needed for a CO molecule to adosorb, not because adsorption is

dissociative. A model is developed to fit the coverage and temperature dependence of CPD measurements for CO. The CPD changes non-linearly with coverage because of the depolarization of the overlayer from dipole-dipole interactions and changes in temperature through the thermal population of translational and vibrational modes. Inferences about the binding sites for CO could be drawn from CPD measurements because the CPD and the translational motion of the adsorbate are related. Also, a method is presented for evaluating the error in measurements of isosteric heats that arise when the Clausius-Clapeyron equation is evaluated for points of constant CPD, rather than constant coverage.

Chapter 4 describes the chemisorption of oxygen on Ir(110). Oxygen may chemisorb molecularly, may chemisorb dissociatively or may bind into the oxide layer. Above 200 K, oxygen chemisorbs dissociatively since the activation energy for dissociating molecularly chemisorbed oxygen is about 8 kcal/mole. Oxygen desorbs molecularly with an activation energy between 45 and 70 kcal/mole, which decreases continuously with increasing coverage. A second-order precursor model consistently describes the chemisorption of oxygen; that is, the logarithm of measured lifetime for the precursor is inversely proportional to the surface temperature between 300 and 700 K. Moreover, oxygen chemisorption is not activated. Measurements of the CPD upon oxygen chemisorption, which were analyzed by extending the methods presented in the previous chapters, indicate that the overlayer undergoes a structural transformation at a fixed coverage, which depends on whether or not the surface is oxidized.

Chapter 5 discusses the reaction of gaseous CO and O_2 to form CO_2 over Ir(110). Four elementary reactions - the desorption of CO, the oxidation of CO via the Langmuir-Hinshelwood mechanism, the adsorption of CO and the adsorption of oxygen - may limit the rate of CO oxidation under equilibrium conditions. The kinetics of each reaction are modeled from measurements involving TDS. The pre-exponential factors and activation energies for CO desorption and CO oxidation depend markedly upon the composition of the adlayer. Moreover, the diffusion of chemisorbed CO and oxygen may limit the kinetics of these processes. The co-adsorption of CO and oxygen is not, in the strictest sense, competitive; that is, CO blocks sites for oxygen adsorption, but oxygen does not block sites for CO adsorption. Despite these complications, a model developed from the kinetic expressions for the elementary reactions qualitatively and quantitatively predicts trends in the rate of CO_2 production under equilibrium conditions. These trends include the dependence of the rate upon surface temperature and the partial pressures of the reactants. However, hysteresis in the "equilibrium" rate of CO_2 production, which cannot be attributed to the oxidation of the Ir(110) surface, limits the applicability of this model.

Appendix B details an interactive programming language, POL, for monitoring experiments with a PDP-11/10 minicomputer. This language allows the user to control data acquisition, experimental parameters, the storage of data and the display of data, as well as to analyze data in an on-line environment. This language was used to expedite all experiments described above.

TABLE OF CONTENTS

CHAPTER 1:	INTRODUCTION	1
	References	10
CHAPTER 2:	A METHOD FOR ASSESSING THE COVERAGE DEPENDENCE OF KINETIC PARAMETERS: APPLICATIONS TO CARBON MONOXIDE DESORPTION FROM IRIDIUM (110)	12
	Introduction	14
	Theory	18
	An Isostere Analysis of CO Desorption from Ir(110)	22
	A Novel Analysis of CO Desorption from Ir(110)	27
	The Applicability of the Equilibrium Assumption	34
	Conclusions	40
	References	42
	Figures	44
CHAPTER 3:	THE CHEMISORPTION OF CO ON CLEAN AND OXIDIZED IR(110)	55
	Introduction	57
	Experimental	58
	Spectroscopic Characterization of the Overlayer	61
	Overlayer Structures	63
	Desorption Kinetics	66
	Adsorption Kinetics	73
	Charge Transfer to the Overlayer	77
	Conclusions	92
	References	95
	Figures	98

CHAPTER 4: THE CHEMISORPTION OF OXYGEN ON THE (110) SURFACE OF IRIDIUM 111

Introduction 113

Experimental 115

The Oxide Layer 118

Overlayer Structures 123

Desorption Kinetics 126

Adsorption Kinetics 130

Charge Transfer to the Overlayer 134

Molecularly Chemisorbed Oxygen 142

Conclusions 148

Appendix 151

References 154

Figures 157

CHAPTER 5: THE OXIDATION OF CARBON MONOXIDE OVER THE (110) SURFACE OF IRIDIUM 180

Introduction 182

Experimental 185

Rate Expressions for CO Desorption and CO Oxidation 194

The Co-adsorption Kinetics of CO and Oxygen 205

Equilibrium Rates of CO_2 Production 211

Conclusions 224

References 228

Figures 230

CHAPTER 6: CONCLUSIONS 250

APPENDIX A: A METHOD OF CONTINUOUSLY MEASURING WORK FUNCTION CHANGES 259

APPENDIX B: POL-AN INTERACTIVE PROGRAMMING LANGUAGE FOR EXPERIMENTAL RESEARCH 275

Part I: POL Users' Manual 276

Part II: POL Software Manual 413

APPENDIX C: THE DESIGN OF A HIGH-CURRENT POWER SUPPLY WITH A SERVO-MECHANISM FOR CONTROLLING THE TEMPERATURE OF A SUBSTRATE 562

CHAPTER 1

INTRODUCTION

The motivation for this thesis was to gain a more fundamental understanding of the thermodynamics, kinetics and chemistry of gas-surface interactions that may influence heterogeneously catalyzed processes. More specifically, this study is concerned with: measuring the thermodynamic properties of an adsorbed phase, assessing the extent of adsorbate-adsorbate and adsorbate-adsorbent interactions, defining the chemical steps in a heterogeneously catalyzed reaction, developing rate expressions for these steps, locating binding sites for adsorbates, and identifying the chemical nature of adsorbed species.

The method of this thesis was to study the heterogeneously catalyzed reaction of gaseous CO and O_2 to form CO_2 over Ir(110), as well as the chemisorption of the reactants. Gas-surface interactions in this system were investigated using four surface sensitive probes - thermal desorption mass spectrometry (TDS), contact potential difference (CPD) measurements, Auger electron spectroscopy (AES), and low energy electron diffraction (LEED). The use of these probes limited this study to total pressures below 10^{-5} torr whereas the surface temperature was varied between 90 and 1800 K. The general experimental plan of this work and the geometry of an Ir(110) surface are given in Fig. 1. The surface of Ir(110), like the surface of the (110) face of other fcc metals, is composed of a series of rows of atoms separated by troughs. The distance between the centers of the rows is 3.84Å, and the distance between nearest neighbors in the rows is 2.72Å. The area enclosed by the square superposed on the surface is termed a unit cell. All fractional coverages in this study are given in monolayers (ML) where

$$CO_{(g)} + O_{2(g)} \xrightarrow[90\text{-}1800K]{IR\,(110)} CO_{2(g)}$$
$$P_T < 10^{-5}\ \text{Torr}$$

FCC (110) SURFACE

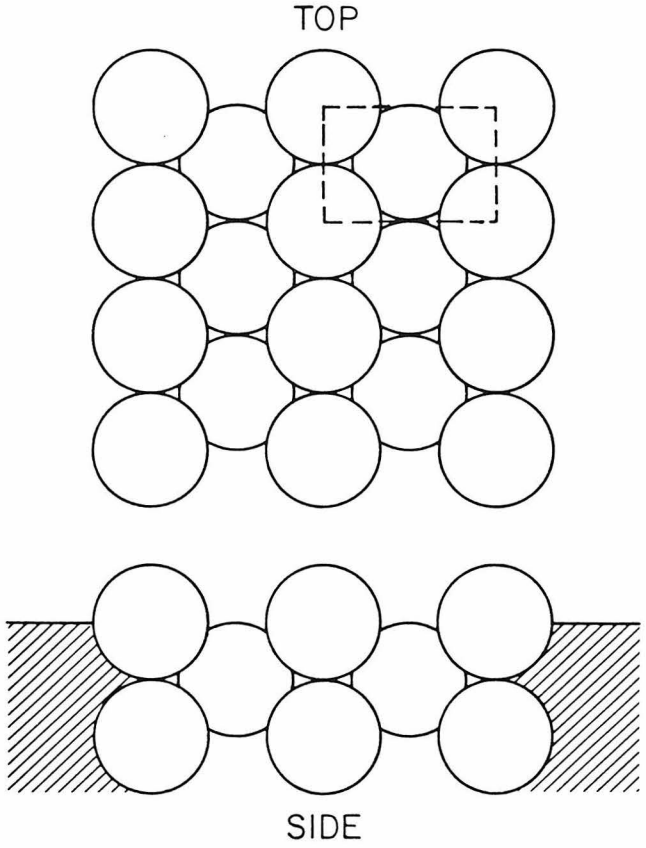


Fig.1

a coverage of one monolayer corresponds to one adsorbate filling each unit cell or to 10^{15} adsorbates covering 1 cm^2 of the Ir(110) surface.

The rational for studying the CO oxidation reaction over Ir(110), rather than other catalytic systems, is as follows. This study concentrated on a single reaction so that several catalytic phenomena could be related. Although the commercial importance of CO oxidation is limited mainly to the control of automobile emissions, the use of surface sensitive probes to study catalysis at low pressures is limited necessarily to gaseous reactants of low molecular weight and simple atomic structure. The data obtained with these probes are difficult to interpret even when the simplest of reactants are studied. Moreover, the oxidation of CO has been studied extensively on other surfaces of the platinum metals besides Ir(110) so that the properties of several catalysts could be compared. The oxidation of CO is studied more easily over iridium, compared to the other platinum metals. As will be shown in subsequent chapters, CO does not dissociate on Ir(110) at low pressures, and adsorbed oxygen does not diffuse into the bulk. A single crystal, rather than a polycrystalline foil, was selected as a catalyst because measurements involving some surface sensitive probes, in particular LEED, CPD measurements and TDS, are more reproducible if the substrate is a single crystal. Moreover, LEED and CPD measurements may be interpreted most easily if a low index orientation, such as the (110) surface of an fcc metal, is chosen as the substrate.

The presentation of this thesis is as follows. Chapter 2 presents a novel method for analyzing thermal desorption spectra, which was used

for developing rate expressions throughout this study. The chemisorption of CO and oxygen on Ir(110) is described in Chapters 3 and 4, respectively. Chapter 5 deals with the heterogeneously catalyzed reaction of CO and O_2 to form CO_2 over Ir(110). Following the conclusions in Chapter 6, three appendices detail the experimental innovations introduced in this work. A method for obtaining reliable and reproducible CPD measurements is presented in Appendix A. Appendix B details POL, an interactive programming language for monitoring experiments with a PDP-11/10 computer. Appendix C describes a high current power supply used for heating the substrate in this study, as well as a servomechanism used for controlling the temperature of the substrate.

To orient the reader to the terminology and the material in the succeeding chapters, the remainder of this introduction will survey the experimental techniques for investigating the chemisorption and reaction of molecules on surfaces at low pressures ($<10^{-5}$ torr) and will review briefly the theoretical techniques for modeling these gas-surface interactions. Specifically, the following topics are discussed: the thermodynamic properties of an overlayer, rates of chemisorption, rates for the activated processes of desorption and reaction, and the structure and chemical nature of the overlayer. If the reader requires background information concerning CO chemisorption, oxygen chemisorption, or the reaction of gaseous CO and O_2 to form CO_2 over the platinum metals, he is referred to the introductions of Chapters 3, 4 and 5, respectively.

Terrell Hill (1) first formulated the thermodynamic variables for surfaces in terms of partial molar quantities. This formulation stressed

the dependence of thermodynamic quantities upon the coverages of various adsorbates in the overlayer. Subsequently, several models were developed to predict the variation in the isosteric heat and the differential entropy of adsorption with coverage (2-4). Experimentally, the isosteric heat of adsorption (q_{st}) for a single adsorbate is evaluated from the Clausius-Clapeyron expression, $[\partial \ell nP_i/\partial(\frac{1}{T})]_{\theta_i}=q_{st}/k$, where k,T,P_i and θ_i are the surface temperature, the Boltzmann constant, the partial pressure of the adsorbing species and the coverage of the adsorbate. Other thermodynamic variables, such as the differential entropy of adsorption and the spreading pressure of a species, usually may be calculated from the same set of measurements from which the isosteric heat is derived (1). Although the partial pressure and the temperature in a thermodynamics experiment may be measured easily, the coverage is more difficult to obtain. Specifically, TDS (5,6) (also referred to as the flash filament technique in early studies), CPD measurements (7,8) and AES (9) have been used for measuring surface coverages in thermodynamics experiments. In these experiments, the gas is seldom in thermal equilibrium with the surface, but the error introduced through this disparity (5) (~1 kcal/mole) is typically small compared to the error introduced when coverages are measured with these techniques.

The rate of chemisorption for a species is the relation between the number of molecules that impinge on the surface to the number that chemisorb. At low pressures, the number that impinge is proportional to the product of time and the partial pressure of the species where

the proportionality constant is given by the statistical-mechanical formulation of the motion of dilute gases (10). The number that stick to the surface is determined with TDS or through molecular beam experiments (11). In the limit of zero coverage, the probability that a molecule impinging on the surface will chemisorbed is termed the initial sticking probability. If the adsorption process is activated, the logarithm of the initial sticking probability is inversely proportional to the surface temperature where the proportionality constant gives the activation energy (12). If the adsorption process is non-activated, the initial sticking probability is independent of temperature. For partially covered surfaces, the probability of adsorption depends on the lifetime of the precursor (13). This is the average time that a physisorbed molecule will remain on the surface to search for an empty chemisorption site. Although physisorbed molecules have been observed with field emission microscopy (14), few studies have tested the validity of precursor models. In such experiments, the temperature dependence of the adosrption kinetics is measured for larger coverages (greater than 25% of saturation coverage). If the lifetime of the precursor is extremely short, the adsorption kinetics are fit by Langmuir models; that is, the rate is proportional to the number of empty sites or the number of empty sites squared (11). The rate depends on the number of empty sites if one empty site is needed for chemisorption to occur, but depends on the number of empty sites squared if two adjacent and empty sites are necessary for chemisorption to occur. Obviously, the ordering of the adlayer following chemisorption may influence second-order behavior so that models that account

for intermolecular forces among adsorbates have been developed (11).

The activated processes of desorption from a surface and of the reaction of two molecules on a surface are modeled by Arrhenius expressions. In this expression, the rate is proportional to the product of a pre-exponential factor, the exponential of the ratio of an activation energy to the surface temperature, and one or more coverages with integral exponents. Rates of activated processes are measured most often with TDS (15). In this technique, the desorption of various molecules from a surface is monitored with a mass spectrometer while the substrate is heated. If the temporal variation in the temperature of the substrate and the removal rates for the gaseous products are known, the rate parameters-the activation energy and the pre-exponential factor - may be determined (16). Although the methods for calculating the temporal variation in coverage for a thermal desorption spectrum have been established for quite sometime (15), methods for determining the coverage dependence of the activation energy have been developed only recently (17). Moreover, the coverage dependence of the pre-exponential factor is only now being tested (18). The activation energy is expected to vary markedly with coverage. This follows since the activation energy of desorption for a single component system nearly equals the isosteric heat of adsorption for the pertinent coverage if the adsorption of the component is not activated (5).

Many theoretical studies have been undertaken to elucidate the structure and the chemical nature of the adsorbed layer (19). Also, several experimental techniques have been developed to examine this

problem. Although low energy electron diffraction (LEED) readily gives the periodicity of an overlayer with respect to the substrate, the registry of the overlayer with respect to the substrate may be obtained only through complex calculations (20). Angle-resolved ultraviolet photoemission spectroscopy (ARUPS) may be used to give the bonding angle of a diatomic molecule with respect to the surface normal, but, once again, complex calculations are involved (21). Recently, the coverage and temperature dependences of CPD measurements have been used in inferring information concerning the binding sites of adsorbates (22,23). These measurements indicate that the concept of a binding site may not be entirely appropriate for some adsorbates, for two or more types of sites may be populated as the surface temperature is increased.

The object of this thesis is to test, extend and innovate these experimental and theoretical methods for examining gas-surface interactions.

References

1. T. L. Hill, J. Chem. Phys. 18, 246 (1950).
2. C. Kemball, Adv. Catalysis 2, 233 (1952).
3. J. H. de Boer, The Dynamical Character of Adsorption, 2nd Ed., Clarendon Press, Oxford, 1968.
4. W. H. Weinberg, in Surface Effects in Crystal Plasticity, R. M. Latanison and J. F. Fourie, Ed., Noordhoff, Leyden, 1977.
5. P. Kislink, J. Chem. Phys. 30, 174 (1959).
6. E. Rideal and F. Swett, Proc. Roy. Soc. A 257, 291 (1960).
7. J. C. Tracy and P. W. Palmberg, J. Chem. Phys. 51, 4852 (1969).
8. H. Conrad, G. Ertl, J. Koch and E. E. Latta, Surface Sci. 43, 462 (1974).
9. R. W. Joyner, J. Rickman and M. W. Roberts, J. C. S. Faraday I 70, 1825 (1974).
10. W. H. Weinberg, C. M. Comrie and R. M. Lambert, J. Catal. 41, 489 (1976).
11. D. A. King and M. G. Wells, Proc. Roy. Soc. A 339, 245 (1974).
12. G. Ehrlich, J. Phys. Chem. Solids 1, 3 (1956).
13. P. Kislink, J. Phys. Chem. Solids 3, 95 (1957); 5, 78 (1958).
14. R. Lewis and R. Gomer, Surface Sci. 12, 157 (1968).
15. G. Ehrlich, J. Appl. Phys. 32, 4 (1961).
16. P. A. Redhead, Vacuum 12, 203 (1962).
17. J. L. Falconer and R. J. Madix, J. Catal. 48, 262 (1977).
18. T. Engel, T. von dem Hagen and E. Bauer, Surface Sci. 62, 361 (1977).
19. G. Doyen and G. Ertl, Surface Sci. 43, 197 (1974).

20. P. J. Estrup and E. G. McRae, Surface Sci. 25, 1 (1971).

21. C. L. Allyn, T. Gustafsson and E. W. Plummer, Chem. Phys. Letters 47, 127 (1977).

22. K. Christmann and G. Ertl, Surface Sci. 60, 365 (1976).

23. G. Ertl, M. Neumann and K. M. Streit, Surface Sci. 64, 393 (1977).

CHAPTER 2

A METHOD FOR ASSESSING THE COVERAGE DEPENDENCE OF KINETIC PARAMETERS: APPLICATIONS TO CARBON MONOXIDE DESORPTION FROM IRIDIUM (110)

(The text of Chapter 2 consists of an article coauthored with W. H. Weinberg that has been submitted to Surface Science for publication.)

Abstract

A method for analyzing thermal desorption mass spectra has been developed for determining the coverage dependence of the pre-exponential factor and the desorption energy in the Arrhenius equation for a one-step desorption process. The method, which involves variable heating schedules, is applicable to spectra in which several features appear. If the desorption process involves multiple steps, or if substantial desorption from multiple sites occurs for any one coverage, this method cannot be used. The experimental difficulties of thermal desorption techniques, particularly the problem of finite removal rate of the desorption products, are treated to show other limitations of this analysis procedure. The method is applied to CO desorption from the (110) surface of Ir. Three features can be resolved in the desorption spectrum. Both the pre-exponential factor and the desorption energy vary strongly with coverage, and a compensation effect occurs.

1. Introduction

Thermal desorption mass spectrometry (TDS) is one of the most common techniques used in studies of gas-surface interactions. General descriptions of this technique may be found in the literature (1). Adsorbate coverages, thermodynamic properties, and chemical rates are accessible through TDS. This study focuses on the measurement and interpretation of desorption rates for systems in which the rate coefficient varies strongly with both the adsorbate concentration and the surface temperature. The modeling of TDS will be described briefly to expose the advantages and limitations of a novel method of data analysis which stresses the coverage dependence of desorption rates.

In TDS an adsorbate is desorbed from a surface by heating the substrate. The gaseous desorption products are detected by a mass spectrometer. Three relations are needed to describe desorption in a single component system. These are a mass balance on the surface [Eq. (1)], a heating schedule [Eq. (2)], and a mass balance on the gas phase [Eq. (3)] (2)

$$-dN_s/dt = R_d(N_s,T)$$

$$= \nu(N_s,T)N_s^n \exp[-E_d(N_s,T)/kT] \qquad (1)$$

$$T = f(t) \qquad (2)$$

$$dN_g/dt = R_d - N_g/\tau \qquad (3)$$

where N_s is the number of adsorbed molecules, N_g is the number of molecules in the gas phase, T is the temperature of the substrate, t is the time, and τ is the pumping time, i.e., the life-time of a gaseous

molecule. Eq. (3) applies for systems in which desorbing species are pumped continuously at a rate proportional to number of molecules in the gas phase. Over the short duration of a TDS experiment, τ is constant.

The mass balance for the surface is the Arrhenius expression. Readsorption of the desorbed species is ignored, since this effect is made negligible through a careful choice of experimental conditions. The desorption rate (R_d) depends on the reaction order (n) and two rate parameters, a pre-exponential factor [$\nu(N_s,T)$], and the activation energy for desorption [$E_d(N_s,T)$]. Use of this expression implies that desorption occurs through a one-step mechanism or through one dominant, rate-determining step. When many steps are important, the analysis of TDS data by any method is difficult, if not impossible. Data analysis is also difficult if the adsorbate desorbs from multiple sites at comparable rates for any <u>particular</u> coverage. Throughout this paper, these two situations are ignored.

If Eq. (1) represents an elementary chemical reaction, n is an integer that is independent of temperature and surface concentration. Otherwise, Eq. (1) is not strictly applicable for the mass balance. A frequently ignored aspect in TDS analysis is the coverage dependence of the rate parameters. Both activated complex theory (<u>3</u>) and the collision theory of reaction rates (<u>4</u>) indicate that the pre-exponential factor and the desorption energy are related to thermodynamic quantities. Therefore, the rate parameters are functions of coverage, temperature, surface area, and volume of the adsorbed phase. Since the last two quantities are fixed in most desorption systems, the dependence of the rate parameters on these variables will be ignored. The temperature

dependence can be ignored also. The desorption energy varies with temperature through the excitation of vibrational modes in the adsorbed phase (4). This contribution to the desorption energy is usually small compared to the depth of the potential energy well. In activated complex theory, the pre-exponential factor is related exponentially to entropy. Typically, entropy terms vary logarithmically with temperature (5) so that the pre-exponential factor varies linearly or to some small power in temperature. This temperature dependence of the desorption rate is overshadowed strongly by the factor $\exp[-E_d(N_s)/kT]$. Since the temperature range in most desorption experiments is limited, the temperature dependence of the two rate parameters has not been resolved.

The coverage dependence of the rate parameters, unlike the temperature dependence, cannot be ignored. In several systems, such as CO adsorption on the Group VIII transition metals, the coverage dependence of the desorption energy frequently is reported (6), but the coverage dependence of the pre-exponential factor seldom is examined.

The analysis of TDS data frequently involves a simplification of Eqs. (2) and (3). The heating schedule often is assumed to be linear in time, $f(t) = \beta t$, where β is the average heating rate (7). In many experimental systems, the sample is heated conductively so that the temperature varies exponentially, rather than linearly, with time (8). If Eq. (1) is integrated with the assumption of a linear heating schedule, deviations from linearity introduce a cumulative error in the result. This paper presents a method of analysis which completely circumvents any approximations concerning the heating schedule.

One other approximation, involving Eq. (3), is made consistently

in analyses of desorption data. The term dN_g/dt is taken to be zero (7). This approximation is referred to as the equilibrium approximation. The lifetime of a desorbed molecule (τ) is assumed to be short compared to the temporal variation of the desorption rate. Few, if any, methods have been developed to assess the validity of this approximation. A method for testing the validity of, and correcting for, the equilibrium approximation is presented in this paper.

Several problem areas in current methods of analyzing desorption spectra have been described above. These areas include a number of questionable assumptions concerning the variation of the rate parameters with surface coverage, the form of the heating schedule, and the pumping speed in the experimental system. Two objectives will be pursued in this study. First, a method for analyzing desorption spectra is presented which involves a minimum of assumptions. Second, criteria are developed for assessing the validity of the few assumptions that are necessary. Other methods of analyzing desorption spectra will be discussed simultaneously with the development of this novel method. In this way, the relative merits of the various techniques are emphasized.

2. Theory

Methods for analyzing thermal desorption spectra are numerous. Most methods can be categorized as requiring two spectral features for determining the pre-exponential factor and the desorption energy (9). Typically, these features are the temperature corresponding to the maximum desorption rate, and either the maximum desorption rate or the thermal width of the desorption peak. The coverage dependence of the rate parameters is ignored generally. A linear or fixed heating schedule must be assumed also. All methods in this class are refinements of the pioneering work of Redhead (7). These refinements are often of limited value since these methods of analysis neglect an important physical phenomenon, namely, the coverage dependence of rate parameters.

A few methods make use of several desorption spectra and many points in each spectrum (10). With this approach, the coverage dependence of the rate parameters may be determined. The most noteworthy of these techniques is the isostere method developed by Falconer and Madix. The method described in this paper and the isostere method embody many similar concepts.

To assess the coverage dependence of the rate parameters, one must realize that a desorption spectrum reflects the temporal variation of surface coverage (11). If Eq. (3) is integrated with the equilibrium assumption, $\frac{dN_g}{dt} = 0$, and if the number of gaseous molecules is proportional to the signal from the mass spectrometer, the number of adsorbed molecules is proportional to the integral of the mass spectrometer signal with time

$$N_s^o - N_s = \eta \int_o^t I dt \qquad (4)$$

where N_s^o is the initial number of adsorbed molecules on the surface. The signal intensity is presumed to be proportional to the number of gaseous molecules. The constant (η) contains this proportionality constant and the lifetime (τ). If the initial number of adsorbed molecules is known (12), then η may be calculated from

$$\eta = N_s^o / \int_0^\infty I dt \tag{5}$$

Once the temporal variation of surface coverage is established, thermal desorption spectra may be analyzed exclusively through the surface mass balance [Eq. (1)]. Moreover, integration of this relation is unnecessary. If the desorption rate, or equivalently the mass spectrometer intensity, and the surface temperature are monitored as independent functions of time, then the number of adsorbed molecules and these two measured parameters are known at any point in time. Then the number of adsorbed species may be considered as the independent variable in Eq. (1), that is,

$$\begin{aligned} R_d(N_s,T) &= \eta I(N_s,T) \\ &= -\nu(N_s)\, N_s^n \exp[-E_d(N_s)/kT] \end{aligned} \tag{6}$$

If several spectra are measured by altering the heating schedule, the desorption rate and the sample temperature may be varied for any particular coverage. An Arrhenius plot of $\ell n(I)$ versus $1/T$ at a fixed coverage gives a line with slope $-E_d(N_s)/k$ and intercept $\ell n[\nu(N_s)N_s^n/\eta]$ (13).

Several subtle points are contained in this approach. If both intensity and sample temperature are monitored as independent functions of time, no special form for the heating schedule need be assumed in the

analysis. Any errors arising from an assumed form are avoided completely. Although the desorption energy is directly accessible, both the constant (n) and the order of the reaction (n) must be known to determine the pre-exponential factor. The order is often accessible through auxiliary chemical information. For example, if CO is adsorbed dissociatively, the expected order is two; if CO is adsorbed molecularly, the expected order is one. Another method for determining the order is to plot $\ell n\ R_d$ versus $\ell n\ N_s$ at points of constant temperature and for low surface coverages (8) where the rate parameters often depend very weakly on coverage (9, 10).

In the isostere method of Falconer and Madix (10), which is somewhat similar to this technique, order plots, $\ell n\ R_d$ versus $\ell n\ N_s$, are used in analyzing desorption spectra. One fundamental difference between the two techniques is that the order, rather than the pre-exponential factor is considered to be a function of coverage in the isostere technique. The coverage dependence of both parameters cannot be determined independently from an Arrhenius plot since both quantities appear in the intercept. If the surface mass balance represents an elementary chemical reaction, the order certainly must be independent of coverage. Otherwise, Eq. (1) alone does not describe the desorption process. When the rate parameters vary markedly with coverage, the order appears to change with coverage, as will be shown.

Another difference between the two methods is the technique used in varying the desorption rate and the surface temperature for a fixed coverage. Falconer and Madix chose to alter the initial surface coverage, rather than the heating rate, to produce this variation (10). The

latter technique produces a larger range of surface temperatures so that the experimental accuracy is improved.

In summary, this new method of thermal desorption analysis has several features that are not found in other methods. The coverage variation of the pre-exponential factor is determined. No special form of the heating schedule need be assumed in the analysis. The method of varying the surface temperature for a particular coverage enhances the accuracy of the analysis.

The remainder of this paper demonstrates the power of this technique in analyzing desorption spectra. Specifically, the desorption of CO from Ir(110) is discussed. Three features appear in the desorption spectra, and the rate parameters are strong functions of coverage. The relative advantages of this method over others, particularly the isostere method will be stressed also. Three topics are presented: (1) an analysis of CO desorption from Ir(110) with the isostere method; (2) an analysis of the same system with this new technique, and (3) the treatment of artifacts arising from the inapplicability of the equilibrium assumption.

3. An Isostere Analysis of CO Desorption from Ir(110)

The experimental system used in these studies has been described elsewhere (14). The method of mounting the substrate was changed, however. Two 10 mil Ta wires, which heat the sample resistively, were spot-welded to the sample and were mounted on the sample manipulator. Both the quadrupole mass spectrometer (UTI-100C) intensity and the thermocouple (Pt - Pt + 10% Rh) EMF were monitored as independent functions of time by a PDP11-10 computer (15). Both variables could be sampled at a frequency of 60 Hz or less. Several data analysis programs enhanced the utility of the computer and the accuracy of the analysis. For example, one program performed the intensity-time integration which fixes the temporal variation in surface coverage. The Ir (supplied by Materials Research Corp.) was oriented to within one degree of the (110) surface through standard preparation and polishing techniques (14). Final *in situ* cleaning of the surface consisted of several oxidation and reduction cycles (16). The CO was obtained from Matheson Gas Products and was of 99.99% purity.

Representative samples of 40 desorption spectra of CO from Ir(110) are shown in Fig. 1. The initial surface coverage for each spectrum was varied by altering the gas exposure to the sample. All exposures were performed at 300 K. The average heating rate was fixed at 36 K/sec.

The data in Fig. 1 were analyzed with the isostere method of Falconer and Madix (10). Order plots, $\ell n\ I$ as a function of $\ell n\ \theta$ for fixed surface temperatures, were constructed from these data and are shown in Fig. 2. The coverage (θ) is the surface population (N_s) divided by the population at saturation (N_s^{max}). The desorption order appears to be one only at

low coverages where the rate parameters might be expected to be weak functions of coverage. At higher coverages, the desorption process appears to be eighth order or may involve multiple, distinct sites. As will be shown, neither artifact occurs. Instead, the rate parameters are strong functions of coverage. On this basis, the usefulness of order plots can be questioned. However, the authors have found that order plots are valuable for examining desorption phenomena at low coverages, in particular oxygen desorption from Ir(110) (17). However, caution must be exercised in interpreting order plots at coverages from 10-100% of saturation.

In a strict application of the isostere method, Arrhenius plots of ℓn I versus 1/T at points of constant coverage are constructed from the order plots to determine the desorption energies. This procedure gives desorption energies with very poor accuracy for two reasons. First, many of the lines in Fig. 2 should be curved since the rate parameters vary with coverage. Second, the temperature changes only slightly for a fixed coverage; that is, errors in the measured parameters are comparable in magnitude to the range over which the parameters are varied. The first problem is overcome by constructing Arrhenius plots from the original data. Yet, desorption energies with uncertainties of 10 to 20%, as determined from the least squares deviation of the plot, were obtained. Better estimates of the desorption energy certainly can be derived from adsorption isosteres (18) if the adsorption is not activated.

The second problem, the weak variation of temperature for any surface coverage, induces the poor accuracy. Two techniques can produce this variation. Either the heating schedule or the initial coverage can

be altered. (The initial sample temperature also can be changed, but this is not practical since the initial temperature is selected so that the desorption rate is negligible.)

The relative merits of the two techniques are illustrated by a simple mathematical analysis of Eqs. (1) and (2). The desorption behavior of a system with first order kinetics was modeled. The rate parameters were considered to be independent of coverage with E_d = 30 kcal/mole and $\nu = 3 \times 10^{12}\ sec^{-1}$. The heating schedule was linear with β = 30 K/sec. This model reduces to Eq. (7) which was solved for θ as a function of T on an HP-25 programmable calculator using the Euler method

$$d\theta/dT = \left(\frac{\nu}{\beta}\right)\ \theta\ \exp\ [-E_d/kT] \qquad (7)$$

The solid curves in Fig. 3 were obtained by varying the initial coverage.

For the isostere method to succeed, the temperature at a particular coverage must be known accurately, compared to the range over which the temperature is varied. For a 10% accuracy, the temperature must be known within two degrees (as will be shown). Errors in coverage, arising from variations in the constant η, are optimistically ±1% of the maximum coverage. These two constraints imply that the slope ($\Delta\theta/\Delta T$) should exceed 0.005 K^{-1} at a point chosen for an isostere plot. Greater accuracy is derived if the slope is larger. If the initial coverage is varied to alter the temperature at a particular coverage, the largest useful temperature range for any coverage is about 25 K. This range is smaller both at high and low coverages. Because curves corresponding to higher initial coverages tend to coalesce, curves with smaller slopes cover much of the temperature range. This coalescence also expands the data set needed for an isostere analysis. A temperature-coverage curve

for a particular initial coverage produces useful data only for coverages near that initial coverage.

A better way to alter the temperature for a fixed coverage is to vary the heating rate (β). The two dashed curves in Fig. 3 are generated from Eq. (7) for heating rates of 3 and 300 K/sec. Since the slope ($d\theta/dT$) remains large for most of the temperature-coverage curve, points for an Arrhenius plot may be selected accurately except for very small or very large coverages. For higher heating rates, the slope for any coverage decreases somewhat, but not substantially. The most conspicuous advantage of varying the heating rate is that a larger temperature range for any coverage is produced. A hundred-fold variation in heating rate generates at least a 80 K temperature range for all coverages. Therefore, one desorption spectrum gives accurate data for all coverages so that the data set required for an Arrhenius plot is smaller than that needed if the initial coverage is varied.

Because varying the heating rate tends to shift the spectra in temperature, rather than alter their shape, temperature variations for a fixed coverage predictably follow the variations in heating rate. Eq. (7) illustrates that the desorption rate at a fixed coverage may be doubled in two ways. Either the temperature may be increased by 10 K, or the heating rate may be halved. Doubling the heating rate shifts the temperature-coverage curve upward in temperature by nearly 10 K.

In principle, varying the heating rate, rather than the initial coverage, has been shown to be a better technique for obtaining Arrhenius plots. The practicality of this technique can be questioned since large heating rates may make the equilibrium assumption, $\frac{dN_g}{dt} = 0$, invalid.

However, the next section demonstrates that varying the heating rate is highly practical for obtaining accurate Arrhenius plots. The following section assesses the validity of the equilibrium assumption for these data.

4. A Novel Analysis of CO Desorption from Ir(110)

Another set of ten desorption spectra of CO from Ir(110) was obtained in the vacuum system described in the previous section. The average heating rate, rather than the initial coverage, was varied for each spectrum. At the beginning of each experiment, the surface at 300 K was exposed to 10.0 L (1L = 10^{-6} torr-sec) of CO. The initial fractional coverage, as determined by desorption kinetics (16), was 0.82 $\pm$ 0.01. The consistency in coverage was verified by comparing values of the constant (n) from experiment to experiment. The average heating rate was varied from 4.5 to 212 K/sec. The linearity of the heating schedule is unimportant in data analysis since the intensity and temperature were sampled independently in time.

Three of the ten spectra and the associated temperature-coverage curves are shown in Fig. 4. These curves were calculated by integrating Eq. (3) with the equilibrium assumption. Increasing the heating rate tends to shift the spectra to higher temperatures and increase the desorption rate, or equivalently the intensity. However, the intensity-time area for any peak is consistent from experiment to experiment. Arrhenius plots for points of constant coverage were constructed from the data in Fig. 4 and are shown in Fig. 5.

This procedure might be incorrect since two features appear in the desorption spectra. Desorption could occur from more than a single type of adsite. However, the plots in Fig. 5 would not be linear if desorption from multiple sites was occurring. Lines with confidence limits of 98% or better can be fit to the data. Desorption from multiple sites is unimportant for a more fundamental reason. If the difference in

desorption energy between two sites is 2 kcal/mole, if the pre-exponential factors for the sites are equal, and if the surface temperature is 500 K; then the desorption rate for the more tightly bound site is about ten times smaller than that of the more weakly bound site. For CO desorption from Ir(110), the peaks are separated by 4 kcal/mole or more so that simultaneous desorption from multiple sites need not be considered.

The desorption energy is directly accessible from the Arrhenius plots. Since the pre-exponential factor is presumed to vary with coverage, the order of the desorption process (n) must be known to fix the pre-exponential factor. The order plot for low coverages (Fig. 2) indicates that the order of the desorption reaction is one. The Auger spectra in Fig. 6 also suggest that the order is one. If CO adsorbs dissociatively, the order is expected to be two; if adsorption is molecular, the order should be one. Adsorption is molecular since the oxygen KLL transition of adsorbed CO is quite similar to that of gaseous CO but is rather different from that of chemisorbed oxygen which adsorbs dissociatively at 300 K (19).

Since the order is clearly one and the constant (η) is known from absolute measurements of coverage, the pre-exponential factor may be derived from the intercept $\left[\ell n\left(\nu(\theta_s)\theta/\eta\right)\right]$. In Fig. 7, the desorption energy and the natural logarithm of the pre-exponential factor are plotted as functions of coverage. Error bars on the rate parameters, which were determined from the uncertainties in least squares fits of the Arrhenius plots, are conservative. Most desorption energies are accurate to ±1% or better. Points at which the rate parameters vary strongly with coverage correspond to the appearance of new desorption

features and to alterations in the overlayer structure. Partially disordered LEED patterns, a p(2x2) and a (4x2), appear near fractional coverages of 0.25 and 0.75, respectively (16).

Fig. 7 demonstrates that the compensation effect (3) occurs for this system. The desorption energy and the pre-exponential factor vary sympathetically with coverage, or, equivalently, the desorption rate coefficient (κ) is nearly independent of coverage for a fixed surface temperature

$$\kappa = \nu(\theta) \exp[-E(\theta)/kT] \tag{8}$$

Bauer and co-workers (20) observed this trend for oxygen desorption from a W surface although the experimental accuracy was inferior to that reported here. They developed the following expression for describing the desorption rate when the compensation effect occurs:

$$R_d = \nu_o N_s \exp\left[\frac{-E_d(\theta)}{k}\left(\frac{1}{T} - \frac{1}{T_s}\right)\right] \tag{9}$$

where $\nu_o = 7 \times 10^5$ sec^{-1} and $T_s = 1230$ K for CO desorption from Ir(110). This equation may be approximated further. The desorption energy varies almost linearly with coverage

$$E_d(\theta) = 35 - 16\theta \text{ kcal/mole} \tag{10}$$

Oxygen desorption from W and CO desorption from Ir are markedly different systems. Yet, the compensation effect occurs in each case. Using activated complex theory (3), one can rationalize that the compensation effect should occur in all desorption systems. The overall rate coefficient from this theory is given by

$$\kappa = (kT/h)\, K_{eq} \tag{11}$$

where k and h are the Boltzmann and Planck constants, respectively. The parameter K_{eq} is the equilibrium coefficient between the adsorbed species and the activated complex. In this theory, a molecule can exist in three phases: the adsorbed phase, the activated phase, and the gas phase. The first two phases are in equilibrium. A species passes from the second to third phase with a rate coefficient (kT/h), which is independent of surface concentration. In demonstrating that the compensation effect occurs, one must show that the overall rate coefficient (κ), or equivalently the equilibrium coefficient (K_{eq}), varies weakly with coverage while other thermodynamic variables are fixed, that is,

$$\left(\frac{\partial K_{eq}}{\partial N_s}\right)_{T,\ P,\ a} = 0 \qquad (12)$$

The subscripts (s) and (*) will denote the adsorbed and activated phases, respectively. Only one type of adsorption site is assumed to exist at any one coverage. The variables H, S, ϕ, a and μ denote enthalpy, entropy, surface pressure, surface area, and chemical potential, respectively. Total molar quantities are represented by a capital letter (H); partial molar quantities, by a lower case letter with a bar ($\bar{h}$). Hill has defined partial molar quantities and has shown how these are necessary in describing adsorbed phases (21).

At equilibrium, the chemical potentials in the adsorbed and activated phases are equal, $\mu_s = \mu_* = \mu$. For each phase, the chemical potential is given by

$$\mu_i = (H_i - TS_i + \phi_i a)/N_i \qquad (13)$$

where i = s or *. The equilibrium coefficient is the ratio of the number

of activated species to adsorbed species, N_*/N_s,

$$K_{eq}(N_s) = (H_* - TS_* + \phi_* a)/(H_s - TS_s + \phi_s a) \qquad (14)$$

The change in the equilibrium coefficient with coverage is given by

$$\frac{\partial K_{eq}}{\partial N_s} = \frac{1}{\mu N_s}\frac{\partial}{\partial N_s}(H_* - TS_* + \phi_* a) - \frac{N_*}{\mu N_s^2}\frac{\partial}{\partial N_s}(H_s - TS_s + \phi_s a) \qquad (15)$$

The variables N_s and N_* are not independent but are related through the equilibrium coefficient. Therefore, the following identity holds

$$\frac{\partial}{\partial N_s} = \left(K_{eq} + N_s \frac{\partial K_{eq}}{\partial N_s}\right)\frac{\partial}{\partial N_*} \qquad (16)$$

Eq. (15) can be reduced using this identity and the definition of partial molar quantities ($\bar{h}_s = \frac{\partial H_s}{\partial N_s}$, etc.) to

$$\frac{\partial \kappa}{\partial N_s} = \frac{N_*}{\mu N_s^2}\left[(\bar{h}_* - T\bar{s}_* + \bar{\phi}_* a) - (\bar{h}_s - T\bar{s}_s + \bar{\phi}_s a)\right] \qquad (17)$$

$$+ \frac{1}{\mu}\left(\frac{\partial \kappa}{\partial N_s}\right)(\bar{h}_* - T\bar{s}_* + \bar{\phi}_* a)$$

At equilibrium, the chemical potential is related to partial molar quantities by

$$\mu = \bar{h}_* - T\bar{s}_* \qquad (18)$$
$$= \bar{h}_s - T\bar{s}_s$$

Eq. (16) is reduced to

$$\left(\frac{\partial K_{eq}}{\partial N_s}\right)_{T,\,P,\,a} = \frac{K_{eq}}{N_s}\left(\frac{\bar{\phi}_s - \bar{\phi}_*}{\bar{\phi}_*}\right) \approx 0 \qquad (19)$$

Throughout these arguments, the area for the adsorbed phase and the area of the activated phase tacitly were assumed to be equal ($a = a_s = a_*$). This follows from the fact that both phases are on the surface. For this same reason, the surface pressure for the adsorbed phase ($\bar{\phi}_s$) and the activated phase ($\bar{\phi}_*$) are equal. Therefore, the equilibrium coefficient (K_{eq}) is independent of the number of adsorbed species (N_s) if the two surface pressures are equal. Consequently, the overall desorption rate coefficient (κ) given by activated complex theory is independent of surface coverage for a fixed temperature and surface area.

A less rigorous appeal to activated complex theory may explain the substantial decrease in the pre-exponential factor with coverage, which is three orders of magnitude in the case of CO desorption from Ir(110). In this theory, the natural logarithm of the pre-exponential factor is preportional to the entropy difference between the activated and adsorbed phases

$$\ell n\ \nu = (S_* - S_s)/k \qquad (20)$$

Either the entropy of adsorption increases with coverage, or the entropy in the activated phase decreases with coverage. Both conjectures are reasonable. If the adsorbate becomes more mobile as coverage increases, the first conjecture is substantiated. The theory of Doyen and Ertl for CO adsorbed on transition metals (22) reveals that the adsorbate binds less strongly on an fcc (110) surface as coverage increases. Hence, the effective barrier to surface diffusion is reduced. The entropy of the activated complex may decrease with increasing coverage. Eq. (19) demonstrates that the number of activated species increases with coverage.

The entropy of adsorbed systems decreases with increasing population (5).

Fig. 8 shows that the isostere method is inapplicable to this system. Rate parameters from Fig. 7 were used in constructing a plot of R_d versus θ for a surface temperature of 600 K. The slope, which should be constant for this first order system, increases rapidly with coverage. This artifact arises because the rate parameters vary markedly with coverage. Since the desorption rate is not proportional to the coverage in Fig. 8, one cannot conclude that the compensation effect does not occur for this system. Certainly, Eq. (19) is not strictly correct, and the rate parameters do vary in sympathy. Activated complex theory is simply an approximate treatment of desorption phenomena and allows one to rationalize why the rate parameters vary in sympathy.

A plot of $\ell n\, R_d$ versus θ for a surface temperature of 600 K is also shown in Fig. 8. Eqs. (9) and (10) demonstrate why this relation, rather than R_d versus θ, is linear. For a plot of $\ell n R_d$ versus θ, a line of slope (E_δ / kT) (where $E_d = E_o - E_\delta \theta$) is predicted if these equations hold. Both the predicted and observed slopes equal 13.

This method of analysis can be applied to systems in which reactive desorption, such as the combination of surface oxygen and CO to form CO_2 occurs. The form of the rate expression changes, and the rate parameters are evaluated for constant coverages of CO and oxygen. If two rate processes are important in the same regime of temperature, this method of analysis is also apposite. For example, both the reaction to form CO_2 and the desorption of CO may take place simultaneously. Then both the initial coverages and the heating rate may be varied to alter the desorption rate and surface temperature for a fixed adsorbate composition.

5. The Applicability of the Equilibrium Assumption

The application of the equilibrium assumption ($\frac{dN_g}{dt} = 0$) to the data for CO desorption from Ir(110) should be examined since the average heating rates were large, as high as 212 K/sec. In assessing the applicability of the equilibrium assumption, two questions will be addressed in this section. How can the pumping time (τ) be estimated, and how are thermal desorption data distorted if the equilibrium assumption does not apply?

The pumping time can be determined from the parameter η in Eq. (4), from experiment or from a phenomenological model of the pumping process. If the equilibrium assumption applies, the pumping time is derived readily from the parameter η. The pumping time may be related to other measured parameters through a straightforward integration of Eq. (3). The ideal gas law ($N_g = PV/kT_g$) applies under vacuum conditions. The volume of the experimental chamber is V≈40L, and the temperature of the gas phase (T_g) is room temperature (as will be shown). The mass spectrometer signal (I) is proportional to the partial pressure of the desorbing species ($P = gI$). Through calibrating the mass spectrometer signal with an ion gauge, the gain (g) was determined to be 4.9×10^{-4} torr/amp. Upon substituting these relations, Eq. (3) becomes

$$dN_s/dt = gIV/kT_g\tau \qquad (21)$$

which, making use of Eq. (5), simplifies to

$$\tau = gV/\eta kT_g \qquad (22)$$

The value of η, 3.9×10^{18} molecules/amp-sec, was deduced from measurements of surface coverages that were derived from desorption kinetics data (16).

Since all parameters on the right-hand-side of Eq. (22) are known, τ is fixed at 160 msec. The pumping time is related to the pumping speed (S) of the vacuum system by $S = V/\tau$. The calculated pumping speed, 250 ℓ/sec, is remarkably close to the value specified for the unit, 220 ℓ/sec. This agreement is fortuitous. Pressure gradients must occur in the belljar so that this analysis is invalid.

The pumping time, the half-life of a gaseous molecule of CO in the belljar, can be measured directly. The gas supply is shut off suddenly and the exponential decay of pressure with time is monitored with a mass spectrometer. Values of 30-70 msec were obtained for the pumping time. Reproducibility was poor since the gas supply was shut off manually. This artifact produces an unrealistically high value for the pumping time, but an upper bound of 30 msec is associated with this quantity.

A lower bound on the pumping time is calculated from a phenomenological model of the pumping process. The belljar is partitioned by a poppet value. The sample and the mass spectrometer are above the valve; the ion pump is located below. In this model, the pressure on the sample side of the valve is assumed to be uniform while a vacuum is assumed to be maintained on the pump side. The areas of the walls on the sample side and of the orifice into the pump side are 5800 and 300 cm^2, respectively. Since the gas in the belljar is rarified, the probability that a gas molecule will be pumped is $300/(5800 + 300) = 0.05$. From a statistical treatment (4), the root-mean-square velocity of a CO molecule at room temperature is 5×10^4 cm/sec. A characteristic distance separating the walls of the belljar is 25 cm. Hence, the collision

frequency of a CO molecule with the wall is 2×10^3 sec^{-1}. A geometric series relates the collision frequency and probability of pumping to the pumping time, or half-life, of the gaseous molecule. A value of 10 msec is obtained for the lower bound on the pumping time. A typical CO molecule undergoes twenty collisions with the wall before it is pumped. Therefore,the gas is at the temperature of the walls, room temperature, not the temperature of the sample. In summary, the pumping time is 20±10 msec, depending upon how the value is estimated.

Since the pumping time may be estimated, a method will be developed for assessing the validity of the equilibrium assumption and estimating how the desorption spectra are distorted when this assumption is violated. Figs. 3 and 4 illustrate that desorption spectra may be approximated as square pulses. That is, the desorption rate may be written as

$$\begin{aligned} dN_s/dt &= c, \quad T_1 \leq T \leq T_2 \\ &= 0, \quad \text{otherwise} \end{aligned} \tag{23}$$

where c is a constant. To relate the number of molecules in the gas phase to the desorption rate, the mass balance for the gas phase [Eq. (3)] is integrated with the initial condition $N_g(0) = 0$. If the heating schedule is linear ($T = \beta t$), the result is

$$\begin{aligned} N_g(T) &= 0 \qquad\qquad\qquad\qquad\qquad\qquad , \; T < T_1 \\ &= \tau c \left\{1 - \exp\left[(T_1 - T)/\beta\tau\right]\right\}, \; T_1 < T < T_2 \\ &= \tau c \exp(-T/\tau\beta)\left[\exp(T_2/\tau\beta) - \exp(T_1/\tau\beta)\right], \; T > T_2 \end{aligned} \tag{24}$$

When the width of the desorption spectrum (T_2-T_1) is greater than the factor $\beta\tau$, the shape of the spectrum is not distorted substantially. However, the observed spectrum, monitored by the mass spectrometer, is

shifted to higher temperatures compared to the true spectrum ($\tau = 0$). The shift (ΔT) of the center of the spectrum (where $N_s = N_s^o/2$) is calculated from a straightforward integration of Eq. (24)

$$\Delta T = T_c - (T_2 + T_1)/2 = \beta\tau\left\{1 - \exp\left[(T_1 - T_c)/\beta\tau\right]\right\} \quad (25)$$

where T_c and $\frac{T_2 + T_1}{2}$ are the centers of the observed and true spectra, respectively. The second term is negligible if $\beta\tau < (T_2 - T_1)$. Under this constraint, the observed spectrum is shifted upward in temperature from the true spectrum by the temperature $\beta\tau$.

This $\beta\tau$ approximation is applicable to spectra that are not square pulses in temperature. A model system in which $\beta = 1000$ K/sec, $\theta_o = 0.82$, and $\tau = 50$ msec will be considered. The surface mass balance approximates that for CO desorption from Ir(110)

$$\frac{d\theta}{dT} = \frac{\nu_o \theta}{\beta} \exp\left[-(E_o - E_\delta \theta)/kT\right] \quad (26)$$

where $\nu_o = 4 \times 10^{10}$ sec^{-1}, $E_o = 35$ kcal/mole, and $E_\delta = 16$ kcal/mole. In this system, $\beta\tau < (T_2 - T_1)$ since $\beta\tau$ is 50 K and $(T_2 - T_1)$ is about 200 K. The observed spectrum ($\tau = 50$ msec) and the true spectrum ($\tau = 0$ msec) are shown in Fig. 9. The observed spectrum is distorted only at high and low temperatures, but the observed variation of coverage with temperature is shifted upward from the true variation by 52 K at $\theta = \theta_o/2$. The theory predicts a shift of 50 K. This model demonstrates that this theory is useful for quantitatively assessing spectral distortions that arise when large heating rates are used.

Little distortion occurs in the spectra reported here. The pumping time was 20 ± 10 msec and the maximum heating rate was 212 K/sec so that

the maximum temperature shifts in these spectra are 4 ± 2 K. As the theory predicts, distortion in the spectral shape was acute at high and low coverages for large heating rates. For this reason, some data points for larger heating rates are excluded from the Arrhenius plots in Fig. 6. No attempt was made to reanalyze the data by accounting for this distortion. The observed spectra could be shifted downward in temperature by a factor of $\beta\tau$. However, since the pumping time is not known accurately, and the necessary shifts are on the order of experimental uncertainties, reanalysis is unwarranted.

This theory for estimating distortion may be extended to systems with multiple, successive time delays. If an X-Y recorder, rather than a computer, collects the desorption spectra, an additional time delay is introduced since the slewing speed of the recorder is finite (23). (The time delay in the acquisition path used in these computer-controlled experiments was 40 μsec.)

Now a theory for evaluating distortions in systems with two successive time delays will be developed. The first time delay results from a finite pumping speed; the second, from a finite response time in the data acquisition path. The second time delay is characterized by

$$\frac{dN_o}{dt} = \frac{N_g - N_o}{\tau_o} \tag{27}$$

The time delay in the acquisition path is τ_o, and N_o is the apparent intensity of the mass spectrometer. In solving for $N_o(t)$, Eq. (27) is integrated with the initial condition $N_o(0) = 0$. If the heating schedule is assumed to be linear, the apparent intensity may be calculated as a function of temperature with Eq. (24) giving $N_g(T)$, that

is,

$$N_o(T) = 0 \qquad , T < T_1$$

$$= c\tau \left\{1 - \left(\frac{\tau}{\tau - \tau_o}\right)\left[\exp\left(\frac{T_1 - T}{\beta\tau}\right) + \exp\left(\frac{T_1 - T}{\beta\tau_o}\right)\right]\right\}, \; T_1 < T < T_2$$

$$= \left\{\frac{c\tau^2}{\tau - \tau_o}\right\}\left\{\exp\left(\frac{-T}{\beta\tau}\right)\left[\exp\left(\frac{T_2}{\beta\tau}\right) - \exp\left(\frac{T_1}{\beta\tau}\right)\right]\right.$$

$$\left. + \exp\left(\frac{T_1 - T}{\beta\tau_o}\right)\left[1 - \exp\left(\frac{T_2 - T_1}{\beta\tau}\right)\right]\right\}, \; T_2 < T \qquad (28)$$

The shift between the centers ($N_s = N_s^o/2$) of the observed and true spectra is determined as in Eq. (25).

$$\Delta T = \beta\left[(\tau + \tau_o) - \left(\frac{\tau^2}{\tau - \tau_o}\right)\exp\left(\frac{T_1 - T_c}{\beta\tau}\right) + \left(\frac{\tau^2}{\tau - \tau_o}\right)\exp\left(\frac{T_1 - T_c}{\beta\tau}\right)\right] \qquad (29)$$

As before, this equation is applicable only when $(T_2 - T_1) < \beta(\tau + \tau_o)$. Under this condition, the second and third terms are negligible. Therefore, the total effective time delay is the sum of the individual time delays in the system. Mathematical induction may be used to prove that this result also applies for three or more time delays. Consequently, this theory may be applied to experimental conditions in which spectral distortions arise from several time lags.

6. Conclusions

A method for analyzing thermal desorption spectra has been developed for determining the coverage dependence of the pre-exponential factor and the desorption energy in the Arrhenius expression for a one-step desorption process. Both rate parameters show a strong coverage dependence in the case of CO desorption from Ir(110). For this system, three features appear in the desorption spectrum, and the compensation effect occurs. Activated complex theory may be used to argue that this effect should be common, rather than exceptional, for desorption systems.

This method of analysis has three limitations. First, it is applicable only to systems in which desorption occurs through a one-step mechanism. Second, the order of the desorption process must be known. Finally, the technique is not appropriate for systems involving desorption from several distinct sites, unless the difference in desorption energies between sites is greater than about 10% of the larger energy. However, this method may be used to extract rate parameters from spectra with several features and may be extended to treat multi-adsorbate systems in which multiple desorption processes occur. Moreover, no special form of the heating schedule need be assumed if the mass spectrometer signal and surface temperature are measured as independent functions of time.

Since this method involves variable heating schedules, a rule of thumb was developed for assessing the spectral distortion resulting from successive time delays. When time delays, such as a finite pumping time, are present, the centers of mass in the observed desorption spectra are shifted upward in temperature from the true spectra. The shift is equal to

the sum of the time delays multiplied by the average heating rate.

Acknowledgment

The authors are grateful to the National Science Foundation for financial support and to Professor R. J. Madix for helpful discussions.

References

1. For a review see: L. D. Schmidt, Catal. Rev. - Sci. Eng. 9, 115 (1974).
2. G. Ehrlich, Advan. Catal. 14, 255 (1963).
3. G. C. Bond, Catalysis by Metals, Academic Press, New York, 1962.
4. R. H. Fowler and E. A. Guggenheim, Statistical Thermodynamics, Macmillan, New York, 1939.
5. C. Kemball, Advan. Catal. 2, 233 (1950).
6. For example, and for other references, see: G. Ertl, M. Neumann and K. M. Streit, Surface Sci. 64, 393 (1977).
7. P. A. Redhead, Vacuum 12, 203 (1962).
8. G. Ehrlich, J. Appl. Phys. 32, 4 (1961).
9. For a review of such methods see: C.-M. Chan, R. Aris, and W. H. Weinberg, Appl. Surface Sci. (in press, 1978).
10. For a review of such methods see: J. L. Falconer and R. J. Madix, J. Catal. 48, 262 (1977).
11. This aspect of TDS was realized first by Ehrlich in ref. 8.
12. P. J. Estrup and E. G. McRae, Surface Sci. 25, 1 (1971).
13. E. Bauer, H. Poppa, G. Todd and F. Bonczek, J. Appl. Phys. 45, 5164 (1974).
14. C. M. Comrie and W. H. Weinberg, J. Chem. Phys. 64, 250 (1976).
15. Details of the interfacing will be described in: J. L. Taylor and W. H. Weinberg, to be published.
16. J. L. Taylor, D. E. Ibbotson, and W. H. Weinberg, to be published.
17. J. L. Taylor, D. E. Ibbotson, and W. H. Weinberg, to be published.
18. J. C. Tracy and P. W. Palmberg, J. Chem. Phys. 51, 4852 (1969).
19. Fuggle et al. used a similar argument to show CO adsorbs molecularly on Ru. Details are found in: J. C. Fuggle, E. Umbach and D. Menzel, Solid State Commun. 20, 89 (1976).

20. E. Bauer, F. Bonczek, H. Poppa and G. Todd, Surface Sci. 53, 87 (1975).

21. T. L. Hill, J. Chem. Phys. 18, 246 (1950).

22. G. Doyen and G. Ertl, Surface Sci. 43, 197 (1974).

23. The slewing speed of most X-Y recorders is about 50 cm/sec. To relate the slewing speed to a time delay, one calculates the proportionality constant between the ordinate and the number of molecules in the gas phase. The time delay is the proportionality constant divided by the slewing speed.

Figure Captions

Figure 1: Thermal desorption spectra of CO from Ir(110) obtained with a constant heating rate, 36 K/sec, and for a variable initial coverage.

Figure 2: Order plots for points of constant surface temperature constructed from the data in Figure 1.

Figure 3: Solutions of the surface mass balance for variable heating schedules and initial coverages. The heating rate is linear, and the equilibrium assumption is made.

Figure 4: Thermal desorption spectra of CO from Ir(110) obtained with a variable heating rate and for a fixed initial surface coverage, $\theta_o = 0.82 \pm 0.01$. The mass balance for the gas phase was integrated under the equilibrium assumption to determine the coverage-temperature relations.

Figure 5: Arrhenius plots for CO desorption from Ir(110) for various coverages. Data points were obtained from desorption spectra like those shown in Figure 4.

Figure 6: Auger KLL transitions of oxygen for chemisorbed CO and oxygen on Ir(110) at 300 K.

Figure 7: Variations in the desorption energy and the pre-exponential factor with surface coverage for CO on Ir(110). The desorption order is unity (see text).

Figure 8: Variations in the desorption rate and the natural logarithm of the desorption rate with surface coverage for CO on Ir(110) at 600 K. The desorption rates were calculated from the rate parameters in Figure 7.

Figure 9: True and observed desorption spectra for a model system. The equilibrium assumption (τ = 0) was employed in calculating the true spectrum. The pumping time was fixed at 50 msec when the observed spectrum was calculated.

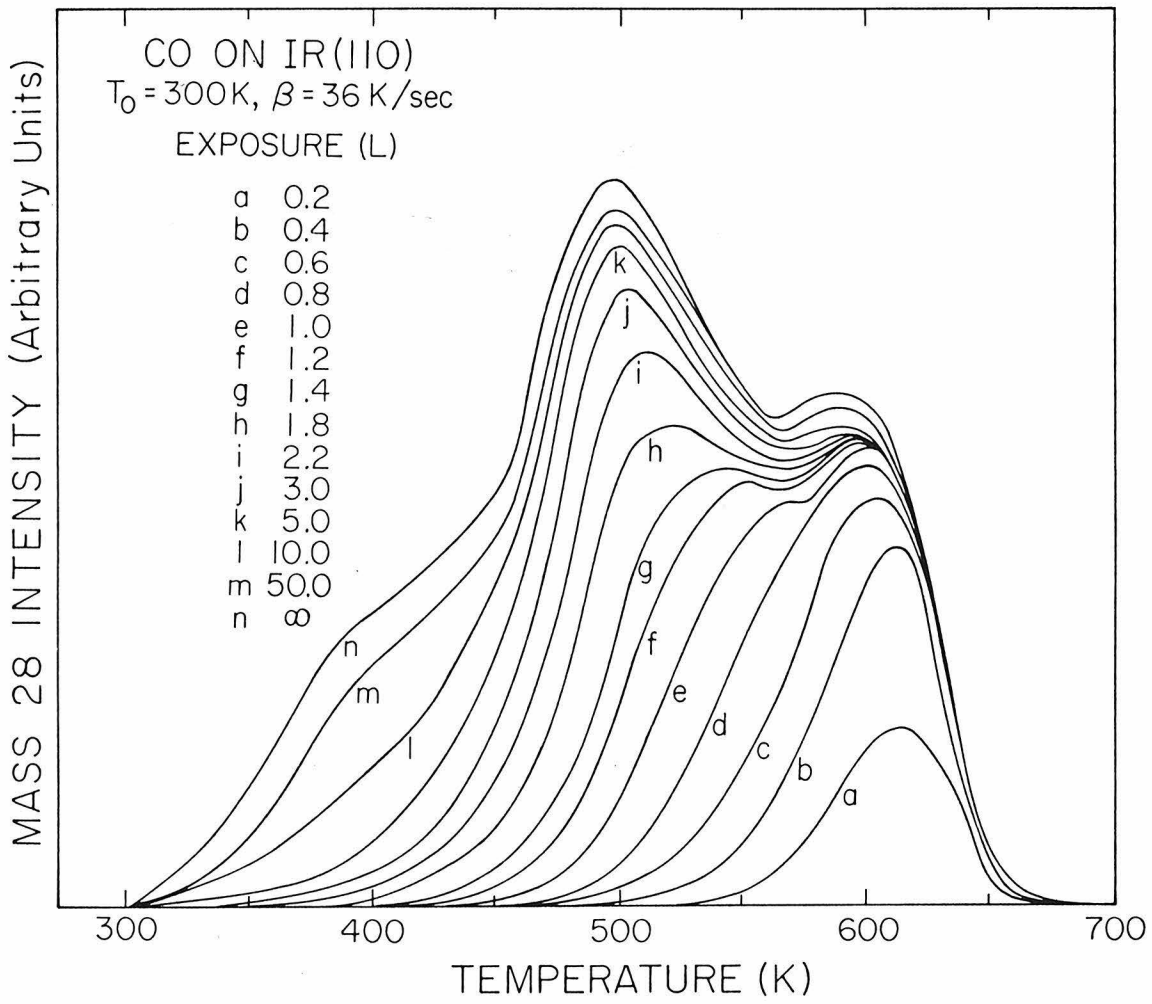

CO ON IR(110)
T_0 = 300K, β = 36 K/sec
EXPOSURE (L)
a 0.2
b 0.4
c 0.6
d 0.8
e 1.0
f 1.2
g 1.4
h 1.8
i 2.2
j 3.0
k 5.0
l 10.0
m 50.0
n ∞
MASS 28 INTENSITY (Arbitrary Units)
TEMPERATURE (K)
300
400
500
600
700

Fig.1

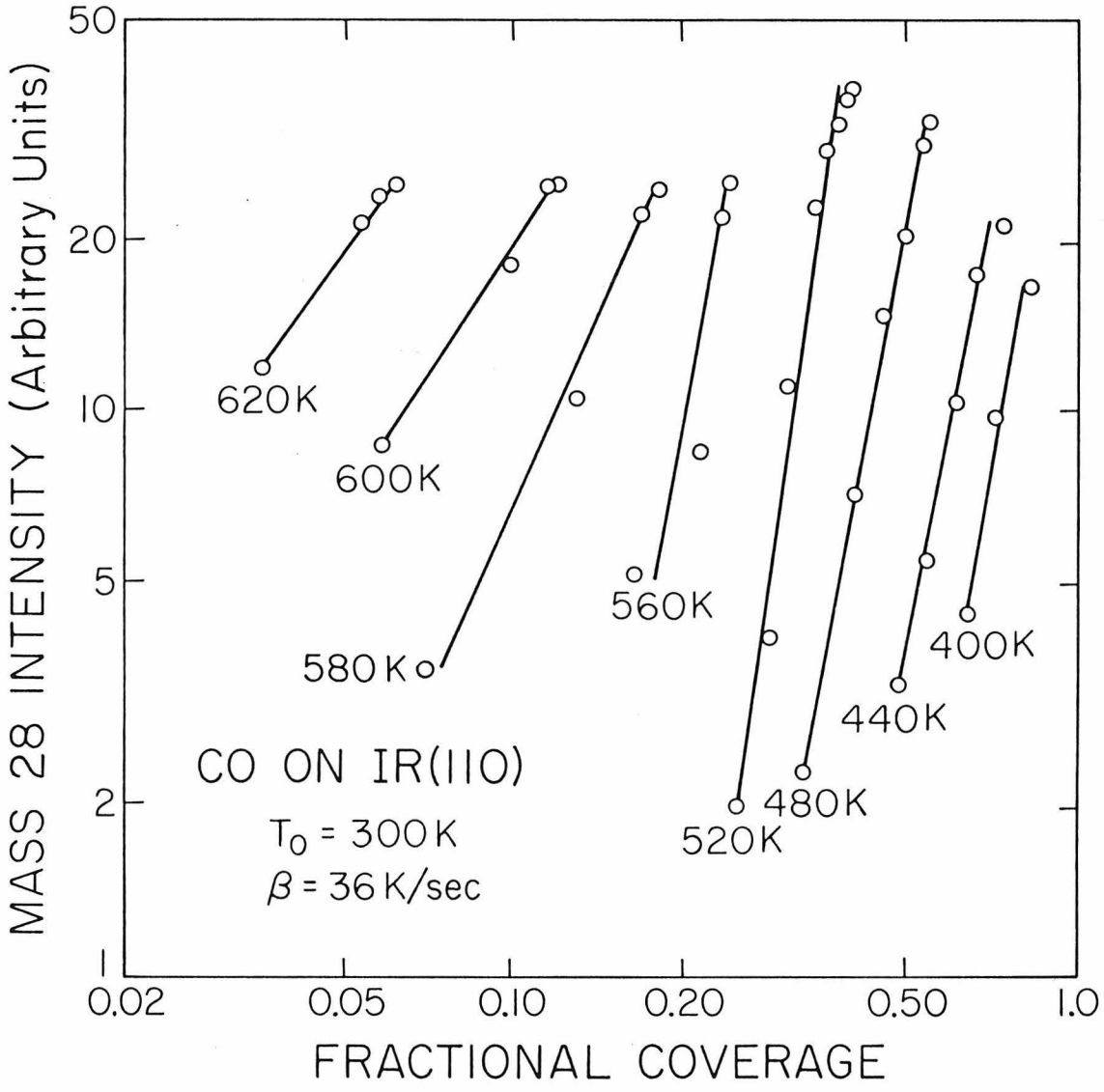

MASS 28 INTENSITY (Arbitrary Units)
50
20
10
5
2
1
0.02
0.05
0.10
0.20
0.50
1.0
FRACTIONAL COVERAGE
620K
600K
580K
560K
520K
480K
440K
400K
CO ON IR(110)
T_0 = 300K
β = 36 K/sec

Fig.2

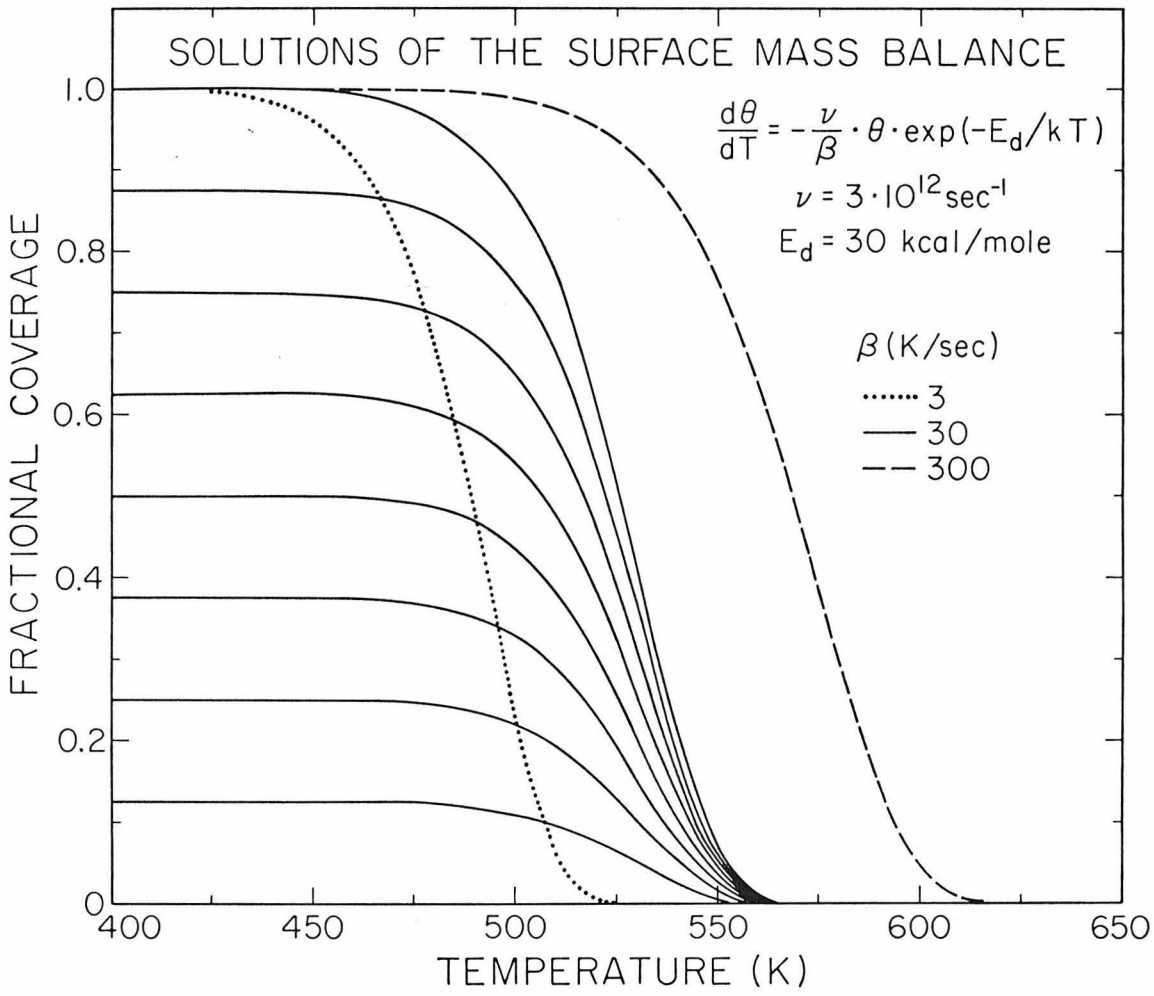

SOLUTIONS OF THE SURFACE MASS BALANCE
$\frac{d\theta}{dT} = -\frac{\nu}{\beta} \cdot \theta \cdot \exp(-E_d/kT)$
$\nu = 3 \cdot 10^{12} sec^{-1}$
$E_d = 30$ kcal/mole
β (K/sec)
3
30
300
FRACTIONAL COVERAGE
1.0
0.8
0.6
0.4
0.2
0
400
450
500
550
600
650
TEMPERATURE (K)

Fig.3

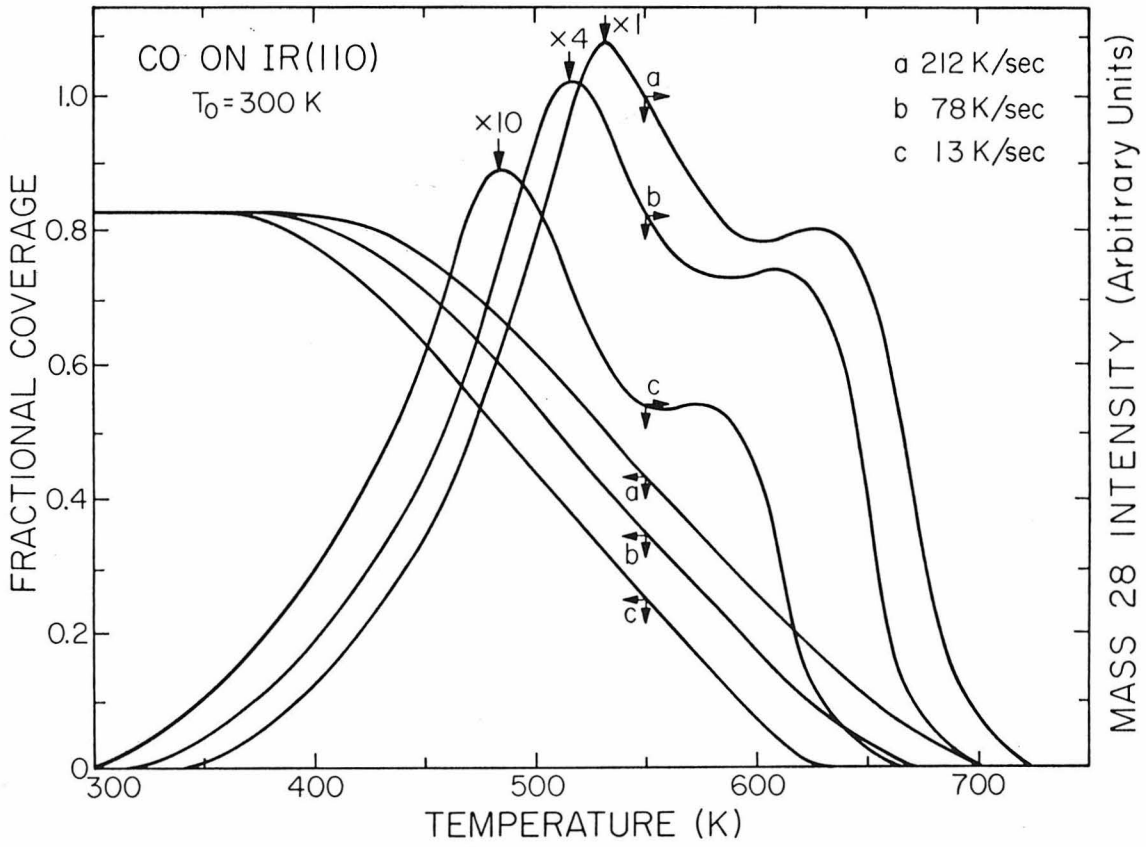

CO ON IR(110)
T_0 = 300 K
×10
×4
×1
a 212 K/sec
b 78 K/sec
c 13 K/sec
a
b
c
a
b
c
FRACTIONAL COVERAGE
MASS 28 INTENSITY (Arbitrary Units)
TEMPERATURE (K)
0
0.2
0.4
0.6
0.8
1.0
300
400
500
600
700

Fig.4

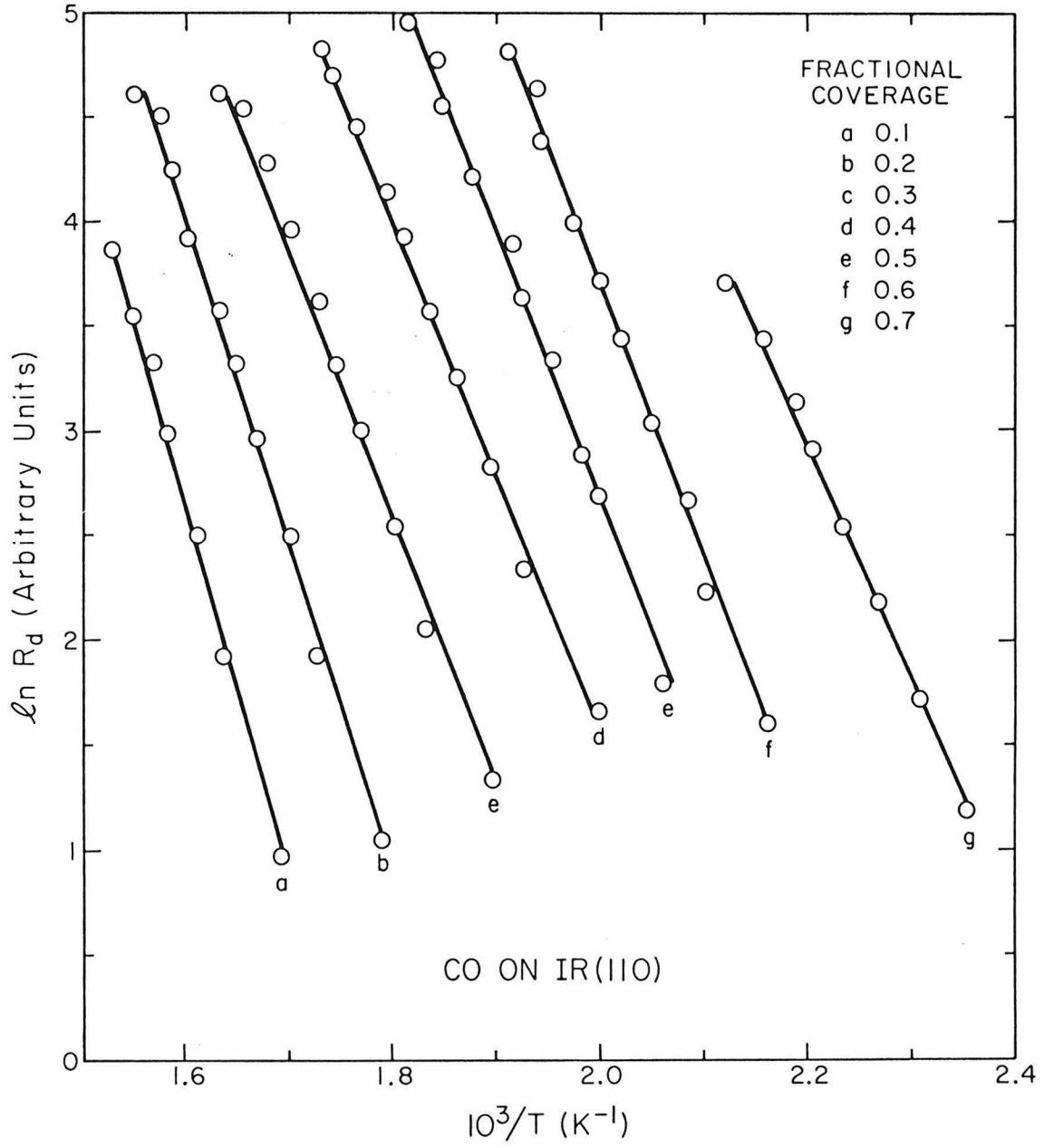

FRACTIONAL COVERAGE
a 0.1
b 0.2
c 0.3
d 0.4
e 0.5
f 0.6
g 0.7
CO ON IR(110)
ℓn R_d (Arbitrary Units)
10^3/T (K^-1)
0
1
2
3
4
5
1.6
1.8
2.0
2.2
2.4
a
b
e
d
e
f
g

Fig.5

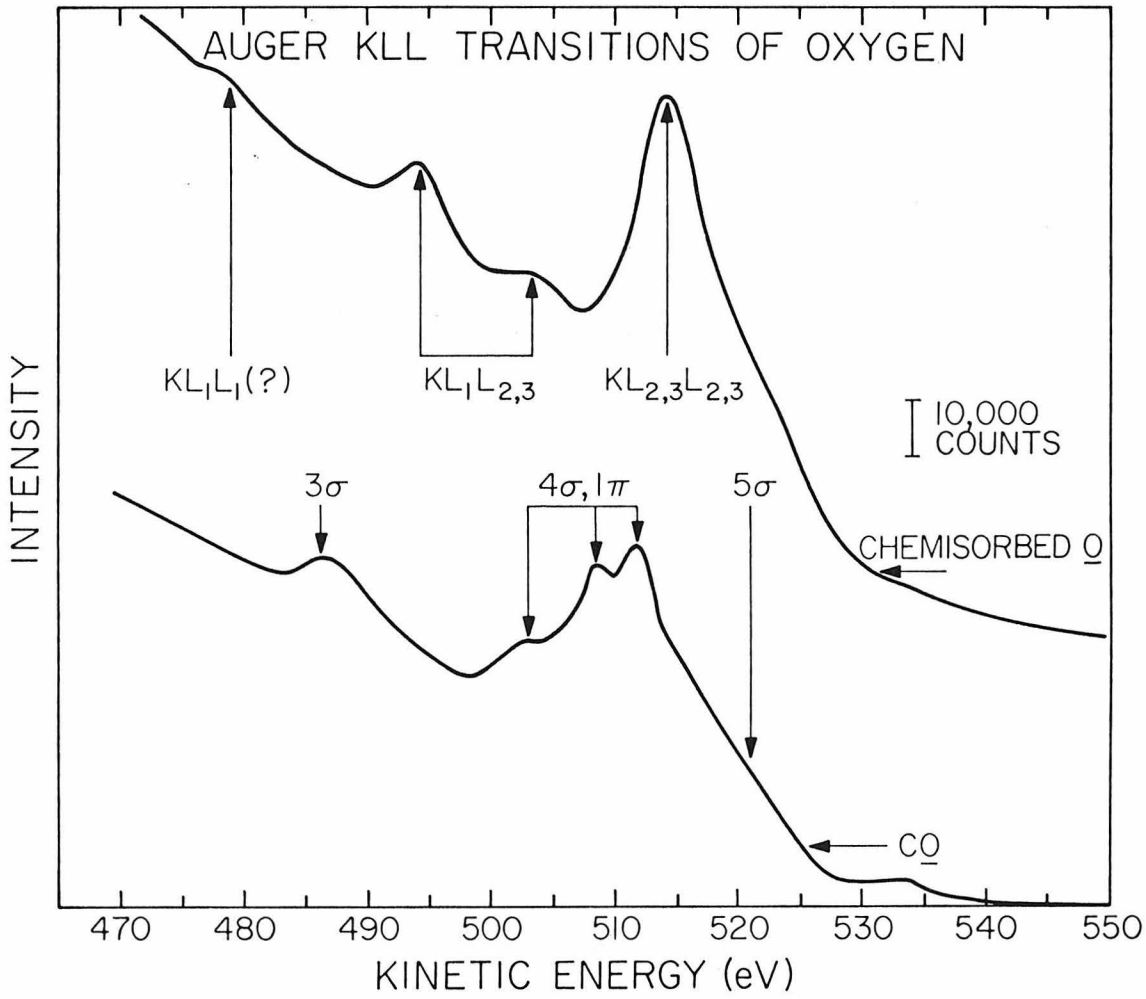

AUGER KLL TRANSITIONS OF OXYGEN
INTENSITY
KL1L1(?)
KL1L2,3
KL2,3L2,3
10,000 COUNTS
3σ
4σ,1π
5σ
CHEMISORBED O
CO
470
480
490
500
510
520
530
540
550
KINETIC ENERGY (eV)

Fig.6

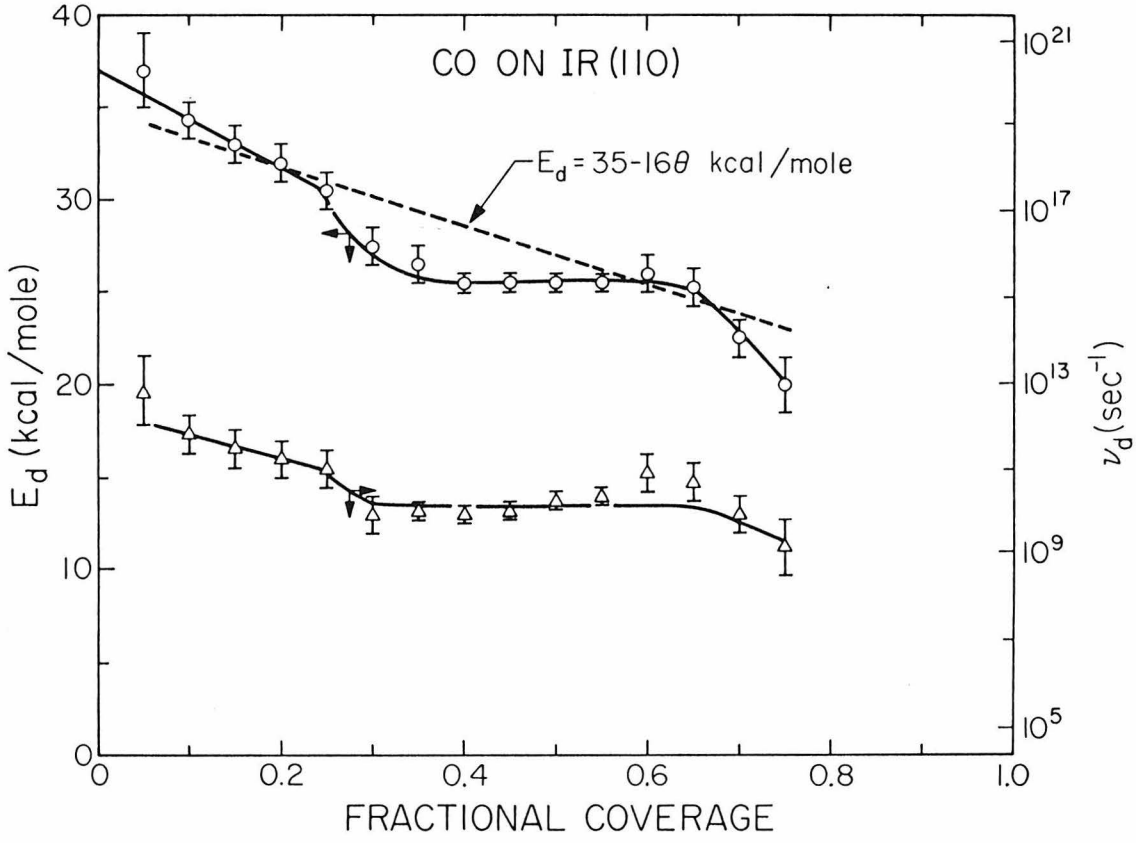

CO ON IR(110)
$E_d = 35-16\theta$ kcal/mole
E_d (kcal/mole)
ν_d (sec^{-1})
FRACTIONAL COVERAGE
0
10
20
30
40
0.2
0.4
0.6
0.8
1.0
10^{5}
10^{9}
10^{13}
10^{17}
10^{21}

Fig. 7

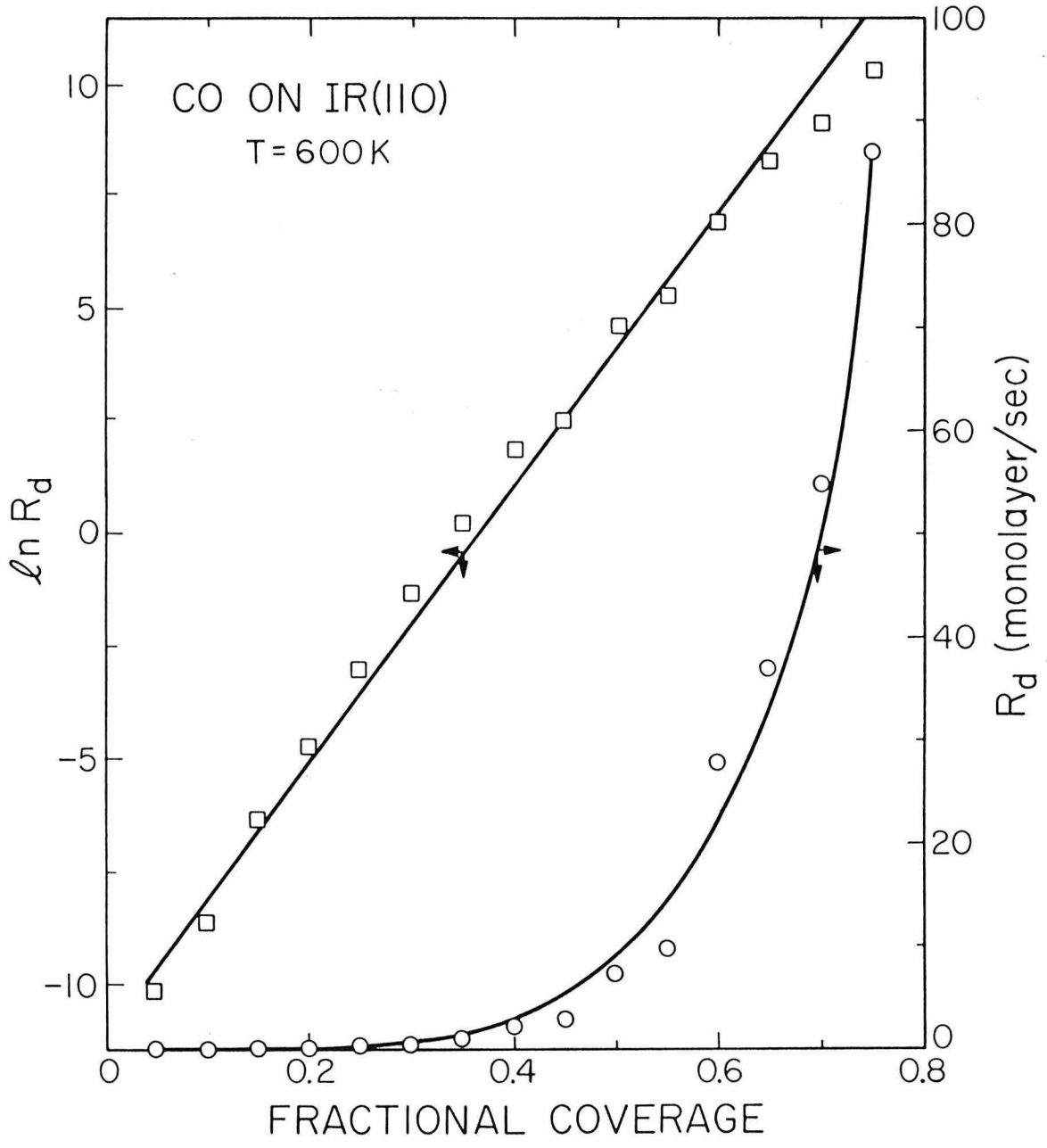

CO ON IR(110)
T = 600K
ℓn R_d
R_d (monolayer/sec)
FRACTIONAL COVERAGE
10
5
0
-5
-10
100
80
60
40
20
0
0
0.2
0.4
0.6
0.8

Fig.8

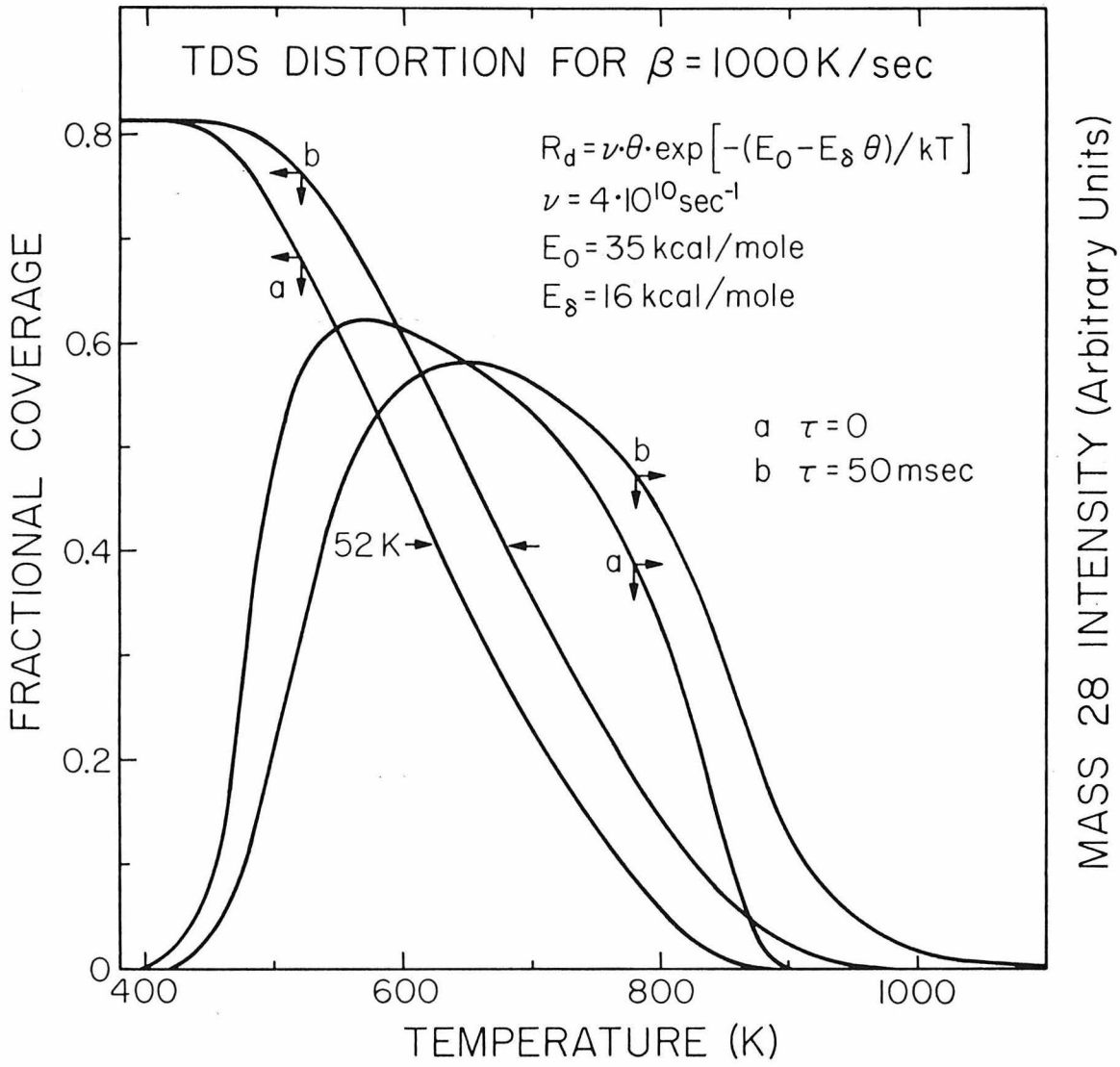

TDS DISTORTION FOR β = 1000 K/sec
$R_d = \nu \cdot \theta \cdot \exp\left[-(E_0 - E_\delta\,\theta)/kT\right]$
$\nu = 4 \cdot 10^{10}\,sec^{-1}$
$E_0 = 35$ kcal/mole
$E_\delta = 16$ kcal/mole
a τ = 0
b τ = 50 msec
52 K
a
b
a
b
FRACTIONAL COVERAGE
MASS 28 INTENSITY (Arbitrary Units)
TEMPERATURE (K)
0
0.2
0.4
0.6
0.8
400
600
800
1000

Fig. 9

CHAPTER 3

THE CHEMISORPTION OF CO ON CLEAN AND OXIDIZED IR(110)

(The text of Chapter 3 consists of an article coauthored with D. E. Ibbotson and W. H. Weinberg that has been submitted to Journal of Chemical Physics for publication.)

Abstract

The chemisorption of CO on clean and oxidized Ir(110) has been investigated under ultra-high vacuum conditions with thermal desorption mass spectrometry, contact potential difference measurements, Auger electron spectroscopy, and LEED. On both surfaces, adsorption is non-dissociative under all conditions examined, i.e., temperatures between 90 and 1300 K and pressures below 10^{-6} torr. Two LEED patterns, a p(2x2) and a (4x2) disordered in the [001] direction, form on the clean surface. A (2x1)p1g1 structure forms on the oxidized surface. Thermal desorption spectra from the clean and oxidized surfaces exhibit three and two features. The desorption energy and pre-exponential factor in the Arrhenius equation describing the desorption kinetics have been measured as functions of surface coverage. The compensation effect occurs on both surfaces. In the limit of zero coverage, the desorption energies are 37 and 31 kcal/mole for clean and oxidized Ir(110). The adsorption kinetics are identical for both surfaces. At 90 K, the sticking probability is unity until the surface coverage nears saturation. At 300 K, the adsorption kinetics display second-order behavior with an initial sticking probability of unity. The saturation coverage is about 10^{15} molecules/cm^2 for both surfaces at 90 and 300 K. Upon adsorption, the work function increases for both surfaces, but is larger for the clean surface and at lower temperatures. A model developed to fit the change in work function with coverage and temperature indicates that binding sites between the topmost rows of surface atoms are preferred energetically.

I. Introduction

This investigation of CO chemisorption was undertaken to give insight into the catalytic behavior of the CO oxidation reaction on Ir(110). Surface sensitive techniques—thermal desorption mass spectrometry (TDS), contact potential difference (CPD) measurements, Auger electron spectroscopy (AES), and LEED—were used to examine various aspects of CO chemisorption on Ir(110) under ultra-high vacuum conditions. These aspects include the spectroscopic characterization of the overlayer, overlayer structures, desorption kinetics, adsorption kinetics, and charge transfer in the overlayer. Each aspect is developed as a separate topic in this article. Inferences drawn from the various results are presented in the discussion that parallels each topic.

Although CO chemisorption has been studied extensively on several surfaces of the platinum metals, including the (110) surfaces of Ni (1, 2), Rh (3), Pd (4, 5), Pt (6, 7), and Ir (8), the different behavior among the various metals justifies this new study. Moreover, many aspects of CO chemisorption have not been treated thoroughly for Ir(110). Examples of different behavior are numerous. CO adsorbs either molecularly or dissociatively depending on the surface and the surface temperature. Although a CO-(2x1)p1g1 LEED pattern appears on all (110) faces of the platinum metals, c(2x2) structures have been observed on Ni, Rh, and Pd; and (4x2) structures on Ni and Pd. First-order adsorption kinetics with an initial sticking probability near unity have been reported in most studies of CO chemisorption, but second-order kinetics have been observed on Ir(111) (9). CPD measurements are understandably

different for CO adsorption on the various metals. However, values varying by several hundred meV have been reported for saturation coverages on surfaces of the same metal and orientation (10).

One explanation for the different observations is that an impurity, a surface oxide, alters the CPD for CO. Often oxygen is used to clean carbon from the surfaces of platinum metals. As this work demonstrates, residual oxides are difficult to detect with AES. Surface oxides, distinct from chemisorbed oxygen, have been observed on Ni (11), Pd (12), Pt (7), and Ir (13) surfaces. A stable, well characterized surface oxide forms on Ir(110) (14). Understanding the chemisorption of CO on this oxidized surface is crucial in this study because the CO oxidation reaction is rapid at temperatures where the oxide forms (15). Therefore, all aspects of CO chemisorption mentioned above are considered for both clean and oxidized Ir(110).

One additional goal of this work is modeling certain facets of CO chemisorption. Specifically, models are developed for describing the influence of inter-adsorbate repulsion upon adsorption kinetics and for explaining the coverage and temperature dependences of CPD measurements.

II. Experimental

The experiments were performed in a stainless steel belljar equipped with ion and Ti sublimation pumps. The base pressure of the vacuum system was 2×10^{-10} torr. The belljar contained several probes for surface analysis, including a quadrupole mass spectrometer (UTI-100C), LEED

optics (Varian), AES (PHI) and XPS (PHI). A double-pass cylindrical mirror analyzer (PHI) performed the energy analysis for the various electron spectroscopies. CPD measurements were recorded by a continuous retarding potential technique (16). The Ir(110) crystal was spot welded to two 10 mil Ta wires, which were mounted on a precision manipulator. The crystal could be cooled to 90 K with liquid nitrogen or could be heated conductively by passing a current through the Ta wires. The surface temperature in the range from 200 - 1800 K was measured with a Pt-Pt10%Rh thermocouple that was spot welded to the crystal. Lower temperatures were measured through the change in the resistance of the Ta wires supporting the crystal.

A computer was interfaced to all probes (17). The pass energy of the analyzer and the filter of the mass spectrometer were controlled by the computer. The intensity of the mass spectrometer, pulse counts from the analyzer, the thermocouple EMF, and CPD measurements were recorded by the computer. A variety of programs analyzed and displayed the data. Although most data were recorded digitally by the computer, smooth curves have been drawn through the data to facilitate the presentation.

The Ir sample was cut from a single crystal (Materials Research Corp.) and was aligned within 1° of the (110) orientation through standard preparation and polishing techniques (18). All data have been reproduced after the sample was repolished, which was done several times during this investigation. Because Ar ion bombardment was inefficient in removing carbon from the crystal *in situ*, the surface was cleaned chemically through a series of oxidation and reduction cycles. The sample

was heated to 900 K in an atmosphere of 10^{-7} torr of O_2 for several minutes, the belljar was evacuated, and the sample was annealed above 1600 K to remove surface oxygen and to drive carbon from the bulk to the surface. Annealing above 1600 K was absolutely necessary for decomposing surface oxides. During the oxidation cycles, the surface temperature and oxygen pressure were optimized to remove carbon rapidly and to inhibit the formation of oxides. The oxygen used for cleaning the crystal, as well as the carbon monoxide for the chemisorption studies, were research grade (99.99% pure).

All experiments of CO chemisorption were performed on both clean and oxidized Ir(110). The clean surface was free from impurities within the detection limits of Auger spectroscopy ($\sim$ 0.01 ML where 1 ML = 10^{15} atoms/cm^2). However, the oxide, which is distinct from chemisorbed oxygen, is relatively insensitive to Auger detection and is characterized as follows (14). At 700 K, the oxide layer forms rapidly until the layer saturates at a coverage of 0.25 ± 0.05 ML. Upon annealing an oxygen covered surface to 1200 K, a p(1x4) LEED pattern slowly appears. Oxygen that does not incorporate into the oxide layer desorbs. Above 1600 K, the oxide desorbs as oxygen molecules; no iridium oxides appear as desorption products. An oxidized surface with reproducible physical properties is formed routinely by adsorbing 0.8 L O_2 (1 L = 10^{-6} torr - sec) on the clean surface at 300 K. This exposure corresponds to a surface coverage of 0.25 ML. To transform the chemisorbed oxygen into an oxide layer, the surface is annealed at 1100 K for one minute.

III. Spectroscopic Characterization of the Overlayer

Contrary to previous studies (8), evidence from Auger spectroscopy indicates that CO adsorbs non-dissociatively on both clean and oxidized Ir(110) at temperatures between 90 and 1300 K. X-ray photoelectron spectroscopy (XPS) is not useful for characterizing the adsorbate because the carbon 1s transition is coincident in energy with the Ir $4d_{5/2}$ line. Moreover, the oxygen transitions of CO, chemisorbed oxygen, and the oxide layer differ neither in shape nor in energy within the resolution of the energy analyzer (0.5 eV).

Fig. 1a shows the oxygen KLL Auger transitions of chemisorbed CO and oxygen at 300 K. For all spectra reported here, the electron gun current was approximately 10 nA at 2000 V. Several spectra were summed to obtain the necessary resolution. To accentuate the Auger features, a linear background was subtracted from each spectrum. The oxygen KLL transition of CO resembles that of gaseous CO in shape but is shifted in energy. Because these spectra are similar in shape, but are quite different from the transition of chemisorbed oxygen, which chemisorbs dissociatively at 300 K, CO adsorption must be at least partially molecular. This comparison was suggested first for CO on Ru by Fuggle et. al. (19), who assigned hole states to the oxygen peaks for CO. Even though the CO and oxygen are adsorbed on different metals, their spectra are nearly identical in shape and energy to those in Fig. 1a.

The carbon KLL transitions of chemisorbed CO and carbon on clean Ir(110) at 300 K are shown in Fig. 1b. Because the carbon KLL transition

lies near the Ir $N_4O_3O_3$ line, each transition has been accentuated by subtracting the Auger spectrum of the clean surface from the spectrum of the chemisorbed system. The transitions are not altered if the surface is oxidized. The spectrum of chemisorbed CO is quite similar to a derivative spectra reported for CO on polycrystalline Ir (20). By comparison to a gas phase spectrum (21), peaks in the CO spectra have been associated with final hole states as noted in Fig. 1b. The peak intensity associated with the 5σ orbital is reduced with respect to the gas phase spectrum because the surface bond is made largely through this orbital.

The CO transition appears to be superimposed on the carbon transition so that adsorption might be partially dissociative. If the surface is annealed to 800 K after the spectrum is recorded, CO and CO_2 desorb. Carbon is the only species remaining so some surface carbon must have been co-adsorbed with molecular CO. However, dissociation is induced by the electron beam, not through a thermally activated process. Auger spectroscopy has been used to prove this statement. If an Ir(110) surface is heated to 1400 K for 10 min. in 10^{-6} torr of CO, less than 0.02 ML of carbon is detected on the surface following evacuation. The coverage is determined by assuming that the area under an Auger transition is proportional to coverage and that the Auger sensitivities of CO and carbon are equal. When the experiment is repeated in vacuum, only slightly less carbon is detected. No carbon is detected when either experiment is repeated at lower temperatures. Therefore, chemisorbed CO does not dissociate detectably through a thermally activated process below 1400 K.

Instead, carbon appears on the surface either by electron stimulated dissociation of CO or by diffusion of carbon from the bulk.

The cross section for electron stimulated dissociation of CO is estimated as follows. When an electron beam of 0.5 μA, 50 V, and approximately 1 mm diameter impinges on an Ir(110) surface saturated with CO at 300 K, 0.05 ML of surface carbon is formed in 5 min. A background pressure of 10^{-7} torr of CO is maintained to replace CO removed by electron stimulated desorption. The electron flux (J) is 8×10^{14} $cm^{-2}sec^{-1}$. The rate of formation of surface carbon is related to the cross section for electron stimulated dissociation (Q) by (22)

$$d\theta_c/dt = JQ\theta \qquad (1)$$

where θ_c is the coverage of carbon and θ is the coverage of CO, which is assumed to be 1 ML. The cross section is $2 \times 10^{-19 \pm 1}$ cm^2, which is the same as that for CO dissociation on Pt(111) under similar conditions (22). This cross section should not vary substantially for higher electron impact energies (22). Although the cross section appears to vary with surface temperature, this subject was not pursued.

IV. Overlayer Structures

Under the conditions examined, below 1000 K, the clean Ir(110) surface reconstructs to give a (1x2) LEED pattern. Some possible surface structures for Pt(110), which also reconstructs to a (1x2) structure, are described elsewhere (6). Formation of an oxide layer diminishes the intensity of the half-

order LEED beams so that the surface appears to relax to the (1x1) structure, and streaks appear between the spots in the [001] azimuth (14). An earlier study of chemisorption on Ir(110) associated the (1x1) pattern, but not the streaks, with surface oxygen (8).

The carbon patterns, which should not be confused with CO patterns, form on both clean and oxidized Ir(110), and also appear on Ni(110) (23). Schematic representations as well as photographs of the p(2x1) and the (4x5) patterns of carbon are shown in Fig. 2. The p(2x1) pattern forms by electron stimulated dissociation of chemisorbed CO. If CO and the oxygen formed from electron beam dissociation are removed from the surface by annealing to 700 K (oxygen reacts with CO to form CO_2, but does not desorb), the pattern sharpens. The p(2x1) pattern transforms into the (4x5) pattern by annealing the surface to 1200 K. As determined by AES, both patterns form above coverages of 0.05 ML. When the crystal is heated to 900 K in an oxygen atmosphere, these carbon patterns vanish rapidly, and CO and CO_2 are formed. An earlier study of Ir(110) associated a c(2x2) pattern with carbon (8). In the present study, this structure is associated with oxygen only (14). The pattern disappears and CO_2 forms when the surface is heated to 600 K in the presence of CO. Moreover, this pattern is never observed if oxygen is absent from the surface.

Different LEED patterns for CO appear on clean and oxidized Ir(110). On the clean surface, two patterns, a p(2x2) and a (4x2) appear near 90 K, but are poorly ordered at room temperature. The (4x2) pattern is disordered somewhat in the [001] azimuth, perhaps as a result

of surface reconstruction in that direction. Both patterns are disordered rapidly by the electron beam and could not be photographed. Drawings of these patterns are shown in Fig. 3. As determined by TDS, the p(2x2) and (4x2) patterns are sharpest at 1/4 and 3/4 of saturation coverage. The overlayer compresses continuously since, as coverage increases, the p(2x2) pattern streaks in the [001] azimuth and broadens in the [110] azimuth. Then the broad streak coalesces to form the enlongated spots of the (4x2) pattern. The (4x2) pattern appears on the (110) surfaces of Ni (1, 2) and Pd (4, 5), but the p(2x2) pattern has not been reported previously. Since the saturation coverage of CO on Ir(110) is one molecule per unit cell (1 ML), as will be shown, surface structures for these patterns may be postulated and are shown in Fig. 3. Although theories of CO chemisorption on transition metals suggest that binding sites between the topmost row of surface atoms are energetically favorable (24), additional experimental evidence is needed to fix the registry of the overlayer with respect to the surface. CPD measurements (q.v.) indicate that CO indeed binds preferentially between the topmost rows of surface atoms.

A (2x1)p1g1 pattern forms on the oxidized surface, and appears from 100 - 500 K and between 5 - 100% of saturation coverage. This pattern has been observed on the (110) face of all platinum metals (1 - 8), including Ir (8). Fig. 4 contains a photograph and a drawing of the (2x1) p1g1 pattern. Since certain spots, represented by dotted circles, are missing, this structure is distinct from the carbon p(2x1) pattern. The missing spots indicate that p1q1 symmetry exists in the overlayer. Lambert (25) proposed the structure in Fig. 4 that is consistent with

this symmetry condition. Because the (2x1)p1g1 pattern is intense at low coverages, CO molecules coalesce to form islands on the oxidized surface. If the oxide layer is partially formed, i.e., the oxide coverage is less than 0.25 ML, the (2x1)p1q1 pattern forms, but streaks in the [001] azimuth. Hence, the oxide layer is crucial in ordering this CO pattern on Ir. Also the overlayer appears to influence the structure of the oxidized surface. When CO adsorbs on an amorphous or an ordered oxidized surface with a (1x1) or a (1x4) structure, respectively, the substrate rapidly displays a sharp (1x2) pattern. Similar observations have been reported for CO chemisorption on Pt(110) (6) and Ir(110) (8), suggesting that those studies were conducted on oxidized surfaces.

V. Desorption Kinetics

Several thermal desorption spectra of CO from clean and oxidized Ir(110) were collected as a function of CO exposure at 300 K. These data are useful for investigating both the desorption and the adsorption kinetics of CO on Ir(110). The latter subject will be treated in the next section. The desorption spectra for both surfaces show no evidence for dissociative adsorption since neither a CO nor a CO_2 feature appears near 900 K where the reaction between chemisorbed carbon and oxygen is rapid (as was shown previously).

Fig. 5a shows representative curves of 40 desorption spectra for CO from clean Ir(110). The average heating rate (β) for each spectrum is 36 K/sec. Three features appear with relative intensities of 1:2:1 for saturation coverage. The completion of the first and second structures

is associated with 1/4 and 3/4 of saturation coverage. Since LEED patterns appearing at these relative coverages may be associated with absolute coverages of 1/4 and 3/4 ML, saturation coverage is 1 ML or 9.6×10^{14} molecules/cm^2. Moreover, this determination coincides with results obtained from adsorption kinetics (q.v.).

Assuming a pre-exponential factor of 10^{13} sec^{-1}, desorption energies of 23, 29 and 35 kcal/mole are associated with the three features, but this approximation is both unnecessary and incorrect (26). These data may be analyzed to yield the coverage dependences of the pre-exponential factor $[\nu_d(\theta)]$ and the desorption energy $[E_d(\theta)]$ in the Arrhenius expression for desorption

$$R_d = -C_s \frac{d\theta}{dt}$$

$$= \nu_d(\theta) C_s \theta \exp\,[-E_d(\theta)/kT] \qquad (2)$$

where R_d, C_s, k, t, and T are the desorption rate, the concentration of CO sites ($9.6 \times 10^{14}/cm^2$), the Boltzmann constant, time and surface temperature. The order of the desorption is assumed to be one since CO adsorption is non-dissociative. Because the experimental chamber is pumped rapidly, the desorption rate is proportional to the intensity of the mass spectrometer signal (I) so that

$$-\frac{d\theta}{dt} = \eta I \qquad (3)$$

where η is the proportionality constant. The coverage for any point in

a desorption spectrum is determined by integrating Eq. (3),

$$\theta_o - \theta(t) = \eta \int_o^t I dt \qquad (4)$$

No special form for the heating schedule need be assumed in performing this integration if the intensity and temperature are monitored as independent functions of time. If the initial coverage (θ_o) is known, the constant (η) may be determined by integrating Eq. (4) over the entire spectrum. Since the desorption rate and temperature for a particular coverage vary with the initial coverage, an Arrhenius plot of $\ell n R_d$ versus 1/T for a particular coverage gives the desorption energy from the slope $[- E_d(\theta)/k]$ and the pre-exponential factor from the intercept $\ell n[\nu_d(\theta) C_s \theta/\eta]$. However, the resulting values are highly inaccurate because the desorption rate and temperature do not vary strongly with coverage if the pre-exponential factor is large ($\sim 10^{13} sec^{-1}$).

Excellent accuracy is obtained if the heating schedule, rather than the initial coverage, is altered to vary the desorption rate and temperature for a particular coverage. Ten desorption spectra of CO from clean Ir(110) were collected by varying the heating rate from 4.5 to 212 K/sec (27). By exposing the surface to 10.0 L of CO at 300 K, the initial surface coverage was fixed at 0.82 ML for each run. Arrhenius plots constructed from these data are shown in Fig. 6a. This procedure could be inaccurate since three features appear in the desorption spectra, implying that desorption may occur from multiple distinct sites at any particular coverage. However, the peaks in the spectra are separated in energy sufficiently that desorption from multiple sites is a second order

effect since the difference in the desorption energies of the peaks is greater than 10% of the desorption energy of the highest temperature peak (26). Moreover, LEED data (q.v.) indicate the overlayer compresses continuously so that the three peaks in the thermal desorption spectra appear as a result of interadsorbate interactions, rather than desorption from multiple sites.

In Fig. 7a, the desorption energy and pre-exponential factor from the Arrhenius plots are shown as a function of surface coverage. Substantial decreases in the desorption energy near 0.25 and 0.75 ML are associated with the completion of peaks in the desorption spectra and with the formation of LEED structures. Because CO adsorption on the platinum metals is not believed to be activated (28), the desorption energy is nearly equal to isosteric heats measured for CO chemisorption on the (110) faces of other platinum metals. In the low coverage limit, the desorption energy 37 kcal/mole, does not differ appreciably from isosteric heats of 30, 31, 40, 31 and 37 kcal/mole measured for Ni (2), Rh (3), Pd (5), Pt (6) and Ir (8). Although the isosteric heats for Ir, Ni (2) and Pd (5) decrease continuously with coverage, substantial decreases in the isosteric heats occur at different coverages. As shown in Fig. 7a, the desorption energy of chemsorbed CO on Ir varies almost linearly with coverage. Some models for chemisorption indicate that a linear decrease in the isosteric heat is associated with intermolecular repulsion (29), but the large decrease for Ir, 16 kcal/mole-ML, cannot be attributed solely to nearest-neighbor repulsive interactions. A model for the adsorption kinetics (q.v.) indicates that these interactions reduce the isosteric heat by about 2 kcal/mole-ML. The additional decrease may be attributed to other phenomena, such as changes in CO binding sites with coverage or longer-range inter-adsorbate interactions. The former conjecture is unlikely since it contradicts the results from

CPD measurements and from the continuous compression seen with LEED (q.v.).

The compensation effect (28) occurs for CO desorption from Ir; that is, the desorption energy and the pre-exponential factor vary in sympathy. As discussed elsewhere (30), this effect should be common to most desorption systems. To describe the compensation effect in desorption kinetics, Bauer and coworkers (31) developed a modified form of the Arrhenius expression,

$$R_d = \nu_o C_s \theta^n \exp\left[-\frac{E_d(\theta)}{k}\left(\frac{1}{T} - \frac{1}{T_s}\right)\right] \qquad (5)$$

where n, ν_o and T_s are constants, and are equal to 1, $7 \times 10^{5 \pm 1}$ sec^{-1} and 1240 ± 50 K, respectively, for CO desorption from clean Ir(110).

Activated complex theory (28) is useful for rationalizing the substantial decrease in the pre-exponential factor with coverage. In this theory, the pre-exponential factor is related to the entropies of the activated and chemisorbed phases, $S_*(\theta)$ and $S_s(\theta)$,

$$\ell n \nu_d(\theta) = [S_*(\theta) - S_s(\theta)]/k \qquad (6)$$

The activated phase may be thought of as a precursor state to desorption. Since the pre-exponential factor decreases with coverage, either the entropy in the activated phase decreases, or the entropy in the adsorbed phase increases. Both conjectures are reasonable. Since the number of molecules in the activated phase is approximately proportional to the number in the adsorbed phase, the population in the activated phase

increases, but the entropy decreases as the number increases (32). If the overlayer becomes more mobile as coverage increases, and the mobility of the activated complex does not change, the entropy of the adsorbed phase increases. This conjecture is substantiated by theories of CO chemisorption on the platinum metals, which indicate that CO binds less deeply in the potential wells of the surface as coverage increases (24). The precursor model for desorption developed by King (33) is not appropriate for describing the change in the pre-exponential factor with coverage. This theory would predict that the pre-exponential factor increases with coverage, but the opposite trend is observed.

Desorption spectra of CO from oxidized Ir(110) are different from those obtained on a clean surface. Representative spectra from the oxidized surface with a heating rate (β) of 40 K/sec are shown in Fig. 5b. CO reacts with the oxide layer to produce both CO and CO_2 as desorption products. Since some CO is created when CO_2 is ionized by the mass spectrometer, this CO signal, which is deduced from the cracking pattern for CO_2 and from the CO_2 spectrum, has been subtracted from each CO spectrum. To account for the different sensitivities of CO and CO_2 in the mass spectrometer, the CO_2 spectra have been multiplied by a weighing factor also. From comparing the area under the CO spectrum for clean Ir(110) to the sum of the CO and CO_2 areas for the oxidized surface, saturation coverage on oxidized Ir(110) is 1 ML. This result is also in agreement with the observed (2x1)p1g1 LEED structure. A maximum of 65% of the oxide layer reacts with CO at 4 L exposure. This fraction decreases slightly for higher exposures. The oxide is distinct from chemisorbed oxygen because all chemisorbed oxygen reacts with CO under similar conditions. Further details of the CO oxidation reaction

involving the two forms of surface oxygen are discussed elsewhere (15).

Two peaks with relative intensities of 1:1 at saturation coverage appear in the CO desorption spectra from the oxidized surface. McCabe and Schmidt (7) observed similar two-peak spectra for CO desorption from an oxidized Pt(110) surface, but reported different spectra for a clean surface. If the pre-exponential factor is assumed to be 10^{13} sec^{-1}, desorption energies of 29 and 36 kcal/mole are associated with the two peaks for oxidized Ir(110). The coverage dependences of the desorption energy and pre-exponential factor are calculated from Arrhenius plots constructed from the desorption spectra and are shown in Fig. 6b. Suitable accuracy is obtained because the pre-exponential factor is relatively small ($\lesssim 10^{11}$ sec^{-1}) (30). The CO coverage is determined from the weighted sum of the integrals [Eq. (4)] of the CO and CO_2 spectra. Although the CO and oxide coverages should be held constant in an Arrhenius plot, the oxide coverage varies for each plot shown in Fig. 6b. Thus, the calculated rate parameters may be somewhat inaccurate.

The coverage dependences of the desorption energy and the pre-exponential factor for the oxidized surface are shown in Fig. 7b. Although the shapes of these curves are nearly identical to those obtained for the clean surface, both the desorption energy and the pre-exponential factor for any coverage are substantially lower on oxidized Ir(110). In the limit of low coverage, these rate parameters assume values of 31 kcal/mole and 10^{11} sec^{-1}. The desorption features from the oxidized surface appear at higher temperatures than those from the clean surface, not because the desorption energy is larger on the oxidized surface, but because the pre-exponential factor is smaller. The desorption energy is

again a strong function of coverage, varying by 16 kcal/mole over 0.5 ML, and the compensation effect occurs once more. The parameters ν_o and T_s in Eq. (5) assume values of $4 \times 10^{\pm 1}$ sec^{-1} and 650 ± 10 K, which differ substantially from those for the clean surface. As Bauer and coworkers have suggested (31), these parameters are quite sensitive to the condition of the surface. Since the desorption kinetics of CO differ markedly for the clean and oxidized surfaces, the two surfaces might be expected to display different catalytic behavior for the CO oxidation reaction. This is indeed the case (15).

VI. Adsorption Kinetics

By integrating Eq. (4) over an entire desorption spectrum, an initial coverage is associated with a gas exposure. In this way, the adsorption kinetics of CO on clean and oxidized Ir(110) at 300 K have been deduced from the spectra in the previous section and are shown in Fig. 8. An absolute coverage may be associated with each exposure since, as shown previously, the saturation coverage on either surface at 300 K is 1 ML (9.6×10^{14} molecules/cm^2). The adsorption kinetics for CO exposures below 10 L are identical for the two surfaces within experimental error. Above 10 L exposure, the coverage of CO increases slowly until the surface saturates at several hundred Langmuirs. This trend is definitely not an artifact resulting from drift in the gain of the mass spectrometer and has been observed previously on Ir(111) (18). Near saturation coverage, the overlayer desorbs slowly at a rate of about 10^{-4} ML/sec at 300 K. This behavior is predicted from the

measurements of desorption kinetics (q.v.) by extrapolating the rate parameters to saturation coverage.

The probability of adsorption (S), or the sticking probability, is the ratio of the number of gaseous CO molecules that chemisorb to the number that impinge on the surface (9),

$$S = (2\pi M_{CO} kT_g)^{\frac{1}{2}} C_s \frac{d\theta}{d\varepsilon} \qquad (7)$$

where M_{CO}, T_g, and ε are the molecular weight, temperature, and exposure of the gaseous CO. After the exposures are adjusted for the CO sensitivity of the ion gauge that monitored the exposures, an initial sticking probability (S_0) of 1.04 ± 0.05 is calculated for CO adsorption on either surface. This result agrees closely with other values reported for CO adsorption on the platinum metals (9). Regardless of the metal or the orientation of the surface, the initial sticking probability is nearly one.

Different models for adsorption kinetics fit the data, depending on the coverage range. The Langmuir model (35) is appropriate below 0.4 ML, a first-order precursor model (36) is appropriate below 0.7 ML, but the best overall fit is obtained with a second-order precursor model developed by King (37). This model has two noteworthy features. A molecule in the precursor state may not chemisorb unless it is above an empty adsorption site having one or more empty nearest neighbors. Second,

the overlayer is ordering and disordering continually because of repulsive or attractive interactions between adsorbates, and thermal randomization of the adlayers. This model is apposite for the non-dissociative adsorption of CO if the probability of chemisorption is zero above an empty site with no empty nearest neighbor and is constant otherwise. This hypothesis does not imply that an adsorbed molecule cannot be surrounded completely by other adsorbates, but does mean that chemisorption must occur in one site in a pair of empty and adjacent sites.

In the second-order precursor model (37), the sticking probability (S) is related to the initial sticking probability (S_0), a constant (κ) that is inversely proportional to the lifetime of the precursor state, and the fraction of empty sites with an empty nearest neighbor (θ_{oo}),

$$\frac{S}{S_0} = \left[(1 - \kappa) + \kappa/\theta_{oo}\right]^{-1} \tag{8}$$

The value of θ_{oo} depends on the fractional coverage and a constant B,

$$\theta_{oo} = (1 - \theta) - \frac{2\theta(1 - \theta)}{\left(1 - 4B\theta(1 - \theta)\right)^{\frac{1}{2}} + 1} \tag{9}$$

but is insensitive to whether the adlayer is twofold, fourfold, or six-fold symmetric (34). The constant B is related to the attractive or repulsive energy between nearest neighbors,

$$\psi = - kT\ell n(1 - B) \tag{10}$$

where a negative value for ψ is associated with an attractive interaction. For CO adsorption on both clean and oxidized Ir(110) at 300 K, the second-order model with S_o = 1.0, κ = 0.2, and B = 0.96 fits the kinetic data well. This model in the integrated form, obtained from Eqs. (7)-(9), is shown as a solid line in Fig. 8.

The repulsive energy determined from this model, ψ = 2 kcal/mole, may be compared to that from a Lennard-Jones potential between two adsorbates (38),

$$\psi(r) = 4kT_\sigma\left[\left(\frac{\sigma}{r}\right)^{12} - \left(\frac{\sigma}{r}\right)^{6}\right] \qquad (11)$$

Values of σ = 3.76 Å and T_σ = 100 K for gaseous CO approximate the values for chemisorbed CO. The distance between the centers of adsorbates (r) may be estimated as 3.3 Å, which is the largest distance consistent with (2x1)p1g1 symmetry of the overlayer on the oxide (see LEED results) (25). Hence, the value for ψ from the Lennard-Jones potential, 2 kcal/mole, equals the value from the kinetic model. If the overlayer has p1g1 symmetry, each adsorbate has two nearest neighbors so that intermolecular repulsion reduces the heat of adsorption by about 2 kcal/mole-ML. The decrease in the heat of adsorption with coverage, 16 kcal/mole-ML, results only in part from intermolecular repulsion.

Second-order behavior for CO adsorption kinetics on the platinum metals is not unique to Ir(110) and has been reported on Ir(111) (9). Moreover, two independent groups have reported the coverage dependence of the sticking probability of CO on Pt(111) (39), which shows second-order behavior. This kinetic behavior might be observed for other

surfaces of the platinum metals if higher exposure regimes (> 10 L) are investigated.

The adsorption kinetics of CO on Ir(110) at 90 K are inferred from CPD measurements (q.v.). The sticking probability is uniformly one up to 0.8 ML and decreases somewhat at higher coverages. The saturation coverage at 90 K is 1 ML. For this reason and since the CPD is not altered after annealing the overlayer to 300 K, CO does not appear to adsorb physically at 90 K. At first glance, the measurements of a uniformly large sticking probability at 90 K might appear to be inconsistent with the second-order kinetics observed at 300 K. If thermal randomization of the overlayer is negligible at 90 K, the parameter B would approach one so that the saturation coverage at 90 K would be 0.5 ML (37). However, since 90 K is near the sublimation point of CO, the lifetime of the precursor state is protracted, and the parameter κ approaches zero. Therefore, the extended lifetime of the precursor state compensates for the poor thermal randomization of the overlayer so that the kinetic behavior at 90 K is consistent with the second-order behavior at 300 K.

VII. Charge transfer to the Overlayer

The contact potential differences (CPD) for clean and oxidized Ir(110) at 90 and 300 K were measured as a function of CO exposure and are shown in Fig. 9. In all cases, the CPD is positive, i.e., the work function of the surface increased with CO coverage. For saturation coverage on the clean surface, the CPDs were 0.28 and 0.23 ± 0.01 eV at 90 and 300 K; on

the oxidized surface, 0.17 and 0.13 ± 0.01 eV at 90 and 300 K. For any temperature, and coverage, the CPD is markedly different on the two surfaces. Previously reported variations in the CPD on identical surfaces under the same conditions, which are as large as 0.2 eV for CO on some platinum metals (10), may result from inadvertent measurements on oxidized surfaces.

The sensitivity of the CPD to the surface oxide is estimated readily. If 1 ML of CO is adsorbed on the oxidized surface and is flashed off at 800 K, a subsequent measurement of CPD versus CO exposure lies between the curves for clean and oxidized Ir(110). The curve for the clean surface is obtained after this cycle is repeated five times. As stated previously, about 65% of the complete oxide layer reacts with 1 ML of CO during a flash. If 65% of the remaining oxide is removed with each additional cycle, then 0.005 ML of oxide remains after five cycles. Hence, the CPD is sensitive to oxide concentrations of less than 1%, which exceeds the detection limit of AES.

MacDonald and Barlow (40) have related the CPD to the apparent polarizability (α) and the apparent dipole moment (μ) for the adsorbate near zero coverage. Because of the image charge in metals, the apparent dipole moment is twice the component of the true dipole moment of the adsorbate that is normal to the surface and must be divided by two before it is compared to gas phase values. The apparent polarizability is the normal component of the true dipole moment divided by the normal component of the effective field at the surface (41). This field equals the average surface field that occurs at a binding site in the absence

of induced charge ($\underset{\sim}{E}_n$) plus the field that is induced by the dipole moment ($\underset{\sim}{E}_i$). Since the induced field produces the image charge of the dipole, the apparent polarizability is the ratio of the apparent dipole moment (μ) to the average field normal to the surface (E_n). These conventions for μ and α, which are used throughout this article and others (40), are convenient because the induced field may be ignored completely if the apparent dipole moment (μ) is used in place of the true dipole moment ($\mu/2$). If the sign of μ is positive, negative charge is transferred from the surface to the adlayer.

The model of MacDonald and Barlow (40), a modified form of the Topping model, accounts for the depolarization of the overlayer resulting from dipole-dipole interactions. For a fixed surface temperature, the ratio of the coverage (θ) to the CPD at that coverage ($\Delta\phi$) is plotted versus coverage. The slope of the resulting line is $(\alpha\Lambda C_s^{\frac{1}{2}}/4\pi\mu)$, and the intercept is $(1/4\pi C_s\mu)$, where Λ is a structural parameter for the overlayer and is equal to 9.46 if the overlayer has the periodicity of the (110) surface (14). This model, one of two developed by MacDonald and Barlow, is appropriate for adsorption at specific sites, rather than on an amorphous surface.

For CO adsorption at 300 K, Topping plots of $\theta/\Delta\phi$ versus θ are constructed by relating a coverage from adsorption kinetics to a CPD through points of constant exposure. At 90 K, the plots are produced by assuming that the initial sticking probability is one, and the plot gives a single straight line. The adsorption kinetics at 90 K were deduced from these plots. The initial sticking probability is one because CO adsorption is not activated. Since the plots at

300 K give a single line, the plots at 90 K presumably behave similarly. Moreover, this construction gives a value of 1 ML for saturation coverage of CO on both clean and oxidized Ir(110) at 90 K. However, Topping plots do not produce a single line always, but sometimes give two or more linear portions, as for oxygen adsorption on Ir(110) (14). Plots for CO adsorption on clean and oxidized Ir(110) at 90 and 300 K are shown in Fig. 10. The calculated values for the normal components of the polarizability and dipole moment for CO, shown in Fig. 10, are temperature dependent.

Two models, involving the thermal randomization of the rotational and the translation motions of the adsorbate, will be developed to describe the temperature dependences of the polarizability and the dipole moment. Although population in a vibrational mode that is normal to the surface varies with temperature, the CPD does not because the mean displacement of the adsorbate from the surface, which is proportional to the dipole moment, is constant (41).

The CPD for any coverage may change as a result of the thermal randomization of a rotational motion (42). The Auger spectra of CO on Ir(110) (q.v.) indicates that CO is attached to the surface through a carbon-metal bond. Assuming that the substrate is flat, the adsorbate may precess about this bond at angles of up to 90° with respect to the surface normal. The field at the surface ($\underset{\sim}{E}$) tends to orient the

apparent permanent dipole moment of the adsorbate ($\underset{\sim}{\mu}_o$) either normal or parallel to the surface, depending on whether the signs of the field and the dipole are either aligned or juxtaposed. The potential energy (U) for orientation is (41)

$$U = \mp \underset{\sim}{\mu}_o \cdot \underset{\sim}{E} = \mp \mu_o E_n \cos\theta \tag{12}$$

where θ is the angle between the dipole and the surface normal. That is, the average field ($\underset{\sim}{E}$) for a flat metal surface equals the normal component ($\underset{\sim}{E}_n$). The sign of U is minus (plus) if the field and the dipole are aligned (juxtaposed).

The apparent polarizability [$\alpha(T)$] of the overlayer equals the normal component of the apparent dipole moment [$\mu = \mu_o \langle\cos\theta\rangle$] divided by the magnitude of the field (E_n) (41)

$$\alpha(T) = \mu_o \langle\cos\theta\rangle / E_n \tag{13}$$

Here $\langle\cos\theta\rangle$ is the thermal average value of $\cos\theta$,

$$\langle\cos\theta\rangle = \frac{\int_\Omega e^{-U/kT} \cos\theta \, d\Omega}{\int_\Omega e^{-U/kT} \, d\Omega}$$

$$= -\frac{1}{x} \pm \frac{e^x}{e^x - 1} \tag{14}$$

where $d\Omega$ is an element of solid angle in the half sphere in which the

adsorbate may precess, and $x = \mu_o E_n/kT$. Because μ_o, E_n and T are typically 1D ($=10^{-18}$esu - cm), 10^6 V/cm, and 100 K; x is on the order of 0.1. Hence, Eq. (14) may be expanded in a Taylor series about x = 0. To first order, $\langle\cos\theta\rangle = \frac{1}{2} \pm \frac{x}{12}$, and

$$\alpha(T) = \frac{1}{2}\alpha_e \pm \frac{\mu_o^2}{12kT} \qquad (15)$$

where the electronic polarizability (α_e) is defined as μ_o/E_n. If the signs of the field and the dipole are aligned (juxtaposed), the sign of the second term is plus (minus).

Except for the numerical coefficients on the two terms, Eq. (15) is identical to the Langevin relation (41), which describes the temperature dependence of the polarizability of a homogeneous dielectric field (41). The mathematical formulation of the two models is identical except for the limits of integration in Eq. (14). If the true dipole moment replaces the apparent value, Eq. (15) reduces to the Langevin relation. However, the Langevin relation in terms of the apparent dipole moment has been applied erroneously to metal surfaces (40). In both cases for a surface, the limiting value of $\langle\cos\theta\rangle$ with increasing temperature is 0.5, which explains why the term $\frac{1}{2}\alpha_e$ rather than α_e appears in Eq. (15).

Since the polarizabilities for CO on Ir(110) decrease with increasing temperature, the signs of the field at the Ir(110) surface and the permanent dipole moment of CO are aligned and a plus sign is associated with the second term in Eq. (15). The term x for these data is of order one so that μ_o and α_e must be determined from Eqs. (13) and (14).

Electronic polarizabilities of 20 and 60 $\AA^3$ and apparent permanent dipole moments of 0.55 and 0.8 D are obtained for CO on clean and oxidized Ir(110), respectively. At any temperature, $\langle\cos\theta\rangle$ is $\mu(T)/\mu_o$ so, in three of the four cases, $\langle\cos\theta\rangle$ is significantly less than 0.5. Because Eq. (14) dictates that $\langle\cos\theta\rangle$ is no less than 0.5 if the signs of the field and the dipole are aligned, this model does not describe the temperature dependences of the CPD for these data consistently. This model might be appropriate if CO binds on the ridges in the (001) azimuth where $\langle\cos\theta\rangle$ could be less than 0.5. However, the diameter of a chemisorbed CO molecule, 3.3 Å (as shown in the last section) is comparable to the distance between the troughs, 3.8 Å, so the surface should appear flat to chemisorbed CO. Therefore, this rotational model does not describe the temperature dependence of the CPD adequately for these data.

In the second model, which is similar to one developed by Ertl (39), the CPD for any coverage changes with temperature, because the adsorbed molecules are energetically partitioned between two states of different electronic polarizabilities and permanent dipole moments. Since the LEED patterns of CO disorder only in the [001] azimuth as the surface temperature increases, thermal randomization must be significant in the [001] azimuth, but may be ignored for the [110] azimuth. The potential variation for chemisorption in the [001] azimuth may be approximated by a periodic square well of height U. The magnitude of the permanent dipole moment is μ_1 for the minima and μ_2 for the maxima. The electronic polarizability varies similarly (43). Hence, by Boltzmann statistics (32), the temperature dependence of the apparent dipole moment is

$$\mu(T) = \frac{\mu_1 e^{U/kT}}{1 + e^{U/kT}} + \frac{\mu_2}{1 + e^{U/kT}} \tag{16}$$

where the rotational motion will be ignored for the time being. This expression may be simplified in the limit of small U/kT,

$$\mu(T) = \bar{\mu} + \frac{\Delta\mu}{4} \frac{U}{kT} \tag{17}$$

where $\bar{\mu} = \frac{\mu_1 + \mu_2}{2}$ and $\Delta\mu = \mu_2 - \mu_1$. The same expression in $\bar{\alpha}$ and $\Delta\alpha$ is obtained for the temperature dependence of the apparent polarizability $[\alpha(T)]$. If the electric fields at both types of sites are equal, the ratio of the dipole moment to the polarizability for any temperature is independent of temperature and equals the field, but this need not be the case.

This ratio is independent of temperature for CO on either clean or oxidized Ir(110), and this result is not surprising. The diameter of chemisorbed CO (d_{CO} = 3.3 Å) is large compared to the depth of the troughs on the surface, which are about one-half the diameter of a Ir atom (d_{Ir} = 2.7 Å). Furthermore, the area occupied by chemisorbed CO ($\frac{\pi}{4} d_{CO}^2 = 9$ Å^2) is comparable to the area of a unit cell ($\sqrt{2}\, d_{Ir}^2 = 10$ Å^2). Hence, the field acting on CO is averaged over a relatively large volume on the surface and is independent of the binding site. This is not the case for oxygen on Ir(110) where the dimensions of the adsorbate are much smaller than the dimensions of a unit cell (14). The average surface fields for CO on clean and oxidized Ir(110) are approximately 10^7 and 3×10^6 V/cm. As an aside, the electronic polarizability depends on the total field normal to the

surface. In field emission microscopy (44), the applied field ($\sim 10^7$ V/cm) is comparable to the naturally occurring field at the surface so the measured CPD is understandably different from values obtained by other methods, which perturb the surface minutely.

Since this translational model has three parameters, but only two dipole moments are available from the data, values for α_1, α_2, μ_1, μ_2 and U may be estimated only. Based upon the electronegativities for C, O, and Ir atoms (45), the dipole of CO is not expected to have a large negative sign; that is, a large transfer of charge toward the carbon atom and the surface is not anticipated. A minimum value of -1 D may be postulated for the dipole moment of CO. This requires that the potential (U) be 0.1 - 1 kcal/mole, which is close to the potential between sites for CO on Pt(111), 0.5 kcal/mole (39). Since the potential is on the order of tenths of kcal/mole, $\alpha_1 \gtrsim \alpha(90\ K)$, $\alpha_2 \lesssim 0$, $1\ D \gtrsim \mu_1 \gtrsim \mu(90\ K)$ and $-1\ D \lesssim \mu_2 \lesssim 0$ for CO on either clean or oxidized Ir(110). The dipole moments on the oxidized surface are understandably smaller than those on the clean surface, for the oxide layer is expected to localize charge closer to the surface.

Because the dipole moment of CO on clean or oxidized Ir(110) is between -1 and 1 D, the overlayer is rotationally randomized above 300 K. In this temperature range, the thermal average angle between CO and the surface normal is 60°, the calculated values of μ_1 and μ_2 are directly comparable to gas phase values, and the apparent polarizability equals the true dipole moment ($\mu/2$) divided by the average surface field (E_n). These results appear to contradict those of an angular resolved UPS study of CO on Ni(100) (46), which indicate that the angle between CO and the

surface normal is no greater than 20° at 300 K. However, the dipole moment and polarizability of CO on Ni(100) are expected to be very different from the values for Ir(110) because the CPD for an equivalent coverage is markedly different (10).

The dipole moments from the translational model are close to the gas phase value, - 0.12 D, where the minus sign indicates that the net flow of charge is toward the C atom (47). Although the bonding electrons move toward the O atom, the non-bonding electrons move toward the C atom. Similar chemical arguments are used to determine whether sites on top of the rows of surface atoms or sites in the troughs are favored energetically. The flow of charge in the C-Ir bond will be ignored because the electronegativities of these atoms are nearly identical (45). From the Pauli exclusion principle, the movement of the non-bonding electrons from CO toward the surface is more likely for sites on the rows than in the troughs where the average density of metal electrons is greater (24). Only sites on the rows are expected to have a vanishing or negative dipole moment. Therefore, trough sites are energetically favored since the dipole moment in the energetically disfavored sites on Ir(110) is no greater than zero. This reasoning agrees with several theoretical studies of CO chemisorption on the platinum metals (24), but contradicts arguments based on the vibrational spectroscopy of CO on these metals (48). However, these arguments have been questioned elsewhere (39), and the experimental results are controversial (49).

Both experiments and theory demonstrate clearly that the CPD for any coverage may vary with temperature. By assuming this variation is

insignificant, isosteric heats of adsorption often are measured for points of constant CPD (50). A method for estimating the error in these measurements will be developed.

If the partial molar volume of the adsorbed phase is small compared to that in the gas phase (v_G), and if the surface area (a) does not vary upon chemisorption, the chemical potential in the adsorbed phase (μ_s) is approximated by (51)

$$d\mu_s = - \bar{s}_s dT + \left(\frac{\partial \mu_s}{\partial \theta}\right)_{T,P,a} d\theta \qquad (18)$$

where $\bar{s}_s$ is the partial molar entropy of the adsorbate. The chemical potential for an ideal gas (μ_G) is

$$d\mu_G = - s_G dT + kT d\ell nP \qquad (19)$$

where s_G and P are the molar entropy and the pressure in the gas phase. (The symbol μ with a subscript of s or G denotes a chemical potential, not a dipole moment.) Since the chemical potentials for the two phases are equal at equilibrium, and since the isosteric heat q_{st} is approximated closely by $(s_G - \bar{s}_s)T$, the Clausius-Clapeyron relation for points of constant CPD is

$$- \left[\frac{\partial \ell nP}{\partial (1/T)}\right]_{\Delta\phi} = \frac{q_{st}}{k} + \frac{T}{k}\left(\frac{\partial \mu_s}{\partial \theta}\right)_{T,P,a} \left(\frac{\partial \theta}{\partial T}\right)_{\Delta\phi} \qquad (20)$$

Eq. (20) is simply the usual Clausius-Clapeyron relation (51) with an

extra term, which equals the error (ε/k) introduced by ignoring the temperature variation of the CPD.

The term $\left(\frac{\partial\theta}{\partial T}\right)_{\Delta\phi}$ is evaluated by assuming that the Topping equation relates the coverage and CPD at any temperature

$$\left(\frac{\partial\theta}{\partial T}\right)_{\Delta\phi} = -\theta^2 \left(\frac{4\pi C_s}{\Delta\phi}\frac{\partial\mu}{\partial T} - \Lambda C_s^{3/2}\frac{\partial\alpha}{\partial T}\right) \tag{21}$$

In the translational model, both the polarizability and the dipole moment of the adsorbate vary with temperature according to Eq. (17). Once again, the Topping equation gives $\Delta\phi$ so Eq. (21) reduces to

$$\left(\frac{\partial\theta}{\partial T}\right)_{\Delta\phi} = \frac{\theta}{T^2}\left(\frac{U}{4k}\right)\left[\frac{\Delta\mu}{\mu_R} + \theta\Lambda C_s^{3/2}\left(\frac{\Delta\mu}{\mu_R}\alpha_R - \Delta\alpha\right)\right] \tag{22}$$

where θ, μ_R, and α_R are the coverage, dipole moment, and polarizability associated with the CPD ($\Delta\phi$) at a fixed reference temperature. At this temperature, the coverage-CPD relation is established to give the coverage dependence of the isosteric heat. The error (ε in kcal/mole) depends upon the choice of reference temperature (usually 300 K), as well as the coverage and the temperature at which the Clausius-Clapeyron expression is evaluated. Because the average surface field for Ir(110) is the same for the two types of binding sites, the second term in brackets vanishes. As argued previously, this uniformity should occur generally on metals if the dimensions of chemisorbed CO are comparable or larger than the dimensions of a unit cell. Assuming that the potential variation between sites (U) is 0.5 kcal/mole, which has been shown to be

a reasonable value, and that the reference temperature is 300 K, μ_R = 0.17 D and $\Delta\mu$ = 0.70 D (from Eq. 16). Because the potential energy for rotation is small for this case ($\mu_o E_n \langle\cos\theta\rangle \sim$ 0.1 kcal/mole), the adlayer is rotationally randomized above 90 K. Little error is introduced by entirely ignoring this motion. For CO on clean Ir(110), Eq. (22) is approximated closely by

$$\left(\frac{\partial\theta}{\partial T}\right)_{\Delta\phi} = \frac{\theta U}{T^2 k} \tag{23}$$

Intuitively, the term $\left(\frac{\partial\mu}{\partial\theta}\right)_{T,P,a}$ in Eq. (20) is approximately $-\left(\frac{\partial E_d}{\partial\theta}\right)$ = 16 kcal/mole-ML, but may be estimated more precisely. The enthalpy in the adsorbed phase (H_s) is related to the entropy (S_s), the surface pressure (ϕ), and the chemical potential (μ_s) by (51)

$$H_s = TS_s - \phi a + \mu_s n_s \tag{24}$$

where n_s is the number of adsorbed molecules. By differentiating with respect to n_s with T, P, and a constant, Eq. (24) becomes

$$\bar{h}_s = T\bar{s}_s - \left(\frac{\partial\phi}{\partial n_s}\right)_{T,P,a} a + \mu_s + \left(\frac{\partial\mu_s}{\partial n_s}\right)_{T,P,a} n_s \tag{25}$$

where $\bar{h}_s = \left(\frac{\partial H_s}{\partial n_s}\right)_{T,P,a}$ and $\bar{s}_s = \left(\frac{\partial S_s}{\partial n_s}\right)_{T,P,a}$ are partial molar

quantities. Because $\overline{h}_s - T\overline{s}_s = \mu_s$, $n_s/aC_s = \theta$, and $\phi = kT \int_0^P (n_s/a)d\ell nP$, which is a good approximation of the Gibbs equation for surfaces (51),

$$\left(\frac{\partial \mu_s}{\partial \theta}\right)_{T,P,a} = \frac{kT}{\theta} \frac{\partial}{\partial \theta} \left[\int_0^P \theta d\ell nP\right]_{T,P,a} \tag{26}$$

Eq. (26) is evaluated directly from thermodynamic data.

When no such data are available, Eq. (26) may be evaluated from kinetic data. At equilibrium, the product of the flux of molecules to the surface and the sticking probability [from Eq. (7)] equals the desorption rate [Eq. (5)]

$$\frac{PS(\theta, T)}{(2\pi M_{CO} kT_g)^{\frac{1}{2}}} = \nu_0 C_s \theta \exp \left[-\frac{E_{d(\theta)}}{k}\left(\frac{1}{T} - \frac{1}{T_s}\right)\right] \tag{27}$$

Substituting Eq. (27) into Eq. (26),

$$\left(\frac{\partial \mu_s}{\partial \theta}\right)_{T,P,a} = \frac{kT}{\theta} - \frac{kT}{\theta}\left[\frac{\partial}{\partial \theta}\int_0^\theta \theta \left(\frac{\partial \ell nS(\theta, T)}{\partial \theta}\right)_T d\theta\right]$$

$$- \left(\frac{1}{\theta} - \frac{T}{T_s}\right)\frac{\partial}{\partial \theta}\left[\int_0^\theta \theta \left(\frac{\partial E_d(\Theta)}{\partial \theta}\right) d\theta\right] \tag{28}$$

Eq. (28) is evaluated at the temperature and coverage at which the error in the isosteric heat is to be calculated. For CO on Ir(110), the adsorption kinetics are second order, $S = S_0(1 - \theta)^2$, and the desorption energy varies linearly, or piecewise linearly, with coverage, $\left(\frac{\partial E_d}{\partial \theta}\right) =$

$- E_\delta$. Hence, Eq. (28) reduces to

$$\left(\frac{\partial \mu_s}{\partial \theta}\right)_{T,P,a} = kT \left[\frac{1 + \theta}{\theta(1 - \theta)}\right] + E_\delta\left(1 - \frac{T}{T_s}\right) \tag{29}$$

If the first term is ignored and T is much less than T_s, $\left(\frac{\partial \mu_s}{\partial \theta}\right)_{T,P,a}$ equals $-(\partial E_d/\partial \theta)$ as stipulated earlier.

From Eqs. (20), (23) and (29), the total error in the Clausius-Clapeyron equation evaluated for points of constant CPD and for a reference temperature of 300 K is

$$\varepsilon = U \frac{(1 + \theta)}{(1 - \theta)} + \frac{UE_\delta \theta}{k}\left(\frac{1}{T} - \frac{1}{T_s}\right) \tag{30}$$

where U = 0.5 kcal/mole and T_s = 1240 K. The coverage variation of the energy of desorption, which nearly equals the isosteric heat of adsorption (52), may be approximated by a piecewise linear fit and is shown as a dashed line in Fig. 11. The error in the isosteric heat at 500 K, given by Eq. (30), is superimposed on the isosteric heat and is represented by a dotted line. At 500 K, the adsorption and desorption rates of CO on Ir(110) are approximately 0.5 ML/sec at 10^{-6} torr, which is a typical pressure for evaluating the isosteric heat under UHV conditions. If the temperature is varied by ± 100 K, the error (ε) is not altered significantly, but the 1/T term in ε introduces some skewness in the Clausius-Clapeyron plots [Eq. (20)]. This skewness is negligible to first order. The solid line in Fig. 11 represents the isosteric heat measured for points of constant

CPD if the discontinuities in the piecewise linear fit are smoothed.

Several artifacts arise if the isosteric heat is evaluated for points of constant CPD. Since the measured heats are too large, pre-exponential factors deduced from these heats are also too large because this factor offsets the overestimation of the isosteric heat. The compensation effect in the desorption kinetics may be masked entirely. The error in the isosteric heat is particularly great at high coverages. Moreover, the error is accentuated near coverages at which $\left(\frac{\partial \mu}{\partial \theta}\right)_{T,P,a}$ is large (for example, near coverages at which the overlayer structure changes).

VIII. Conclusions

The chemisorption of CO on a clean surface and a well characterized oxidized surface of Ir(110) has been studied to give insight into the CO oxidation reaction on Ir(110). CO adsorbs non-dissociatively on both surfaces at temperatures between 90 and 1300 K. The saturation coverage on both surfaces at 90 and 300 K is 1 ML (9.6×10^{14} molecules/cm^2). Different LEED patterns form on each surface. On the clean surface, a p(2x2) pattern and a (4x2) pattern disordered in the [001] azimuth, form near 0.25 and 0.75 ML. Between these coverages, the overlayer appears to compress continuously. A $(2x1)_{p1g1}$ pattern forms on the oxidized surface between 0.05 and 1.0 ML, indicating that the overlayer coalesces into islands.

The thermal desorption spectra of CO from the two surfaces differ also. Three peaks with relative intensities of 1:2:1 appear in the spectra from the clean surface; two peaks with relative intensities of 1:1 appear in the spectra from the oxidized surface. Up to 65% of the oxide layer reacts with CO to form CO_2. The desorption energy and the pre-exponential factor in the Arrhenius expression describing the desorption kinetics have been measured as functions of coverage for both surfaces. The compensation effect occurs for both cases, and the shape of the desorption energy versus coverage relations are similar. The desorption energy decreases continuously with increasing coverage up to 0.25 ML, drops off sharply there, and levels off up to 0.6 ML. In the limit of zero coverage, the desorption energy and the pre-exponential factors are 37 kcal/mole and 10^{13} sec^{-1} for the clean surface, and are 31 kcal/mole and 10^{11} sec^{-1} for the oxidized surface. Although the desorption kinetics for the two surfaces differ, the adsorption kinetics are the same. At 90 K, the sticking probability is one up to 0.8 ML, and CO does not appear to adsorb physically. At 300 K, the adsorption kinetics display a second-order behavior with an initial sticking probability of one. This behavior does not result from dissociative adsorption, but from intermolecular repulsion.

The contact potential difference (CPD) increases with CO coverage on both surfaces at 90 and 300 K. The CPD is greater on the clean surface and at lower temperatures. A Topping model, which relates the coverage to the CPD for a fixed temperature, was used to determine the normal components of the dipole moment and the polarizability of CO. The dipole

moment is larger on the clean surface although the polarizability is greater on the oxidized surface. On both surfaces, these quantities decrease with temperature. Two models, involving the thermal randomization of the rotational and the translational motions of CO, were developed to explain the temperature dependence of the CPD. The second model, which is appropriate for CO on both surfaces, suggests that binding sites between the topmost rows of surface atoms are preferred energetically. A method was presented to estimate the error in the isosteric heat of adsorption caused by evaluating the Clausius-Clapeyron equation at points of constant CPD, rather than constant coverage.

Acknowledgment

The authors are grateful for helpful correspondence with Dr. R. M. Lambert and discussions with Professor W. A. Goddard. This research was supported by the National Science Foundation.

References

1. T. N. Taylor and P. J. Estrup, J. Vac. Sci. Technol. 10, 26 (1973).
2. H. H. Madden, J. Küppers and G. Ertl, J. Chem. Phys. 58, 3401 (1973).
3. R. A. Marbrow and R. M. Lambert, Surface Sci. 67, 489 (1977).
4. G. Ertl and P. Rau, Surface Sci. 15, 443 (1969).
5. H. Conrad, G. Ertl, J. Koch and E. E. Latta, Surface Sci. 43, 462 (1974).
6. C. M. Comrie and R. M. Lambert, JCS Faraday I 72, 1659 (1976).
7. R. W. McCabe and L. D. Schmidt, Surface Sci. 60, 85 (1976).
8. K. Christmann and G. Ertl, Z. Naturforsch. 28a, 1144 (1973).
9. W. H. Weinberg, C. M. Comrie and R. M. Lambert, J. Catal. 41, 489 (1976).
10. S. Andersson, Solid State Commun. 21, 75 (1977).
11. S. W. Johnson and R. J. Madix, Surface Sci. 66, 189 (1977).
12. C. T. Cambell, D. C. Foyt and J. M. White, J. Phys. Chem. 81, 491 (1977).
13. V. P. Ivanov, G. K. Boreskov, V. I. Savchenko, W. F. Egelhoff, Jr. and W. H. Weinberg, J. Catal. 48, 269 (1977).
14. J. L. Taylor, D. E. Ibbotson and W. H. Weinberg (in preparation).
15. J. L. Taylor, D. E. Ibbotson and W. H. Weinberg (in preparation).
16. J. L. Taylor, Ph.D. Thesis, California Institute of Technology, 1978.
17. J. L. Taylor and W. H. Weinberg (in preparation).
18. C. M. Comrie and W. H. Weinberg, J. Chem. Phys. 64, 250 (1976).
19. J. C. Fuggle, E. Umbach and D. Menzel, Solid State Commun. 20, 89 (1972).
20. M. P. Hooker and J. T. Grant, Surface Sci. 62, 21 (1977).
21. W. E. Moddeman, T. A. Carlson, M. O. Krause, B. P. Pullen, W. E.

Bull and G. K. Schweitzer, J. Chem. Phys. 55, 2317 (1971).

22. R. M. Lambert and C. M. Comrie, Surface Sci. 38, 197 (1973).

23. N. M. Abbas and R. J. Madix, Surface Sci. 62, 739 (1977).

24. G. Doyen and G. Ertl, Surface Sci. 43, 197 (1974).

25. R. M. Lambert, Surface Sci. 49, 325 (1975).

26. The methods presented in the next two paragraphs are reviewed thoroughly in: J. L. Taylor and W. H. Weinberg, Surface Sci. (submitted).

27. No artifacts were introduced by using these large heating rates. See Ref. 26.

28. G. C. Bond, Catalysis by Metals, Academic Press, New York, 1962.

29. D. H. Everett, Proc. Chem. Soc. 38, 38 (1957).

30. The details are discussed in Ref. 26.

31. E. Bauer, F. Bonczek, H. Poppa and G. Todd, Surface Sci. 53, 87 (1975).

32. R. H. Fowler and E. A. Guggenheim, Statistical Thermodynamics, Macmillan, New York, 1939.

33. D. A. King, Surface Sci. 64, 43 (1977).

34. This statement is proved by calculating θ_{∞}^{o} for twofold and sixfold symmetries according to King's formulation on p. 250 in . 37.

35. J. B. Taylor and I. Langmuir, Phys. Rev. 44, 423 (1933).

36. P. Kisliuk, J. Phys. Chem. Solids 3, 95 (1957).

37. D. A. King and M. G. Wells, Proc. R. Soc. A339, 245 (1974).

38. J. O. Hirschfelder, C. F. Curtis and R. B. Bird, The Molecular Theory of Gases and Liquid, Wiley, New York, 1954.

39. G. Ertl, M. Neumann and K. M. Streit, Surface Sci. 64, 393 (1977).

40. J. R. MacDonald and C. A. Barlow, J. Chem. Phys. 39, 412 (1963).

41. J. D. Jackson, Classical Electrodynamics, Wiley, New York, 1962, Chapter 4.

42. The motion described in this paragraph is a vibration, rather than

a true rotation. It is called a rotation because the motion is formulated analogously to a rotation in the gas phase.

43. This model is not rigorously correct for describing the polarizability. Because the order in the adlayer changes with temperature, the structural parameter Λ must be temperature dependent. However, ignoring the temperature dependence of Λ introduces little error in the analysis. This is discussed in Ref. 14.

44. G. Ertl and J. Küppers, Low Energy Electrons and Surface Chemistry, Verlag Chemie, Weinheim, 1974.

45. C. L. Allyn, T. Gustafsson and E. W. Plummer, Chem. Phys. Letters 47, 127 (1977).

46. R. M. Stevens and M. Karplus, J. Chem. Phys. 49, 1094 (1968).

47. L. Pauling, Nature of the Chemical Bond, Cornell University Press, Ithaca, 1960.

48. H. Froitzheim, H. Hopster, H. Ibach and S. Lehwald, Appl. Phys. 13, 147 (1977).

49. R. A. Shigeishi and D. A. King, Surface Sci. 58, 379 (1976).

50. J. C. Tracy and P. W. Palmberg, J. Chem. Phys. 51, 4852 (1969).

51. T. L. Hill, J. Chem. Phys. 18, 246 (1950)

52. P. Kisliuk, J. Chem. Phys. 30, 174 (1959).

Figure Captions

Figure 1: Auger KLL transitions for CO, oxygen and carbon chemisorbed on clean Ir(110), (a) oxygen transitions, (b) carbon transitions.

Figure 2: LEED patterns of carbon on clean or oxidized Ir(110), (a) p(2x1) pattern at 60 eV, (b) (4x5) pattern at 60 eV.

Figure 3: LEED patterns and proposed structures for CO on clean Ir(110), (a) p(2x2) pattern and structure, (b) (4x2) pattern and structure.

Figure 4: (2x1)p1g1 LEED pattern at 102 eV and proposed structure for CO on oxidized Ir(110). The spots represented by dashed circles are missing for all voltages.

Figure 5: Thermal desorption spectra of CO on Ir(110), (a) clean surface, (b) oxidized surface. Both CO and CO_2 appear as desorption products from the oxidized surface.

Figure 6: Arrhenius plots for CO desorption from Ir(110) for various coverages, (a) clean surface, (b) oxidized surface.

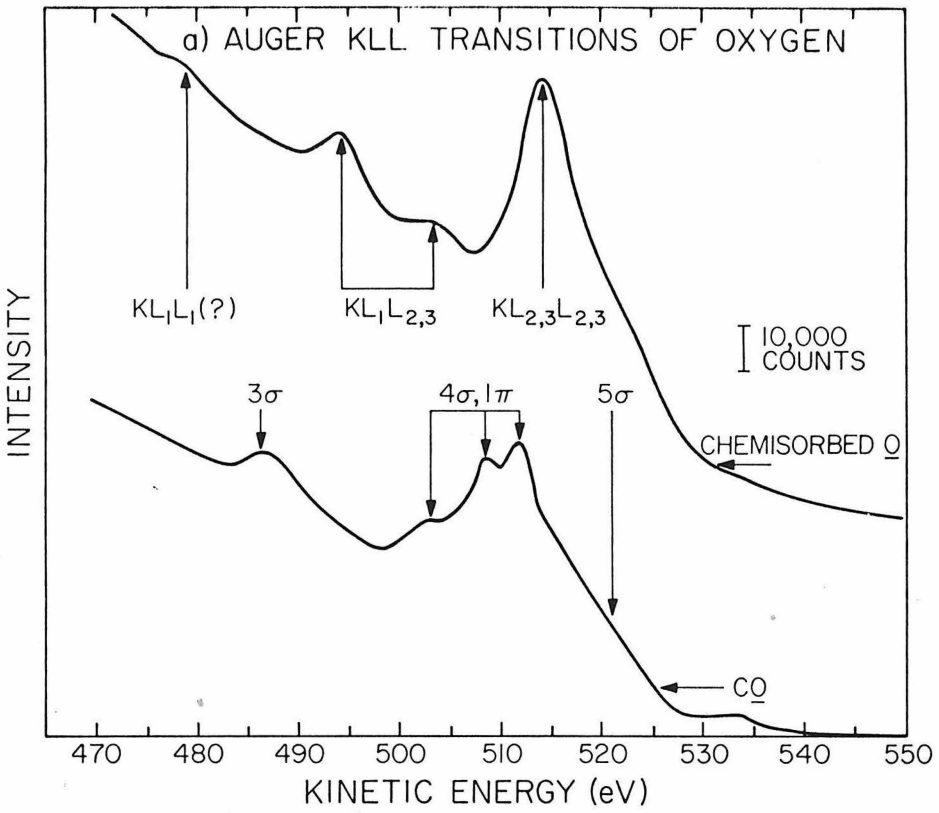
Figure 7: Variations in the desorption energy and the pre-exponential factor with surface coverage for CO on Ir(110), (a) clean surface, (b) oxidized surface.

Figure 8: Adsorption kinetics of CO on clean and oxidized Ir(110) at 300 K.

Figure 9: CPD as a function of CO exposure on Ir(110) at 90 and 300 K, (a) clean surface, (b) oxidized surface.

Figure 10: Topping plots for CO on Ir(110) at 90 and 300 K, (a) clean surface, (b) oxidized surface.

Figure 11: The coverage dependence of the isosteric heat, for CO on clean Ir(110), (---) true value, (•••) value obtained from the piecewise linear approximation (see text), (—) value obtained by evaluating the Clausius-Clapeyron equation at points of constant CPD.


a) AUGER KLL TRANSITIONS OF OXYGEN
INTENSITY
KL1L1(?)
KL1L2,3
KL2,3L2,3
10,000 COUNTS
3σ
4σ,1π
5σ
CHEMISORBED O
CO
470
480
490
500
510
520
530
540
550
KINETIC ENERGY (eV)

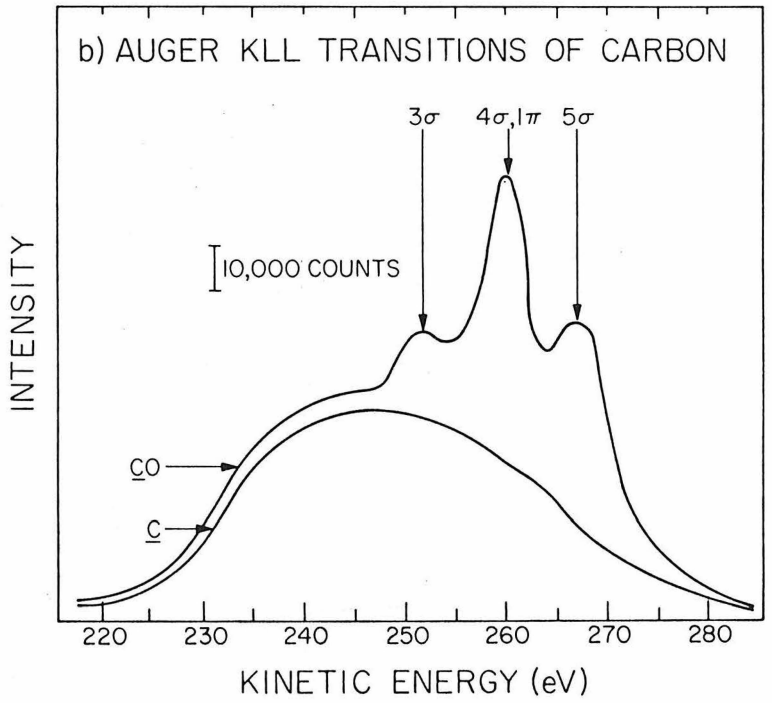

b) AUGER KLL TRANSITIONS OF CARBON
3σ
4σ,1π
5σ
INTENSITY
10,000 COUNTS
CO
C
220
230
240
250
260
270
280
KINETIC ENERGY (eV)

Fig. 1

a) C-(2×1) ON IR(110)

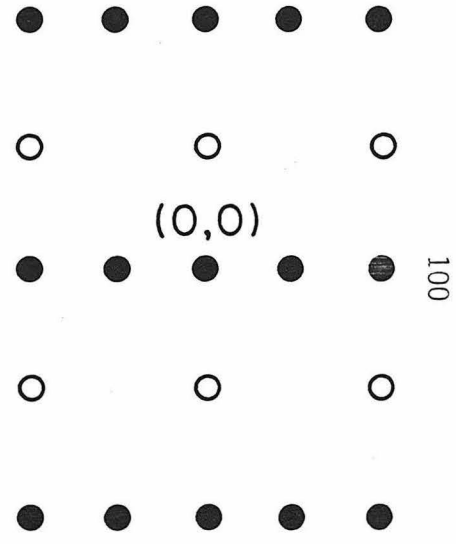
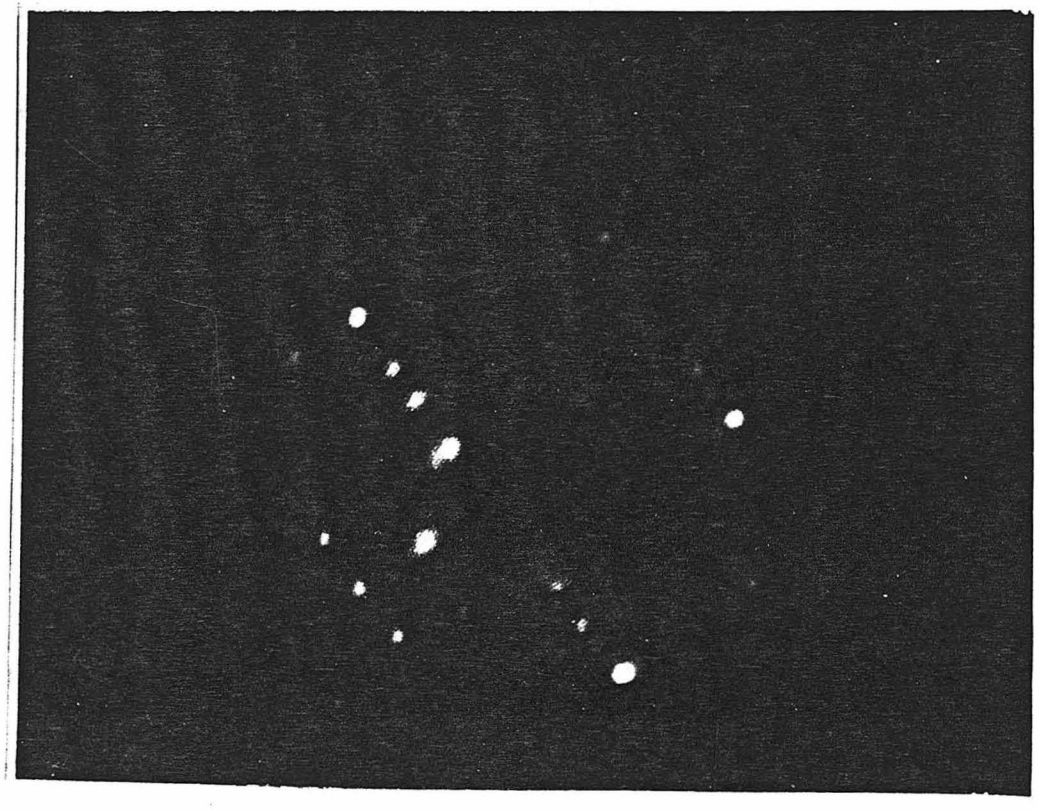


Fig. 2a

b) C-(4×5) ON IR(110)



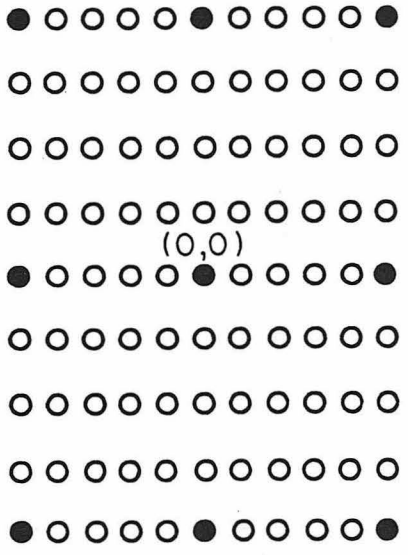
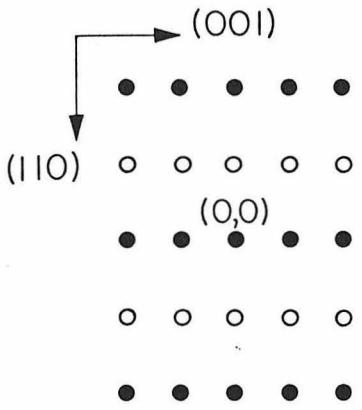
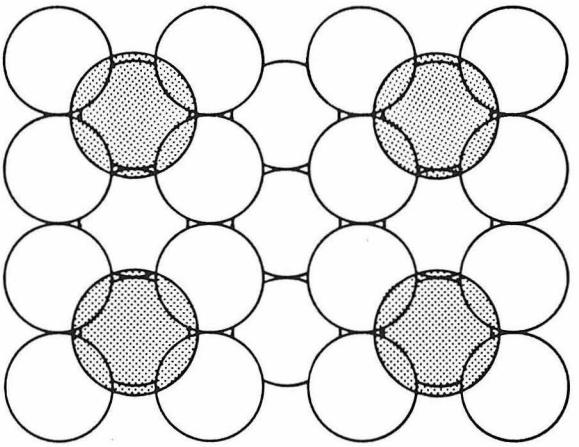


Fig. 2b

a) CO-p (2×2) ON IR(110)

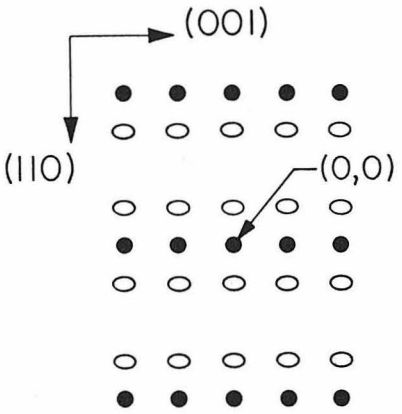


LEED PATTERN

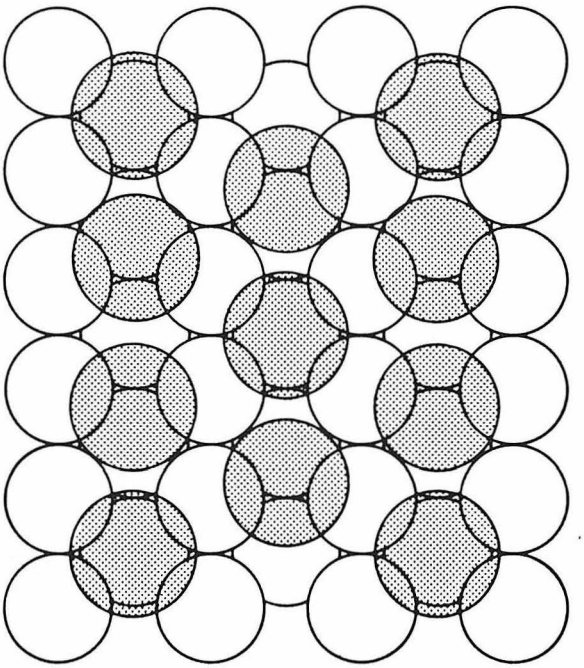


STRUCTURE

b) CO-(4×2) ON IR(110)



LEED PATTERN



STRUCTURE

Fig. 3

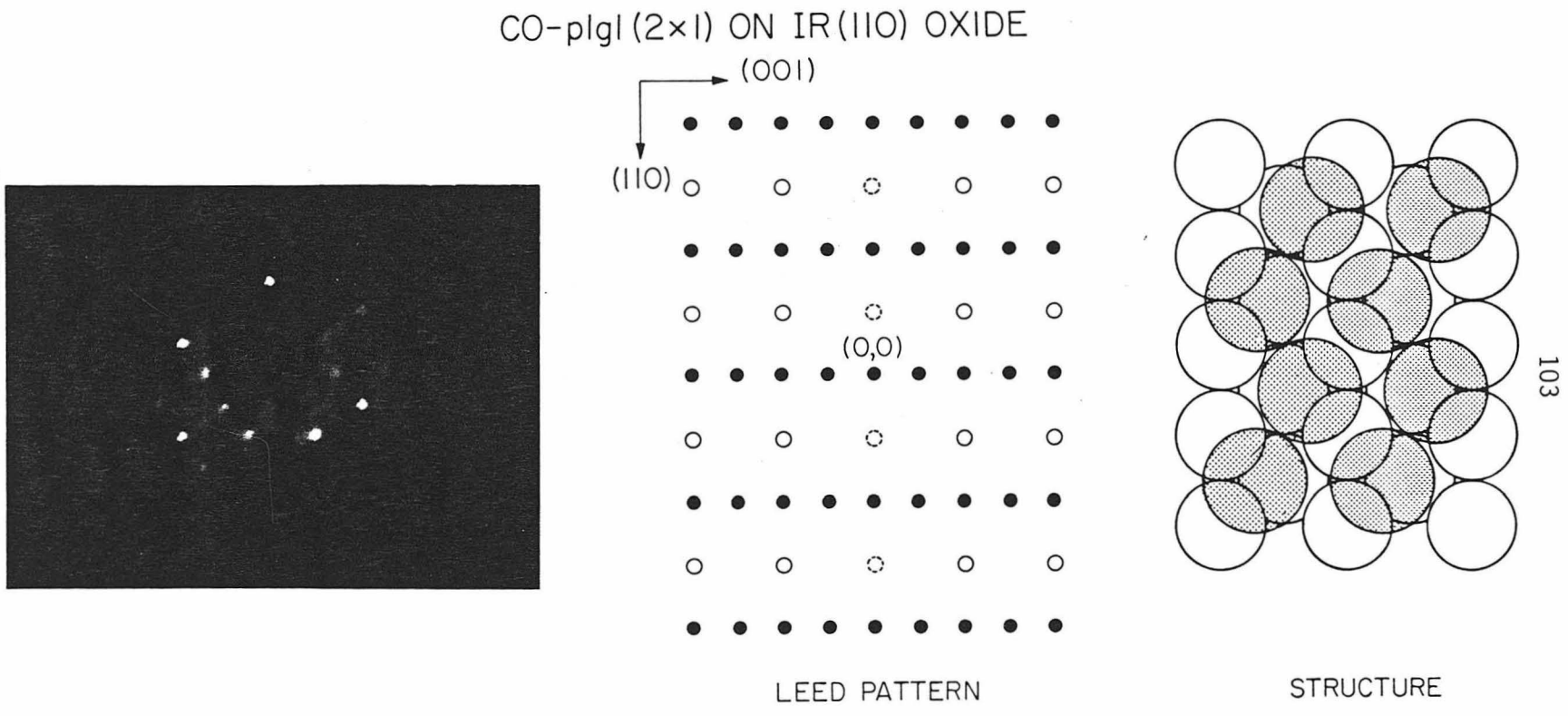


Fig. 4

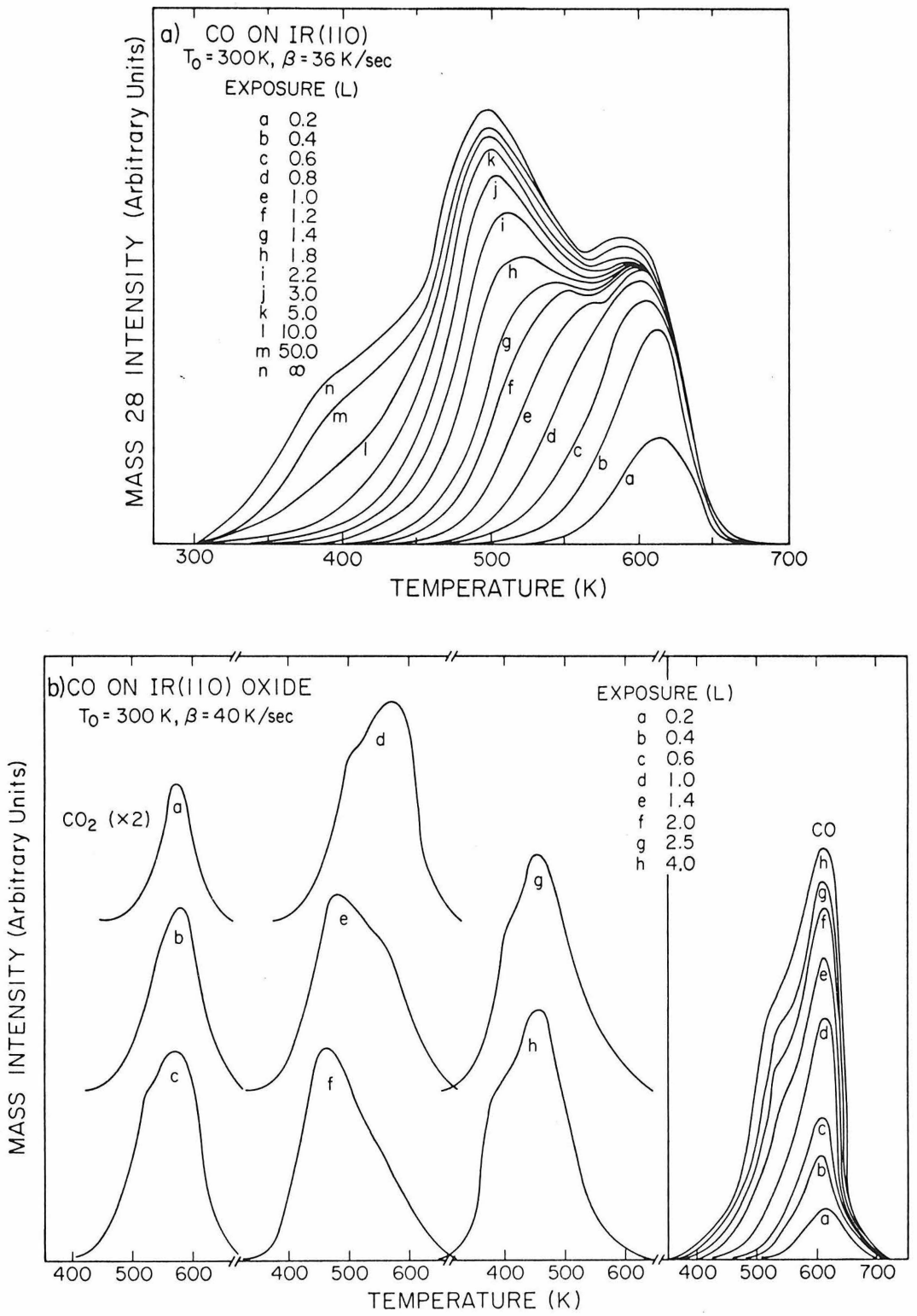

a) CO ON IR(110)
T0 = 300K, β = 36 K/sec
EXPOSURE (L)
a 0.2
b 0.4
c 0.6
d 0.8
e 1.0
f 1.2
g 1.4
h 1.8
i 2.2
j 3.0
k 5.0
l 10.0
m 50.0
n ∞
MASS 28 INTENSITY (Arbitrary Units)
TEMPERATURE (K)
b)CO ON IR(110) OXIDE
T0 = 300 K, β = 40 K/sec
EXPOSURE (L)
a 0.2
b 0.4
c 0.6
d 1.0
e 1.4
f 2.0
g 2.5
h 4.0
CO2 (×2)
CO
MASS INTENSITY (Arbitrary Units)
TEMPERATURE (K)

Fig. 5

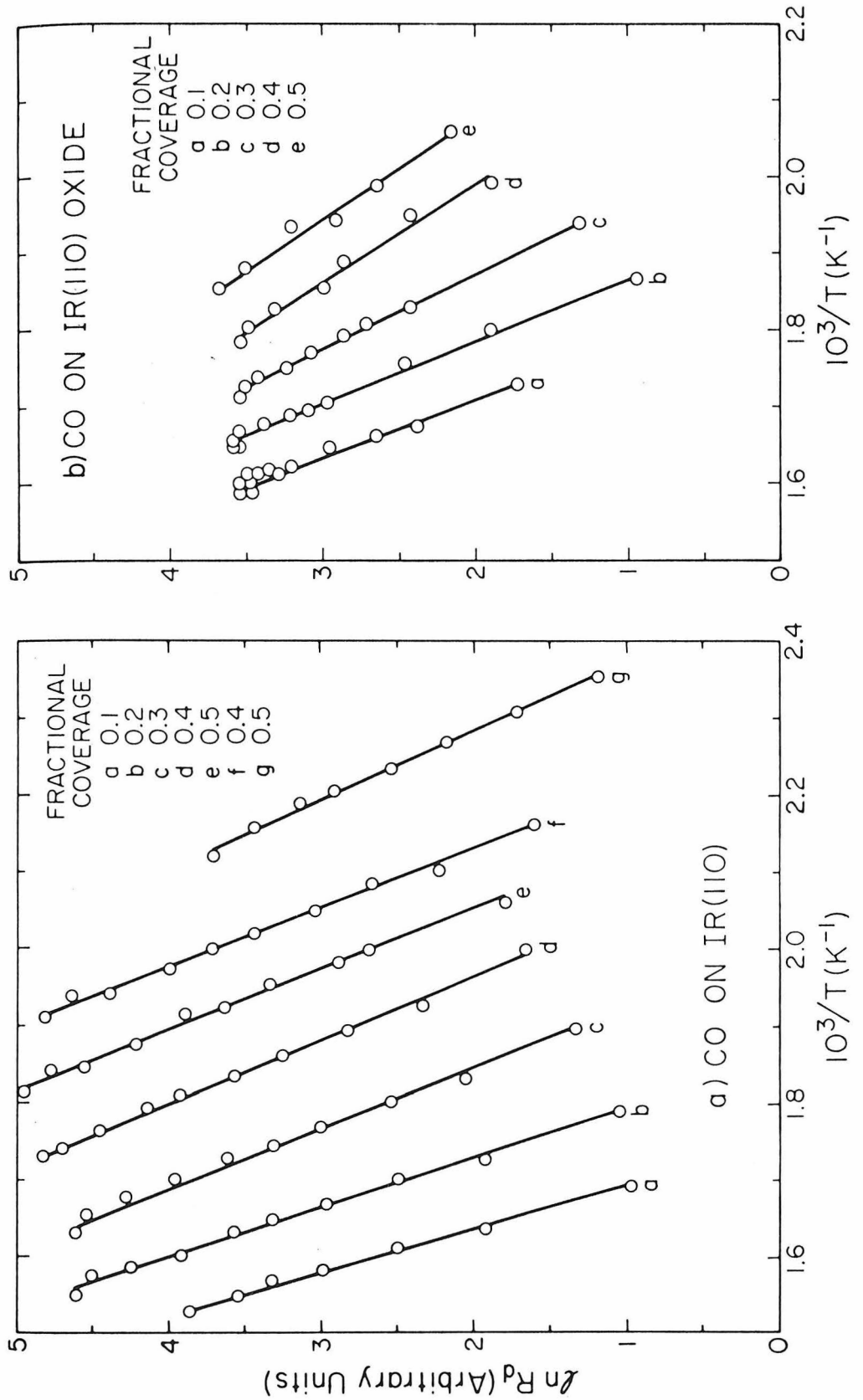

b) CO ON IR(110) OXIDE
FRACTIONAL COVERAGE
a 0.1
b 0.2
c 0.3
d 0.4
e 0.5
$10^3/T\,(K^{-1})$
a) CO ON IR(110)
FRACTIONAL COVERAGE
a 0.1
b 0.2
c 0.3
d 0.4
e 0.5
f 0.4
g 0.5
$\ell n\, R_d$ (Arbitrary Units)
$10^3/T\,(K^{-1})$

Fig. 6

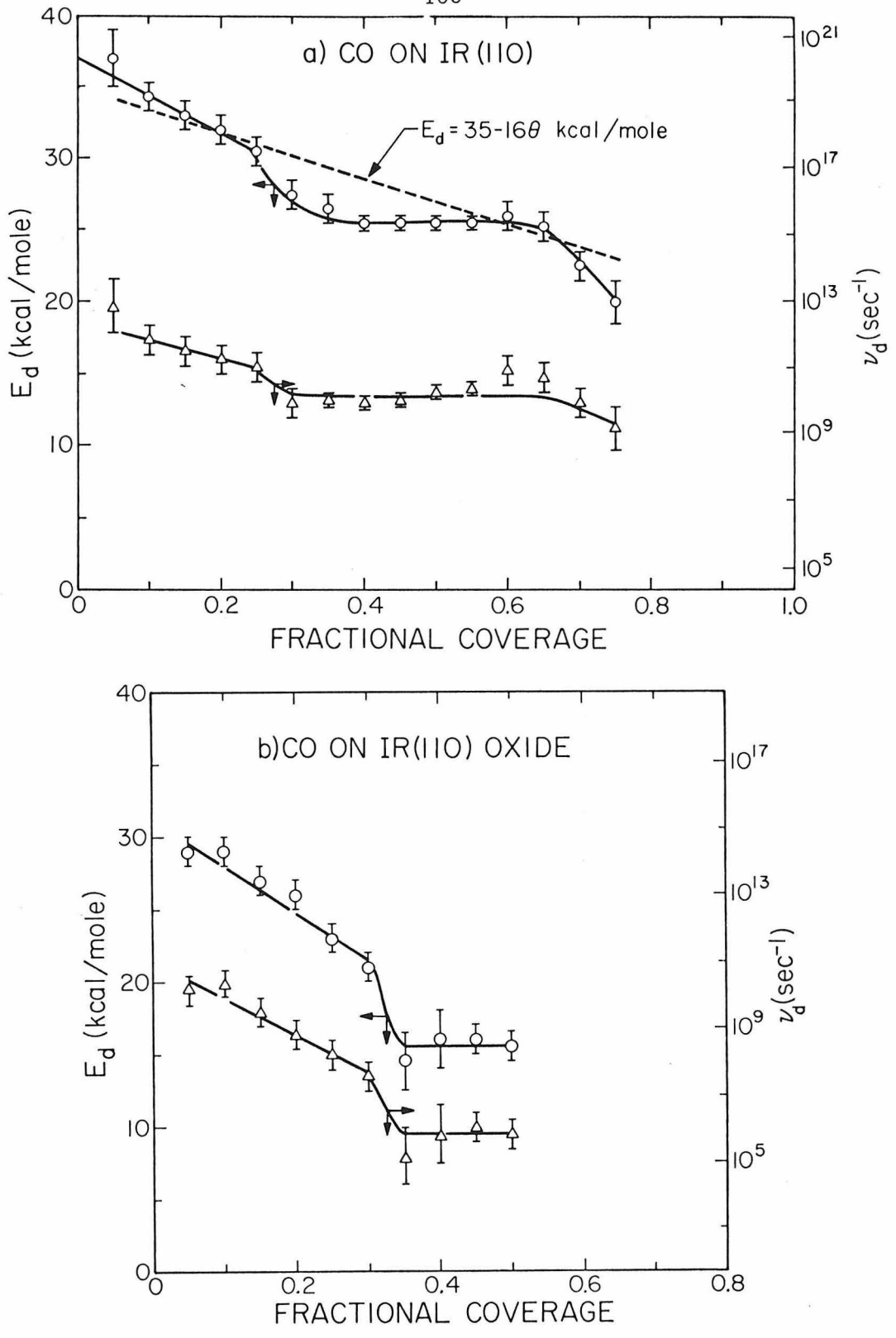

a) CO ON IR(110)
$E_d = 35-16\theta$ kcal/mole
E_d (kcal/mole)
ν_d (sec^{-1})
FRACTIONAL COVERAGE
b)CO ON IR(110) OXIDE
E_d (kcal/mole)
ν_d (sec^{-1})
FRACTIONAL COVERAGE

Fig. 7

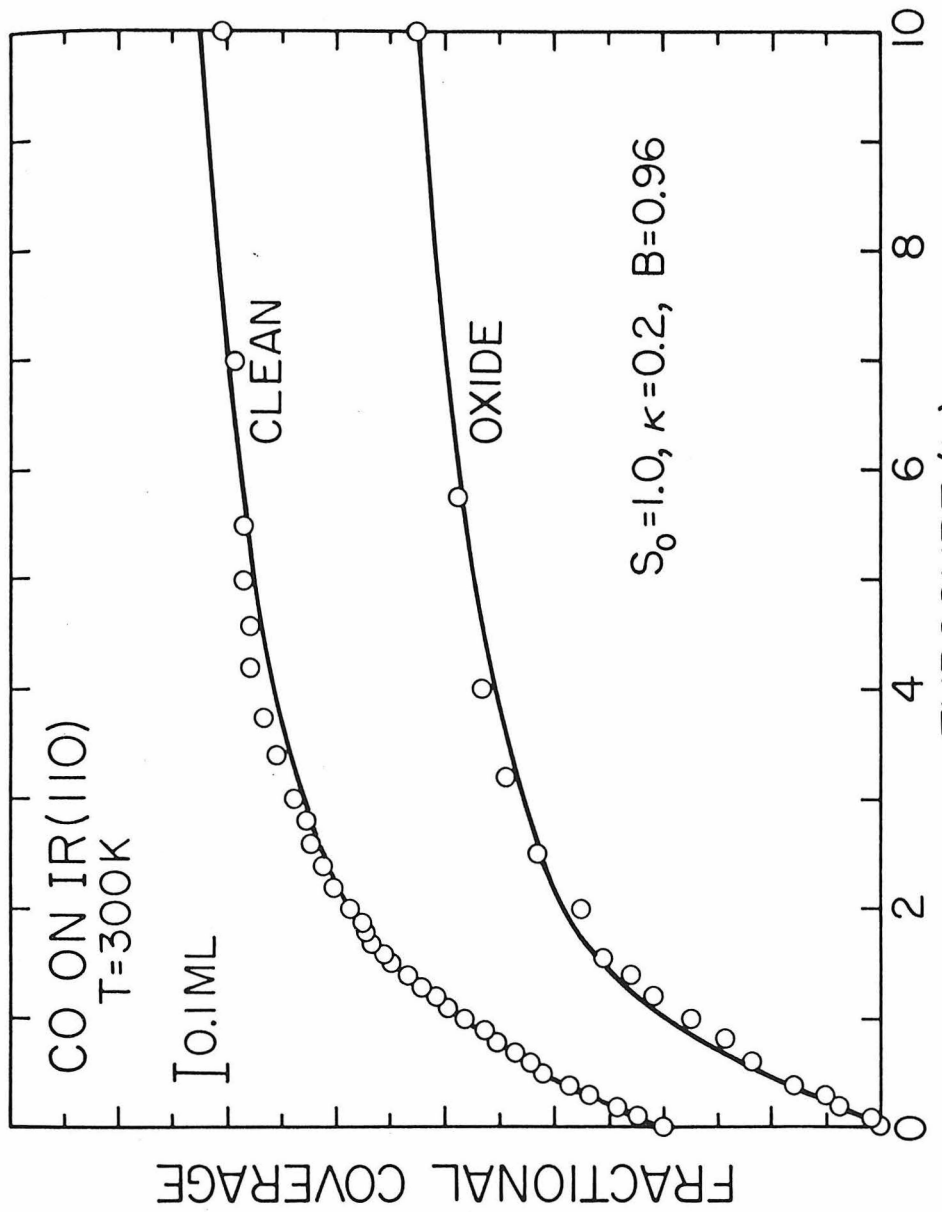

CO ON IR(110)
T=300K
0.1ML
CLEAN
OXIDE
S_o=1.0, κ=0.2, B=0.96
FRACTIONAL COVERAGE
EXPOSURE (L)
0
2
4
6
8
10

Fig. 8

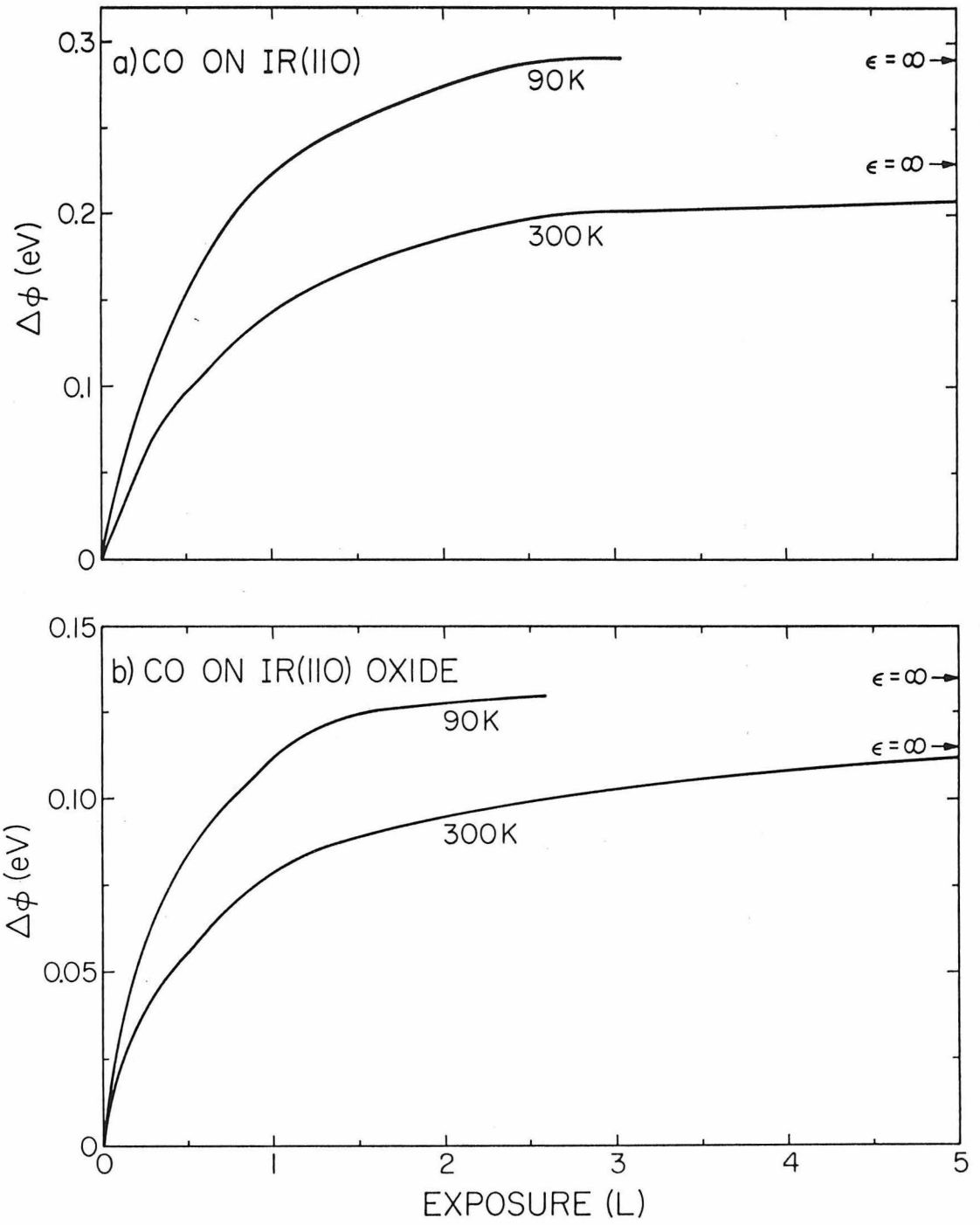

a) CO ON IR(110)
90K
300K
ε = ∞
ε = ∞
Δφ (eV)
0.3
0.2
0.1
0
b) CO ON IR(110) OXIDE
90K
300K
ε = ∞
ε = ∞
Δφ (eV)
0.15
0.10
0.05
0
0
1
2
3
4
5
EXPOSURE (L)

Fig. 9

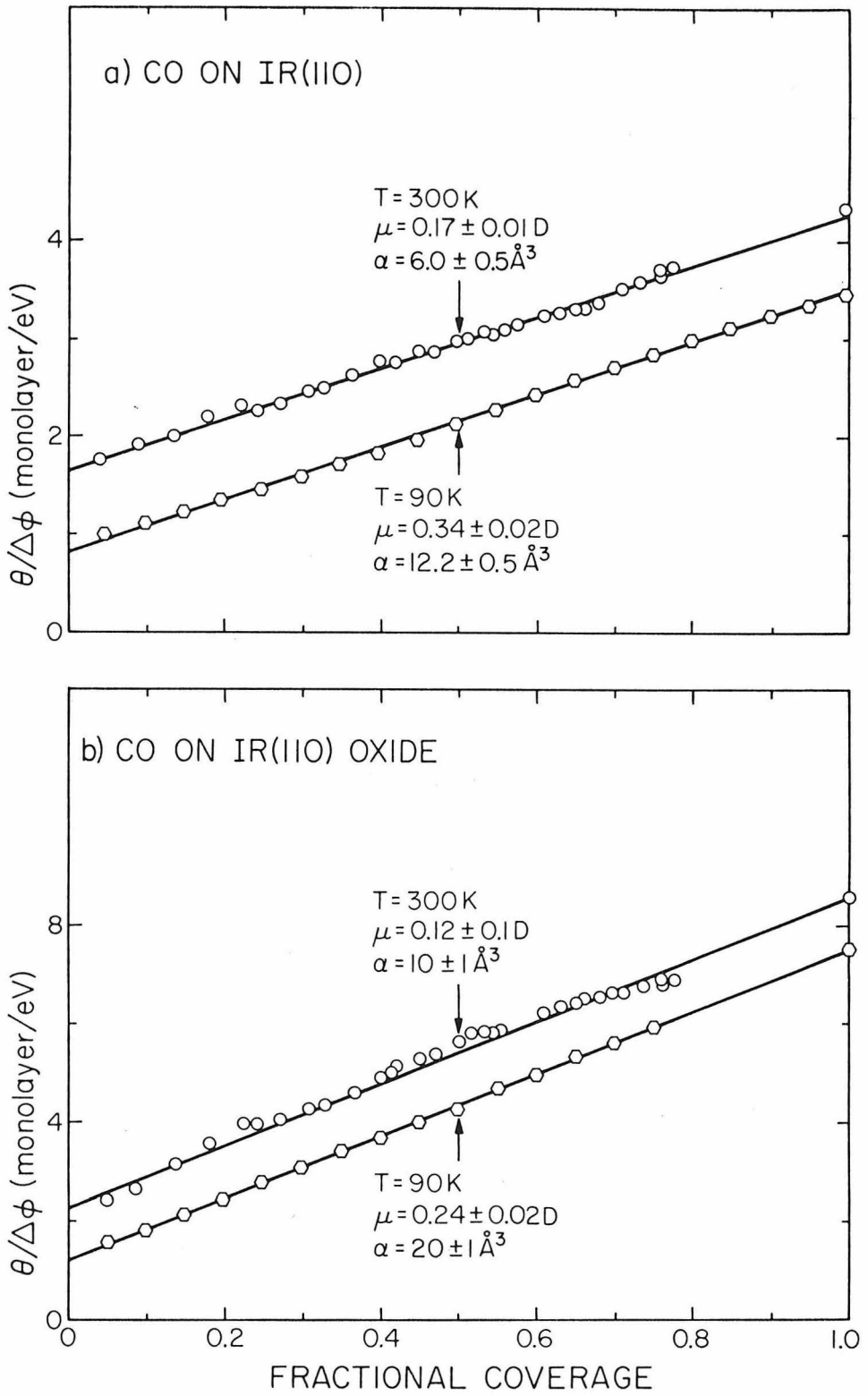

a) CO ON IR(110)
T = 300 K
μ = 0.17 ± 0.01 D
α = 6.0 ± 0.5 Å³
T = 90 K
μ = 0.34 ± 0.02 D
α = 12.2 ± 0.5 Å³
θ/Δφ (monolayer/eV)
b) CO ON IR(110) OXIDE
T = 300 K
μ = 0.12 ± 0.1 D
α = 10 ± 1 Å³
T = 90 K
μ = 0.24 ± 0.02 D
α = 20 ± 1 Å³
FRACTIONAL COVERAGE

Fig. 10

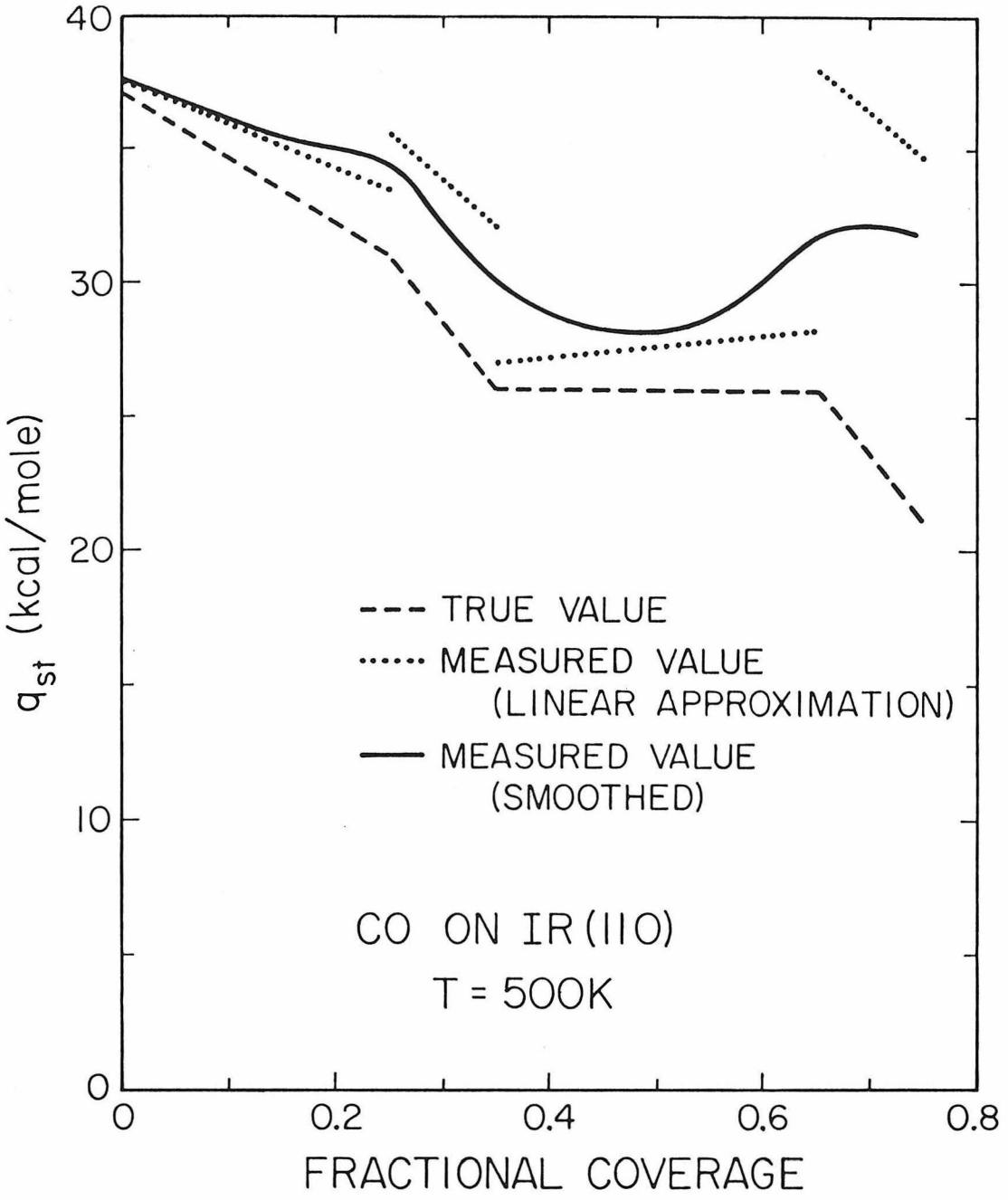

40
30
20
10
0
0
0.2
0.4
0.6
0.8
q_{st} (kcal/mole)
FRACTIONAL COVERAGE
TRUE VALUE
MEASURED VALUE
(LINEAR APPROXIMATION)
MEASURED VALUE
(SMOOTHED)
CO ON IR(110)
T = 500K

Fig. 11

CHAPTER 4

THE CHEMISORPTION OF OXYGEN ON THE (110) SURFACE OF IRIDIUM

(The text of Chapter 4 consists of an article coauthored with D. E. Ibbotson and W. H. Weinberg that has been submitted to Surface Science for publication.)

ABSTRACT

The chemisorption of oxygen on Ir(110) has been investigated under ultra-high vacuum conditions with thermal desorption mass spectrometry, contact potential difference measurements, Auger electron spectroscopy, and LEED. Oxygen may adsorb in three distinct chemical states: a molecularly chemisorbed species that is stable below 100 K, a dissociatively chemisorbed species, and a surface oxide that forms rapidly above 700 K. The oxide layer saturates at a coverage of 0.25 ML (1 ML = 9.6×10^{14} atoms/cm^2) and orders to form a (1x4) LEED pattern. Different LEED patterns of dissociatively chemisorbed oxygen are observed on clean and oxidized Ir(110). A p(2x2) pattern forms on the clean surface near 0.25 ML coverage whereas a c(2x2) pattern forms on the oxidized surface near 0.5 ML coverage. Oxygen desorbs molecularly from Ir(110) with an activation energy of 45 - 70 kcal/mole, decreasing continuously with increasing coverage. The adsorption kinetics are described by a second-order precursor model for surface temperatures between 300 - 700 K. Oxygen chemisorption is not activated since the initial sticking probabilities on clean and oxidized Ir(110) equal 0.28 and 0.4, irrespective of the surface temperature. The dipole moment and polarizability of dissociatively chemisorbed oxygen change at 0.25 ML coverage on the clean surface and at 0.5 ML coverage on the oxidized surface. Although the dipole moment for any coverage is independent of temperature, the polarizability is inversely proportional to temperature. The activation energy for the dissociation of molecularly chemisorbed oxygen is 8 kcal/mole.

I. Introduction

This study of oxygen chemisorption was undertaken to give insight into the catalytic behavior of the CO oxidation reaction on Ir(110). Surface sensitive probes--thermal desorption mass spectrometry (TDS), contact potential difference (CPD) measurements, LEED, Auger electron spectroscopy (AES) and X-ray photoelectron spectroscopy (XPS)--were used for studying oxygen chemisorption under ultra-high vacuum (UHV) conditions. Since the understanding of catalytic phenomena was the impetus for this study, the characterization of the kinetic properties and the chemical states of adsorbed oxygen was emphasized. This study reveals that oxygen may adsorb in three distinct chemical states on Ir(110): a dissociatively chemisorbed species, a surface oxide and a molecularly chemisorbed species. The latter two species are described in separate sections of this article; the dissociatively chemisorbed species is described in four sections, which focus on overlayer structures, desorption kinetics, adsorption kinetics and charge transfer in the overlayer. In each section, a discussion parallels the presentation of experimental results.

A surface oxide, which modifies several aspects of oxygen chemisorption, forms on Ir(110). Oxide formation on the platinum metals, i.e., the incorporation of oxygen into the surface lattice, was reported first for Rh(110) (1) and subsequently has been observed on other platinum metals (2 - 8). However, some ambiguity exists in the definition of a surface oxide. A surface oxide is defined here as a second dissociatively chemisorbed state of oxygen, which forms irreversibly through thermal activation and has different physical properties than the first state of

dissociatively chemisorbed oxygen. Moreover, the formation of the oxide layer may cause the total saturation coverage of oxygen to increase (4, 5, 7). The oxide atoms presumably bind in the subsurface region, but this region is difficult to define for rough surface orientations, such as the (110) face of the fcc metals. Since the oxide may be incorporated into the surface lattice, oxides are not necessarily inert to reaction with other adsorbates. For example, oxides formed on the (111) surfaces of Ni (3), Pd (7) and Ir (5) are inert to reduction by CO for pressures below 10^{-6} torr and surface temperatures below 1000 K, whereas oxides formed on Rh(110) (1) and polycrystalline Pt (8) are reactive. Previous studies of oxide formation on Ni (3), Rh (1), Pd (7), Pt (2, 4, 6, 8) and Ir (5, 9) have focused on characterizing the oxide layer, but few studies have described how the oxide layer modifies oxygen chemisorption with respect to the clean surface. Both subjects are treated in this work.

Several aspects of oxygen chemisorption are similar among the platinum metals. Oxygen generally chemisorbs dissociatively for surface temperatures above 300 K, but adsorbs molecularly on polycrystalline Pt near 100 K (10). Oxygen contamination on the (110) surfaces of Ni (11), Rh (1), Pd (12), Pt (13) and Ir (14) introduces streaks between the substrate spots in the [001] azimuth of the LEED pattern. If the contamination is sufficiently large, the streaks coalesce upon annealing to form (1x2) patterns on Rh (1) and Pd (12), and (1x3) patterns on Rh (1), Pd (12) and Pt (4). Several overlayer structures of dissociatively chemisorbed oxygen--including the c(2x2) pattern on Rh (1), Pt (4) and Ir (14), and the c(2x4) pattern on Rh (1) and Pd (12)--are common among

the (110) surfaces of the platinum metals. Depending on the metal, the surface orientation and the coverage, the activation energy for oxygen desorption from the platinum metals is between 36 - 80 kcal/mole (12, 13). The initial sticking probability of oxygen is reported generally to be between 0.1 and 1.0 (3 - 5, 7, 8, 13, 15, 16). The adsorption kinetics usually are described by a dissociative Langmuir model or a second-order precursor model, and are independent of the surface temperature below 700 K (3, 4, 7, 8, 13, 15, 16). Although the CPD has not been reported widely for oxygen chemisorption on the platinum metals, a curious phenomenon is observed on Ni(100) and Ni(111) (3). The CPD changes sign at a coverage associated with the completion of a LEED pattern. Although the results of this study are generally in concurrence with those described previously, several new findings are reported.

To avoid confusion, each of the three chemical states of adsorbed oxygen will be denoted differently in this work. The molecularly chemisorbed species will be termed molecular oxygen. Chemisorbed oxygen will refer to the dissociatively chemisorbed species, and the oxide phase will be designated as the oxide or the oxide layer. Oxidized Ir(110) will denote an Ir(110) surface covered with a saturated oxide layer. Irrespective of the chemical state of oxygen, all coverages will be reported on an atomic basis where 1 ML equals 9.6×10^{14} atoms/cm^2.

II. Experimental

The experiments were performed in a stainless steel belljar with a base pressure below 2×10^{-10} torr. Several surface sensitive probes--a quadrupole mass spectrometer, LEED optics, a CPD apparatus and Auger

electron and X-ray photoelectron spectrometers--were contained in the experimental chamber. CPD measurements were obtained by a continuous retarding potential method (17). A double-pass cylindrical mirror analyzer (CMA) performed the energy analysis for the electron spectroscopies. All probes were interfaced to a PDP 11/10 computer (17). Although all data reported here, except the LEED data, were obtained in digital form, the data are shown in analog form to facilitate the presentation. A more detailed description of the experimental system is reported elsewhere (18).

The substrate was cut from a single crystal of Ir and was polished within 1° of the (110) orientation using standard polishing techniques. During the course of this investigation, the substrate was repolished several times, and the results of many experiments were reproduced after repolishing. Carbon was cleaned from the substrate through a series of oxidation and reduction cycles. Annealing above 1600 K was necessary to remove O, Ca and K impurities from the substrate. Prior to each experiment, the surface was free from impurities within 0.01 ML as determined by Auger spectroscopy and CPD measurements. The substrate was heated conductively by passing current through two 10 mil Ta wires, on which the crystal was mounted. Temperatures in the range between 200 and 1800 K were measured with a Pt-Pt10%Rh thermocouple that was spot-welded to the Ir crystal. Lower temperatures were measured by noting the change in the resistance of the Ta wires. All surface temperatures reported in this work are accurate to ± 5 K, except when noted otherwise.

When thermal desorption was monitored with the surface in the line-of-sight of the mass spectrometer, no iridium oxides--IrO, IrO_2 or IrO_3--

appeared as desorption products. Also, from a line-of-sight experiment, oxygen desorption was determined to be molecular. The ratio of the mass 16 to mass 32 signals for any surface temperature was 0.12, which exactly equaled the measured ratio for gaseous oxygen. Moreover, atomic desorption is not expected on a thermodynamic basis (19). Following the completion of a thermal desorption spectrum, no oxygen was detected on the surface by Auger spectorscopy or CPD measurements.

The measured heating schedule during desorption was exponential, rather than linear, because the sample was heated conductively by passing a constant current through the Ta support wires that held the Ir(110) surface (20). In the temperature regime where oxygen desorbed, the heating rate (β) varied between 80 and 300 K/sec. Since the intensity of the mass 32 signal and the surface temperatures were measured as independent functions of time, this variation introduced no artifacts when the data were analyzed. Moreover, the pumping speed was sufficiently large that these large heating rates did not distort the spectra (21). When the substrate received large exposures to oxygen, the gain of the electron multiplier in the mass spectrometer, which was quite sensitive to oxygen contamination, drifted by as much as a factor of four. The gain was calibrated repeatedly by comparing the total intensities for spectra obtained under the same conditions but at different times.

Absolute coverages of adsorbed oxygen were determined by comparing the mass 28 and 32 signals of the thermal desorption spectra of CO and oxygen for saturation coverages. Since oxygen desorbs molecularly, the mass 16 signal for oxygen was not monitored. The absolute sensitivity

of the mass spectrometer to CO was determined from the CO spectrum for saturation coverage, which is 9.6×10^{14} molecules/cm^2 on Ir(110) (18). The relative sensitivities for CO and O_2 were determined by calibrating the mass spectrometer signals against an ion gauge where the ionization efficiencies for the two species were known. The experimental chambers contained a poppet valve, which was closed sufficiently that the ratio of the pumping speeds for O_2 and CO equaled the square root of the ratio of their molecular weights (22, 23). Then absolute coverages of oxygen were determined from twice the product of the absolute sensitivity of the mass spectrometer to CO, the ratio of the relative sensitivities for CO and O_2, the ratio of the pumping speeds for O_2 and CO, and the ratio of the pressure-time integrals for the O_2 and CO spectra. The factor of two is included above because the coverage of CO is given in molecules/cm^2, whereas the coverage of O_2 is given in atoms/cm^2.

III. The Oxide Layer

At 300 K, the saturation coverage of chemisorbed oxygen on clean Ir(110) was measured as 1.0 ± 0.1 ML by TDS; but the total coverage of adsorbed oxygen could be increased to 1.25 ± 0.1 ML as follows. First, a surface covered with more than 0.25 ML of oxygen was annealed between 700 and 1200 K in an oxygen atmosphere. When the surface was placed in an oxygen atmosphere again, the saturation coverage was 1.25 ML at 300 K. The saturation coverage was increased by no more than 0.25 ML if the surface temperature and the oxygen pressure were maintained below 1200 K and 10^{-6} torr. The additional 0.25 ML of oxygen coverage, which is obtained through thermal activation, is associated with the oxide layer.

The formation of this surface oxide is particularly important in studying the CO oxidation reaction because the rates of CO_2 production are different for clean and oxidized Ir(110) (22). The saturation coverage of oxygen on Rh(110) has been reported as 1.25 ML (1). Although annealing in an oxygen atmosphere to 600 K was necessary to obtain this coverage, it has not associated with an oxide layer.

A clean Ir(110) surface is reconstructed into a (1x2) structure (14); extended annealing of the oxide layer reconstructs the surface further to a (1x4) structure. When CO or oxygen is chemisorbed on the oxidized surface, the (1x4) pattern, shown in Fig. 1, is transformed into a (1x2) pattern. Hence, the (1x4) pattern is not associated with a coincidence lattice of oxygen, but with a composite structure involving both oxygen and Ir. The (1x4) pattern was observed after a surface covered with more than 0.25 ML of oxygen was annealed at 1200 K in vacuum for several minutes. Alternatively, a clean surface was annealed to 1200 K in an oxygen atmosphere for several minutes, and the oxygen was evacuated prior to cooling. When the (1x4) pattern was formed in either of these ways, the coverage of adsorbed oxygen was 0.25 ML, as determined by TDS. If the initial oxygen coverage exceeded 0.25 ML, the excess oxygen desorbed during annealing. The (1x4) pattern did not form for coverages below 0.20 ML. Since both the oxide layer and the (1x4) pattern correspond to coverages of 0.25 ML and are formed through thermal activation, the (1x4) structure is associated with the oxide layer.

One possible structure for the (1x4) pattern, which is consistent with an oxide coverage of 0.25 ML, is shown in Fig. 1 where the oxide atoms are shaded. If the rows of surface atoms move in the directions

indicated by the arrows, further oxidation beyond 0.25 ML coverage may be inhibited by the pairwise approach of the surface atoms. Moreover, this movement coincides with the notion that the (1x4) pattern is formed through a reconstruction of the surface. The (1x4) pattern orders at elevated temperatures like the (1x2) patterns of oxygen on the (110) surfaces of Rh (1) and Pd (12), and the (1x3) patterns on Rh (1), Pd (12) and Pt (4). Hence, these structures may be associated with oxide layers also. When Pd(110) is oxidized, a similar pairwise approach of the rows of surface atoms may occur. At elevated temperatures, no oxygen dissolves into the bulk of Pd(110) whereas dissolution is rapid on Pd(111) (7).

The oxide layer may be either ordered or amorphous. Both forms have identical chemical properties. The amorphous oxide was observed when a surface covered with 0.25 ML of chemisorbed oxygen was annealed rapidly between 800 and 1200 K. Upon cooling, the LEED pattern had streaks between the substrate beams in the [001] azimuth. Also, the half-order beams of the substrate were reduced in intensity and were quite diffuse. Similar streaks, associated with oxygen contamination, have been observed on the (110) surfaces of Ni (11), Rh (1), Pd (12), Pt (4) and Ir (14).

The activation energy for forming the oxide layer was estimated from a coadsorption experiment involving CO. First, a clean surface at 300 K was exposed to 2.7 L (1 L = 10^{-6} torr-sec) of oxygen, which corresponds to an oxygen coverage of 0.5 ML (see Section VI). After the surface was annealed at fixed temperature for one minute, the surface was exposed to 0.6 L CO at 300 K, which corresponds to a CO coverage of 0.25 ML (22). Then the thermal desorption spectra for CO and CO_2 were obtained with an average heating rate (β) of 40 K/sec.

In Fig. 2, the CO_2 spectra are shown as a function of the annealing temperature. For temperatures above 800 K, the spectra are identical to the spectrum for 800 K. The CO spectra give a single peak near 600 K; only the intensity of the peak changes with the annealing temperature. A second state in the CO_2 spectrum near 600 K, which is associated with the growth of the oxide layer, appears for annealing temperatures above 600 K. When the second state appears, the amount of CO_2 produced decreases substantially. For annealing temperatures near 700 K, the oxide layer forms almost completely, as evidenced by the CO_2 spectra, and grows at a rate of approximately 0.25 ML/min. Assuming the formation kinetics are described by a first-order Arrhenius expression with a pre-exponential factor of 10^{13} sec^{-1}, the energy of formation for the oxide layer is 45 kcal/mole. The energies of formation for the oxide layers on polycrystalline Pt (8) and Ir(111) (5) are similar since these oxides form rapidly near 700 K.

In contrast to the oxide layer on the (111) surfaces of Ni (3), Pd (7) and Ir (5), the oxide formed on Ir(110) can be reduced by CO. Oxides formed on Rh(110) (1) and polycrystalline Pt (8) can be reduced also. A second type of oxide, which is unreactive to CO reduction, may form on Ir(110). This is reasonable, for two types of oxide layers have been observed on Ir(111) (9). On Ir(110), the unreactive oxide was formed by annealing the surface above 1400 K in an oxygen atmosphere and was desorbed by heating the surface to 1800 K in vacuum for extended times. Because this oxide does not form in the temperature regime from 400 to 900 K where the CO oxidation reaction is rapid (22), it was not characterized more extensively. Since the LEED data show no evidence of a

rutile structure, neither type of oxide on Ir(110) is associated with the most stable Ir oxide, IrO_2.

The Auger KLL transitions are different for chemisorbed oxygen and the oxide, indicating that these two forms of surface oxygen represent truly different chemical states. XPS was not useful for characterizing the different chemical states because the 1s transitions for each species differed neither in shape nor in energy. Similar behavior for chemisorbed oxygen and the oxide was observed in the X-ray spectra for Ni(110), but not Ni(111) or Ni(100) (6). The Auger KLL spectra of chemisorbed oxygen and the oxide layer on Ir(110), shown in Fig. 3, were recorded for 0.25 ML coverage at 300 K. The oxide layer was formed when a surface covered with 0.25 ML of chemisorbed oxygen was annealed between 700 and 1200 K. When a surface covered with chemisorbed oxygen was annealed between 300 and 500 K, the Auger spectrum was not changed. For the spectra in Fig. 3, the incident current density was 10^{-6} A/cm^2 at 2 kV. In obtaining each spectrum, several individual spectra were summed to attain the desired resolution and to minimize the effects of background contaminants. A linear background has been subtracted from each spectrum to accentuate the oxygen transitions.

The spectrum of oxygen chemisorbed on Ir(110) at 300 K is typical of dissociatively chemisorbed oxygen (24). As noted in Fig. 3, Fuggle et al. (24) have associated transitions to peaks in that spectrum. The spectrum of the oxide layer and chemisorbed oxygen differ in three respects: (1) the slope of the inelastic background, which has been subtracted from the spectra in Fig. 3, is larger for the oxide; (2) the peaks are broader in the oxide spectrum; and (3) the $KL_{2,3}L_{2,3}$ peak for

the oxide is shifted by more than 2 eV with respect to chemisorbed oxygen. The shift is much larger than the resolution of the analyzer, 0.5 eV, and cannot be attributed solely to differences in the CPD for the two overlayers (see Section VII). The first two observations indicate that the oxide atoms bind more deeply in the Ir lattice than chemisorbed oxygen. For a more deeply bound species, inelastic electron scattering should be more pronounced, and the peaks should be more band-like and broadened (3, 6). Hence, the positions for the oxide atoms in Fig. 1 are reasonable. Derivative spectra for chemisorbed oxygen and an oxide layer on polycrystalline Pt (8) are similar in shape and energy to those in Fig. 3.

Since the oxide layer on Ir(110) is a second state of dissociatively chemisorbed oxygen that is formed irreversibly by thermal activation, it is truly a surface oxide according to the definition given in Section I. In the following sections, the term oxide or oxidized surface will refer to an Ir(110) surface with a saturated oxide layer (0.25 ML coverage). Furthermore, the distinction between an ordered oxide, which gives a (1x4) LEED pattern, and an amorphous oxide, which gives a (1x2) LEED pattern with streaks in the [001] azimuth, is unnecessary since both forms have identical chemical properties otherwise. In this work, the oxide layer was made routinely by adsorbing 0.25 ML of oxygen on clean Ir(110) and then annealing the surface to 1100 K for one minute.

IV. Overlayer Structures

Different overlayer structures of oxygen were observed on clean and oxidized Ir(110). The p(2x2) pattern, reported previously for Ir(110) (14), but no other platinum metals, appeared sharpest for 0.25 ML coverage

on clean Ir(110). The coverage was determined by TDS (see Section II). A photograph of the p(2x2) pattern is shown in Fig. 4. The overlayer beams were sharpened by slowly cooling the substrate below 300 K. When clean Ir(110) at 300 K was exposed to oxygen initially, the overlayer structure was disordered in the [001] azimuth. The disorder disappeared after a few minutes, indicating that a substantial activation barrier to diffusion exists in the [001] azimuth. A similar slow ordering of the oxygen (2x2) pattern on Ir(111) has been observed near room temperature (5). In that work, an activation energy for diffusion of 17 ± 2 kcal/mole was calculated by assuming that the kinetics for diffusion could be described by a zero-order Arrhenius expression with a pre-exponential factor of 10^{13} sec^{-1}. Subsequently, this value for the pre-exponential factor has been measured directly for oxygen diffusion on Rh(111) (25). Using a pre-exponential factor of 10^{13} sec^{-1} and a half-life of one minute for ordering the overlayer at 300 K, an activation energy of 20 kcal/mole is estimated for diffusion in the [001] azimuth of clean Ir(110). A possible structure for the p(2x2) pattern, which is consistent with 0.25 ML coverage, is shown in Fig. 4. As CPD measurements indicate (see Section VII), oxygen binds above the topmost rows of surface atoms at this coverage so that the registry of the overlayer that is shown in Fig. 4 is reasonable.

Two oxygen patterns, a c(2x2) and a (3x2) disordered in the [001] azimuth, were observed on oxidized Ir(110). These patterns are shown in Fig. 5. When oxygen was chemisorbed on an ordered oxide layer, the quarter order beams of the substrate disappeared entirely. Similarly, when oxygen was chemisorbed on an amorphous oxide layer, the streaks

between the substrate beams in the [001] azimuth vanished. Hence, the chemisorbed layer reorders the substrate.

For oxygen coverages between 0.2 and 0.6 ML at 300 K, a streak appeared in the [001] azimuth between the substrate spots. After the overlayer was annealed to 700 K for a few minutes, the streak coalesced to form the c(2x2) pattern. The activation energy for the diffusion of oxygen in the [001] azimuth may be calculated as was done for the clean surface. Assuming a pre-exponential factor of 10^{13} sec^{-1} and a half-life of 1 min for ordering the overlayer at 700 K, the diffusion energy is 45 kcal/mole on the oxidized surface. The oxygen c(2x2) structure has been reported previously on the (110) surfaces of Rh (1), Pt (4) and Ir (14). In all cases, annealing between 700 and 800 K was necessary for ordering this structure so the c(2x2) pattern may be associated with chemisorbed oxygen on oxidized surfaces of these metals. The c(2x2) pattern on Ir(110) appeared sharpest near 0.5 ML coverage. A possible structure for this pattern, consistent with this coverage, is shown in Fig. 5. Once again, CPD measurements (q.v.) indicate that chemisorbed oxygen binds above the topmost rows of surface atoms at this coverage.

The partially disordered (3x2) pattern was observed on oxidized Ir(110) after a clean or oxidized surface was annealed near 1000 K in an oxygen atmosphere. The pattern appeared sharper for lower surface temperatures. Certain overlayer beams, indexed $(\frac{1}{3}\ h,\ 0)$ where h is an integer, are missing from the LEED pattern, indicating that the overlayer is symmetric about a glide line running parallel to the [110] azimuth. Because of the disorder in the [001] azimuth, any structure suggested for this pattern is subject to question, but a structure for a (3x2) pattern

with glide line symmetry has been proposed elsewhere (1). Except for the (3x1) pattern on Ni (11), no oxygen patterns resembling the (3x2) have been observed on the (110) surfaces of other platinum metals.

V. Desorption Kinetics

To study the desorption and adsorption kinetics of chemisorbed oxygen, thermal desorption spectra were collected as a function of oxygen exposure on clean and oxidized Ir(110) at 300 K. The adsorption kinetics will be treated in the next section. Representative spectra are shown in Fig. 6. The spectrum for the oxidized surface gives a finite intensity for zero exposure because the oxide layer, which is associated with an oxygen coverage of 0.25 ML, contributes intensity in addition to that of the chemisorbed oxygen. Since the oxide forms rapidly at temperatures where oxygen begins to desorb, the spectra for the clean and oxidized surfaces are not markedly different in shape.

Three peaks, including a high temperature shoulder, appear in the desorption spectra of oxygen from clean and oxidized Ir(110). For saturation coverage, the intensity of the low temperature peak is increased for the oxidized surface because the total oxygen coverage at saturation is 1.25 ML on the oxidized surface, but is 1.0 ML on the clean surface (see Section III). Hence, the ratios of the intensities of the three peaks are 1:2:1 and 2:2:1 for saturation coverage on clean and oxidized Ir(110), respectively. The total coverages upon completion of the higher temperature peaks are 0.25 and 0.75 ML, which are associated with the (1x4) LEED pattern of the oxide and the c(2x2) pattern on the oxidized surface, respectively.

The desorption rate (R_d in atoms/cm^2 sec) may be modeled by the Arrhenius expression

$$R_d = - C_s \frac{d\theta}{dt} \qquad (1)$$

$$= \nu_d(\theta) C_s^n \theta^n \exp [- E_d(\theta)/kT]$$

where C_s, θ, t, T, n, k, $\nu_d(\theta)$ and $E_d(\theta)$ are the concentration of adsites (9.6×10^{14}/cm^2 for chemisorbed oxygen), total fractional coverage of oxygen, time, surface temperature, order of the desorption process, Boltzmann constant, pre-exponential factor and desorption energy. However, Eq. 1 is applicable to oxygen desorption only if the oxide atoms enter the chemisorbed phase before desorbing. This must be the case on Ir(110) since oxygen desorbs molecularly (see Section II).

If the rate parameters, $\nu_d(\theta)$ and $E_d(\theta)$, vary weakly with coverage, the order of the desorption process (n) is given by the slope of a plot of $\ell n\ R_d$ versus $\ell n\ \theta$ at points of constant temperature (26). Because the pumping speed in these experiments was rapid (21), the desorption rate (R_d) is proportional to the intensity of the mass 32 signal (I) at any instant. Substituting this relation into Eq. (1) and integrating, the coverage for any temperature in the desorption spectrum [$\theta(T)$] is given by (21)

$$\theta_o - \theta(T) = n \int_o^{t'} I dt \qquad (2)$$

where t' is the time when the sample reaches the temperature T, and θ_o is

the initial coverage of oxygen. Since the saturation coverage of oxygen on clean Ir(110) is known to be 1 ML at 300 K, the proportionality constant η is the ratio of 1 ML to the total intensity-time integral for saturation coverage.

Order plots of $\ell n\ R_d$ versus $\ell n\ \theta$, constructed from the spectra in Fig. 6, are shown in Fig. 7. For total coverages below 0.1 ML where all oxygen is in the oxide phase, the desorption order is one. Before two oxygen atoms may combine to desorb as a molecule, atoms in the oxide layer must enter the chemisorbed phase. This first-order process is presumably rate limiting below 0.1 ML coverage. Between 0.1 and 0.2 ML coverage, the order is two, which is expected for the chemisorbed atoms, which desorb as molecules. Above 0.2 ML coverage, the order appears to exceed two because the desorption energy is a strong function of coverage.

Plots of $\ell n\ I$ versus T^{-1} for points of constant coverage were constructed from the desorption spectra to determine the coverage dependences of the desorption energy $[E_d(\theta)]$ and the pre-exponential factor $[\nu_d(\theta)]$ (21). From Eq. (1), such plots should give lines with slope $[-\ E_d(\theta)/R]$ and intercept $\ell n[\nu(\theta)C_s^{n-1}\theta^n/\eta]$. For oxygen desorption from Ir(110), n is one for coverages below 0.1 ML, but is two otherwise. No artifacts associated with desorption from multiple states should appear at any coverage since the peaks in Fig. 6 are separated sufficiently in energy (27). Because of a problem of accuracy in this analysis (21), this procedure gave reliable values of the rate parameters between 0.3 and 0.4 ML coverage only. In this range, the desorption energy $[E_d(\theta)]$ and the pre-exponential factor $[\nu_d(\theta)]$ are 64 ± 2 kcal/mole and $3.5 \times 10^{-4} \overset{x}{\div} 3$ cm^2/sec.

If the pre-exponential factor varies weakly with coverage, the

desorption energy $[E_d(\theta)]$ is given by $kT\ell n(I\eta/\nu_d C_s^{n-1}\theta^n)$, which may be averaged over several spectra at points of constant coverage. Using this method and $\nu_d = 3.5 \times 10^{-4}$ cm^2/sec, the variation in desorption energy with total coverage of oxygen, was calculated, and the result is shown in Fig. 8. For coverages between 1.00 and 1.25 ML on the oxidized surface, the desorption energy equals 45 ± 3 kcal/mole. Substantial decreases in the energy near 0.25 and 0.75 ML are associated with the completion of the higher temperature peaks in the desorption spectra. The desorption energies of 45 - 70 kcal/mole measured for oxygen on Ir(110) do not differ appreciably from the energies of 48 - 80 and 36 - 59 kcal/mole measured on Pd(110) (12) and polycrystalline Pt (8), respectively.

Ignoring the coverage dependence of the pre-exponential factor probably introduces a substantial error in the calculated desorption energies, for the compensation effect, a sympathetic variation in the desorption energy and the pre-exponential factor with coverage, has been observed in other desorption systems (21). For such systems, the desorption kinetics are described by a modified Arrhenius equation (28)

$$R_d = \nu_o C_s^n \theta^n \exp\left[-\frac{E_d}{k}\left(\frac{1}{T} - \frac{1}{T_s}\right)\right] \qquad (3)$$

where ν_o and T_s are constants, and could not be obtained directly from the data in Fig. 6. If T_s is large compared to the temperature range where desorption is observed, little error is introduced by neglecting the coverage dependence of the pre-exponential factor. However, if T_s is assumed to be 2100 K, which approximates the value for oxygen desorption

from W(110) (28), the calculated desorption energies for Ir(110) are substantially in error, particularly at the extremes of coverage. At 0.05 ML, the energy is underestimated by 10 kcal/mole; at 0.95 ML, the energy is overestimated by 10 kcal/mole.

VI. Adsorption Kinetics

By integrating Eq. (2) over the entire desorption spectrum, the absolute coverage of chemisorbed oxygen was fixed for a given gas exposure since the saturation coverage of oxygen on clean or oxidized Ir(110) at 300 K was determined to be 1 ML (= 9.6×10^{14} atoms/cm^2). The adsorption kinetics for oxygen were derived in this way. Although the reaction of chemisorbed oxygen with the background impurities H_2 and CO may distort coverages measured in this way (15), these artifacts are insignificant in this work because these experiments were performed rapidly. Even if the reaction probability is unity for each CO and H_2 molecule impinging on the surface, the calculated coverages are in error by no more than ± 0.05 ML. The coverage exposure relations for oxygen on clean and oxidized Ir(110) at 300 K, obtained from the spectra in Fig. 6, are shown in Fig. 9. In determining the coverage of chemisorbed oxygen on the oxidized surface, the coverage of the oxide layer (0.25 ML) was subtracted from the total coverage.

The adsorption kinetics of oxygen on clean and oxidized Ir(110) are somewhat different. By evaluating the zero-coverage limit of the sticking probability (S) (29)--

$$S = \frac{1}{2} C_s (2\pi M k T_g)^{\frac{1}{2}} \frac{d\theta}{d\varepsilon} \qquad (4)$$

where M, T_g and ε are the molecular weight, the temperature and the

exposure (the pressure-time product) of gaseous oxygen--initial sticking probabilities (S_0) of 0.28 ± 0.02 and 0.36 ± 0.09 were calculated for oxygen chemisorption on the clean and oxidized surfaces at 300 K. The factor of $\frac{1}{2}$ appears in Eq. (4) because each adsorbed molecule occupies two sites. In calculating S_0, the exposures in Fig. 9 were multiplied by 1.22, which is the relative sensitivity of oxygen for the ion gauge that monitored the pressure.

For larger exposures, the data in Fig. 9 were fit by a second-order precursor model developed by Kisliuk (30). In this model, the sticking probability depends on the initial sticking probability, the coverage and a parameter κ that is inversely proportional to the lifetime of the precursor state,

$$\frac{S}{S_0} = \frac{(1 - \theta)^2}{1 - (1 + S_0 - \kappa)\theta + S_0\theta^2} \tag{5}$$

or in integrated form,

$$\frac{2S_0\varepsilon}{C_s(2\pi MkT_g)^{\frac{1}{2}}} = S_0\theta + \frac{\kappa\theta}{1 - \theta} - (1 - S_0 - \kappa)\ell n(1 - \theta) \tag{6}$$

The value of κ is different for clean and oxidized Ir(110). Values of zero and 0.25 ± 0.07 for the respective surfaces were obtained from a least squares fit of the data above 3.0 L exposure. The kinetic models for the two surfaces, given by Eq. (6), are shown as solid lines in Fig. 9.

A more complex second-order model (31), which includes an additional

parameter to account for the ordering of the adlayer by interadsorbate repulsion or attraction between nearest neighbors, is not needed to describe the data in Fig. 9. Repulsion between nearest neighbors is not expected to be significant for these data because the hard-sphere diameter of a chemisorbed oxygen atom, d_O = 1.5 Å (32), is small compared to the size of an adsorption site, $C_s^{-\frac{1}{2}} \sim 3$ Å.

The dependence of the adsorption kinetics of oxygen upon the surface temperature of clean Ir(110) was measured to determine the activation energy for oxygen chemisorption and to access the validity of the precursor model. The kinetic behavior for oxygen chemisorption on the platinum metals is not well established since, in studies of oxygen chemisorption on polycrystalline Pd (16) and Pt (8), and Pt(111) (13), the kinetics appeared to vary randomly with surface temperature. In the present work, the coverage-exposure relations for surface temperatures between 293 and 723 K were derived from TDS, as was done for the data in Fig. 9. These relations are shown in Fig. 10. Surface temperatures for these data are accurate to ± 5 K. The coverages for any surface temperature depended on the oxygen exposure only, but not the total pressure. Because the gain of the electron multiplier in the mass spectrometer drifted markedly for larger oxygen exposures, the gain was calibrated frequently by comparing the total intensities for spectra obtained for the same exposures at 300 K, but at different times. This calibration procedure was crucial for obtaining reproducible results and was also necessary for determining absolute surface coverages. Oxygen chemisorption is irreversible for the temperature and exposure ranges in Fig. 10. That is, desorption of the chemisorbed layer was negligible for these

data, as evidenced by the desorption energies calculated in the previous section.

The initial sticking probability was calculated to be 0.28 ± 0.04, irrespective of the surface temperature. Therefore, oxygen chemisorption on clean Ir(110) is not activated for gaseous molecules that impinge with a temperature of 300 K. Kinetic models, calculated from Eq. (6) with S_o = 0.28 for each surface temperature, are shown as solid lines in Fig. 10. The parameter κ for any temperature was determined by a least-square fit to the kinetic data above 3.0 L exposure. The temperature dependence of κ is shown in Fig. 11.

Because the initial sticking probability for oxygen chemisorption is constant with temperature, $\ell n\ \kappa$ should be inversely proportional to temperature (30). The proportionality constant is E_p/k where E_p is the difference between the activation energies for the desorption and diffusion of the precursor (33). The plot of $\ell n\ \kappa$ versus T^{-1} in Fig. 11 gives a line between 323 and 573 K with E_p = 2.1 kcal/mole. This value for oxygen chemisorption on Ir(110) is close to the value of 1.4 kcal/mole measured for oxygen on Pt field emission tips (2). The value of κ = 0 calculated from the kinetic data at 293 K does not fall on the line because diffusion in the chemisorbed layer is slow at this temperature, but is rapid at slightly higher temperatures (see Section IV). Diffusion and the subsequent ordering or disordering of the chemisorbed layer influences adsorption kinetics markedly (30, 31). Oxide formation, which has been shown to modify the kinetics of oxygen chemisorption on Ir(110), occurs above 573 K. Hence, values of κ for surface temperatures above 573 K also deviate from the line in Fig. 11. Despite these deviations,

the second-order precursor model given by Eq. (6) describes the kinetics of oxygen chemisorption consistently.

The kinetic behavior observed in the present work is quite similar to the behavior reported for oxygen chemisorption on other surfaces of the platinum metals. At 300 K, oxygen adsorbs dissociatively with a saturation coverage near 10^{15} atoms/cm^2 (4, 5, 7, 15). Generally, the adsorption kinetics are described accurately by a second-order Langmuir model or a second-order precursor model (3, 4, 6-8, 13, 15, 16). For surface temperatures near 300 K, the initial sticking probability for oxygen decreases with increasing molecular wieght of the metal and on smoother surface orientations. The initial sticking probability approximately equals unity on Ni(111), Ni(110), Ni(100) (3, 6) and polycrystalline Pd (16); 0.3 on Pd(111) (7), Pt(110) (4) and polycrystalline Pt (8); and 0.1 on Pt(111) (15) and Ir(111) (5). The formation of an oxide layer on Ni surfaces (3, 6), polycrystalline Pt (8) and Pt(110) (4) alters the adsorption kinetics of oxygen perceptibily. Moreover, oxidation of these surfaces increases the total coverage of oxygen.

VII. Charge transfer to the Overlayer

The contact potential differences (CPD) on clean and oxidized Ir(110) at 90 and 300 K were measured as a function of oxygen exposure. In Fig. 12, the data are shown as curves I and III for the respective temperatures. The CPD, ($\Delta\phi$), or equivalently, the work function,

increased monotonically with oxygen coverage. For all CPD measurements reported in this work, the CPD for any surface temperature varied with the oxygen exposure only, but not the partial pressure of oxygen. When the substrate was annealed from 90 to 300 K, the CPD for an overlayer chemisorbed at 90 K changed irreversibly. This suggests that another state of chemisorbed oxygen, presumably molecularly chemisorbed oxygen, is stable at low temperatures. The data for adsorption at 90 K will be treated in the next section.

To obtain accurate coverages at 90 K for the chemisorbed phase that is entered irreversibly upon annealing from 90 to 300 K, the substrate was exposed to oxygen at 300 K. Then the CPD was calculated from the difference between the surface potentials for the chemisorbed layer and the clean or oxidized surface at 90 K. The corresponding data points are shown as circles in Fig. 12; curves IIa and IIb approximate the data. The coverages for these data may be somewhat inaccurate because the surface temperature varied by ± 10 K during the exposures. As shown in the last section, the adsorption kinetics, especially for larger exposures, are quite sensitive to the surface temperature near 300 K.

When the coverages obtained from the adsorption kinetics are related to CPD measurements at points of constant exposure, the apparent dipole moment at infinite dilution (μ) and the apparent polarizability (α) of the adsorbate may be calculated from a modified Topping model (34). This model accounts for the depolarization of the overlayer resulting from dipole-dipole interactions. For any surface temperature, the ratio of the coverage to the CPD at that coverage ($\theta/\Delta\phi$) is plotted versus coverage (θ). The slope and intercept of the resulting line are

$(\Lambda\alpha C_s^{\frac{1}{2}}/4\pi\mu)$ and $(4\pi\mu C_s)^{-1}$. Here Λ is a structural parameter for the overlayer and is equal to 9.46 for an overlayer with periodicity of the Ir(110) surface. This value was obtained from a lattice sum over a (101x101) matrix according to a procedure described elsewhere (34). The modified Topping model is appropriate for adsorption at specific sites, rather than on an amorphous surface (34). For the latter case, the ratio $(\theta/\Delta\phi)$ is plotted versus $\theta^{3/2}$, but the slope and intercept assume the same values as for adsorption at specific sites.

Since the apparent dipole moment is the sum of the normal components of the true dipole moment of the adsorbate and the image dipole of the adsorbate in the surface, the apparent dipole moment at infinite dilution (μ) equals twice the component of the true dipole moment that is normal to the surface. Similarly, the apparent polarizability of the adsorbate (α) equals the apparent dipole moment divided by the normal component of the average field at the surface (E_n), which orients the apparent dipole moment in the absence of induced charge. The image dipole of the adsorbate accounts for the field resulting from the induced charge (18).

Topping plots for the CPD measurements in curves II and III of Fig. 12 were constructed using the coverage-exposure relations for oxygen chemisorption on clean and oxidized Ir(110) at 300 K, shown in Fig. 9. The plots, as well as the apparent dipole moments and polarizabilities, deduced from the intercepts and slopes, are shown in Fig. 13. The dipole moments of oxygen on the oxidized surface are smaller than those on the clean surface because the oxide layer draws charge toward the surface. The Topping plots for the clean surface have discontinuities at 0.25 ML coverage where the p(2x2) LEED pattern is formed completely. Similarly,

the plots for the oxidized surface have discontinuities at 0.50 ML coverage where the c(2x2) pattern is formed completely. In all cases, the apparent dipole moments and polarizabilities decrease for coverages exceeding the point of discontinuity. A pronounced change in the CPD, or equivalently, the dipole moments with coverage has been observed for oxygen chemisorption on Ni(100) (3) and Ru(001) (19) also. For these systems, the CPD also changes discontinuously for coverages where an ordered overlayer structure appears with maximum intensity.

The binding sites for oxygen on either clean or oxidized Ir(110) may be inferred from the pronounced change in the dipole moment with coverage. On either surface, the dipole moment for the low coverage state is substantially larger. If the binding sites of oxygen are above the topmost row of surface atoms, the apparent dipole moment, i.e. the normal component of the true dipole moment of the adsorbate, is expected to be larger than the dipole moment for sites between the rows. That is, for sites above the rows, the movement of the bonding electrons would be more nearly normal to the surface. Therefore, the low coverage states on clean and oxidized Ir(110) are associated with binding sites above the topmost rows of surface atoms whereas the high coverage states are associated with binding sites between the rows. Since the p(2x2) pattern on clean Ir(110) and the c(2x2) pattern on oxidized Ir(110) correspond to the low coverage states of oxygen, the binding sites of oxygen in Figs. 4 and 5 were fixed above the topmost rows of surface atoms (35). Although chemisorbed oxygen may bind in different states at low and high coverages, no differences in the Auger or X-ray spectra of oxygen could be resolved for the two states when the spectra were normalized to the same intensity.

The data in Fig. 12 show that the CPD for oxygen chemisorption is a function of temperature as well as coverage. The temperature dependence of the CPD was investigated more thoroughly by measuring the CPD as a function of oxygen exposure on clean Ir(110) for temperatures between 300 and 723 K. The data are shown in Fig. 14. When the surface temperature was maintained between 300-673 K, the CPD changed reversibly, within experimental error (± 0.01 eV) upon rapidly annealing or cooling the sample. The CPD is altered irreversibly upon annealing above 673 K because the oxide layer forms rapidly at these temperatures. Topping plots, shown in Fig. 15, were constructed from the coverage-exposure relations in Fig. 11 and the CPD measurements in Fig. 14.

The Topping plots for surface temperatures below 673 K give a discontinuity near 0.25 ML coverage, where the p(2x2) LEED pattern of the overlayer appears with maximum intensity on the clean surface. At 723 K, the discontinuity vanishes because the oxide layer forms rapidly and changes the dipole moment and polarizability of oxygen. In this case, these quantitites are 0.43 ± 0.04 D (1D = 1 esu-cm) and 25 ± 2 $\mathring{A}^3$, respectively. For temperatures below 673 K, where the oxide forms slowly, the apparent dipole moments for the low coverage ($\lesssim$ 0.25 ML) and the high coverage (> 0.25 ML) states are 1.0 ± 0.1 D and 0.25 ± 0.02 D as calculated from a least squares fit to the data. Moreover, the dipole moment for either state is independent of temperature. One exception to this generality occurs for the dipole moment of the high coverage state at 90 K, which is 0.39 ± 0.05 D. However, this value may be inaccurate because, as noted previously, some error was introduced in determining the coverage for this particular case.

Although the dipole moments for oxygen in either coverage state are independent of temperature, the polarizabilities are not. The polarizability for either state is inversely proportional to temperature, as shown in Fig. 16. A model will be developed now to explain this temperature dependence. The polarizability may change with a temperature as a result of the thermal randomization of the chemisorbed species among different electronic, vibrational or translational modes. Because the dipole moment for either coverage state is independent of temperature, only the energy of chemisorbed oxygen and the average surface field ($E_n = \frac{\mu}{\alpha}$) vary among the modes. Since the thermal energy required to populate low-lying electronic levels is usually quite large (35), the temperature dependence of the polarizability cannot be explained in this way. When the first vibrational mode is entered, the equilibrium separation of nuclei typically changes by $\sim$ 0.1 Å (35), which is small compared to the diameter of a chemisorbed oxygen atom (d_0 = 1.5 Å) (32). Since the volume, over which the surface field is averaged, does not vary appreciably for the various vibrational modes, the temperature dependence of the polarizability cannot be explained by this motion either. Therefore, this phenomenon can be attributed only to the thermal randomization of the translational motion. The translational motion for the two coverage states will be treated separately.

Motion in the [001] azimuth may be ignored since the barrier for diffusion in this direction is relatively large, as will be shown. The potential variation for chemisorption in the [110] azimuth will be approximated by a periodic square well of height U. Although the dipole moment does not vary for the sites at the top and bottom of the well, the

polarizability does change because the average surface field (E_n) is different for the two sites. The field is expected to vary between the sites since the field acting on an oxygen atom is averaged over a relatively small volume, $\frac{\pi}{6} d_0^3 \approx 2$ Å^3, compared to the volume of a unit cell, $C_s^{-3/2} \approx 30$ Å^3. However, the field acting on chemisorbed CO on Ir(110) does not change among the binding sites (18), for the volume of the CO molecule is comparable to the size of a unit cell. The apparent polarizabilities (μ/E_n) for sites in the bottom and on the top of the well are denoted α_1 and α_2.

The apparent polarizability of chemisorbed oxygen [$\alpha(T)$] is a thermal average between the polarizabilities for the two types of sites, i.e.,

$$\alpha(T) = \alpha_1 \left[\frac{\exp(U/kT)}{1 + \exp(U/kT)}\right] + \alpha_2 \left[\frac{1}{1 + \exp(U/kT)}\right] \qquad (7)$$

where the terms in brackets are the Boltzmann weighting factors for the two sites (36, 37). Although this model will be used to describe the temperature dependence of the polarizability for the two coverage states of oxygen, the values of α_1, α_2 and U are expected to be different for the two states since the dipole moments are different. Eq. (7) may be simplified by a Taylor series expansion about any temperature T_α

$$\alpha(T) = i + s/T \qquad (8)$$

$$i = \alpha_1 + (\alpha_2 - \alpha_1)\left(1 + \frac{U}{kT_\alpha}\right)\exp(-U/kT_\alpha)$$

$$s = \frac{U}{k}(\alpha_2 - \alpha_1)\exp(-U/kT_\alpha)$$

where s and i are the slope and intercept in a plot of $\alpha(T)$ versus T^{-1},

as in Fig. 16. This analysis is not rigorously valid, for the structural parameter Λ was assumed to be constant when the polarizabilities in Fig. 16 were calculated. The value of Λ is expected to vary with temperature because the order in the adlayer is changed. However, as is shown in Appendix I, ignoring the temperature dependence of Λ introduces a negligible error in this analysis.

The polarizability (α_1) for the energetically favored sites of oxygen on clean Ir(110) may be approximated as the apparent polarizability at 90 K, which is 37 $\mathring{A}^3$ for the low coverage state and is 3.8 $\mathring{A}^3$ for the high coverage state. The potential variation between the sites (U) is $k\left[\left(\frac{i - \alpha_1}{s}\right) - \frac{1}{T_\alpha}\right]^{-1}$. If this expression is evaluated at 400 K, which corresponds to the mean value of T^{-1} in Fig. 16, the potential variations for the high and low coverage states are 0.42 and 2.5 kcal/mole. As stated previously, the potential variation in the [110] azimuth, approximately 1 kcal/mole, is small compared to the variation in the [001] azimuth, approximately 20 kcal/mole (see Section IV), so diffusion in the [001] azimuth may be ignored. By evaluating Eq. (8) at 400 K, the polarizabilities (α_2) for the energetically disfavored sites are 94 and 35 $\mathring{A}^3$ for the low and high coverage states. These calculated values of α_1, α_2 and U for chemisorbed oxygen on clean Ir(110) are accurate to ± 20%.

The dipole moments for the low and high coverage states of oxygen on oxidized Ir(110), which equal 0.24 ± 0.03 and 0.15 ± 0.03 D from Fig. 13, also appear to be independent of temperature. Hence, this translational model may be used to describe the temperature dependence of the polarizability in this system. If the potential variations (U) for the low and high coverage states on oxidized Ir(110) are approximated by the variations

on the clean surface, values of α_1 = 4 and 0.4 A^3, and α_2 = 17 and 27 A^3 are calculated for the respective states. For any site on either surface, the field ($E_n = \mu/\alpha$) that orients the dipole moment of chemisorbed oxygen is on the order of 10^7 V/cm, which is comparable to the field acting on chemisorbed CO on Ir(110) (18).

The dipole moment and polarizability for the oxide atoms also were determined from CPD measurements. To fix the coverage in the oxide layer, a clean surface at 300 K was exposed to oxygen. Then the oxide layer was formed by annealing the overlayer to 1100 K for one minute. The CPD for the oxide layer ($\Delta\phi$) was calculated from the difference in the surface potentials for the annealed layer and the clean surface at 300 K. Data obtained from this experiment are shown in the Topping plot in Fig. 17. The concentration of sites in the oxide layer is 2.4 x 10^{14} cm^{-2} (0.25 ML), so the dipole moment and the polarizability for an oxide atom at 300 K are 0.38 ± 0.04 D and 8 ± 4 A^3. Since these values for the oxide are much different from those obtained for chemisorbed oxygen at the same coverage and temperature, the chemical states of the two forms of oxygen must be different.

VIII. Molecularly Chemisorbed Oxygen

The irreversible change in the CPD, which occurs when an oxygen overlayer that is chemisorbed at 90 K is annealed to 300 K, must result from a change in the chemical state of oxygen. A similar change in the CPD is observed when chemisorbed oxygen atoms enter the oxide phase. The irreversibility at low temperatures cannot be associated with the ordering of the overlayer by diffusion since the irreversibility is observed at

high coverages where the ordering of the adlayer is expected to produce a negligible change in the CPD (38). In the remainder of this section, evidence will be presented to show that oxygen chemisorbs molecularly, rather than dissociatively, on clean or oxidized Ir(110) at 90 K. The existence of this third chemisorbed state of oxygen not only explains the irreversible change in the CPD, but also relates other experimental data consistently. A molecularly chemisorbed state of oxygen has not been reported previously on the platinum metals, except on polycrystalline Pt (10). The TDS spectrum of oxygen adsorbed on polycrystalline Pt at 100 K shows a feature near 200 K, which has been attributed to molecularly chemisorbed oxygen.

Several TDS spectra of oxygen adsorbed on clean and oxidized Ir(110) at 90 K were collected in the present study. Below room temperature, less than 0.1 ML of oxygen was desorbed. Because the yield of desorbed oxygen fluctuated randomly with the exposure of oxygen and the heating rate, the low temperature feature was associated with desorption from the manipulator, rather than from the Ir(110) surface. Hence, the desorption of molecular oxygen on Ir(110) occurs at a negligibly slow rate compared to the dissociation of this species.

Because artifacts appeared in the desorption spectra at low temperatures, these spectra could not be used for measuring the adsorption kinetics of oxygen on Ir(110) at 90 K. However, the initial sticking probability of molecularly chemisorbed oxygen could be determined from CPD measurements. The surface was exposed to oxygen at 90 K and then was annealed to 300 K. Using the Topping plots for dissociatively chemisorbed oxygen at 90 K, which are shown in Fig. 13, the coverage for each exposure

was determined from the difference in the surface potentials for the annealed overlayer and the clean or oxidized surface at 90 K. This technique was not used for larger exposures (> 2 L) since the coverages associated with these exposures are somewhat inaccurate, as was shown in the last section. From these CPD measurements, the initial sticking probabilities for oxygen chemisorption on clean and oxidized Ir(110) at 90 K are 0.28 ± 0.03 and 0.54 ± 0.10. These data are further evidence that oxygen chemisorption on either surface is not activated. The discrepancy in the initial sticking probabilities on the oxidized surface at 90 and 300 K may be accounted for entirely by experimental errors. Moreover, since the initial sticking probability on the oxidized surface appears to decrease with temperature, an activation barrier cannot exist, for the opposite trend should be observed.

Using the initial sticking probabilities derived above, Topping plots for oxygen chemisorption on clean and oxidized Ir(110) at 90 K were constructed from the CPD-exposure relations of curves Ia and Ib in Fig. 12. The plots, as well as values for the dipole moments and polarizabilities (per atom) of molecular oxygen, are shown in Fig. 18. The concentration of adsorption sites was assumed to be 9.6×10^{14} cm^{-2} in determining these values. Although the Topping plots were fit to a single line, no discontinuities in the plots were detected experimentally when the coverage for any CPD was measured with the method described previously.

At 90 K, the dipole moments (per atom) of molecular oxygen on clean and oxidized Ir(110) nearly equal those of dissociatively chemisorbed oxygen in the respective low coverage states. This suggests that molecular oxygen binds above the topmost rows of surface atoms, like dissociatively

chemisorbed oxygen in the low coverage states, and fills two adjacent adsorption sites. Moreover, the interatomic axis of molecular oxygen is parallel, rather than normal to the surface. This assignment of a parallel orientation is supported by XPS data since the 1s line for molecular oxygen on clean or oxidized Ir(110) is not split into two peaks. Although the dipole moments (per atom) do not differ for molecularly and dissociatively chemisorbed oxygen at 90 K, the polarizabilities do differ. This, as well as the absence of discontinuities in Topping plots in Fig. 18, is further evidence that oxygen may chemisorb into two distinct chemical states, depending on the surface temperature.

The adsorption kinetics of molecular oxygen on clean and oxidized Ir(110) at 90 K were deduced from the Topping plots in Fig. 18 and the CPD-exposure relations in Fig. 12. The coverage-exposure relations are shown in Fig. 19. A second-order precursor model, which accounts for the ordering of a mobile overlayer by intermolecular repulsion (31), describes the kinetics data for either surface accurately. If the adsorbates fill two adjacent sites, as is the case for molecular oxygen on Ir(110), this model is appropriate for describing non-dissociative adsorption. In this model (31), the sticking probability at any coverage, S, is related to the initial sticking probability; a parameter, κ, that is identical to the parameter κ in Eq. (5) (q.v.); and a parameter, B, that is related to the repulsive energy between nearest neighbors [$- kT\ell n(1 - B)$]

$$S/S_0 = [(1 - \kappa) + \kappa/\theta_{00}]^{-1} \qquad (9)$$

where

$$\theta_{oo} = (1 - \theta) - \frac{2\theta(1 - \theta)}{[1 - 4B\theta(1 - \theta)]^{\frac{1}{2}} + 1} \qquad (10)$$

For clean Ir(110), this second-order model with $S_o = 0.28$, $\kappa = 0.15$ and $B = 0.995$ fits the kinetic data well; for oxidized Ir(110), $S_o = 0.54$, $\kappa = 0.30$ and $B = 0.995$. The integrated forms of models with these parameters from Eqs. (4), (9) and (10) are shown as solid lines in Fig. 19. The repulsive energy between nearest-neighbors equals 1 kcal/mole on either surface. This intermolecular repulsion reduces the sticking probabilities for higher coverages so that the adsorption kinetics at 90 K for higher coverages are slow compared to the rate at 300 K, where the repulsion is negligible.

The activation energy for the dissociation of molecular oxygen on Ir(110) also was determined from CPD measurements. When molecular oxygen was chemisorbed on clean or oxidized Ir(110) at 90 K, the CPD of the overlayer decayed exponentially to the value obtained by annealing the sample to 300 K. The half-life for the decay was independent of coverage and was calculated to be about 20 min , as extrapolated from experimental data. In this experiment, care was exercised to insure that the adsorption of background impurities did not affect the CPD. If the rate of dissociation is described by a first-order Arrhenius expression with a pre-exponential factor of 10^{13} sec^{-1}, the activation energy for dissociation is approximately 8 kcal/mole. This activation energy cannot be associated with diffusion. In the previous section, the activation energies for diffusion in the [001] and [110] azimuths were determined to be approximately 30 and 1 kcal/mole. Moreover, slow dissociation did not influence the CPD measurements shown in Fig. 12 because these

experiments were completed after 100 sec.

Oxygen chemisorption above 200 K is effectively dissociative since the rate of dissociation may be estimated as 10^5 ML/sec at 200 K. Although the activation energy for the dissociation of molecular oxygen is substantial, chemisorption need not be activated. The maximum in the barrier between the molecularly and dissociatively chemisorbed states must be substantially below the vacuum potential since little, if any, molecular oxygen desorbs from the surface below room temperature. Hence, this barrier does not influence the adsorption kinetics of oxygen on Ir(110). Assuming that no more than 0.1 ML desorbs upon annealing the surface from 90 to 300 K, the activation energy for desorption must be at least 10 kcal/mole.

The Auger KLL transitions for molecularly and dissociatively chemisorbed oxygen on clean Ir(110) are different in shape and energy. The spectra for the respective chemical states, shown in Fig. 20, were recorded upon oxygen chemisorption at 90 and 300 K. When an oxygen overlayer was chemisorbed at 90 K and then was annealed to 300 K, the spectrum observed at 90 K was identical to the spectrum for dissociatively chemisorbed oxygen, as expected. The spectra in Fig. 20 were obtained under conditions similar to those for the spectra in Fig. 3 (q.v.). The coverages for the two chemical states were nearly the same value, 0.5 ML. To inhibit electron beam dissociation, the spectrum for molecular oxygen was obtained by summing fifty individual spectra of 120 sec duration each. By comparison to a gas phase spectrum (39), peaks in the spectrum of molecular oxygen have been associated with final hole states as noted in Fig. 20. The intensity of the peak near 495 eV is increased substantially for molecular oxygen. This increase may result from a shift

in the energy of the $2\sigma_u$ orbital toward the $2\sigma_g$ orbital, as compared to the gas phase energies. However, further evidence from UPS experiments is needed to verify this conjecture.

IX. Conclusions

The chemisorption of oxygen on Ir(110) has been studied to give insight into the CO oxidation reaction on Ir(110). Oxygen may adsorb into three distinct chemical states: (1) a molecularly chemisorbed species that is stable below 100 K, (2) a dissociatively chemisorbed species, and (3) a surface oxide that forms rapidly above 700 K. The oxide layer saturatures at 0.25 ML (1 ML = 9.6×10^{14} atoms/cm^2) coverage and forms with an activation energy of 45 kcal/mole. The oxide, compared to dissociatively chemisorbed oxygen, is less reactive to reduction by CO. Although the oxide layer may be either ordered, giving a (1x4) LEED pattern, or amorphous, giving a (1x1) LEED pattern with streaks between the beams in the [001] azimuth, the chemical properties for these two forms of oxide are otherwise identical.

The saturation coverage of oxygen on clean or oxidized Ir(110) at 300 K is 1 ML. Different LEED patterns are observed on the two surfaces. A p(2x2) pattern forms near 0.25 ML coverage on the clean surface; a c(2x2) pattern forms near 0.5 ML coverage on the oxidized surface. In both cases, the overlayer must be annealed to order the pattern in the [001] azimuth. The activation energy for diffusion of dissociatively chemisorbed oxygen on either surface is approximately 30 kcal/mole in the [001] azimuth and 1 kcal/mole in the [110] azimuth.

Oxygen desorbs molecularly from Ir(110) with an activation energy of 45 - 70 kcal/mole, decreasing continuously with coverage. The pre-exponential factor for the desorption rate coefficient is 3.5×10^{-4} cm^2/sec near 0.35 ML coverage. The oxide forms rapidly while chemisorbed oxygen desorbs. The desorption kinetics are second-order above 0.1 ML coverage, but are first-order at lower coverages since the decomposition of the oxide, rather than the union of chemisorbed atoms, is rate limiting.

For surface temperatures between 300 and 700 K, the adsorption kinetics are described by a second-order precursor model. The difference in the activation energies for the desorption and the diffusion of the precursor is 2.1 kcal/mole on clean Ir(110). Oxygen chemisorption is not activated on clean or oxidized Ir(110). That is, the initial sticking probability is independent of the surface temperature, and is equal to 0.28 and 0.4 on the respective surfaces.

The CPD on Ir(110) always increases with oxygen coverage, regardless of the chemical state. The dipole moment and polarizability of dissociatively chemisorbed oxygen on clean and oxidized Ir(110) change at coverages of 0.25 and 0.50 ML, respectively. For lower coverages, the dipole moment and polarizability for either surface are larger, indicating that oxygen adsorbs above the topmost rows of surface atoms at lower coverages and between the rows at higher coverage. Although the dipole moment for any coverage of dissociatively chemisorbed oxygen on either surface is independent of temperature, the polarizability is not. The polarizability changes because, as the surface temperature increases, oxygen diffuses in the [110] azimuth and fills sites with different energies and polarizabilities.

If an overlayer of oxygen that is chemisorbed at 90 K is annealed to 300 K, the CPD changes irreversibly indicating that oxygen chemisorbs molecularly at 90 K. The activation energy for the dissociation of molecular oxygen is 8 kcal/mole whereas the desorption energy is at least 10 kcal/mole. On clean and oxidized Ir(110), the dipole moment (per atom) of molecular oxygen equals that of dissociatively chemisorbed oxygen in the low coverage state on the respective surfaces. This suggests that molecular oxygen chemisorbs with the molecular axis parallel to the surface and fills two adjacent sites above the topmost row of surface atoms. The adsorption kinetics of molecular oxygen are described by a second-order precursor model, which accounts for intermolecular repulsion in the overlayer.

Acknowledgement

This research was supported by the National Science Foundation under Grant Number DMR 77-14976.

Appendix I

The translational model in Section VII, which describes the temperature dependence of the polarizabilities in Fig. 16, is not rigorously correct. These polarizabilities were calculated for a fixed value of the structural parameter, Λ_1, where Λ_1 is the structural parameter for an overlayer with the same periodicity as the (110) surface of an fcc metal. In the translational model, the order in the adlayer changes with temperature and, consequently, the structural parameter [$\Lambda(T)$], as well as the polarizability, varies with temperature. Therefore, the polarizabilities in Fig. 16 (α_o) are rigorously equal to

$$\alpha_o = \alpha(T)\Lambda(T)/\Lambda_1 \tag{11}$$

where $\alpha(T)$ is given by Eq. (7).

The structural parameter [$\Lambda(T)$] for oxygen on Ir(110) is estimated as follows. Motion in the [001] direction is ignored as was done in Section VII. At low temperatures, the overlayer is assumed to have the periodicity of the Ir(110) surface. The position of an adsorbate in the [110] direction is δr in any unit cell. The value of δ varies between $-\frac{1}{2}$ and $\frac{1}{2}$, and r, which equals 2.7 Å for Ir(110), is the distance between the centers of nearest unit cells in the [110] direction. Step changes in the square-well potential occur for hr/2 where h is an integer. When the adsorbate is in an energetically favored site, δ lies between 0 and $+\frac{1}{2}$; for an energetically disfavored site, δ lies between $-\frac{1}{2}$ and 0. Since the hard-sphere diameter of oxygen (1.5 Å) (32) approximately equals r/2, intermolecular repulsion is accounted for by constraining the

motion of the center of an adsorbate between $-\frac{1}{4} < \delta < +\frac{1}{4}$. The mean positions of the adsorbate in energetically favored and disfavored sites are $\delta = +\frac{1}{8}$ and $\delta = -\frac{1}{8}$. To simplify the calculation of lattice parameters, the adsorbate will be constrained to these two positions. This may introduce some error in the calculation, but intermolecular repulsion, which correlates the motion of adsorbates, is treated approximately also.

The dipole-dipole interaction between any two adsorbates is described by one of three situations. (1) Both adsorbates are in energetically favored sites with $\delta = \frac{1}{8}$. The probability of this case occurring is p_1^2 where p_1 is the Boltzmann weighting factor for the energetically favored sites, $[1 + \exp(-U/kT)]^{-1}$. (2) Both adsorbates are in energetically disfavored sites with $\delta = -\frac{1}{8}$. The probability for this situation is p_2^2, where p_2 is the Boltzmann weighting factor for the energetically disfavored sites, $[1 + \exp(U/kT)]^{-1}$. (3) One adsorbate is in an energetically favored site, and one is in an energetically disfavored site. Because this case has a degeneracy of two, the probability is $2p_1p_2$. The structural parameter is Λ for cases (1) and (2), but is Λ_2 for case (3) where Λ_2 is calculated from a lattice sum around one adsorbate at $\delta = -\frac{1}{8}$, but with all other adsorbates at $\delta = +\frac{1}{8}$ (34, 40). Then the structural parameter $[\Lambda(T)]$ is a weighted sum of the structural parameters for the three cases

$$\Lambda(T) = p_1^2\Lambda_1 + p_2^2\Lambda_1 + 2p_1p_2\Lambda_2 \tag{12}$$

Upon substituting Eq. (12) and $\alpha(T) = \alpha_1p_1 + \alpha_2p_2$, Eq. (11)

becomes

$$\alpha_o = (\alpha_1 p_1 + \alpha_2 p_2)(p_1^2 + p_2^2 + 2p_1 p_2 \Lambda_2/\Lambda_1) \tag{13}$$

If the Boltzmann weighting factors in Eq. (13) are expanded in a Taylor series about T_α, Eq. (13) is approximated by

$$\alpha_o = i' + s'/T \tag{14}$$

$$i' = \alpha_1 + (\alpha_2 - \alpha_1\Delta)(1 + U/kT_\alpha)\exp(-U/kT_\alpha)$$

$$s' = \frac{U}{k}(\alpha_2 - \alpha_1\Delta)\exp(-U/kT_\alpha)$$

where $\Delta = (3 - 2\Lambda_2/\Lambda_1)$. Here s' and i' remain the slope and intercept from the plots in Fig. 16, but Eq. (14) accounts for the temperature dependence of the structural parameter whereas Eq. (8) does not. By evaluating lattice sums over a (101x101) overlayer matrix (34, 40), Λ_1 equals 9.46 and Λ_2 equals 11.00. If α_1, α_2 and U are evaluated with Eq. (14), as was done with Eq. (8) in Section VII, the values for α_1 and U are not altered from those in Section VII, whereas the values for α_2 are lower by 10%. Imperfections in the surface may distort α_1, α_2 and U by a larger amount. Therefore, the temperature dependence of the structural parameter Λ is negligible for oxygen on Ir(110).

References

1. C. W. Tucker, Jr., J. Appl. Phys. 37 4147 (1966); 38, 2696 (1967).
2. R. Lewis and R. Gomer, Surface Sci. 12, 157 (1968).
3. P. H. Holloway and J. B. Hudson, Surface Sci. 43, 123 (1974); 43, 141 (1974).
4. R. Ducros and R. P. Merrill, Surface Sci. 55, 227 (1976).
5. V. P. Ivanov, G. K. Boreskov, V. I. Savchenko, W. F. Egelhoff, Jr. and W. H. Weinberg, Surface Sci. 61, 207 (1976).
6. P. R. Norton, R. L. Tapping and J. W. Goodale, Surface Sci. 65, 13 (1977).
7. H. Conrad, G. Ertl, J. Küppers and E. E. Latta, Surface Sci. 65, 245 (1977).
8. T. Matsushima, D. B. Almy and J. M. White, Surface Sci. 67, 89 (1977).
9. H. Conrad, J. Küppers, F. Nitschké and A. Plagge, Surface Sci. (submitted).
10. R. P. Norton, Surface Sci. 47, 98 (1975).
11. L. H. Germer, J. W. May and R. J. Szostak, Surface Sci. 8, 430 (1967).
12. G. Ertl and P. Rau, Surface Sci. 15, 443 (1969).
13. H. P. Bonzel and R. Ku, Surface Sci. 33, 91 (1972).
14. K. Christmann and G. Ertl, Z. Naturforsch. 28a, 1144 (1973).
15. H. P. Bonzel and R. Ku, Surface Sci. 40, 85 (1973).
16. T. Matsushima and J. M. White, Surface Sci. 67, 122 (1977).
17. J. L. Taylor, Ph.D. Thesis, California Institute of Technology.
18. J. L. Taylor, D. E. Ibbotson and W. H. Weinberg, J. Chem. Phys. (submitted).
19. T. E. Madey, H. A. Engelhardt and D. Menzel, Surface Sci. 48, 304 (1975).

20. G. Erhlich, J. Appl. Phys. 32, 4 (1961).

21. J. L. Taylor and W. H. Weinberg, Surface Sci. (submitted).

22. J. L. Taylor, D. E. Ibbotson and W. H. Weinberg, in preparation.

23. Pumping by the walls was negligible since these surfaces, as well as the substrate, received saturation coverages of the adsorbates during exposure.

24. J. C. Fuggle, E. Umbach and D. Menzel, Solid State Commun. 20, 89 (1972).

25. P. A. Thiel, J. T. Yates, Jr. and W. H. Weinberg, to be published.

26. J. L. Falconer and R. J. Madix, J. Catal. 48, 262 (1977).

27. From Ref. 21, two peaks are separated sufficiently in energy if the difference in the energies at the peaks is greater than 10% of the larger energy.

28. T. Engel, T. von dem Hagen and E. Bauer, Surface Sci. 62, 361 (1977).

29. W. H. Weinberg, C. M. Comrie and R. M. Lambert, J. Catal. 41, 489 (1976).

30. P. Kisliuk, J. Phys. Chem. Solids 5, 78 (1958).

31. D. A. King and M. G. Wells, Proc. R. Soc. A 339, 245 (1974).

32. P. M. Marcus, J. E. Demuth and D. W. Jepsen, Surface Sci. 53, 501 (1975).

33. More precisely, $\ell n \kappa$ equals $a - \ell n[1 + b \exp(E_p/kT)]$ where a and b are constants if chemisorption is not activated. For E_p = 2.1 kcal/mole, $\ell n \kappa$ is proportional to T^{-1} between 300 and 600 K. Also, E_p is defined rigorously as the difference in the activation energies for desorption and diffusion of the precursor when the precursor is above a site filled with an adsorbate. See Ref. 30.

34. J. R. MacDonald and C. A. Barlow, J. Chem. Phys. 39, 412 (1963).

35. A LEED study of the c(2x2) pattern, which followed this work, has shown that oxygen binds above the topmost rows of surface atoms and between adjacent Ir atoms, as shown in Fig. 5. C.-M. Chan, S. L. Cunningham, K. L. Luke, W. H. Weinberg and S. W. Withrow, in preparation.

36. R. H. Fowler and E. A. Guggenheim, Statistical Thermodynamics, MacMillan, New York, 1939.

37. A similar expression has been used to model the temperature dependence of the dipole moment in: G. Ertl, M. Neumann and K. M. Streit, Surface Sci. 64, 393 (1977)

38. At lower coverages, the order affects the term Λ in the modified Topping model since the number and position of nearest neighbors for each adsorbate depends on the order. However, the registry of the overlayer, instead of the order, fixes these quantities at higher coverages so that Λ, i.e., the CPD, is not influenced by the order at higher coverages. Also see Ref. 34.

39. W. E. Moddeman, T. A. Carlson, M. O. Krause, B. P. Pullen, W. E. Bull and G. K. Schweitzer, J. Chem. Phys. 55, 2317 (1971).

40. Rigorously, $\Lambda_1 = 2^{3/4} \times \sum_{i,j} [i^2 + 2j^2]^{-3/2}$ where i and j are integers, but the term i=j=o is excluded from the sum. Also, $\Lambda_2 = 2^{3/4} \times \sum_{i,j} [(i + \frac{1}{4})^2 + 2j^2]^{-3/2}$, where i and j assume the same limits as before. Imperfections in the surface distort the values of Λ_1 and Λ_2. Also see Ref. 34.

Figure Captions

Figure 1: (1x4) LEED pattern at 56 eV and proposed structure for an oxidized Ir(110) surface.

Figure 2: Thermal desorption spectra of CO_2 formed from the reaction of adsorbed oxygen and CO on Ir(110). Oxygen was adsorbed on the surface, and then the substrate was annealed to one of the temperatures shown. The CO_2 spectrum was measured after CO was adsorbed on the surface at 300 K.

Figure 3: Auger KLL transitions of an oxide layer and dissociatively chemisorbed oxygen on Ir(110) at 300 K. The coverage in each spectrum is 0.25 ML.

Figure 4: p(2x2) LEED pattern at 55 eV and proposed structure for oxygen chemisorbed on clean Ir(110).

Figure 5: LEED patterns of oxygen chemisorbed on oxidized Ir(110).
(a) c(2x2) pattern at 90 eV, and proposed structure;
(b) (3x2) pattern at 40 eV.

Figure 6: Thermal desorption spectra of adsorbed oxygen on Ir(110).
(a) clean surface;
(b) oxidized surface.

Figure 7: Order plots for oxygen desorption from Ir(110).

Figure 8: The coverage dependence of the activation energy of desorption of oxygen from Ir(110). The pre-exponential factor is fixed at 3.5×10^{-4} cm^2/sec for all coverages.

Figure 9: Adsorption kinetics of oxygen on clean and oxidized Ir(110) at 300 K.

Figure 10: Adsorption kinetics of oxygen on clean Ir(110) at various temperatures.

Figure 11: The temperature dependence of the parameter κ calculated from the data in Fig. 10. Here κ is inversely proportional to the lifetime of the precursor state.

Figure 12: CPD as a function of oxygen exposure on Ir(110).
(a) clean surface;
(b) oxidized surface.
The exposures and CPD measurements were performed at 90 K for curves I; at 300 and 90 K for curves II (see text); and at 300 K for curves III.

Figure 13: Topping plots for dissociatively chemisorbed oxygen on Ir(110) at 90 and 300 K.
(a) clean surface;
(b) oxidized surface.

Figure 14: CPD as a function of oxygen exposure on clean Ir(110) at various surface temperatures.

Figure 15: Topping plots for dissociatively chemisorbed oxygen on clean Ir(110) at various temperatures.

Figure 16: The temperature dependences of the polarizabilities of oxygen chemisorbed on clean Ir(110), calculated from the data in Fig. 15.
(a) low coverage state ($\theta < 0.25$ ML);
(b) high coverage state ($\theta > 0.25$ ML).

Figure 17: Topping plot for the oxide layer on Ir(110) at 300 K.

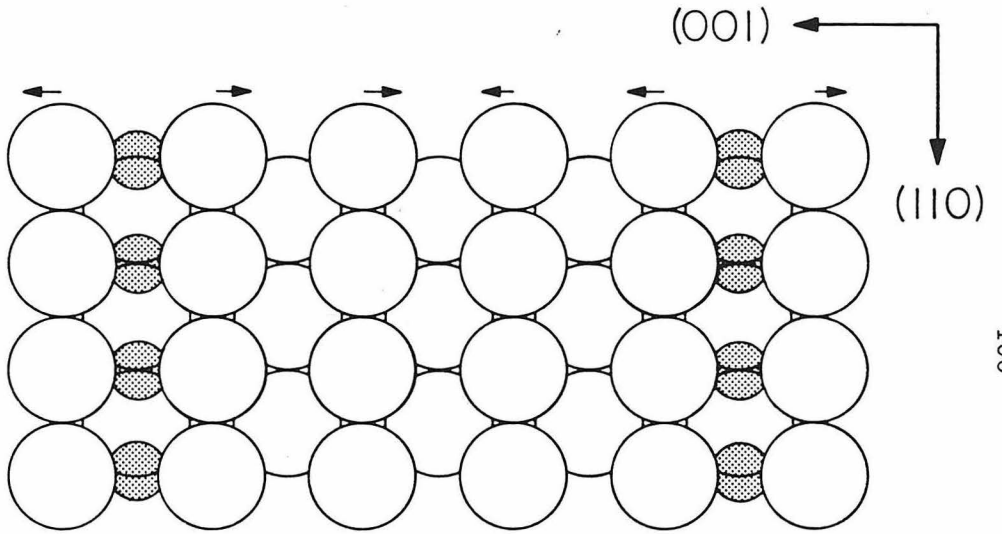
Figure 18: Topping plots for molecularly chemisorbed oxygen on clean and oxidized Ir(110) at 90 K. Dipole moments and polarizabilities shown here were calculated on a per atom basis.

Figure 19: Adsorption kinetics of oxygen on clean and oxidized Ir(110) at 90 K.

Figure 20: Auger KLL transitions of molecularly and dissociatively chemisorbed oxygen on clean Ir(110). The spectra for the respective overlayers were measured following oxygen exposures at 90 and 300 K. The coverage in each spectrum is 0.5 ML.

OXIDE (1×4) ON IR(110)

LEED PATTERN



STRUCTURE

160

Fig. 1

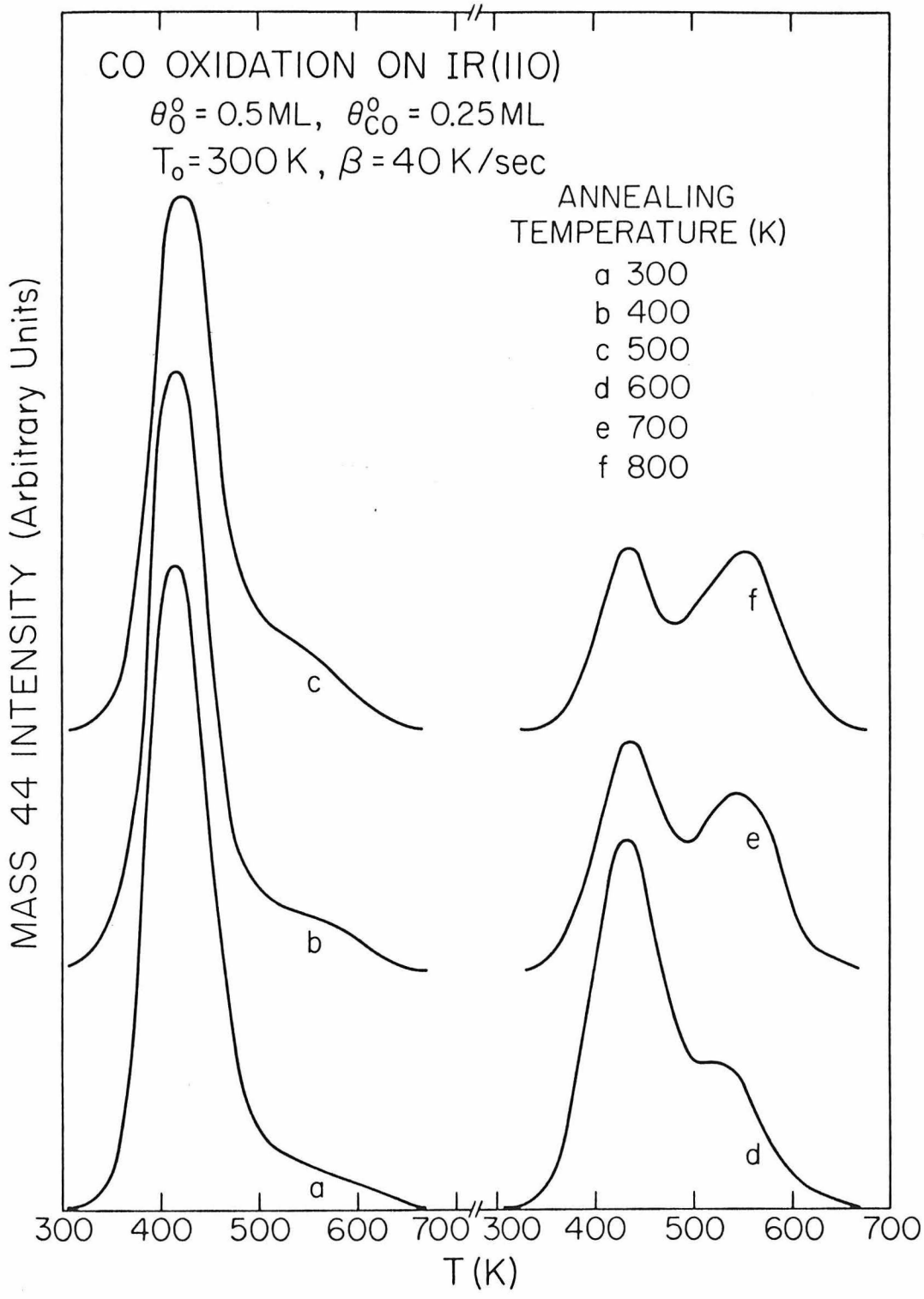

CO OXIDATION ON IR(110)
θ_O^o = 0.5 ML, θ_{CO}^o = 0.25 ML
T_o = 300 K, β = 40 K/sec
ANNEALING TEMPERATURE (K)
a 300
b 400
c 500
d 600
e 700
f 800
MASS 44 INTENSITY (Arbitrary Units)
T (K)
300 400 500 600 700
300 400 500 600 700

Fig. 2

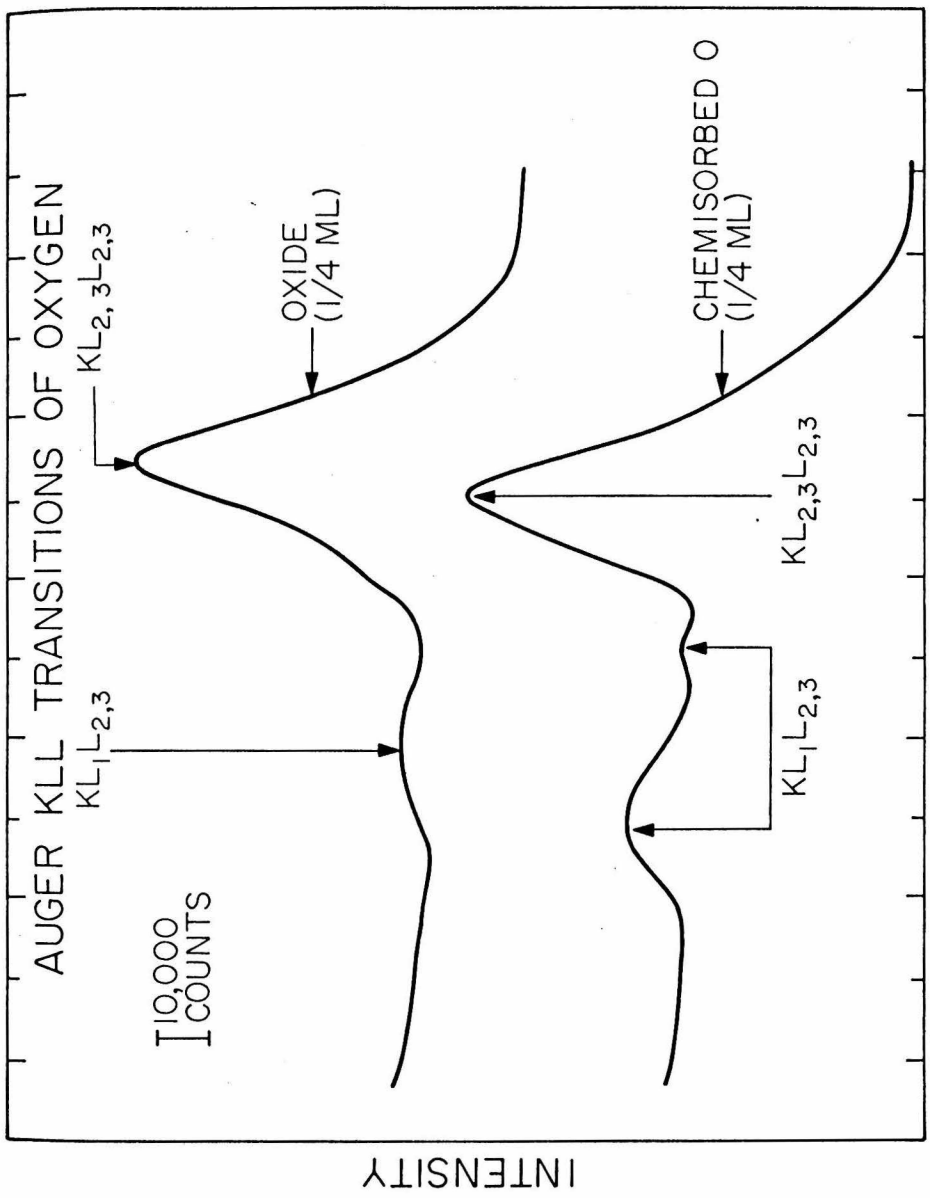
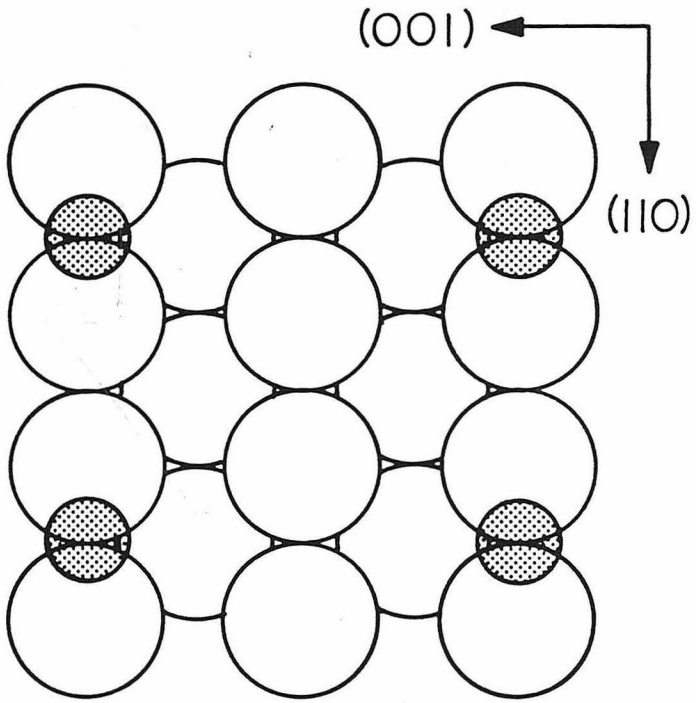

AUGER KLL TRANSITIONS OF OXYGEN
$KL_1L_{2,3}$
$KL_{2,3}L_{2,3}$
OXIDE (1/4 ML)
10,000 COUNTS
CHEMISORBED O (1/4 ML)
$KL_{2,3}L_{2,3}$
$KL_1L_{2,3}$
480
490
500
510
520
530
540
KINETIC ENERGY (eV)
INTENSITY

Fig.3

O-p(2×2) ON IR(110)

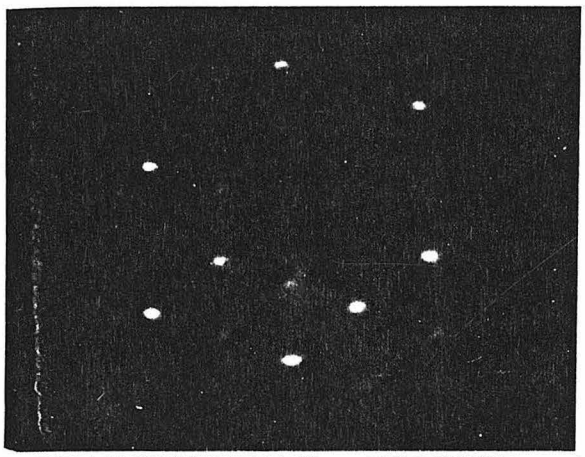
LEED STRUCTURE



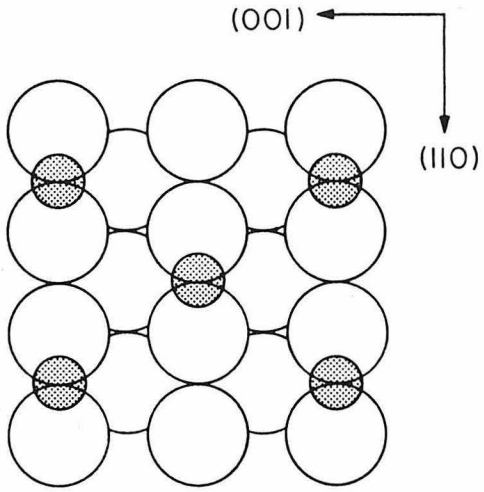
STRUCTURE

Fig. 4

a) O-c(2×2) ON IR(110) OXIDE



LEED PATTERN



STRUCTURE

b) O-(3×2) ON IR(110) OXIDE

LEED PATTERN

Fig. 5

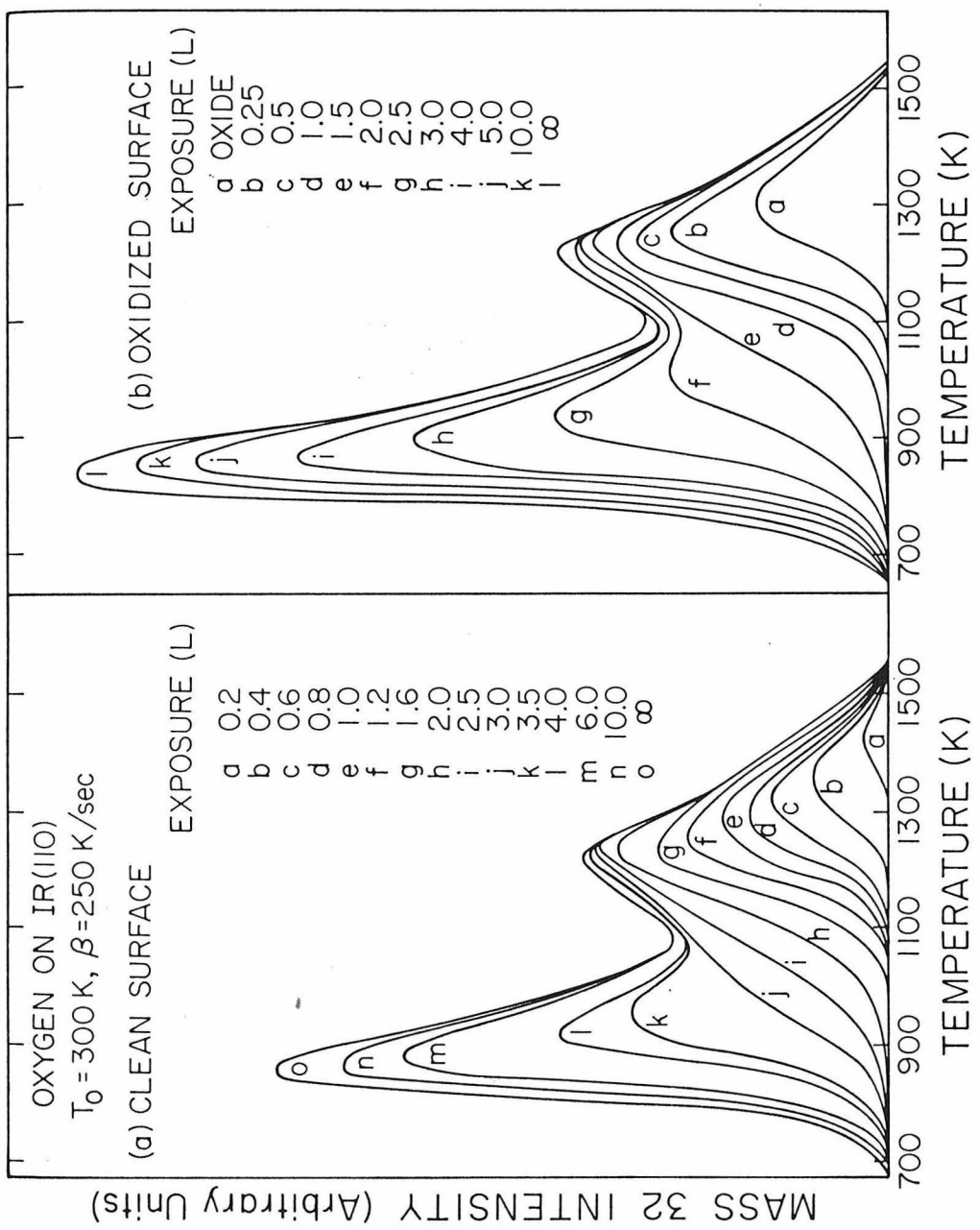

OXYGEN ON IR(110)
T_0 = 300 K, β = 250 K/sec
(a) CLEAN SURFACE
EXPOSURE (L)
a 0.2
b 0.4
c 0.6
d 0.8
e 1.0
f 1.2
g 1.6
h 2.0
i 2.5
j 3.0
k 3.5
l 4.0
m 6.0
n 10.0
o ∞
(b) OXIDIZED SURFACE
EXPOSURE (L)
a OXIDE
b 0.25
c 0.5
d 1.0
e 1.5
f 2.0
g 2.5
h 3.0
i 4.0
j 5.0
k 10.0
l ∞
MASS 32 INTENSITY (Arbitrary Units)
700
900
1100
1300
1500
TEMPERATURE (K)

Fig. 6

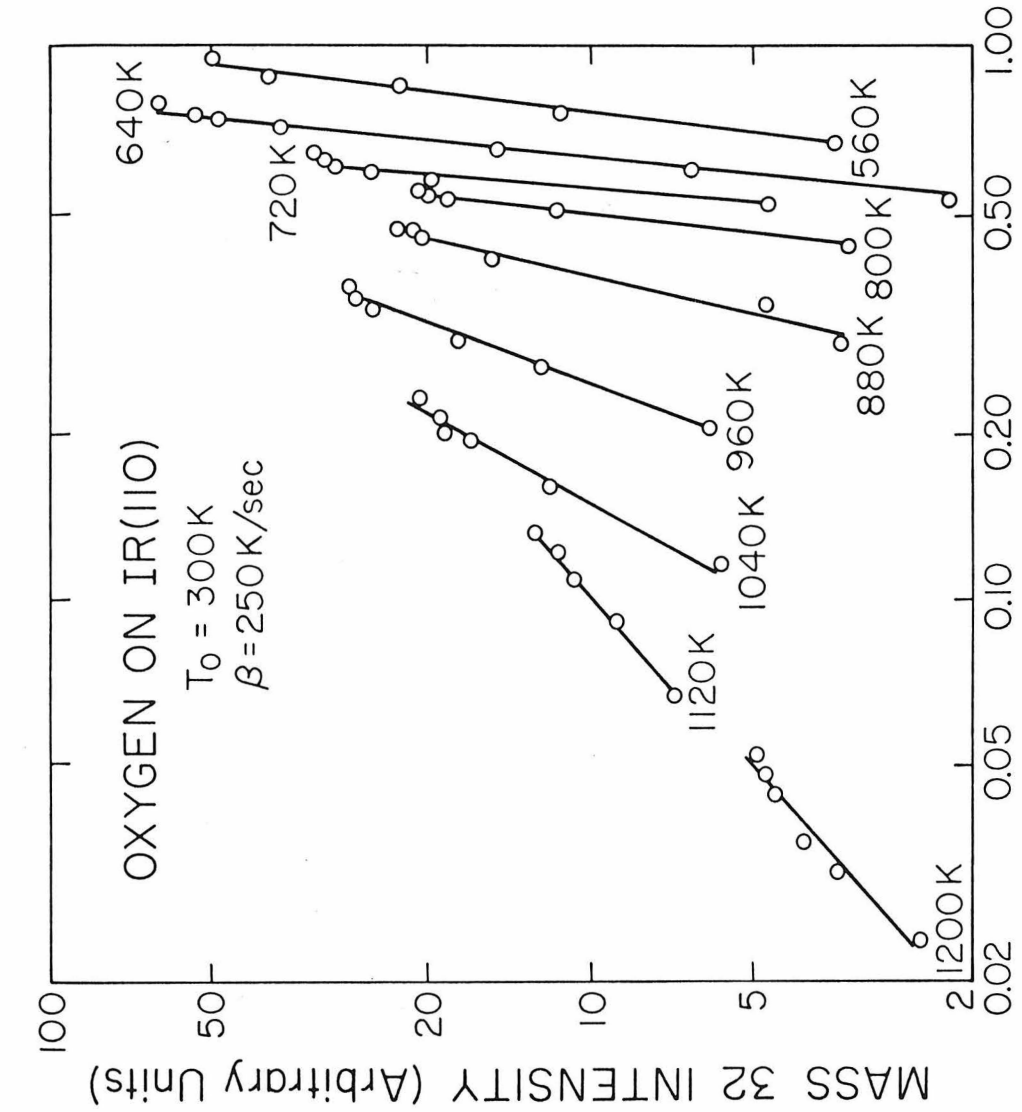

OXYGEN ON IR(110)
T_0 = 300K
β = 250K/sec
640K
720K
560K
800K
880K
960K
1040K
1120K
1200K
MASS 32 INTENSITY (Arbitrary Units)
FRACTIONAL COVERAGE
100
50
20
10
5
2
0.02
0.05
0.10
0.20
0.50
1.00

Fig.7

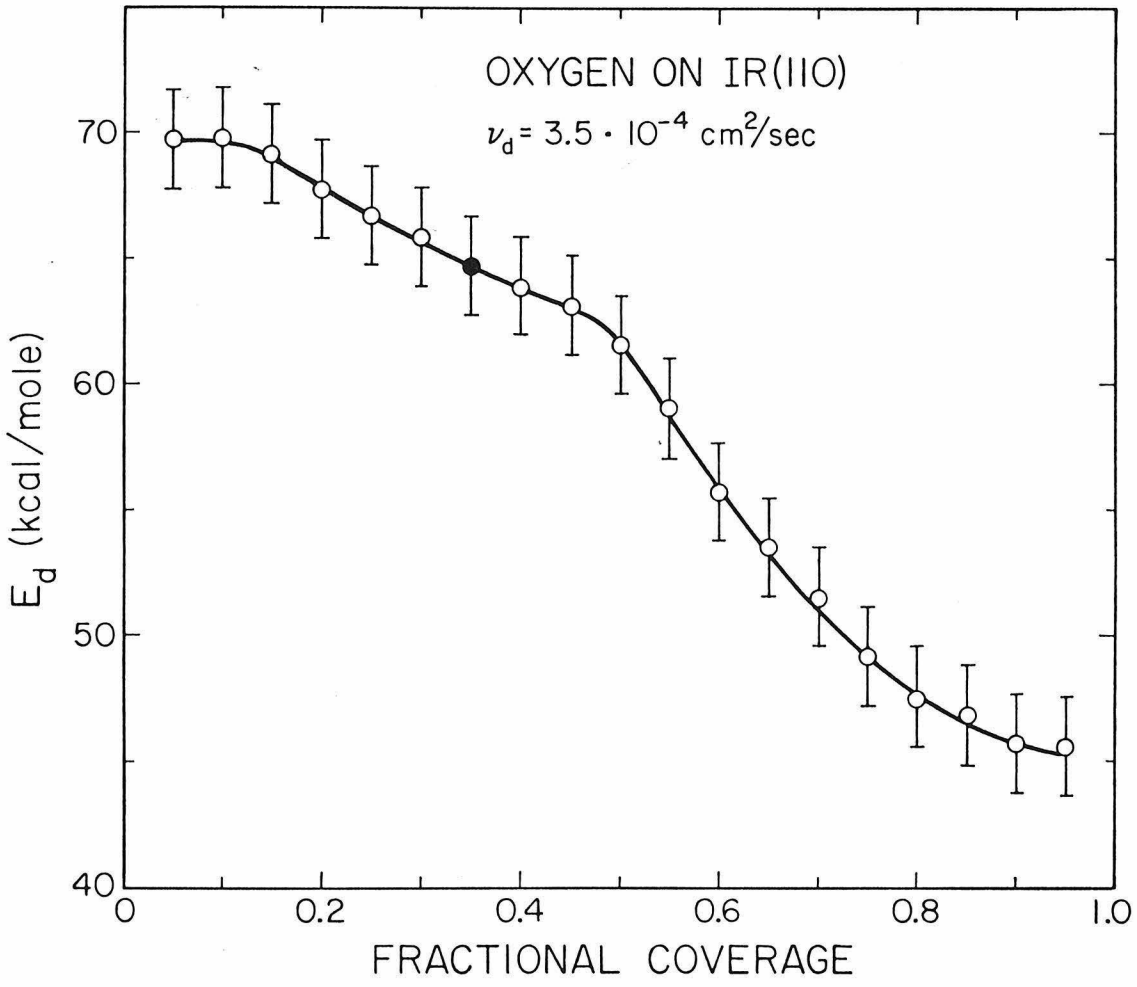

OXYGEN ON IR(110)
$\nu_d = 3.5 \cdot 10^{-4}$ cm^2/sec
E_d (kcal/mole)
70
60
50
40
0
0.2
0.4
0.6
0.8
1.0
FRACTIONAL COVERAGE

Fig. 8

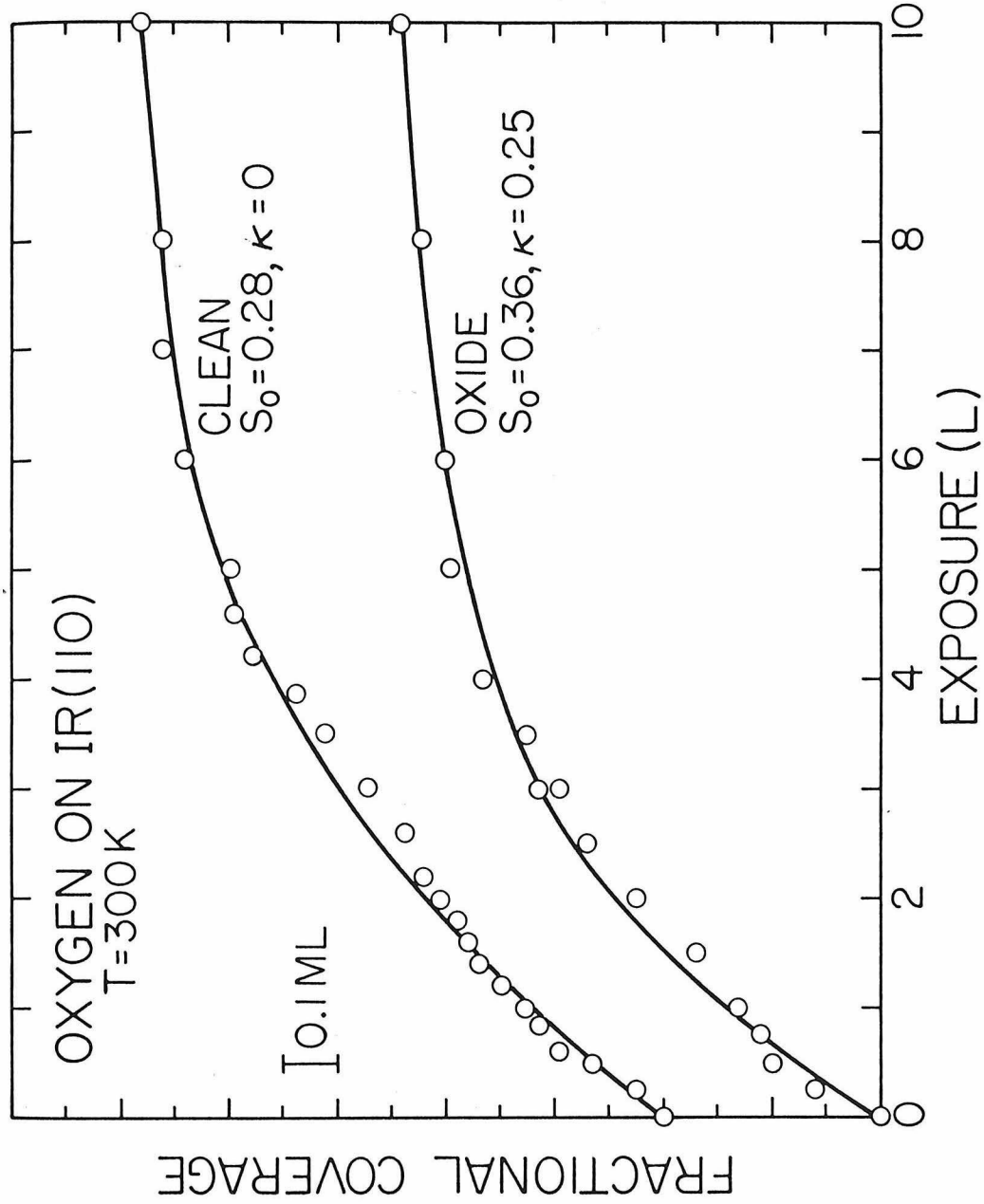

OXYGEN ON IR(110)
T=300K
0.1 ML
CLEAN
S_0=0.28, κ=0
OXIDE
S_0=0.36, κ=0.25
FRACTIONAL COVERAGE
EXPOSURE (L)
0
2
4
6
8
10

Fig. 9

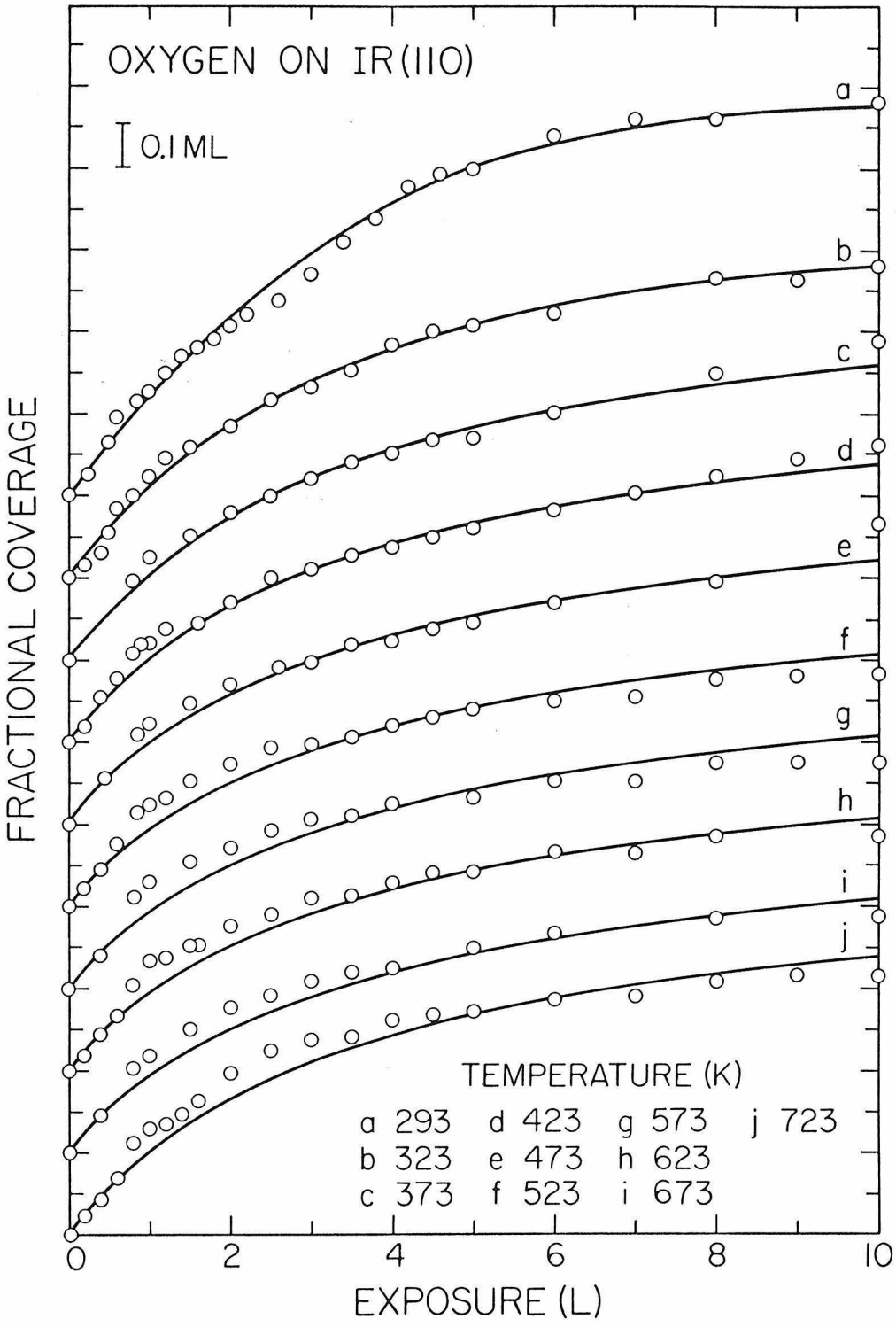

OXYGEN ON IR(110)
0.1 ML
FRACTIONAL COVERAGE
a
b
c
d
e
f
g
h
i
j
TEMPERATURE (K)
a 293 d 423 g 573 j 723
b 323 e 473 h 623
c 373 f 523 i 673
0
2
4
6
8
10
EXPOSURE (L)

Fig.10

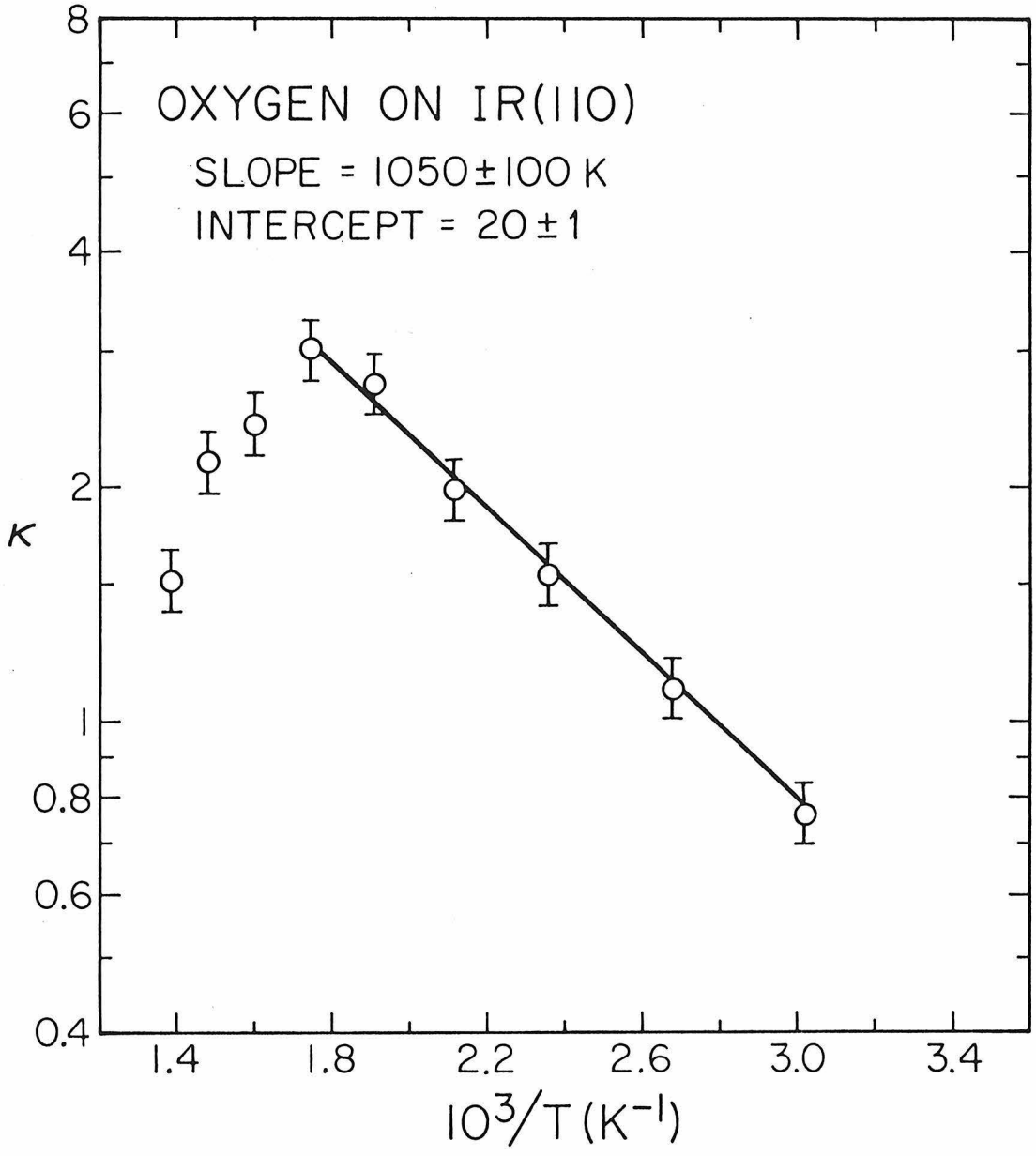

OXYGEN ON IR(110)
SLOPE = 1050±100 K
INTERCEPT = 20±1
κ
8
6
4
2
1
0.8
0.6
0.4
1.4
1.8
2.2
2.6
3.0
3.4
10³/T (K⁻¹)

Fig.11

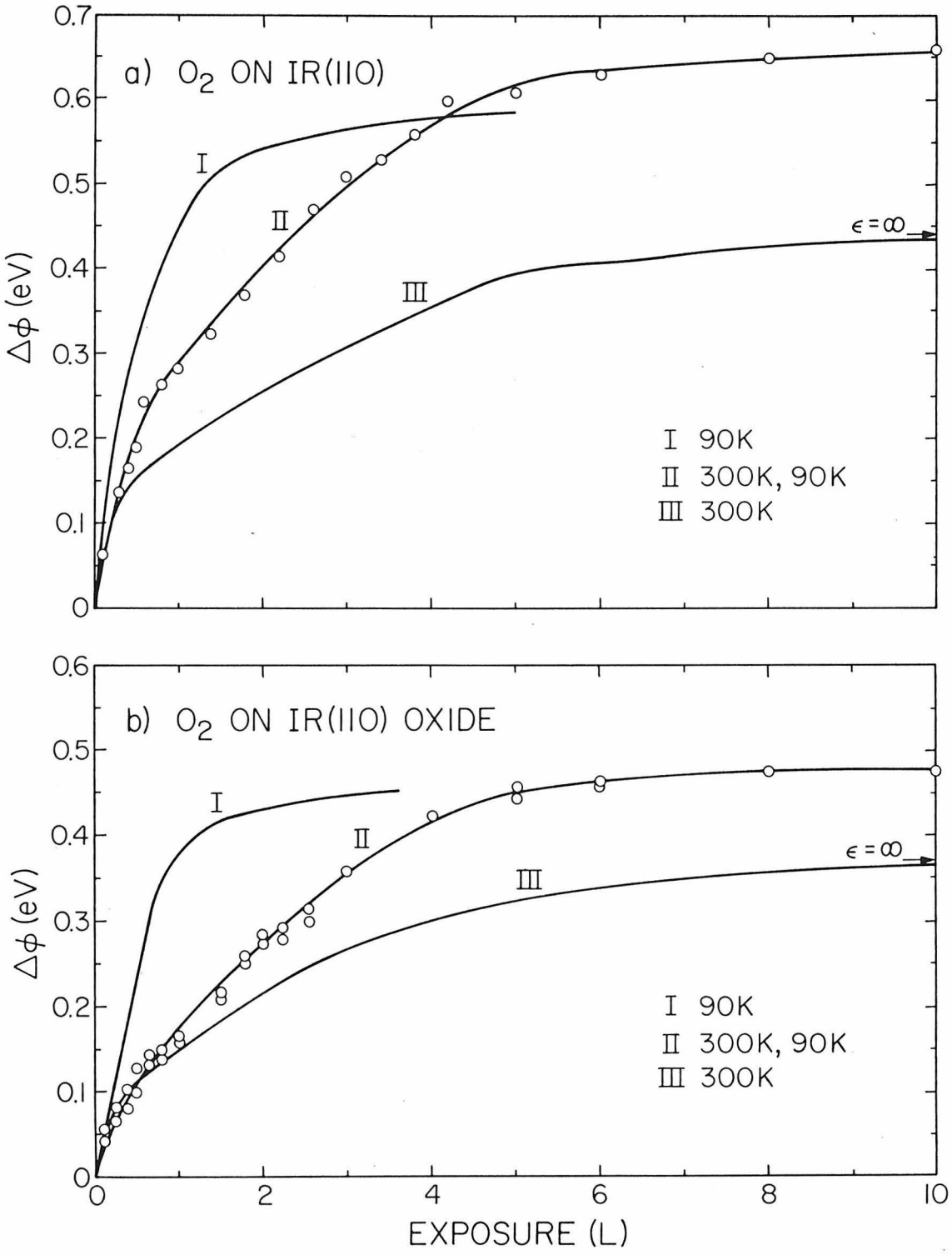

a) O_2 ON IR(110)
I
II
III
$\epsilon = \infty$
I 90K
II 300K, 90K
III 300K
$\Delta\phi$ (eV)
b) O_2 ON IR(110) OXIDE
I
II
III
$\epsilon = \infty$
I 90K
II 300K, 90K
III 300K
$\Delta\phi$ (eV)
EXPOSURE (L)

Fig. 12

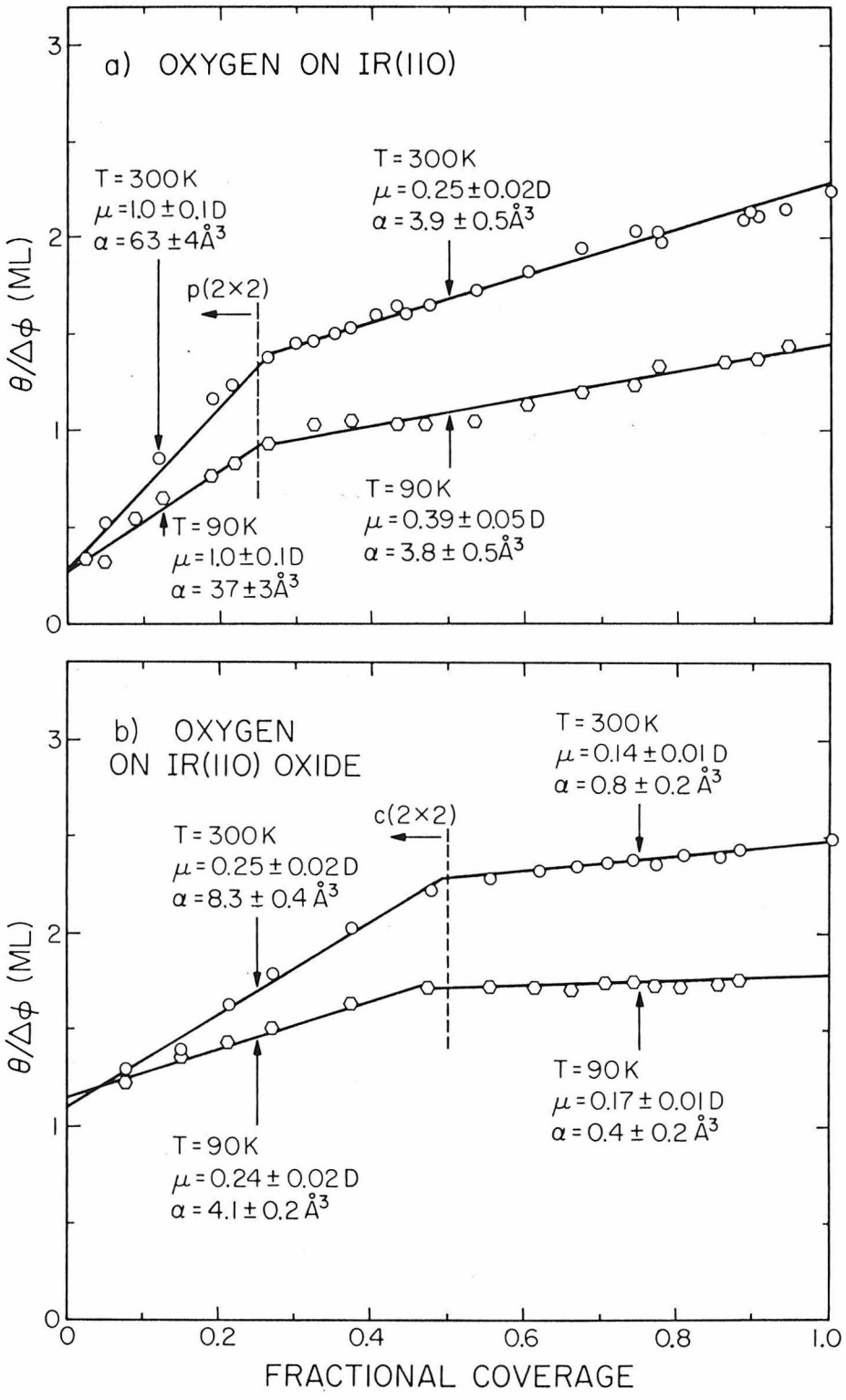

a) OXYGEN ON IR(110)
T = 300K
μ = 1.0 ± 0.1D
α = 63 ± 4Å³
p(2×2)
T = 300K
μ = 0.25 ± 0.02D
α = 3.9 ± 0.5Å³
T = 90K
μ = 1.0 ± 0.1D
α = 37 ± 3Å³
T = 90K
μ = 0.39 ± 0.05D
α = 3.8 ± 0.5Å³
b) OXYGEN
ON IR(110) OXIDE
T = 300K
μ = 0.14 ± 0.01D
α = 0.8 ± 0.2 Å³
c(2×2)
T = 300K
μ = 0.25 ± 0.02D
α = 8.3 ± 0.4 Å³
T = 90K
μ = 0.17 ± 0.01D
α = 0.4 ± 0.2 Å³
T = 90K
μ = 0.24 ± 0.02D
α = 4.1 ± 0.2 Å³
θ/Δφ (ML)
FRACTIONAL COVERAGE
0
1
2
3
0.2
0.4
0.6
0.8
1.0

Fig.13

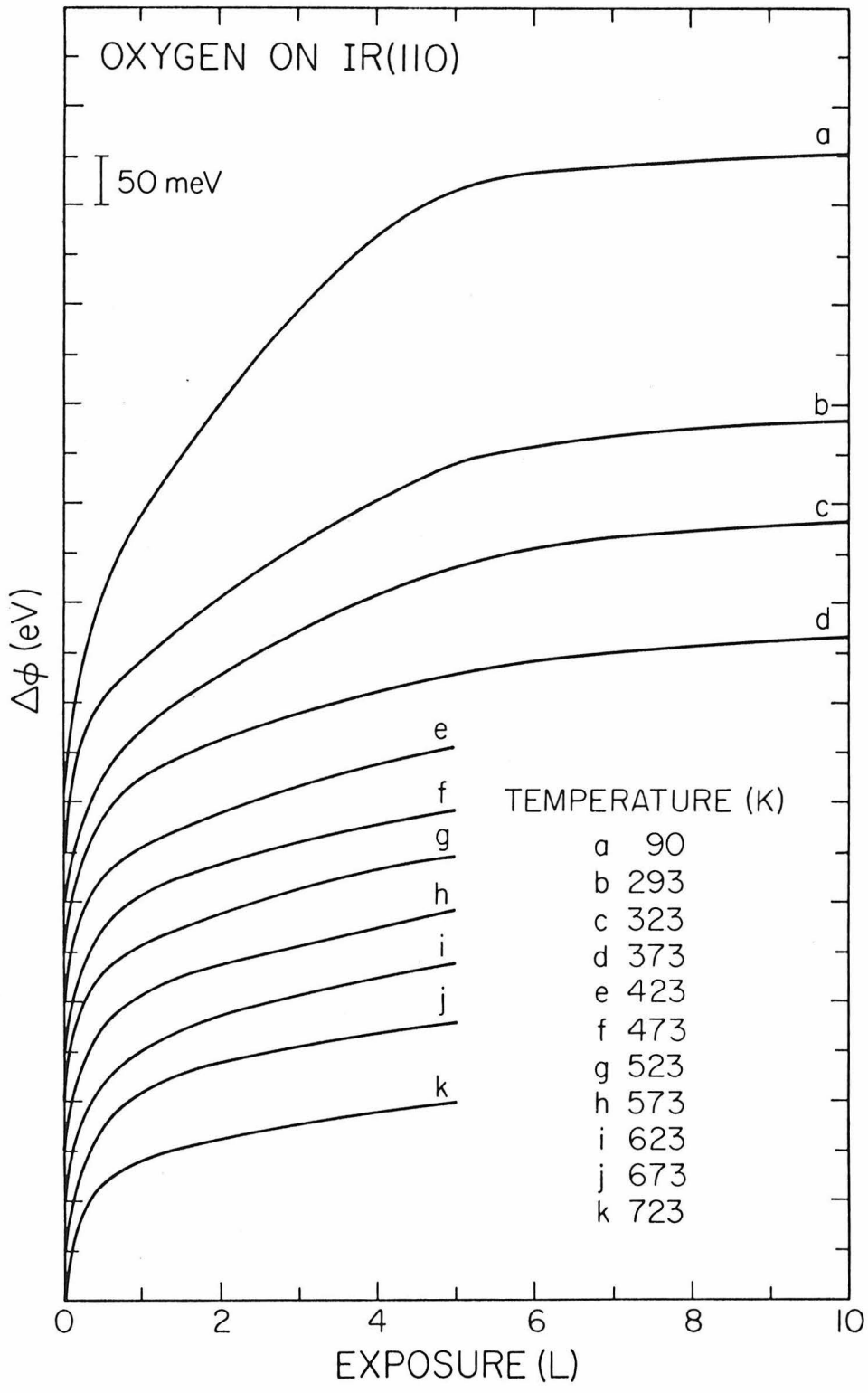

OXYGEN ON IR(110)
50 meV
a
b
c
d
e
f
g
h
i
j
k
TEMPERATURE (K)
a 90
b 293
c 323
d 373
e 423
f 473
g 523
h 573
i 623
j 673
k 723
Δφ (eV)
0
2
4
6
8
10
EXPOSURE (L)

Fig. 14

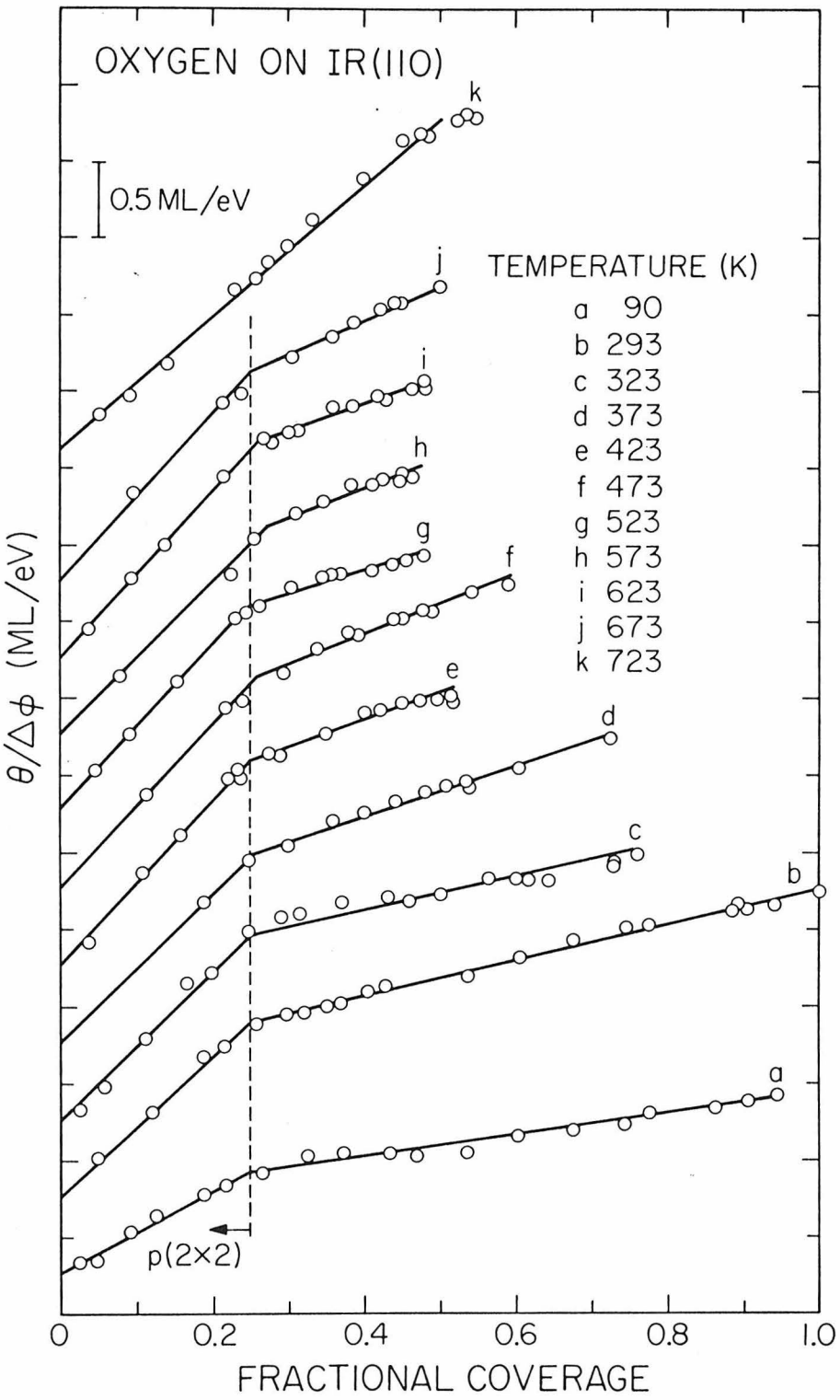

OXYGEN ON IR(110)
0.5 ML/eV
TEMPERATURE (K)
a 90
b 293
c 323
d 373
e 423
f 473
g 523
h 573
i 623
j 673
k 723
θ/Δφ (ML/eV)
p(2×2)
0
0.2
0.4
0.6
0.8
1.0
FRACTIONAL COVERAGE

Fig.15

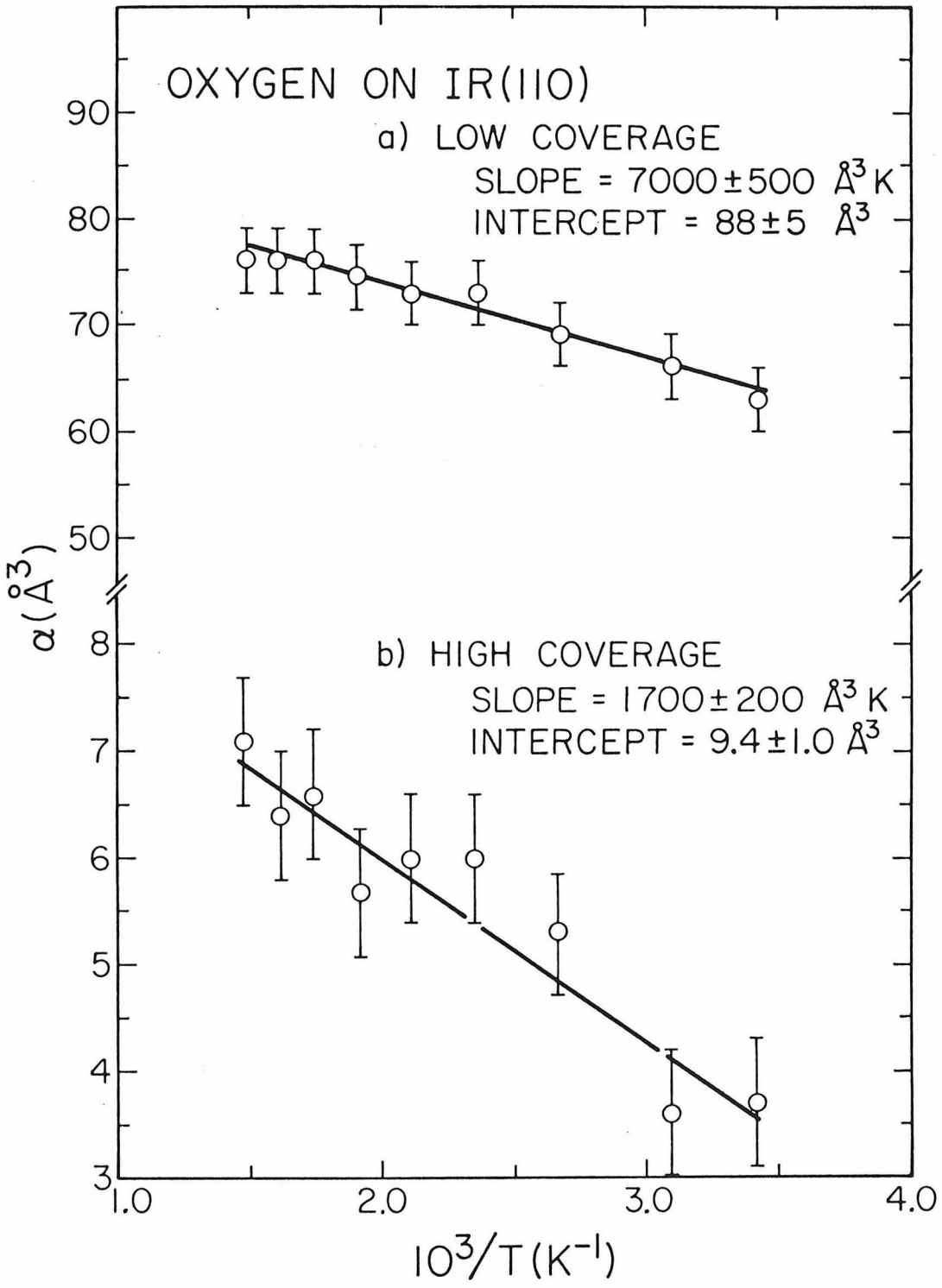

OXYGEN ON IR(110)
a) LOW COVERAGE
SLOPE = 7000±500 Å³ K
INTERCEPT = 88±5 Å³
b) HIGH COVERAGE
SLOPE = 1700±200 Å³ K
INTERCEPT = 9.4±1.0 Å³
α(Å³)
10³/T(K⁻¹)
90
80
70
60
50
8
7
6
5
4
3
1.0
2.0
3.0
4.0

Fig.16

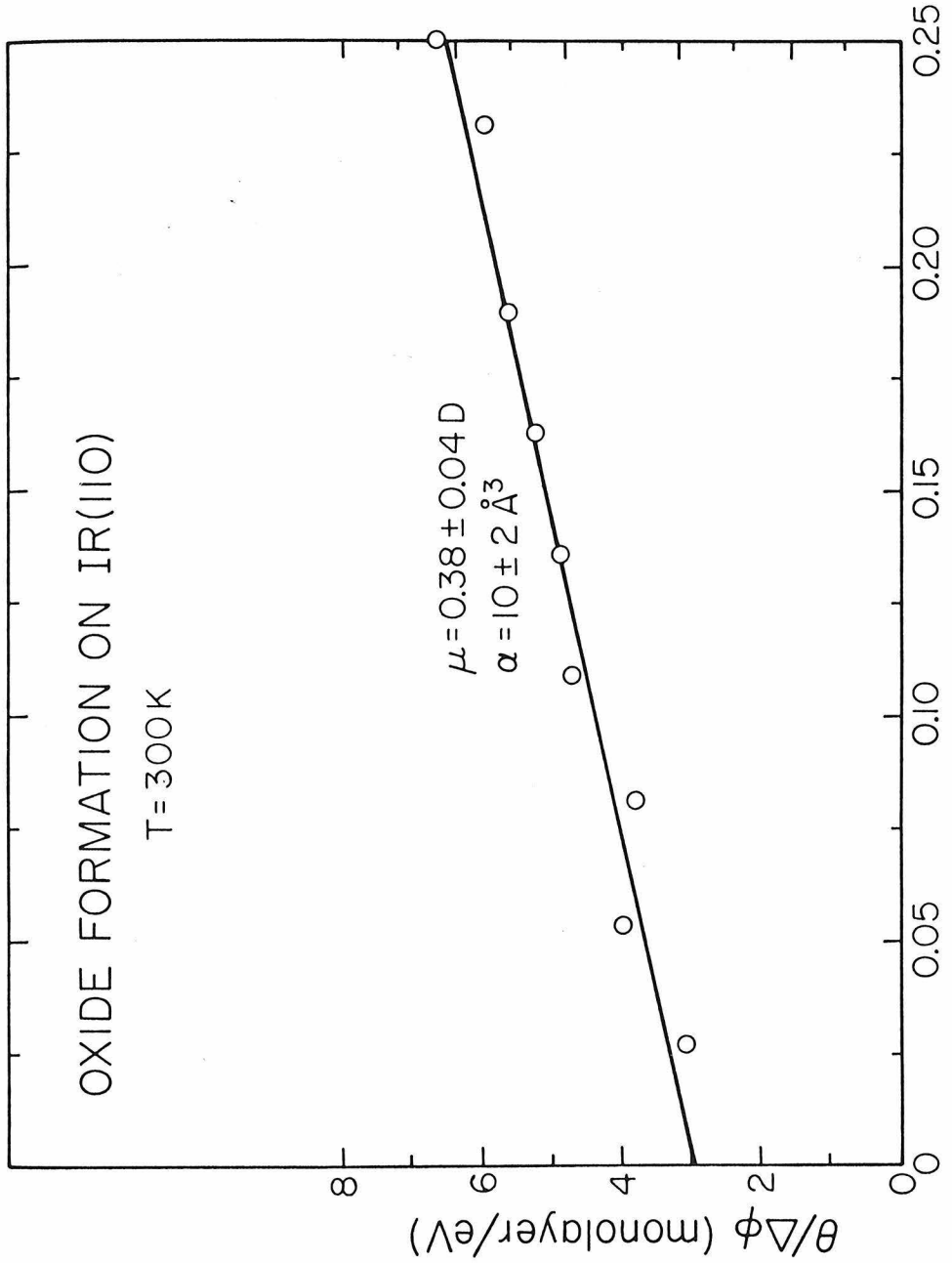

OXIDE FORMATION ON IR(110)
T = 300K
μ = 0.38 ± 0.04 D
α = 10 ± 2 Å³
θ/Δφ (monolayer/eV)
FRACTIONAL COVERAGE
0
2
4
6
8
0.05
0.10
0.15
0.20
0.25

Fig. 17

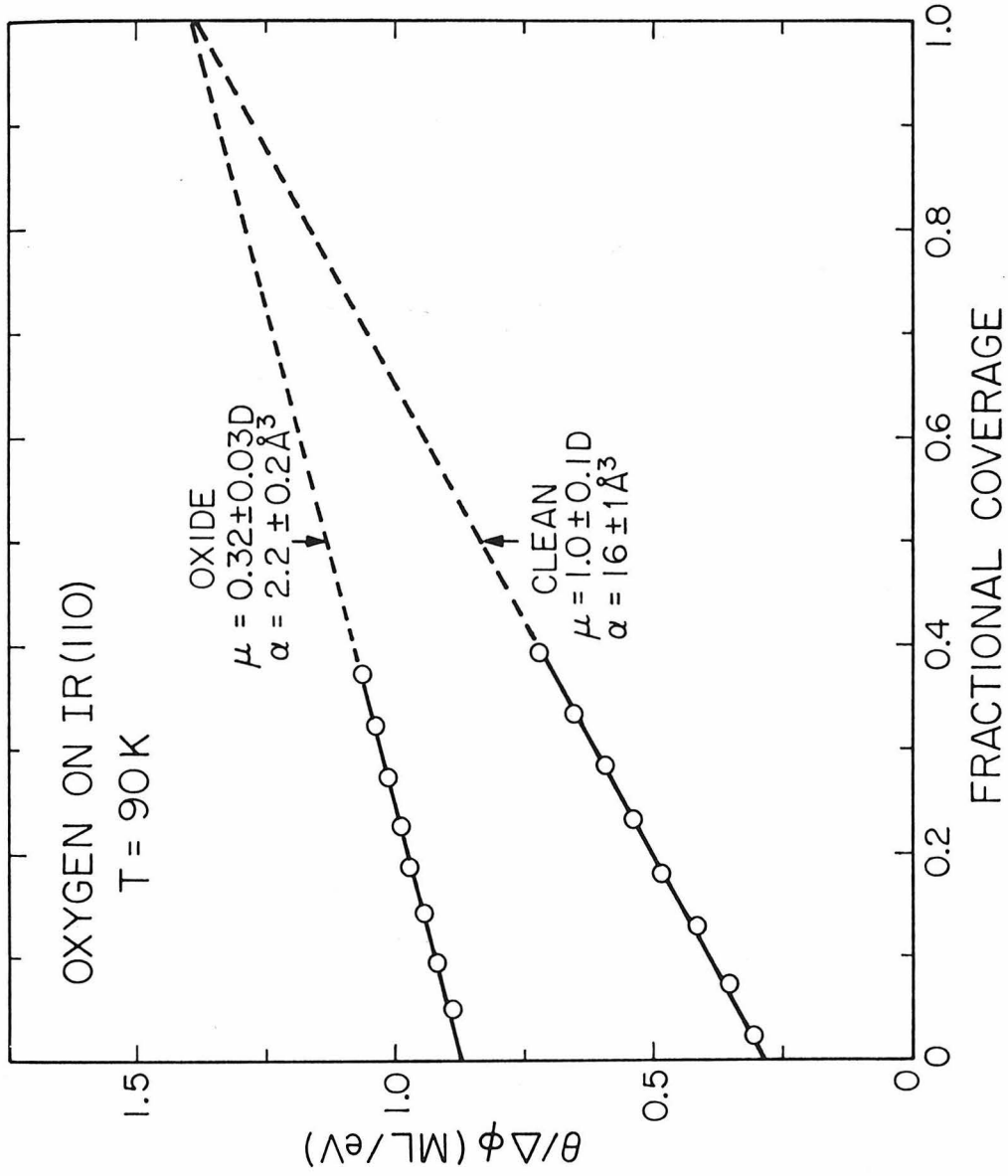

OXYGEN ON IR(110)
T = 90 K
OXIDE
μ = 0.32±0.03D
α = 2.2 ±0.2Å³
CLEAN
μ = 1.0 ± 0.1D
α = 16 ±1Å³
θ/Δφ (ML/eV)
FRACTIONAL COVERAGE
0
0.5
1.0
1.5
0.2
0.4
0.6
0.8
1.0

Fig.18

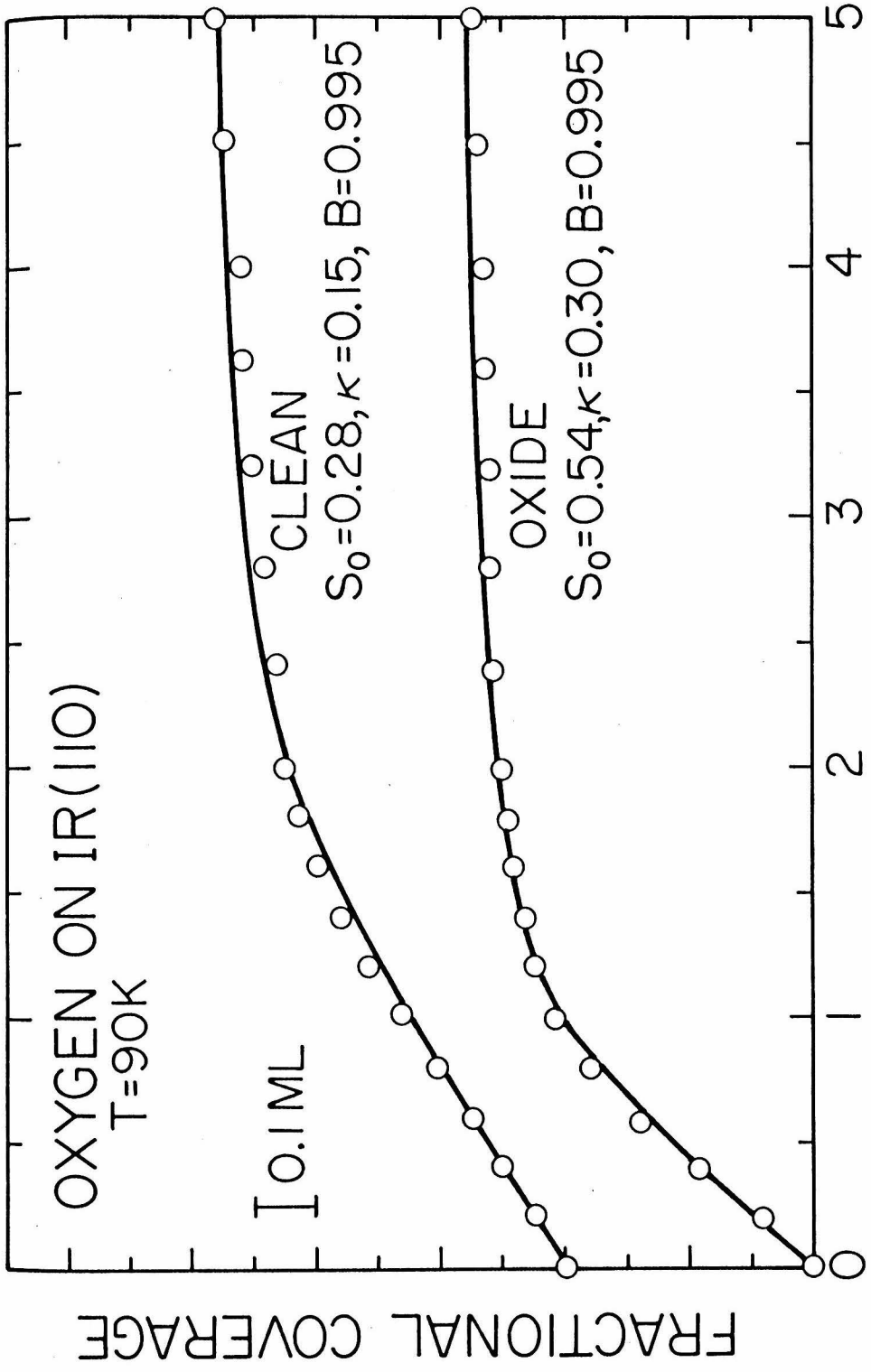

OXYGEN ON IR(110)
T=90K
0.1 ML
CLEAN
S_0=0.28, κ=0.15, B=0.995
OXIDE
S_0=0.54, κ=0.30, B=0.995
FRACTIONAL COVERAGE
EXPOSURE (L)
0
1
2
3
4
5

Fig. 19

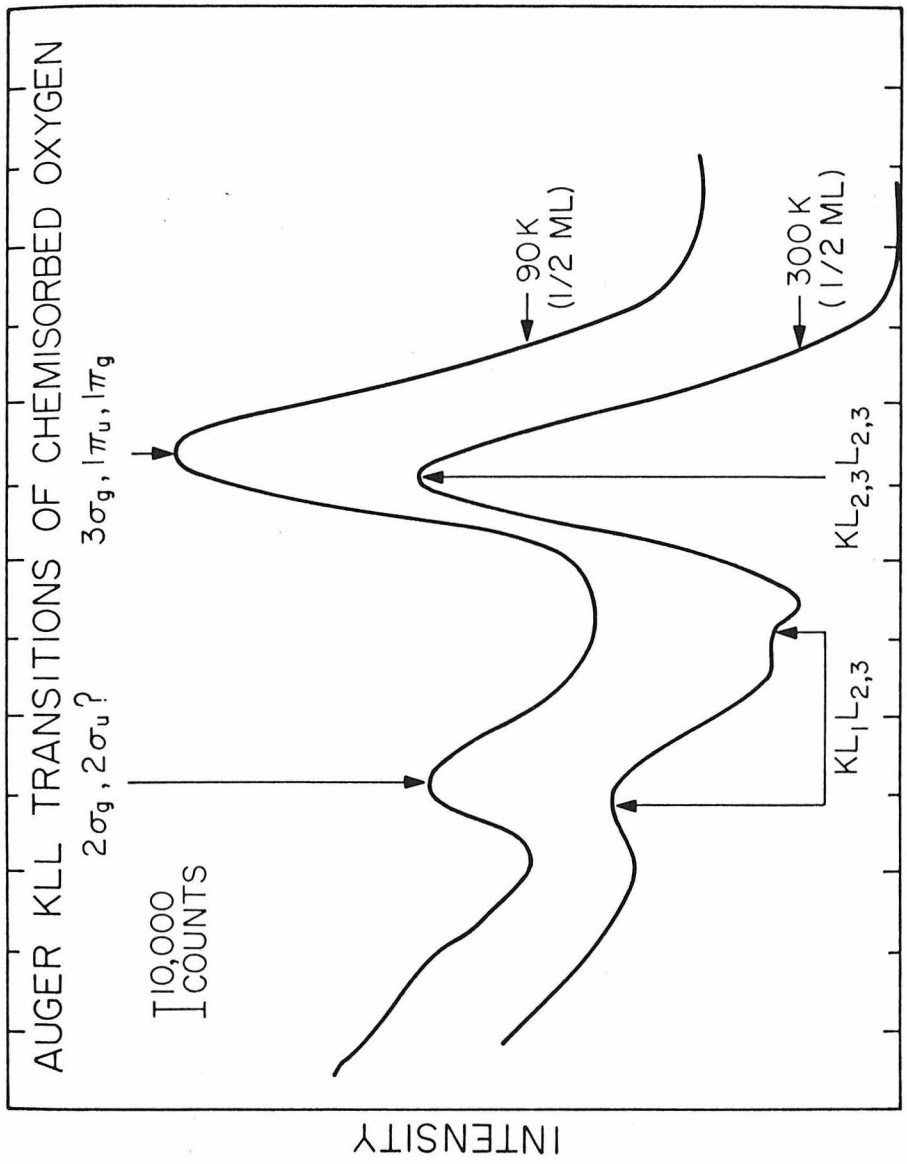

AUGER KLL TRANSITIONS OF CHEMISORBED OXYGEN
$2\sigma_g, 2\sigma_u$?
$3\sigma_g, 1\pi_u, 1\pi_g$
10,000 COUNTS
90 K (1/2 ML)
300 K (1/2 ML)
$KL_1L_{2,3}$
$KL_{2,3}L_{2,3}$
480
490
500
510
520
530
540
KINETIC ENERGY (eV)
INTENSITY

Fig. 20

CHAPTER 5

THE OXIDATION OF CARBON MONOXIDE OVER THE (110) SURFACE OF IRIDIUM

(The text of Chapter 5 consists of an article coauthorized with D. E. Ibbotson and W. H. Weinberg that has been submitted to Journal of Catalysis for publication.)

ABSTRACT

The heterogeneously catalyzed reaction of gaseous CO and O_2 to form CO_2 over Ir(110) has been studied through measurements of the equilibrium rate of CO_2 production and of the transient kinetics of the elementary reactions that may limit the equilibrium rate. Rate expressions for these elementary reactions--the desorption of CO, the oxidation of CO via the Langmuir-Hinshelwood mechanism, the adsorption of CO and the adsorption of oxygen--were developed using thermal desorption mass spectrometry. Mass spectrometry also was used to measure the equilibrium rate of CO_2 production for surface temperatures between 300 and 1000 K and for partial pressures of the reactants between 10^{-8} and 3×10^{-6} torr. The rate expressions developed from the measurements of transient kinetics were combined into a model that both qualitatively and quantitatively predicts trends in the equilibrium measurements. However, four phenomena limit the applicability of this model. (1) The activation energies for CO desorption and CO oxidation depend markedly upon the composition of the adlayer on the catalyst. (2) Diffusion in the adlayer may limit the rates of CO desorption and CO oxidation, but may be included empirically in the rate expressions for these processes. (3) The catalyst surface is oxidized at temperatures where the equilibrium rate of CO oxidation is rapid; that is, a second species of atomically adsorbed oxygen forms irreversibly near 700 K. (4) Hysteresis occurs in the "equilibrium" measurements of CO_2 production.

I. Introduction

The reaction of gaseous CO and O_2 to form CO_2 over platinum metal catalysts has been studied extensively to gain a fundamental insight into heterogeneous catalysis. Because of the relative simplicity of this reaction, investigators have been able to isolate individual elementary reactions in the chemical mechanism for CO oxidation and to describe the influence of surface phenomena on the chemical rate. Two objectives, which supplement those of earlier investigations, are pursued in this experimental study of CO oxidation over Ir(110). The first is to determine how the coverages of various adsorbed species influence the rates of individual elementary reactions that may limit the equilibrium rate of CO_2 production. The second goal is to model the dependence of the equilibrium rate upon these elementary reactions under various conditions. The kinetics of the elementary reactions were measured using thermal desorption mass spectrometry under ultra-high vacuum conditions. In addition, the mass spectrometer was used to monitor the equilibrium rates of CO_2 production. In these studies, the partial pressures of the reactants, CO and O_2, were varied between 10^{-8} and 3×10^{-6} torr, and the temperature of the catalyst was varied between 300 and 1000 K. To determine how ordering phenomena influence the oxidation rate, the structure of the chemisorbed layer also was investigated with LEED. The results of this study of CO oxidation over Ir(110) are discussed in three sections that focus on the following: (1) the dependence of the rate expressions for CO desorption and CO oxidation upon the composition of the adlayer, (2) the competitive adsorption of CO and oxygen, and (3) the interpretation of equilibrium oxidation

rates based on the kinetic expressions for the elementary reactions that are developed in the first two sections.

Equilibrium rates of CO_2 production at low pressures ($<10^{-5}$ torr) have been measured as a function of surface temperature for several catalysts made from a pure platinum metal (1 - 7). The rate of production is observed to increase rapidly with temperature until a maximum is reached near 600 K and then to decrease slowly at higher temperatures. In the regime where the rate increases with temperature, the desorption of chemisorbed CO, and the subsequent competitive adsorption of oxygen and CO have been thought to be the rate-limiting since Langmuir first described this process (1). However, the elementary reaction that depicts the combination of oxygen and CO to form CO_2 has been the subject of controversy until recently (6). Although other mechanisms have been suggested (8,9), the reaction of chemisorbed oxygen atoms with either chemisorbed or physisorbed CO molecules is thought to be the most probable path for CO oxidation (1). For the first case, termed the Langmuir-Hinshelwood (L-H) reaction, the rate of oxidation is proportional to the concentration of chemisorbed CO molecules; for the second case, termed the Eley-Rideal (E-R) reaction, the rate is proportional to the partial pressure of CO. In either instance, the rate is proportional to the coverage of chemisorbed oxygen atoms. At first, the dominant path for CO oxidation was thought to be the E-R reaction (1). The decrease in the equilibrium rate of CO_2 production with increasing surface temperature was attributed to a decrease in the coverage of oxygen (2 - 4). Early studies with modulated molecular beams (10,11) demonstrated that the decrease in the rate with temperature above 600 K

could be explained only by the L-H reaction, for the coverage of CO, rather than oxygen, decreased with increasing temperature. A subsequent study with molecular beams (6,12) showed that CO oxidation on Pd(111) occurs through the L-H mechanism for surface temperatures below 600 K. Hence, the adsorption of oxygen and CO, the desorption of CO, and the L-H reaction to form CO_2 are the four elementary reactions that may limit the rate of CO oxidation over the platinum metal catalysts. For surface temperatures between 300 and 1000 K, where the equilibrium rate is greatest at low pressures, the E-R reaction and the desorption of oxygen and CO_2 are thought to affect the rate to second order at most (1,6).

Several surface phenomena complicate the interpretation of the equilibrium rate data for CO oxidation. Although the overall oxidation rate on Pd is insensitive to the geometric arrangement of the metal atoms at the catalyst surface, LEED studies indicate that the structure of the chemisorbed layer influences the rate (2,6,13). That is, the rate may be limited by the surface diffusion of CO molecules between islands of oxygen atoms. Oxidation of the catalyst surface, the thermally activated process of forming a second chemical state of adsorbed oxygen atoms, markedly changes the equilibrium rate with respect to the clean surface (5,6,13,14). Moreover, the activation energies for the L-H reaction and CO desorption are quite sensitive to the composition of the adlayer and may vary by two-fold on Pd(111) (6). Hence, the variation in the oxidation rate resulting from the coverage dependence of an activation energy may overshadow completely the coverage variation resulting from the leading coefficients in rate expressions that enter as concentrations to integral exponents. Although many of these phenomena are reported in

this work, CO oxidation on Ir(110) can be modeled adequately by a single, simple chemical mechanism.

II. Experimental

The experiments were conducted in a stainless steel belljar that was evacuated continuously by an ion pump. The base pressure was less than 2×10^{-10} torr. Thermal desorption spectra and partial pressures in the equilibrium reactions were measured with a quadrapole mass spectrometer (UTI 100-C). The order of the overlayer under static conditions and during the equilibrium reactions was detected by LEED. The substrate, which was held on a precision manipulator by two 10 mil Ta wires, could be cooled with liquid nitrogen or heated conductively by passing a current through the Ta wires. The temperature of the substrate was measured with a Pt-Pt10%Rh thermocouple. The intensity from the mass spectrometer, as well as the EMF from the thermocouple, were read by a PDP-11/10 computer (15). Although these data were recorded digitally, they are presented in analog form for convenience.

The catalyst was cut from a single crystal of Ir and was polished within 1° of the (110) orientation. Carbon was cleaned from the surface through a series of oxidation and reduction cycles whereas oxygen was removed by annealing the substrate to 1800 K. Prior to each experiment, the surface was free from impurities within 0.01 ML (where 1 ML = 10^{15} cm^{-2}), as detected by Auger spectroscopy. The gases used in this study, CO and O_2, were research grade (99.99% pure).

Although CO does not dissociate on Ir(110) below 1300 K (16), the measurement of equilibrium rates was complicated by the formation of a

surface oxide at surface temperatures where the rate is rapid (17). The oxide, which is distinct chemically from chemisorbed oxygen, is less reactive to reduction by CO. When a surface that is covered with chemisorbed oxygen is heated to 700 K, the oxide forms irreversibly with a saturation coverage of 0.25 ± 0.05 ML. By annealing a saturated oxide layer at 1200 K, the oxide orders to give a p(1x4) LEED pattern. The chemical properties for an ordered and a disordered oxide layer are identical. In this study, the oxide layer was formed routinely by annealing a surface covered with 0.25 ML of chemisorbed oxygen to 1100 K. This coverage was obtained after a clean surface of Ir(110) at 300 K was exposed to 0.8 L (1L = 10^{-6} torr-sec) of oxygen. The oxide was decomposed by annealing the surface to 1800 K. Since the presence of the oxide layer is known to affect the chemisorption of CO (16) and oxygen (17) on Ir(110), many experiments in this study were performed on clean and oxidized surfaces of Ir(110). In this article, oxidized Ir(110) will refer to a surface covered with a saturated oxide layer, whereas clean Ir(110) will refer to a surface having no oxide layer.

Several thermal desorption spectra of CO, O_2 and CO_2 were obtained by flashing co-adsorbed layers of CO and oxygen from clean and oxidized Ir(110). Activation energies for CO desorption and CO oxidation were determined from a detailed analysis of the desorption spectra (18), which is discussed in the next section. The co-adsorbed layers were formed by first exposing the surface to either O_2 or CO and by subsequently exposing the surface to the other gas. During the exposures, the surface temperature was 300 K. For the data reported in this study, the adsorption of either gas was irreversible; that is, the measured rates of CO desorption and oxidation

were insignificant over the time scale of these experiments, which was about 1 min. Hence, the adsorption kinetics of oxygen (CO) on a surface partially covered with CO (oxygen) could be measured since the coverage of the second component in the exposure sequence could be determined from the thermal desorption spectra. This procedure is detailed below. The coverage of the first component in the exposure sequence was fixed from separate measurements of the adsorption kinetics of CO (16) and oxygen (17) on clean and oxidized Ir(110). The spectrum for each desorption product in any sequence of exposures was obtained in a separate flash. This introduced no problems in analyzing the data since the heating shcedules for any set of flashes were nearly invariant. Hence, the spectra for each exposure sequence could be analyzed accurately by aligning the spectra at points of constant temperature.

The coverage of the second component introduced in an exposure sequence was determined as follows. Chemisorbed CO may desorb as CO or CO_2 whereas oxygen may desorb as O_2 or CO_2. Neither oxygen atoms nor iridium oxides desorb from Ir(110) (17). Since the average heating rate in these experiments, 40 K/sec, was sufficiently small (18), the instantaneous desorption rate for any component was proportional to the intensity of its signal from the mass spectrometer (I). For CO, O_2 and CO_2, the proportionality constants are η_{28}, η_{32} and η_{44} where the subscript refers to the mass of the desorbing species. Hence, the coverage of oxygen that desorbs as CO_2 $[\theta_0^o(CO_2)]$ is obtained by integrating the relation (18,19)

$$-\frac{d\theta_0(CO_2)}{dt} = \eta_{44} I_{44} \qquad (1)$$

where t is time. That is,

$$\theta_0^o(CO_2) = \eta_{44} \int_0^\infty I_{44} dt \qquad (2)$$

Because the intensity (I) and the surface temperature (T) were measured as independent functions of time, this expression does not depend implicitly on the heating schedule (the temporal variation of the surface temperatures). Expressions similar to Eq. (2) may be derived for the coverage of oxygen that desorbs as O_2, the coverage of CO that desorbs as CO, and the coverage of CO that desorbs as CO_2. In evaluating the coverage of CO that desorbs as CO, 12% of the I_{44} signal must be subtracted from the I_{28} signal since the mass spectrometer ionizes CO_2 into CO^+, as well as CO_2^+. The value of 12% was obtained directly from the cracking pattern of CO_2 measured in this work. Hence, the initial coverage of CO equals the sum of the fractions that desorb as CO and CO_2,

$$\theta_{CO}^o = \eta_{28} \int_0^\infty (I_{28} - 0.12 I_{44}) dt + \eta_{44} \int_0^\infty I_{44} dt \qquad (3)$$

whereas the initial coverage of oxygen equals the sum of the fractions that desorb as O_2 and CO_2,

$$\theta_0^o = \eta_{32} \int_0^\infty I_{32} dt + \eta_{44} \int_0^\infty I_{44} dt \qquad (4)$$

All coverages of adsorbed oxygen are reported here in units of monolayers where 1 ML = 10^{15} atoms/cm^2; all CO coverages are reported in ML also, but 1 ML = 10^{15} molecules/cm^2. That is, oxygen chemisorption is

dissociative on Ir(110) above 200 K (17), but CO chemisorption is molecular (16). At any instant in a desorption spectra, the coverage of CO is given by (18)

$$\theta_{CO}(t) = \theta^{o}_{CO} - \eta_{28} \int_0^t (I_{28} - 0.12I_{44})dt' - \eta_{44} \int_0^t I_{44}dt' \qquad (5)$$

A coverage is related to a temperature through the heating schedule, which was measured experimentally. The temporal variation of the coverage of oxygen is given by

$$\theta_{O}(t) = \theta^{o}_{O} - \eta_{32} \int_0^t I_{32}dt' - \eta_{44} \int_0^t I_{44}dt' \qquad (6)$$

If η_{28}, η_{32} and η_{44} are evaluated as follows, Eqs. (3 - 6) may be solved directly to fix the coverage of the second component introduced in the exposure sequence. Because the separate coverage-exposure relations for CO (16) and oxygen (17) on Ir(110) at 300 K are known in terms of absolute coverage, η_{28} and η_{32} were determined from the desorption spectra of the single component systems. That is, η_{28} and η_{32} are the only unknowns in expressions similar to Eq. (2). If 0.25 ML of oxygen is adsorbed on clean Ir(110) at 300 K, and then the overlayer is exposed to 2 L CO, all the chemisorbed oxygen desorbs as CO_2 so η_{44} may be fixed from Eq. (2). These experimentally determined values for η were stable over several hours as long as the oxygen exposures were less than 10 L. Larger oxygen exposures induced drift in the gain of the mass spectrometer (17). Quite consistently, η_{44} equaled $1.3\eta_{28}$, and η_{32} equaled $3.4\eta_{28}$.

To summarize the method of analysis described above, it is applied now to a specific example. A clean Ir(110) surface at 300 K was exposed first to 2.7 L (1 L = 10^{-6} torr-sec) of O_2, which corresponds to a coverage of 0.50 ML (17), and then to 2 L of CO. The desorption spectra of the individual products--CO, O_2 and CO_2--were obtained in separate flashes. The spectra of CO and CO_2 are shown in Fig. 1; the O_2 spectrum, which appears above 1000 K, is not shown. To account for the fraction of CO_2 that is ionized as CO^+, 12% of the intensity of the CO_2 spectrum has been subtracted from the intensity of the CO spectrum at points of constant time, or equivalently, temperature. Also, the CO_2 spectrum has been multiplied by 1.3 so that the desorption rates for the products CO and CO_2 are directly comparable. These two modifications have been performed on all CO and CO_2 spectra shown subsequently. Also, I^*_{28} will be used in place of I_{28} and will refer to the intensity in a CO spectrum where 12% of the I_{44} signal has been substracted from the I_{28} signal. The initial coverage of CO was obtained from Eq. (3) and was equal to 0.41 ML. By solving Eq. (4), the value for the initial coverage of chemisorbed oxygen, 0.50 ML, was verified so that adsorption was irreversible under these conditions. The temporal variations in the coverages of CO and oxygen, used for determining the activation energies for CO desorption and CO oxidation (18), were determined from Eqs. (5) and (6), and are shown in Fig. 1.

The mass spectrometer also was used to monitor steady-state rates of CO_2 production as a function of the catalyst temperature and of the composition of the reactant gas. The total pressure of the reactants was varied between 10^{-7} and 3×10^{-6} torr whereas the partial pressure of

each reactant was varied between 10^{-8} and 3×10^{-6} torr. Although the rates for introducing the reactants into the belljar were controlled accurately by two leak valves, the pumping speed of the ion pump was unstable in time at these pressures. This instability made the measurement of equilibrium rates difficult. Hence, the poppet valve separating the pump from the spectrometer and the catalyst was closed sufficiently that the pumping speed was fixed by the flux of gas through the orifice of the valve. Because the total pressure on the sample side of the valve was at least 100 times larger than the pressure on the pump side, the pumping speed was quite stable. The lifetime of a gaseous CO molecule in the belljar (τ_{CO}), which is inversely proportional to its pumping speed, was determined by filling the sample side with 10^{-6} torr of CO and then closing the leak valve quickly. The exponential decay of the pressure with time, monitored with a mass spectrometer, gives the lifetime. For molecular flow through an orifice, the relative pumping speed for any gas is inversely proportional to the square root of its molecular weight so that $\tau_{CO_2} = \sqrt{44/28}\,\tau_{CO}$. Turnover numbers for CO_2 production (R_r in molecules/cm^2 sec) were determined from the expression

$$R_r = \Delta N_{CO_2}/A\tau_{CO_2} \tag{7}$$

where A is the area of the sample (0.62 cm^2). The number of gaseous CO_2 molecules produced on the catalyst (ΔN_{CO_2}) is derived from the ideal gas law, $\Delta N_{CO_2} = \Delta P_{CO_2} V/kT_g$, where V is the volume of the belljar on the sample side of the poppet valve (40 L), T_g is the temperature of the gas (300 K), k is the Boltzmann constant, and ΔP_{CO_2} is the difference of the

CO_2 partial pressures when the catalyst is at a given temperature and when the reaction is quenched. That is, the filaments in the mass spectrometer and the ion gauge also produced CO_2 from the reactants. However, the filaments and the catalyst did not compete for the reactants since the extents of reaction were small. Typically less than 10%, and never more than 20%, of either reactant admitted through the leak valve was converted into CO_2.

When the pumping speed was fixed with the method described above, the partial pressures of the reactants could not be fixed by adjusting the leak rates of the reactants. Reactions with the filaments changed the partial pressures, but equilibrium was obtained in 1 to 10 min. When the reaction on the catalyst was quenched, the composition of the gas phase was determined using an ion gauge to read the total pressure and a mass spectrometer to monitor masses 28, 32 and 44 for CO, O_2 and CO_2, respectively. The apparent total pressure at the ion gauge (P_T) is related to the partial pressure of each component (P_i) by

$$P_T = \sum_i i_i P_i \tag{8}$$

where i_i is the ionization efficiency of component i. The signal intensity from the mass spectrometer for any component (I_i) is related to its partial pressure (P_i) by

$$I_i = P_i i_i f_i T_i G \tag{9}$$

where T_i is the relative gain of the spectrometer for the mass of the component, which is given in the specifications for the spectrometer, and

G is the absolute gain of the spectrometer, which is to be determined. The fraction of the ion current appearing as a given mass for any species (f_i) is determined from the cracking pattern for that molecule,

$$f_i = (s_i/T_i)/\sum_j (s_j/T_j) \tag{10}$$

where s_j is the relative intensity of mass j in the cracking pattern. To account for the fraction of CO_2 that is ionized into CO^+, 12% of the intensity for mass 44 is subtracted from the intensity for mass 28. Hence, the absolute gain is derived from Eqs. (8) and (9),

$$G = \sum_i (I_i/f_i T_i P_T) \tag{11}$$

To obtain the partial pressure of each component, this value for G is substituted into Eq. (9).

Equilibrium rates of CO_2 production were measured through the following procedure. A clean Ir(110) crystal at 300 K was exposed first to CO. Oxygen was not admitted into the belljar until the surface received 100 L of CO. Following this exposure, the surface is covered with CO, which is typically the situation at equilibrium as is shown in Section V. After the partial pressures of CO, O_2 and CO_2 had equilibrated through reactions with the filaments, the partial pressure of each gas was determined using the procedure described above. To obtain turnover numbers, the partial pressure of CO_2 was measured as the temperature of the catalyst was varied at a rate of 80 K/min. The measured oxidation rate at any temperature did not vary for slower heating rates. The

temperature first was increased to 1000 K and then was returned to 300 K. As the temperature of the catalyst changed, the partial pressures of the reactants, CO and O_2, were not monitored since these could be determined directly from the change in the partial pressure of CO_2. Moreover, the extents of reaction were small, less than 20%, so that these partial pressures did not change significantly.

III. Rate Expressions for CO Desorption and CO Oxidation

Representative thermal desorption sepctra of CO and CO_2 obtained by flashing co-adsorbed layers of CO and oxygen from Ir(110) are shown in Fig. 2. The spectra for the desorption product O_2 appear above 1000 K and are identical to those obtained by flashing chemisorbed oxygen layers alone (17). For the data shown in Fig. 2, a clean Ir(110) surface at 300 K was exposed first to 0.8 L of oxygen, which corresponds to a coverage of 0.25 ML (17), and then to varying amounts of CO, which are noted in the figure. Regardless of whether the surface is clean or oxidized prior to the exposure sequence, and regardless of which component is introduced first in the exposure sequence, the spectra are similar in shape to those shown in Fig. 2. The spectra of CO give a single peak near 600 K whereas the spectra of CO_2 give two peaks near 400 and 550 K. The desorption rate for CO is correlated with the oxidation rate around 550 K.

The CO desorption spectra in Fig. 2 appear to vary as zero-order in the coverage of CO. The maximum of the peak shifts to higher temperatures with increasing exposure of CO, and the intensity rises exponentially with temperature (19). If CO desorption is described by an Arrhenius expression that is zero-order in the coverage of CO, a plot of $\ell n\ I_{28}^{*}$ versus

T^{-1} for any spectrum should give a line with slope - E_d/k and intercept $\ell n(\nu_d/\eta_{28})$ where E_d and ν_d are the activation energy and the pre-exponential factor for desorption. Plots constructed from the data in Fig. 2 are shown in Fig. 3. For initial CO coverages of 0.25 and 0.45 ML, the desorption energy is 11 ± 1 kcal/mole and the pre-exponential factor is $10^{17\pm1}$ cm^{-2} sec^{-1}, but for an initial CO coverage of 0.75 ML, the respective parameters are 7 ± 1 kcal/mole and $10^{16\pm1}$ cm^{-2} sec^{-1}. Hence, the desorption energy varies significantly with the coverage of CO. Moreover, the order of the desorption process may not be zero since the coverage dependence of the rate resulting from the variation in the activation energy may overshadow completely the dependence resulting from leading coefficients that vary as coverages to integral exponents. This is indeed the case as will be shown now.

For small CO coverages, the order of the desorption process (n) is given by the slope of a plot of $\ell n\ \Theta_{CO}$ versus $\ell n\ I^*_{28}$ at points of constant temperature (20). The coverage of CO at any instant in a spectrum is given by Eq. (5), and time is related to the temperature through the heating schedule. An order plot of $\ell n\ I^*_{28}$ versus $\ell n\ \theta_{CO}$ constructed from the spectra in Fig. 2 for a temperature of 573 K is shown in Fig. 4. The slope in Fig. 4 is 0.9 ± 0.1 so the order of the desorption process in the coverage of CO is unity. This is expected because CO chemisorbs molecularly on Ir(110). Also, one might have guessed that the order is not zero since the desorption rate above the maximum does not fall off rapidly (19).

For Ir(110) covered with co-adsorbed oxygen and CO, the desorption rate of CO (R_d in molecules/cm^2 sec) is modeled best with the first-order

Arrhenius expression (18)

$$R_d = C_s \eta_{28} I_{28}$$

$$= - \nu_d(\theta_{CO}, \theta_O) C_s \theta_{CO} \exp [- E_d(\theta_{CO}, \theta_O)/kT] \quad (12)$$

where θ_{CO} and θ_O are obtained from Eqs. (5) and (6), and the concentration of adsites (C_s) is $10^{15}/cm^2$ for either CO or oxygen. Also, the rate parameters, $\nu_d(\theta_{CO}, \theta_O)$ and $E_d(\theta_{CO}, \theta_O)$, depend on the coverages of both oxygen and CO. Once again, I_{28}^* refers to the intensity in a CO spectrum where 12% of the I_{44} signal has been subtracted from the I_{28} signal to account for the ionization of CO_2 into CO^+. Arrhenius plots of $\ell n\ I_{28}^*$ versus T^{-1} at points of constant CO and oxygen coverages should give lines with slope $- E_d(\theta_{CO}, \theta_O)/k$ and intercept $\ell n[\nu_d(\theta_{CO}, \theta_O)\theta_{CO}/\eta_{28}]$. In constructing each plot, a large number of spectra would be required to obtain sufficiently accurate rate parameters. Hence, only the coverage of CO is held constant for the Arrhenius plots presented here. The dependence of the rate parameters upon the coverage of oxygen was investigated by varying the initial coverage of oxygen from one set of desorption spectra to the next. Although some error is introduced by this procedure, the coverage of oxygen varies typically by less than a factor of two in the Arrhenius plots presented here. Fig. 5a shows Arrhenius plots for CO desorption from a clean Ir(110) surface initially covered with 0.5 ML of oxygen. These plots are evaluated for points of constant CO coverage. If Arrhenius plots of $\ell n(I_{28}^*/\theta_{CO})$ versus T^{-1} are evaluated at points of constant oxygen coverage, the plots do not yield

lines as those in Fig. 5a do.

The rate parameters for CO oxidation may be obtained like those for CO desorption. For CO oxidation from co-adsorbed layers of oxygen and CO, the rate (R_r in molecules/cm^2 sec) is given by the Langmuir-Hinshelwood (L-H) expression (1),

$$R_r = C_s \eta_{44} I_{44}$$

$$= - \nu_r(\theta_{CO},\theta_O) C_s^2 \theta_{CO} \theta_O \exp [- E_r(\theta_{CO},\theta_O)/kT] \qquad (13)$$

where $E_d(\theta_{CO},\theta_O)$ and $\nu_r(\theta_{CO},\theta_O)$ are the activation energy and the pre-exponential factor for CO oxidation. The rate is truly first-order in the coverages of CO and oxygen since order plots of $\ell n\ I_{44}$ versus $\ell n(\theta_{CO}\theta_O)$ at 423 and 573 K for the data in Fig. 3 give a slope of one for small coverages ($\theta_{CO}\theta_O < 0.01$). The apparent order decreases to zero for larger coverages ($\theta_{CO}\theta_O > 0.04$), indicating that the rate parameters for CO oxidation depend on the composition of the overlayer. Arrhenius plots for CO oxidation, graphs of $\ell n[I_{44}/\theta_{CO}]$ versus T^{-1} at points of constant oxygen coverage, are shown in Fig. 5(b). As in Fig. 5(a), the surface was covered initially with 0.5 ML of oxygen, and the exposure of CO was varied. The slopes of the lines in Fig. 5(b) give $-E_r(\theta_O)/k$ whereas the intercepts give $\ell n[\nu_r(\theta_O) C_s \theta_O/\eta_{44}]$. As before, this construction is only approximate since the concentration of CO is not fixed in each plot. Moreover, Arrhenius plots of $\ell n(I_{44}/\theta_O)$ versus T^{-1} at points of constant CO coverage do not yield lines. Intensities from the two CO_2 peaks in the desorption spectra must be graphed in separate Arrhenius plots;

otherwise, linearity is not obtained. This indicates that a thermally activated process other than CO oxidation, such as the diffusion of CO or oxygen (6), may limit the oxidation rate on the surface for some range of temperature.

The dependence of the rate parameters for CO desorption from clean Ir(110) upon the coverage of CO and the initial coverage of oxygen is shown in Fig. 6(a). The rate parameters were derived from Arrhenius plots using the technique described previously. Oxygen was the first component introduced in the exposure sequence for these data, and the initial coverage is denoted by θ_O^o. If the surface first is covered with 0.25 ML of CO and then is exposed to oxygen, the activation energy for CO desorption calculated from the associated desorption spectra is 16 ± 1 kcal/mole for CO coverages between 0.05 and 0.20 ML. The data in Fig. 6(a) show that the desorption energy decreases as the coverage of CO or oxygen is increased. For any initial coverage of oxygen, the desorption energy varies almost linearly in the coverage of CO; that is, $E_d(\theta_{CO}) = E_o - E_\delta \theta_{CO}$. Regardless of the initial coverage of oxygen, E_δ equals 15 kcal/mole-ML, but E_o is 35, 21, 16 and 15 kcal/mole for initial oxygen coverages of 0, 0.25, 0.50 and 0.75 ML, respectively. If the Ir(110) surface is covered partially with oxygen, the observed desorption energies for CO are relatively small for a desorption feature appearing near 600 K (21), but are comparable to desorption energies calculated from Pd(111) for co-adsorbed layers of CO and oxygen (6). The small values for the desorption energy indicate that another thermally activated process, such as the diffusion of CO, may be rate limiting for CO desorption (6). That is, chemisorbed CO molecules might diffuse to a certain type of site before

desorption may occur.

Although the rate parameters shown in Fig. 6(a) depend complexly upon the composition of the adlayer, an average value for the desorption energy, pertinent for the equilibrium rate of CO oxidation over a large range of catalyst temperatures, may be estimated readily. As described in Section V, the onset of the oxidation reaction is limited by the desorption of CO from the surface; under these conditions, the coverage of CO approaches 1 ML whereas the coverage of oxygen approaches zero. At temperatures above 600 K, the coverage of CO approaches zero, as evidenced by Figs. 1 and 2, whereas the coverage of oxygen approaches 0.5 ML (17). The desorption energies for these two extremes are 20 and 14 kcal/mole, respectively. Since the desorption energy decreases as the coverage of either CO or oxygen is increased, the desorption energy is expected to lie between the values for the extremes. Therefore, the average activation energy for CO desorption associated with the equilibrium oxidation of CO on clean Ir(110) is near 17 kcal/mole.

The dependence of the rate parameters for CO desorption from oxidized Ir(110) upon the composition of the overlayer is shown in Fig. 6(b). Prior to the exposure sequence, the surface was oxidized according to the procedure described in Section II. Oxygen was the first component introduced in the exposure sequence. For the rate parameters labeled A, the chemisorbed layer of oxygen was annealed to 1000 K before the surface was exposed to CO; for the rate parameters labeled B, no annealing occurred. These experiments were undertaken to determine if the ordering of the chemisorbed layer of oxygen influences the transient rates of CO desorption and oxidation. That is, the overlayer of chemisorbed oxygen

does not order into a c(2x2) structure on oxidized Ir(110) until it is annealed to 700 K (17). From the rate parameters shown in Fig. 6(b), the structure in the oxygen adlayer has little effect on the desorption rate for CO. However, the fraction of CO that desorbs, compared to the fraction that reacts to form CO_2, is slightly larger for a surface with an ordered oxygen adlayer.

For any coverage of CO and any initial coverage of oxygen, the rate parameters E_d and ν_d are smaller for oxidized Ir(110). Since the desorption kinetics of CO affect the equilibrium rate of CO oxidation (1), the rates are expected to be different on clean and oxidized Ir(110). This is indeed the case as will be shown in Section V. Again, the desorption energy of CO from oxidized Ir(110) changes markedly with the composition of the adlayer, but an average desorption energy associated with the equilibrium rates of CO oxidation may be estimated. Using the procedure described for the clean surface, an average desorption energy of 14 kcal/mole is estimated for the oxidized surface.

The pre-exponential factors and activation energies for CO desorption vary in sympathy; that is, the compensation effect occurs. A modified Arrhenius expression may be used to describe this effect (22). The appropriate rate expression for CO desorption is

$$R_d = -\nu_{od} C_s \theta_{CO} \exp\left[-\frac{E_d(\theta_{CO},\theta_O)}{k}\left(\frac{1}{T}-\frac{1}{T_{sd}}\right)\right] \qquad (14)$$

where ν_{od} and T_{sd} are constants, and only E_d depends on the composition of the adlayer. If the overlayer contains no adsorbed oxygen, ν_{od} and T_{sd} are approximately 10^6 sec^{-1} and 1240 K, respectively; if the surface is

covered initially with only the oxide layer or with only 0.25 ML of chemisorbed oxygen, ν_{od} and T_{sd} are 3 sec^{-1} and 650 K; for all other cases, ν_{od} and T_{sd} are 0.3 sec^{-1} and 540 K. These values for ν_{od} and T_{sd} were obtained by a least squares fit of the rate parameters shown in Fig. 6, and are accurate within a factor of 3 sec^{-1} and ± 30 K, respectively. Regardless of whether ν_{od} and T_{sd} are chosen as 3 sec^{-1} and 650 K, or as 0.3 sec^{-1} and 540 K, the pre-exponential factor associated with the average desorption energy for the clean surface, 17 kcal/mole, is 10^6 sec^{-1}. Similarly, the pre-exponential factor associated with the average desorption energy for the oxidized surface, 14 kcal/mole, is 10^5 sec^{-1}. These average values of the rate parameters for CO desorption will be used for modeling the equilibrium rate of CO oxidation.

As shown in Fig. 7, the rate parameters for CO oxidation, E_r and ν_r, vary weakly with the total coverage of adsorbed oxygen, θ_O. For the abscissae of these data only, θ_O is the sum of the coverages in the chemisorbed and oxide layers. The parameters for CO oxidation on clean and oxidized Ir(110) are shown in Figs. 7(a) and 7(b), respectively, and are appropriate for describing oxidation by the Langmuir-Hinshelwood mechanism only. The procedure used for calculating these rate parameters from Arrhenius plots was described previously. Rate parameters calculated from the low and high temperature peaks in the CO_2 spectra are shown by empty and filled symbols, respectively. Both oxygen and CO were introduced first in the exposure sequences for these data; for the respective cases, the initial coverages are denoted by θ_O^o and θ_{CO}^o in Fig. 7. The calculated rate parameters do not depend upon the exposure sequence or upon the order in the chemisorbed layer of oxygen. As in Fig. 6(b), the

points labeled A in Fig. 7(b) pertain to rate parameters for an oxygen overlayer that is ordered into the c(2x2) pattern initially, whereas the points labeled B pertain to rate parameters for an oxygen overlayer that is disordered initially. In general, the activation energy for CO oxidation via the L-H mechanism does not depend strongly on the composition of the chemisorbed layer or whether the surface is clean or oxidized. In most instances, the activation energy may be approximated as 12 kcal/mole, which is the value that will be used for modeling the equilibrium rate of CO oxidation.

The compensation effect occurs for CO oxidation, as well as CO desorption. The rate equation that describes the sympathetic variation in the activation energy and the pre-exponential factor for CO oxidation is (22)

$$R_r = -\nu_{or} C_s^2 \theta_{CO} \theta_O \exp\left[-\frac{E_r(\theta_{CO},\theta_O)}{k}\left(\frac{1}{T}-\frac{1}{T_{sr}}\right)\right] \qquad (15)$$

where ν_{or} and T_{sr} are constants, and, once again, only E_r varies with the compensation of the adlayer. Although E_r does not depend strongly on any parameters, including the compensation of the adlayer, ν_{or} and T_{sr} differ for the two peaks in the thermal desorption spectrum of CO_2. From a least squares fit of the rate parameters in Fig. 7, ν_{or} and T_{sr} are $7 \times 10^{-16\pm1}$ cm^2/sec and 390 ± 30 K for the peak at lower temperatures, but are $8 \times 10^{-17\ 1}$ cm^2/sec and 430 ± 30 K for the peak at higher temperatures. Using 12 kcal/mole as an average activation energy for CO oxidation, the average pre-exponential factors (ν_r) are 3×10^{-9} cm^2/sec and 3×10^{-10} cm^2/sec for the respective

peaks. Since the equilibrium oxidation rate on Ir(110) is slow below 450 K, the latter value of the pre-exponential factor is more appropriate for analyzing equilibrium data. Relatively low values of the rate parameters, like those determined here, are reported most frequently for CO oxidation via the L-H mechanism on the platinum metals, and are associated with the diffusion kinetics of chemisorbed CO onto islands of oxygen atoms (2,6,13). On Pt(110) (23), Pd(111) (6,24), Ir(111) (13) the activation energy is about 10 kcal/mole whereas the pre-exponential factor is near 10^{-10} cm^2/sec.

When oxygen is the first component introduced in the exposure sequence, the coverage of CO that desorbs as CO_2 (θ_{CO_2}) depends predictably upon the initial coverage of CO (θ^o_{CO}). This dependence for various initial coverages of oxygen (θ^o_O) on clean and oxidized Ir(110) is shown in Fig. 8. If the coverage of chemisorbed oxygen exceeds that of CO prior to the flash, 85% of the CO desorbs as CO_2. The same trend is observed for the reaction of chemisorbed oxygen on the oxidized surface. However, if the chemisorbed layer is ordered, as for the points labeled A in Fig. 8(b), rather than random, as for the points labeled B, slightly less CO_2 is formed. The oxide layer alone is less reactive than chemisorbed oxygen to reduction by CO. If the oxide coverage exceeds the initial CO coverage, about 50% of the CO is oxidized. In a single flash, no more than 70% of the oxide will be reduced by CO whereas all chemisorbed oxygen will be reduced if a sufficient amount of CO is available. The trends similar to those noted above are observed when CO is the first component introduced in the exposure sequence. That is, the product distribution in these transient experiments depends on the composition of the overlayer, but not

upon the exposure sequence.

In the temperature regime where CO oxidation is rapid on the platinum metals, the desorption of oxygen and CO_2 are thought to be extremely slow and extremely fast, respectively, so that these rates generally are ignored in the analysis of equilibrium oxidation rates. These two conjectures have been verified experimentally for Ir(110). From data presented elsewhere (17), oxygen desorption from clean or oxidized Ir(110) is negligibly slow below 1200 K if the coverage of chemisorbed oxygen is less than 0.5 ML. When CO is absent from the adlayer, the saturation coverage of chemisorbed oxygen is 0.5 ML between 500 and 1000 K (17), which is the temperature regime where CO oxidation over Ir(110) is rapid under equilibrium conditions (see Section V). Therefore, the desorption of oxygen from Ir(110) may be ignored in analyzing the equilibrium data. Contact potential difference (CPD) measurements, which were performed with a retarding potential method (15), were used in an attempt to detect the adsorption of CO_2 on clean Ir(110) between 90 and 300 K. Surface temperatures below 200 K were calculated by measuring the temperature dependence of the resistance of the Ta wires that held the substrate. The CPD changed only slightly after the crystal was exposed to several hundred Langmuirs of CO_2. Upon flashing the surface after the exposure, only CO desorbed, indicating that the background impurity CO, rather than CO_2, had adsorbed to change the CPD. Therefore, either CO_2 must desorb rapidly from Ir(110), even at 90 K, or CO_2 adsorption is strongly activated. Because CO_2 adsorbs non-dissociatively on polycrystalline Pt at 100 K (26), the activation barrier for non-dissociative adsorption must be small so that CO_2 must desorb rapidly from Ir(110).

IV. The Co-Adsorption Kinetics of CO and Oxygen

The adsorption kinetics of CO on clean Ir(110) surfaces that were partially covered with chemisorbed oxygen were measured using thermal desorption spectrometry. These data were obtained for a surface temperature of 300 K and are shown in Fig. 9(a). For each set of CO, O_2 and CO_2 spectra, the coverage of CO for each exposure (the product of pressure and time) was determined from Eq. (3).

Since the initial sticking probability of CO is unity, regardless of the initial coverage of chemisorbed oxygen, oxygen does not block sites for CO adsorption. Similar behavior for CO chemisorption has been observed on Pd(111) (6) and polycrystalline Pt (11). At higher coverages the kinetic data for Ir(110) may be fit by a second-order precursor model that accounts for the ordering of the adlayer by intermolecular forces (27). This model depends on two assumptions. (1) The overlayer of the adsorbing species orders continually because of nearest-neighbor intermolecular forces, but also disorders because of thermal randomization. (2) A molecule in the precursor state may chemisorb only if two empty and adjacent sites for chemisorption are available. This model may pertain to non-dissociative adsorption (16), as is the case for CO, as well as dissociative adsorption (27). In this second-order precursor model (27), the ratio of the sticking probability (S) to the initial sticking probability (S_0) depends on the fraction of empty sites with empty nearest-neighbors (θ_{oo}) and a parameter κ that is inversely proportional to the lifetime of the precursor state,

$$S/S_0 = [(1 - \kappa) + \kappa/\theta_{oo}]^{-1} \qquad (16)$$

The sticking probability is related to the coverage (θ) and the exposure (ε) of the adsorbing species by (28)

$$S = (2\pi M_g kT_g)^{\frac{1}{2}} C_s \frac{d\theta}{d\varepsilon} \tag{17}$$

where M_g and T_g are the molecular weight and temperature of the adsorbing gas. The initial sticking probability is calculated by evaluating Eq. (17) in the limit of small coverage. For a CO overlayer on the Ir(110) surface, the fraction of empty sites with empty nearest-neighbors is given by (16)

$$\theta_{oo} = (1 - \theta) - \frac{2\theta(1 - \theta)}{[1 - 4B\theta(1 - \theta)]^{\frac{1}{2}} + 1} \tag{18}$$

The intermolecular force between nearest-neighbors (ψ) is related to the parameter B by (27)

$$\psi = - kT\ell n(1 - B) \tag{19}$$

where a positive value of ψ is associated with a repulsive interaction.

Chemisorbed oxygen does not block sites for CO chemisorption so that the pertinent coverage in Eq. (18) is the CO coverage, rather than the sum of the coverages of CO and oxygen. However, the repulsive interaction between chemisorbed CO molecules increases and the lifetime of the CO precursor decreases as the initial coverage of chemisorbed oxygen is increased. The parameters B and κ, which are related to the respective phenomena, vary with the initial oxygen coverage according to the

expressions

$$- kT\ell n(1 - B) = 2 + 0.4\theta_0^o \tag{20}$$

$$\kappa = \exp\,[1.6(\theta_0^o - 1)] \tag{21}$$

that were derived empirically from the data in Fig. 9(a). There, kinetic models with these parameters are shown as solid lines. At 300 K, the lifetime of the CO precursor on Ir(110) decreases with increasing coverage of chemisorbed oxygen; at temperatures where the equilibrium rate is rapid, the lifetime decreases even further. Regardless of the value of B, the adsorption kinetics for lower CO coverages (< 0.5 ML) approach the simple limit $S(CO)/S_o(CO) = (1 - \theta_{CO})^2$ as the lifetime of the precursor becomes small (27). As will be shown, CO blocks oxygen chemisorption almost completely for larger CO coverages so the oxidation reaction is inhibited. Hence, from Eq. (17), the adsorption rate of CO (in molecules/cm^2 sec), associated with the equilibrium oxidation reaction on clean Ir(110), may be approximated by

$$C_s \frac{d\theta_{CO}}{dt} = \frac{P_{CO}(1 - \theta_{CO})^2}{(2\pi M_{CO} kT_g)^{\frac{1}{2}}} \tag{22}$$

regardless of the coverage of chemisorbed oxygen. Here, the partial pressure of CO is P_{CO}, and the initial sticking probability for CO is taken to be unity.

As shown in Fig. 9(b), the adsorption kinetics of CO on oxidized Ir(110) at 300 K also exhibit second-order behavior when oxygen is chemisorbed on

the surface. Once again, oxygen does not block sites for CO chemisorp- because the initial sticking probability of CO is unity, regardless of the coverage of oxygen. In the absence of chemisorbed oxygen, the parameters B and κ are identical for clean and oxidized Ir(110). When 0.25 ML of oxygen is chemisorbed on the oxidized surface, both B and κ in Eq. (16) assume values of unity. Kinetic models with these parameters are shown by the solid lines in Fig. 9(b). Hence, the second-order behavior for the kinetics of CO chemisorption is even more pronounced on the oxidized surface. Moreover, ordering the chemisorbed layer of oxygen, into a c(2x2) structure, as in curve c in Fig. 9(b), changes the adsorption kinetics only slightly with respect to a disordered layer, as in curve b. Therefore, Eq. (22) may be used to describe the adsorption kinetics of CO on oxidized Ir(110) when the oxidation reaction is proceeding under equilibrium conditions. The second-order behavior for CO chemisorption is not unique to Ir(110) and has been observed in other studies of CO oxidation (6,11), including the early work of Langmuir (1).

Although chemisorbed oxygen does not block sites for CO adsorption, chemisorbed CO blocks sites for oxygen adsorption. For a clean surface initially covered with 0.75 ML of CO, no detectable amount of oxygen was chemisorbed following an oxygen exposure of 10 L. When the experiment was repeated with an initial CO coverage of 0.5 ML, only 0.1 ML of oxygen chemisorbs, but the oxygen coverage would be near the saturation limit if the CO were absent (17). Fig. 10 shows the kinetic data for oxygen chemisorption on clean and oxidized Ir(110) surfaces that were initially covered with 0.25 ML of CO. The surface temperature was 300 K. The coverage of oxygen associated with each exposure was determined from thermal desorption spectra with Eq. (4). Although the data for the clean surface are not fit well by the second-order Langmuir model,

$$S(O_2)/S_o(O_2) = (1 - \theta_O - \theta_{CO})^2 \qquad (23)$$

the data for the oxidized surface are, as shown by the solid line in Fig. 10. In this case, the initial sticking probability of oxygen [$S_o(O_2)$] is 0.035, which is ten times smaller than the value obtained when CO is absent from the surface.

The kinetic data for oxygen chemisorption on clean Ir(110) may be modeled by a second-order Langmuir model if the CO overlayer orders into a p(2x2) array, rather than a random configuration. The p(2x2) LEED pattern of CO, which forms near 0.25 ML coverage on clean Ir(110) only, is ordered at 90 K, but is disordered somewhat at 300 K (16). In developing this kinetic model of oxygen adsorption, a random number generator was used to select the pair of sites where oxygen might chemisorb. The surface consisted of a (20x20) array of adsorption sites, upon which the p(2x2) array of CO was superposed and filled 25% of the adsorption sites. If a gaseous oxygen molecule struck a pair of empty sites, it might chemisorb with probability $S_o(O_2)$ and would fill the two sites. Otherwise, the gaseous molecule was reflected from the surface. For an initial sticking probability [$S_o(O_2)$] of 0.27, which nearly equals the value obtained when CO is absent from clean Ir(110) (17), this model accurately fits the data for the clean surface in Fig. 10. There, the model is shown as a solid line.

Because the p(2x2) structure of CO disorders as the surface temperature increases (16), the adsorption kinetics of oxygen on clean Ir(110) are expected to follow Eq. (23) for temperatures where the equilibrium rate of CO oxidation is large. Under these conditions, the rate of oxygen adsorption on either clean or oxidized Ir(110) may be approximated further by $S(O_2)/S_o(O_2) = (1 - \theta_O)^2(1 - \theta_{CO})^2$, or

$$C_s \frac{d\theta_O}{dt} = \frac{2S_o(O_2)P_{O_2}(1 - \theta_O)^2(1 - \theta_{CO})^2}{(2\pi M_{O_2} kT_g)^{\frac{1}{2}}} \qquad (24)$$

Moreover, this approximation for Eq. (23) may be necessary since the sum of the coverages of chemisorbed oxygen and CO may exceed unity, as in Fig. 9, but either coverage alone cannot exceed unity (16,17). The factor of two appears in Eq. (24) because oxygen chemisorption is dissociative.

Because chemisorbed oxygen does not block sites for CO adsorption, but chemisorbed CO does block sites for oxygen adsorption, oxygen and CO do not, in the strictest sense, compete for adsorption sites on Ir(110). In previous studies on other platinum metals (14,29), the E-R reaction was thought to be rate-limiting for the rapid reduction of an oxygen overlayer by gaseous CO, but the L-H reaction was thought to be rate-limiting for the slow oxidation of a CO overlayer by gaseous O_2. Both processes may be described in terms of the L-H mechanism if the adsorption kinetics of CO and oxygen are modeled like the kinetics on Ir(110). That is, CO oxidation via the L-H reaction is rate limiting in the first case whereas oxygen adsorption is rate-limiting in the second case. Also, other studies (9) have indicated that CO oxidation may occur through the reaction of chemisorbed oxygen with a reactive form of chemisorbed CO, which is present only when the coverage of CO is small. This phenomenon might be explained better by the fact that adsorbed CO strongly blocks oxygen chemisorption. Again, oxygen adsorption, rather than the CO oxidation reaction, is rate-limiting.

The measured kinetics for oxygen adsorption on oxidized Ir(110)

partially covered with a CO are modeled accurately by Langmuir kinetics for a disordered CO overlayer, given by Eq. (23). This is surprising because chemisorbed CO orders into a (2x1)p1g1 structure on oxidized Ir(110) below 500 K and above 0.05 ML coverage. Moreover, the chemisorbed oxygen does not change the order in the CO overlayer. When oxygen is adsorbed on either the CO-p(2x2) structure on clean Ir(110) or the CO-(2x1)p1g1 structure on oxidized Ir(110) (16), the order in the overlayer is not changed although the background intensity of the LEED pattern increases. Similarly, when CO is adsorbed on either the O-p(2x2) structure on clean Ir(110) or the O-c(2x2) structure on oxidized Ir(110) (17), the order is not affected. The LEED patterns for these four coadsorbed systems sharpen when the substrate is cooled to 90 K, but vanish when the overlayer is annealed to 400 K. Upon annealing, the overlayer is lost through CO oxidation. No LEED patterns that are unique to coadsorbed systems were detected in this study even though these structures were sought extensively.

V. Equilibrium Rates of CO_2 Production

The equilibrium rate of CO_2 production (R_r in molecules/cm^2 sec) was measured as a function of the catalyst temperature (T) and the partial pressures of the reactants, CO and O_2. Representative data for total pressures (P_T) of 3.0×10^{-7} and 3.0×10^{-6} torr are shown in Fig. 11. Oxidation rates were measured for total pressures of 1.0×10^{-7} and 1.0×10^{-6} torr also. For each total pressure, the ratio of the CO partial pressure to the O_2 partial pressure ($Q = P_{CO}/P_{O_2}$) was varied between 0.1 and 10 in separate experiments, which will be referred to as

cycles. The arrows drawn on the data for each cycle point to the right if the catalyst temperature is increasing and to the left if the temperature is decreasing. Since typically less than 10% of either reactant gas reacted to form CO_2 in these experiments, the ratio Q did not vary significantly with the catalyst temperature. The values of Q shown in Fig. 11 were fixed at 300 K using the procedure described in Section II.

Several qualitative features are common among the cycles shown in Fig. 11. The variation in the equilibrium rate with the catalyst temperature for any cycle is similar to that reported for CO oxidation over other surfaces of the platinum metals (1 - 7). A maximum in the rate appears near 600 K. At this temperature, CO begins to desorb rapidly from the platinum metals, indicating that CO desorption limits the oxidation rate at lower temperatures (1). On Ir(110), the temperature associated with the maximum rate is independent of the total pressure in the gas phase, but is shifted to higher temperatures as Q is increased. Because the rate of CO adsorption (Eq. 22) increases with Q, this shift is further indication that the desorption of CO limits the rate below 600 K. Two features, which generally are not reported for CO oxidation, appear in the cycles shown in Fig. 11. (1) A second local maximum in the oxidation rate occurs near 800 K for larger values of Q. (2) The oxidation rates exhibit hysteresis as the catalyst temperature is varied. Moreover, a change in the maximum rate of CO_2 production coincides with the hysteresis.

The hysteresis might result from the oxidation of the surface because, as shown in the previous two sections, the rates of several elementary reactions differ for clean and oxidized Ir(110). To test this

hypothesis, the maximum temperature of the catalyst, rather than Q, was varied among cycles. Several cycles for values of Q of 2.7/1 and 1/4.4 are shown in Fig. 12. In these experiments, the surface was cleaned before the first cycle only. The total pressure was 1.0×10^{-6} torr. In the absence of CO, the catalyst is oxidized near 700 K (17), but, in the presence of CO, the catalyst may be oxidized at higher temperatures because the processes of CO oxidation and oxide formation may compete for chemisorbed oxygen.

For a value of Q of 2.7, the maximum rate of CO oxidation did not change if the maximum catalyst temperature in the first cycle was maintained below 750 K. However, hysteresis still occurred, as is shown in Fig. 12(a). This hysteresis is not an artifact; the oxidation rates at the two maxima were stable over 30 min, which was the longest time that the stability was tested. After the catalyst temperature was returned to 300 K and the reactant gases were evacuated, CO was the only desorption product observed when the catalyst was flashed to 1800 K. During the second cycle, the second local maximum in the equilibrium rate appeared when the catalyst temperature first exceeded 850 K, but did not appear subsequently. As the temperature was returned to 300 K, the maximum rate of CO_2 production increased. Moreover, the increase was irreversible since the rate at the maximum did not change in succeeding cycles as long as the sample was not cleaned. When the reactants were evacuated following the second or succeeding cycles, a (2x1)p1g1 LEED pattern, characteristic of CO chemisorption on oxidized Ir(110) only (16), was observed on the catalyst. Upon flashing the catalyst to 1800 K, CO, O_2 and CO_2 were observed as desorption products so that the catalyst was oxidized

during the second cycle, but not during the first cycle. Therefore, the hysteresis cannot be attributed solely to oxide formation on the catalyst, but the second local maximum in the equilibrium rate and the change in the maximum equilibrium rate of CO_2 production are associated with oxide formation on the catalyst. Moreover, although the oxide layer may be reduced by CO (16), at least a portion of the layer is stable between 300 and 1000 K while CO_2 is produced under equilibrium conditions. For values of Q greater than two, the hysteresis is less pronounced following the oxidation of the catalyst. Also, the maximum equilibrium rate of CO_2 production is increased following the formation of the oxide. Both trends are evident in Fig. 12(a).

Data for the experiment described above are shown in Fig. 12(b), but here $Q(= P_{CO}/P_{O_2})$ is 1/4.4. After the catalyst temperature first exceeded 700 K, the catalyst was oxidized and had to be cooled below 300 K to quench the production of CO_2. As before, the oxidation of the catalyst was detected by LEED and by the thermal desorption spectrum of the catalyst. As Q was decreased, the oxide layer generally formed at lower temperatures. As shown in the curve for the first cycle, the maximum equilibrium rate of CO_2 production was larger before the catalyst was oxidized. In succeeding cycles, the maximum rate decreased irreversibly after the catalyst was oxidized. For values of Q less than one, the maximum equilibrium rate of CO_2 production decreased after the catalyst was oxidized. Moreover, hysteresis was not observed for temperatures above the maximum, but was quite pronounced at lower temperatures. Again, hysteresis cannot be attributed solely to differences in the catalytic behavior on the clean and oxidized surfaces. Hysteresis is observed in

cycles for which the surface is always clean, as in the first cycle in Fig. 12(a), or always oxidized, as in the succeeding cycles in Fig. 12(b).

Hysteresis may appear in the CO oxidation rate because the kinetics of the elementary reactions that may limit the equilibrium rate may vary non-linearly in the coverage of either CO or oxygen. Hence, two or more minima in the total Gibbs energy, each associated with different coverages, may exist for any catalyst temperature. Although the free energy for one minimum may be substantially smaller than the energy for all others, a large activation energy may prevent the system from obtaining the lowest possible free energy. (As an analogy, two stable crystalline structures of carbon exist, but a considerable activation barrier prevents the instantaneous transformation of graphite into diamonds, which are more stable thermodynamically.) The minimum that is entered as the temperature changes depends upon the composition of the adlayer prior to the change and, therefore, depends upon the rate and the sign of the change in temperature. The "equilibrium" rates shown in Figs. 11 and 12 did not vary with the heating rate as along as it was maintained below 4 K/sec, but did vary considerably with the sign of the heating rate. Although two different rates at the same temperature were stable typically for thirty minutes, the rates for larger values of Q tended to fall uncontrollably as the reaction was quenched. In this and other respects, the hysteresis reported for CO oxidation over polycrystalline Pt (7) is similar to that observed in this study.

LEED was not only useful for indicating that the surface was oxidized, but was also useful for showing that chemisorbed oxygen clusters into islands when the equilibrium rate is large. On clean Ir(110), a p(2x2)

pattern, which appears for oxygen coverages near 0.25 ML in the absence of CO, formed for temperatures near the maximum in the equilibrium rate. On oxidized Ir(110), a c(2x2) pattern, which appears for oxygen coverages between 0.2 and 0.6 ML in the absence of CO, formed for temperatures near the maximum also. Hence, chemisorbed oxygen atoms on either clean or oxidized Ir(110) may cluster into islands when the oxidation reaction is proceeding under equilibrium conditions. Alternatively, the coverage of CO may be negligibly small for temperatures near the maximum in the equilibrium rate, but this conjecture is unreasonable since CO oxidation occurs via the L-H reaction. Moreover, the former conjecture is supported by previous studies of CO oxidation on Pd (2,6) and by results presented in Section III.

To obtain apparent activation energies for CO oxidation under equilibrium conditions, the logarithm of the equilibrium rate was graphed against the reciprocal of the catalyst temperature for each cycle. Representative graphs are shown in Fig. 13. For all cycles shown, the catalyst was clean initially, but was oxidized as the temperature was decreased (i.e., for curves where the arrow points to the right). The slope of the linear portion of each graph gives an apparent activation energy divided by the Boltzmann constant (E_a/k), whereas the intercept gives the logarithm of the apparent pre-exponential factor ($\ell n\ \nu_a$).

In the region where the rate increases with temperature of the clean surface, the apparent activation energy (E_1) is between 13 and 16 kcal/mole, regardless of the total pressure and the value of Q. In the same region for the oxidized surface, the activation energy is between 5 and 20 kcal/mole, and tends to increase with Q. In the region where the

oxidation rate decreased with the temperature of clean or oxidized Ir(110) the apparent activation energy (E_2) is between -6 and -3 kcal/mole, regardless of the total pressure and the value of Q.

The apparent pre-exponential factors for the oxidation reaction may be related empirically to the reactant partial pressures (P_{CO} and P_{O_2}) by

$$\nu_a = \nu_i P_{CO}^m P_{O_2}^n \tag{25}$$

where ν_i, m and n are constants. In the region where the rate increases with temperature on the clean surface, the apparent pre-exponential factor is proportional to $P_{O_2}^2/P_{CO}$ (= P_{O_2}/Q), as shown in Fig. 14. For this region, the oxidation rate (R_r in molecules/cm^{2}sec) is given by

$$R_{r1} = \nu_1 (P_{O_2}^2/P_{CO}) \exp [- E_1/dT] \tag{26}$$

where E_1 is between 13 and 16 kcal/mole as was noted above, and ν_1 from the intercept in Fig. 14 is 2 x $10^{25\pm1}$ molecules/cm^{2}torr-sec. The scatter in the data of Fig. 14 is not surprising since an error of 2 kcal/mole in the apparent activation energy causes a ten-fold variation in the apparent pre-exponential factor. The apparent pre-exponential factors in Fig. 14 were proportional to $P_T Q$ and $P_{O_2}^{3/2}$ also, but these correlations fit poorly at the extremes in the abscissa. In the region where the rate increased with temperature on the oxidized surface, no correlation between the partial pressures and the apparent pre-exponential factor could be found because the apparent activation energy varied markedly with the partial pressures of the reactants.

In the region where the rate decreases with the temperature of clean or oxidized Ir(110), the apparent pre-exponential factor is proportional to P_{CO}, as shown in Fig. 15. There, the factors for the clean surface are shown as empty symbols whereas the factors for the oxidized surface are shown as filled symbols. No other correlation in the form of Eq. (25) that fits these data could be found. For this temperature regime, the oxidation rate (R_{r2}) is given by

$$R_{r2} = \nu_2 P_{CO} \exp [- E_2/dT] \qquad (27)$$

where E_2 is between -6 and -3 kcal/mole as was shown above, and ν_2 from the intercept of Fig. 15 is $2 \times 10^{19\pm1}$ molecules/cm^2 torr-sec.

As is evidenced from Fig. 13, Eqs. (26) and (27) intersect at temperatures near the maximum in the equilibrium oxidation rate. If the intersection coincides exactly with the maximum, the temperature at the maximum (T_{MAX}) would be a function of $Q(= P_{CO}/P_{O_2})$ only and would be given by

$$Q^2 = \nu_3 \exp [- E_3/kT_{MAX}] \qquad (28)$$

where ν_3 equals ν_1/ν_2 or approximately 10^6, and E_3 equals $(E_1 - E_2)$ approximately 19 kcal/mole. This equation does not predict T_{MAX} correctly since T_{MAX} is to the left of the intersection for values of Q less than two and is to the right of the intersection for values of Q greater than two. However, as shown in Fig. 16, a plot of ℓn Q versus T_{MAX}^{-1} gives lines for both clean and oxidized Ir(110). There, points for the respective surfaces are empty and filled. In either case, the slope is nine, which nearly equals E_3/k. Also, the intercepts, 3×10^6 for the

clean surface and 1×10^7 for the oxidized surface, nearly equal ν_3. Hence, the temperature at the maximum in the equilibrium rate is approximated by Eq. (28) if the exponent of Q is changed to one. Moreover, the rate at the maximum is generally within a factor of two lower than the rate given by the intersection of Eqs. (26) and (27).

A kinetic model will be developed now to interpret the equilibrium oxidation rates in terms of the transient studies of the previous two sections. The rates of three elementary reactions--the desorption of CO_2, the desorption of oxygen and the oxidation of CO via the E-R mechanism--are not included in this model. In the temperature range where the equilibrium rates were measured, the rate of CO_2 desorption is extremely rapid and the rate of oxygen desorption is extremely slow, so that both processes do not affect the overall oxidation rate. Both desorption rates were determined experimentally for Ir(110), as was discussed in Section III. Studies with modulated molecular beams (6,10, 11) have shown that CO oxidation over the platinum metals between 300 and 1000 K occurs predominantly through the L-H mechanism whereas the oxidation rate via the E-R mechanism is undetectable. Moreover, the decrease in the oxidation rate observed above 600 K on Ir(110) cannot be attributed to a decrease in the coverage of oxygen and can be explained only if oxidation occurs by the L-H mechanism.

Four processes--the desorption of CO, the oxidation of CO via the L-H mechanism, the adsorption of CO and the adsorption of oxygen--may limit the equilibrium rate of CO oxidation. The rate expressions for the respective processes, determined from the transient experiments in the previous two sections, are given by Eqs. (12), (13), (22) and (24). The

diffusion of oxygen and CO on the catalyst also may influence the equilibrium rate, but these effects are included empirically in Eqs. (12) and (13). When CO oxidation is proceeding under equilibrium conditions, the time derivatives of the oxygen and CO coverages are each zero. That is, the rate of CO oxidation equals the rate of oxygen adsorption, and the sum of the rates of CO oxidation and CO desorption equals the rate of CO adsorption. Hence, Eqs. (12), (13), (22) and (24) may be reduced to three equations--

$$R_r + k_2\theta_{CO} = k_1 P_{CO}(1 - \theta_{CO})^2 \tag{29}$$

$$R_r = k_3 P_{O_2}(1 - \theta_{CO})^2(1 - \theta_O)^2 \tag{30}$$

$$R_r = k_4 \theta_O \theta_{CO} \tag{31}$$

in three unknowns--θ_{CO}, θ_O and R_r. Here,

$$k_1 = 1/(2\pi M_{CO} kT_g)^{\frac{1}{2}}$$

$$k_2 = \nu_d C_s \exp\,[-E_d/kT]$$

$$k_3 = 2S_o(O_2)/(2\pi M_{O_2} kT_g)^{\frac{1}{2}}$$

$$k_4 = \nu_r C_s^2 \exp\,[-E_r/kT]$$

The temperature in the gas phase for the equilibrium data is 300 K, $S_o(O_2)$

equals roughly 0.25 (17), and C_s is 10^{15} cm^{-2}. Hence the values of k_1 and k_3 are approximately 4×10^{20} and 2×10^{20} molecules/cm^2 torr-sec, respectively. As discussed in Section III, average values of ν_d, E_d, ν_r and E_r that are pertinent for the steady state data are $10^{5.5\pm0.5}$ sec^{-1}, 14 - 17 kcal/mole, 3×10^{-10} cm^2/sec and 12 kcal/mole. Although Eqs. (29) - (31) may be solved exactly for the equilibrium rate under any conditions, the steady state data are analyzed best by using these equations to describe the asymptotic rates above and below the maximum.

In the region where the equilibrium oxidation rate increases with the catalyst temperature, the reaction rate is limited by the desorption of CO and by the competitive adsorption of oxygen and CO. Evidence presented in this and other studies (1,2,3,6,10,11) supports this interpretation. In this region, the coverage of oxygen is small and may be ignored in Eq. (30) because more chemisorbed CO is present to block sites for oxygen chemisorption. Also, the flux of CO molecules to the surface always exceeds the measured oxidation rate by one to two orders of magnitude. Thus, the desorption rate must be large compared to the oxidation rate, except perhaps near temperatures where the reaction is nearly quenched so that the coverage of CO is large. (From the average values of the rate parameters for CO oxidation and desorption given above, the predicted ratio of the rates is around one between 500 and 1000 K. However, an error of 2 kcal/mole in either of the activation energies would change the ratio by ten-fold so the experimentally determined rates are more appropriate for interpreting the equilibrium data.) If the coverage of oxygen is small, and if the desorption rate of CO substantially exceeds the oxidation rate, Eqs. (30) and (31) reduce to

$$R_r = (k_3 \nu_d C_s / k_1) Q^{-1} \exp [- E_d/kT] \qquad (32)$$

In deriving Eq. (32), the coverage of CO was assumed to vary negligibly in temperature and to be near one. These assumptions are appropriate since, as was shown in Section III, the variation in the rate resulting from the change in the activation energy with coverage may overshadow completely the variation resulting from coefficients that depend on coverages to integral exponents. Moreover, the coverage of CO is near unity for the onset of the equilibrium reaction. Eq. (32) is identical to the expression derived by Langmuir (1) to describe the oxidation rate in the same temperature regime. Although the adsorption kinetics given by Eqs. (22) and (24) differ from the expressions used by Langmuir, the same expression describes the equilibrium rate in either case because the desorption of CO is rate-limiting in either model.

For clean Ir(110), the empirical activation energy from Eq. (26), 13 - 16 kcal/mole, agrees closely with the predicted value from Eq. (32), 17 kcal/mole, which was estimated from the transient experiments in Section III. However, the predicted and empirical dependences of the rate upon the reactant partial pressures are different. The rate is proportional to the pressure of oxygen in the model expression, but varies as the pressure of oxygen squared in the empirical expression. This squared dependence, which also is observed for CO oxidation over Pd(111) for similar temperatures (6), may be related to the diffusion of chemisorbed oxygen. Oxygen diffusion on Ir(110) becomes rapid at temperatures where the oxidation

rate becomes rapid (17). Moreover, chemisorbed oxygen must diffuse and subsequently coalesce into islands because the O-p(2x2) and O-c(2x2) LEED patterns are observed during the equilibrium reaction. Oxygen atoms may diffuse in pairs so that the diffusion rate depends on the coverage squared (30). In the regime where oxygen competes with CO for adsorption sites, the coverage of oxygen is proportional to the pressure of oxygen as indicated by Eqs. (30) and (32). Hence, if the diffusion of oxygen limits the overall oxidation rate, the rate would vary as the partial pressure of oxygen squared. (This variation in the rate cannot be explained by the reaction being limited to the edges of islands of chemisorbed oxygen atoms, for the rate would vary as the square root of the oxygen pressure.) Information concerning the kinetics of oxygen diffusion is needed to relate the empirical pre-exponential factor in Eq. (26) to a predicted value, like that in Eq. (32).

For oxidized Ir(110), no empirical expression for the overall oxidation, similar to Eq. (32), could be found. The data for the oxidized surface were obtained in the part of the hysteresis cycles where the catalyst temperature is decreasing, rather than increasing, so that the model presented in the previous two paragraphs may not be appropriate for this region. Moreover, the oxide, which is reactive to reduction by CO, may be decomposed partially as the temperature is decreased, but this conjecture was not tested.

In the region where the equilibrium rate is decreasing with the catalyst temperature, the empirical expression for the rate over clean and oxidized Ir(110) [Eq. (27)] may be derived readily from Eqs. (29)-(31). As evidenced by desorption rates of CO calculated in Section III, the

coverage of CO is small for this temperature regime. As determined previously, the desorption rate of CO is much larger than the oxidation rate. Hence, Eq. (29) is approximated closely by $\theta_{CO}k_2 = k_1P_{CO}$. Using this expression to give the coverage of CO, Eq. (31) becomes

$$R_r = (k_1C_s\theta_0\nu_r/\nu_d)P_{CO} \exp [(E_d - E_r)/kT] \quad (33)$$

Both the activation energy (6) and the pressure dependence (5 - 7,14,31,32) for the oxidation rate in this expression have been derived previously for other platinum metal catalysts. The coverage of CO is too small to inhibit the adsorption of oxygen, and, as determined from the data in Fig. 11, the oxidation rate is not limited by the flux of oxygen to the surface. Thus, the coverage of oxygen in this temperature regime approaches the saturation value, 0.5 ML. Using the average values for ν_d and ν_r that were given previously, the predicted value of the pre-exponential factor from Eq. (33), $k_1C_s\theta_0\nu_r/\nu_d = 2 \times 10^{20\pm1}$ molecules/cm^2 torr-sec, is quite close to the empirical value from Eq. (27), $2 \times 10^{19\pm1}$ molecules/cm^2 torr-sec. Moreover, the pressure dependences of the two expressions are identical, and the predicted value for the activation energy, -5 to -2 kcal/mole, is quite close to the observed value, -6 to -3 kcal/mole.

VI. Conclusions

The heterogeneously catalyzed reaction of gaseous CO and O_2 to form CO_2 over Ir(110) has been studied through measurements of the equilibrium rate of CO_2 production and of the transient kinetics of elementary reactions that may limit the equilibrium rate. Both clean and oxidized

Ir(110) were used as catalysts. The surface oxide, which is distinct chemically from chemisorbed oxygen, forms irreversibly at temperatures where the equilibrium rate of CO oxidation is rapid. The desorption of CO, the oxidation of CO via the Langmuir-Hinshelwood mechanism, the adsorption of CO and the adsorption of oxygen may limit the equilibrium rate of CO oxidation whereas the desorption of oxygen, the desorption of CO_2 and the oxidation of CO via the Eley-Rideal mechanism do not affect the rate significantly for catalyst temperatures between 300 and 1000 K. The rates of the four elementary reactions that may limit the equilibrium rate depend complexly on the composition of the adlayer. Moreover, the equilibrium rate may depend critically upon the diffusion rates of chemisorbed oxygen and CO, which were not measured directly in this study.

The kinetics of CO desorption and CO oxidation via the L-H mechanism may be modeled accurately by Arrhenius expressions. However, the dependence of the rates upon leading coefficients that vary as coverages to integral exponents may be overshadowed completely by the variation in the pre-exponential factors and activation energies with the composition of the adlayer. The activation energy for CO desorption, which depends on the coverages of both CO and oxygen, may vary from 10 to 36 kcal/mole, but is fixed near 16 kcal/mole under conditions where the equilibrium rate of CO oxidation is rapid. The pre-exponential factor associated with this activation energy is 10^6 sec^{-1}. The activation energy for CO oxidation via the Langmuir-Hinshelwood mechanism is relatively insensitive to the composition of the adlayer and is equal to approximately 12 kcal/mole. The pre-exponential factor for the oxidation reaction is 3×10^{-9} or 3×10^{-10} cm^2/sec, depending on the surface temperature. This variation,

as well as the small value for the pre-exponential factor and other evidence indicates that the diffusion of the reactants, rather than the Langmuir-Hinshelwood reaction, may limit the oxidation rate at the surface.

Because the initial sticking probability of CO on clean or oxidized Ir(110) at 300 K is unity, regardless of the coverage of chemisorbed oxygen, oxygen does not block sites for CO chemisorption. However, if a sufficient quantity of oxygen is chemisorbed on the surface, the adsorption kinetics of CO are of the Langmuir type and are second-order in the coverage of CO. Chemisorbed oxygen and CO both block sites for oxygen chemisorption so that the adsorption kinetics of oxygen are of the Langmuir type, vary to second-order in the sum of the coverages of CO and oxygen, but also depend on the structure of the chemisorbed layer. If the sites filled with CO are accounted for, the initial sticking probability for oxygen adsorption at a surface temperature of 300 K is 0.27 on the clean surface, which is close to the value obtained in the absence of CO, but is 0.035 on the oxidized surface, which is ten times lower than the value obtained in the absence of CO.

Models for the equilibrium oxidation rate, based on the rate expressions developed from the transient experiments, give a consistent interpretation of the rate data for surface temperatures from 300 to 1000 K and for partial pressures of the reactants between 1×10^{-8} and 3×10^{-6} torr. Below 600 K, where the oxidation rate increases with the catalyst temperature, the rate is inversely proportional to the partial pressure of CO, but is proportional to the partial pressure of oxygen squared. The unusual variation in the rate with the oxygen pressure may be associated

with the diffusion of chemisorbed oxygen. If the equilibrium rate is measured as the catalyst temperature is increased below 600 K, the apparent activation energy and pre-exponential factor for the equilibrium rate on clean Ir(110) are 14 ± 2 kcal/mole and 2 x $10^{25\pm1}$ molecules/cm^2 torr-sec. However, if the rate is measured as the temperature is decreased, the apparent activation energy and pre-exponential factor vary considerably because of hysteresis in the rate of CO_2 production. Since the rates of the elementary reactions that limit the equilibrium rate are non-linear functions of the coverages of CO and oxygen, the Gibbs energy for any catalyst temperature may have several local minima so that hysteresis may occur. Above 600 K, where the equilibrium rate decreases with the catalyst temperature, the oxidation rate is proportional to the partial pressure of CO, but is independent of the partial pressure of oxygen. Regardless of whether the surface is clean or oxidized, and whether the catalyst temperature is being increased or decreased, the apparent activation energy and pre-exponential factor for the overall rate coefficient are - 4 ± 2 kcal/mole and 2 x $10^{19\pm1}$ molecules/cm^2 torr-sec. The temperature associated with the maximum oxidation rate at eqiulibrium occurs near the intersections of the asymptotic trends above and below 600 K, and is inversely proportional to the logarithmof the ratio of the CO partial pressure to the oxygen partial pressure.

Acknowledgement

The authors are grateful for the support of the National Science Foundation under Grant No. DMR 77-14976.

References

1. I. Langmuir, Trans. Faraday Soc. 17, 621 (1922).
2. G. Ertl and P. Rau, Surface Sci. 15, 443 (1969).
3. H. P. Bonzel and R. Ku, J. Vac. Sci. Technol. 9, 663 (1971).
4. K. Christmann and G. Ertl, Z. Naturforsch. 28 a, 1144 (1973).
5. J. S. Close and J. M. White, J. Catal. 36, 185 (1975).
6. T. Engel and G. Ertl, submitted to J. Chem. Phys.
7. A. Golchet and J. M. White, submitted to J. Catal.
8. H. Heyne and F. C. Tompkins, Proc. Royal Soc. A 292, 460 (1966).
9. W. L. Winterbottom, Surface Sci. 36, 205 (1973).
10. R. L. Palmer and J. N. Smith, Jr., J. Chem. Phys. 60, 1453 (1973).
11. N. Pacia, A. Cassuto, A. Pentenero and B. Weber, J. Catal. 41, 455 (1976).
12. In Ref. 10, Palmer and Smith first reported that, when a modulated molecular beam of CO impinges on polycrystalline Pt, the phase lag for CO_2 production decreases with increasing surface temperature and O_2 pressure in the background. However, they did not associate this trend with the residence time of chemisorbed CO. This was done in Ref. 6 to prove conclusively that CO oxidation on Pd(111) occurs via the Langmuir-Hinshelwood mechanism.
13. V. P. Ivanov, G. K. Boreskov, V. I. Savchenko, W. F. Egelhoff, Jr. and W. H. Weinberg, J. Catal. 48, 269 (1977).
14. T. Matsushima and J. M. White, J. Catal. 39, 265 (1975).
15. J. L. Taylor, Ph.D. Thesis, California Institute of Technology, 1978.
16. J. L. Taylor, D. E. Ibbotson and W. H. Weinberg, submitted to J. Chem. Phys.

17. J. L. Taylor, D. E. Ibbotson and W. H. Weinberg, submitted to Surface Sci.

18. J. L. Taylor and W. H. Weinberg, submitted to Surface Sci.

19. G. Ehrlich, J. Appl. Phys. 32, 4 (1961).

20. J. L. Falconer and R. J. Madix, J. Catal. 48, 262 (1977).

21. P. A. Redhead, Vacuum 12, 203 (1962).

22. T. Engel, T. von dem Hagen and E. Bauer, Surface Sci. 62, 361 (1977).

23. H. P. Bonzel and J. J. Burton, Surface Sci. 52, 223 (1975).

24. G. Ertl and M. Neumann, Z. Physik. Chem. NF 90, 127 (1974).

25. C. A. Becker, J. P. Cowin, L. Wharton and D. J. Auerback, J. Chem. Phys. 67, 3394 (1977).

26. P. R. Norton, Surface Sci. 44 624 (1974).

27. D. A. King and M. G. Wells, Proc. R. Soc. Lond. A 339, 245 (1974).

28. W. H. Weinberg, L. M. Comrie and R. M. Lambert, J. Catal. 41, 489 (1976).

29. H. P. Bonzel and R. Ku, Surface Sci. 33, 91 (1972).

30. P. A. Thiel, J. T. Yates, Jr. and W. H. Weinberg, to be published.

31. T. Matsushima, D. B. Almy and J. M. White, Surface Sci. 67, 89 (1977).

32. T. Matsushima and J. M. White, Surface Sci. 67, 122 (1977).

Figure Captions

Figure 1: A typical thermal desorption spectrum of co-adsorbed CO and oxygen from Ir(110). The spectra for CO and CO_2, as well as the temporal variations in the coverages of CO and oxygen, are shown here. The spectrum of O_2 appears above 1000 K.

Figure 2: Representative thermal desorption spectra of CO and CO_2 from Ir(110). A clean Ir(110) surface at 300 K initially was exposed to 0.8 L of oxygen, which corresponds to a coverage of 0.25 ML, and then received varying exposures of CO as noted. The average heating rate for these spectra was 40 K/sec.

Figure 3: Zero-order Arrhenius plots of individual desorption spectra of CO from Ir(110). These plots were derived from the data in Fig. 2 for various initial coverages of CO as noted.

Figure 4: An order plot for CO desorption at 573 K. The slope of the line gives the order of the desorption process in the coverage of CO. The plot was derived from the data in Fig. 2.

Figure 5: Arrhenius plots for: (a) CO desorption, and (b) CO oxidation over Ir(110). The surface initially was covered with 0.5 ML of oxygen and then was covered with varying amounts of CO. The plots for CO desorption were derived by evaluating the CO desorption spectra at points of constant CO coverage. The plots for CO oxidation were derived by evaluating the CO_2 desorption spectra at points of constant oxygen coverage.

Figure 6: The variation in the activation energy and pre-exponential factor for CO desorption with the CO coverage on: (a) clean Ir(110) and (b) oxidized Ir(110).

Figure 7: The variation in the activation energy and pre-exponential factor for CO oxidation with the oxygen coverage on: (a) clean Ir(110) and (b) oxidized Ir(110). Rate parameters that pertain to the peak at lower temperature in the CO_2 spectra are hollow whereas parameters that pertain to the peak at higher temperature are solid. The parameters are appropriate for describing CO oxidation via the Langmuir-Hinshelwood mechanism only.

Figure 8: The total amount of CO_2 produced in the thermal desorption spectra for various co-adsorbed adlayers of oxygen and CO from: (a) clean Ir(110) and (b) oxidized Ir(110). The initial coverages of CO and oxygen are as shown. The oxide layer is composed of 0.25 ML of oxygen, which may be reduced also.

Figure 9: The adsorption kinetics of CO on: (a) clean Ir(110), and (b) oxidized Ir(110) surfaces that initially were covered with varying amounts of oxygen. Curves derived from the second-order precursor model are shown as solid lines.

Figure 10: The adsorption kinetics of oxygen on clean and oxidized Ir(110) at 300 K. The surfaces initially were exposed 0.6 L of CO, which corresponds to a coverage of 0.25 ML. Curves derived from second-order Langmuir models are shown as solid lines.

Figure 11: Equilibrium rates of CO_2 production as a function of the catalyst temperature and the ratio of the partial pressures of the reactants ($Q=P_{CO}/P_{O_2}$). Data are shown for total pressures of: (a) 3.0×10^{-7}, and (b) 3.0×10^{-6} torr. The arrows on the curves point to the right if the temperature is increasing and to the left if the temperature is decreasing.

Figure 12: Equilibrium rates of CO_2 production as a function of the catalyst temperature and of the catalyst history. Data are shown for a total pressure of 1.0×10^{-6} torr and for ratios of the partial pressures of the reactants ($Q=P_{CO}/P_{O_2}$) of:(a) 2.7/1 and (b) 1/4.4. The arrows on the curves point to the right if the temperature is increasing and to the left if the temperature is decreasing.

Figure 13: Zero-order Arrhenius plots for the variation in the equilibrium rates of CO_2 production with the catalyst temperature. Data are shown for various ratios of the partial pressures of the reactants ($Q=P_{CO}/P_{O_2}$) and for total pressures of: (a) 3.0×10^{-7}, and (b) 3.0×10^{-6} torr. The arrows on the curves point to the left if the temperature is increasing and to the right if the temperature is decreasing.

Figure 14: The variation in the apparent pre-exponential factor with the ratio of the oxygen pressure squared to the CO pressure. The apparent pre-exponential factors were derived from plots similar to those in Fig. 13 and pertain to the region where the apparent activation energy is positive. Only points

associated with an increasing surface temperature are shown.

Figure 15: The variation in the apparent pre-exponential factor with the partial pressure of CO. The apparent pre-exponential factors were derived from plots similar to those in Fig. 13 and pertain to the region where the apparent activation energy is negative. Points associated with an increasing surface temperature are hollow whereas points associated with a decreasing surface temperature are solid.

Figure 16: The variation in the ratio $Q(=P_{CO}/P_{O_2})$ with the temperature associated with the maximum equilibrium rate of CO_2 production. Points pertaining to clean Ir(110) are hollow whereas points pertaining to oxidized Ir(110) are solid.

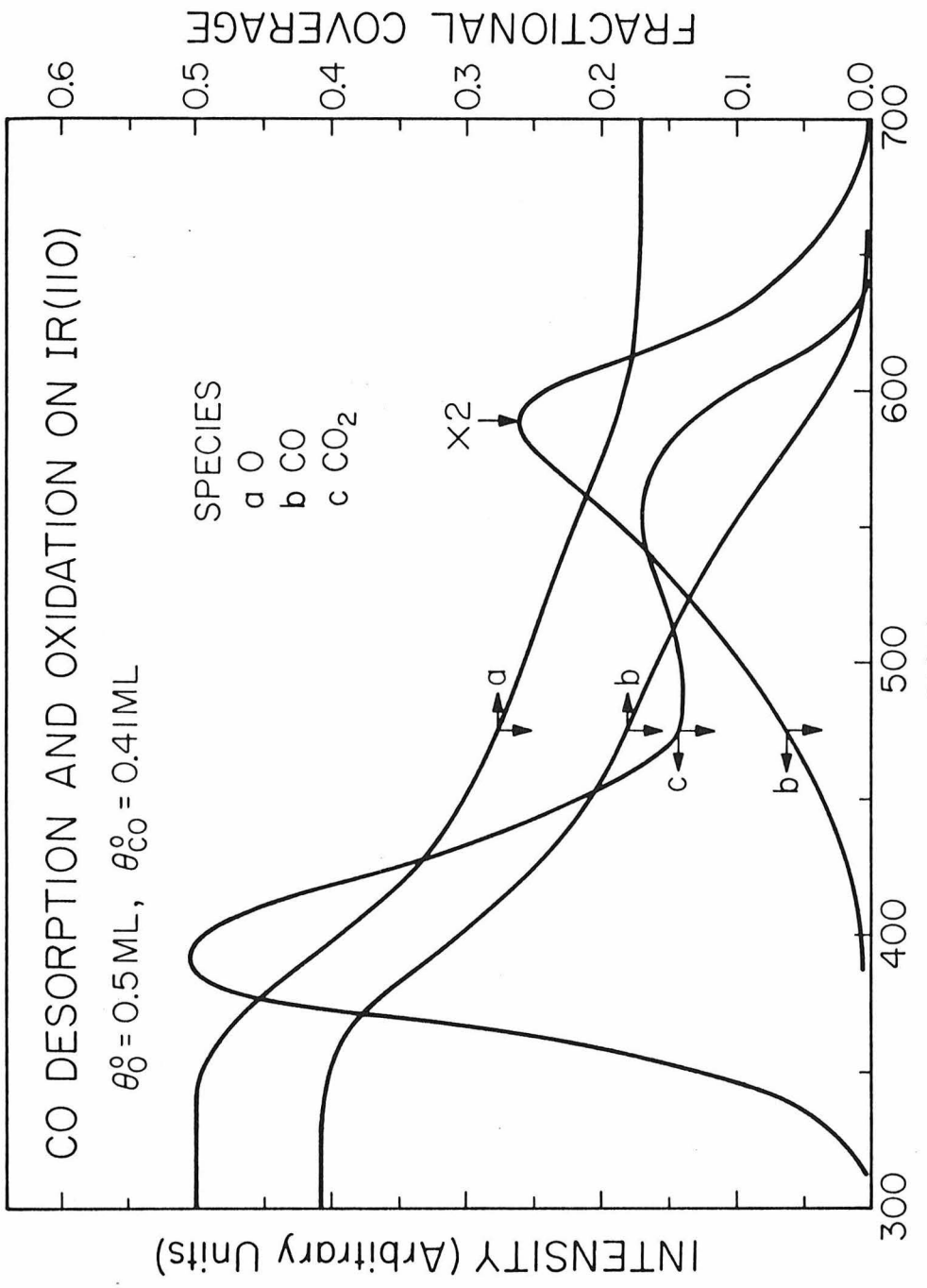

CO DESORPTION AND OXIDATION ON IR(110)
θ_O^o = 0.5 ML, θ_{CO}^o = 0.41 ML
SPECIES
a O
b CO
c CO_2
X2
a
b
c
b
INTENSITY (Arbitrary Units)
T(K)
FRACTIONAL COVERAGE
300
400
500
600
700
0.0
0.1
0.2
0.3
0.4
0.5
0.6

Fig. 1

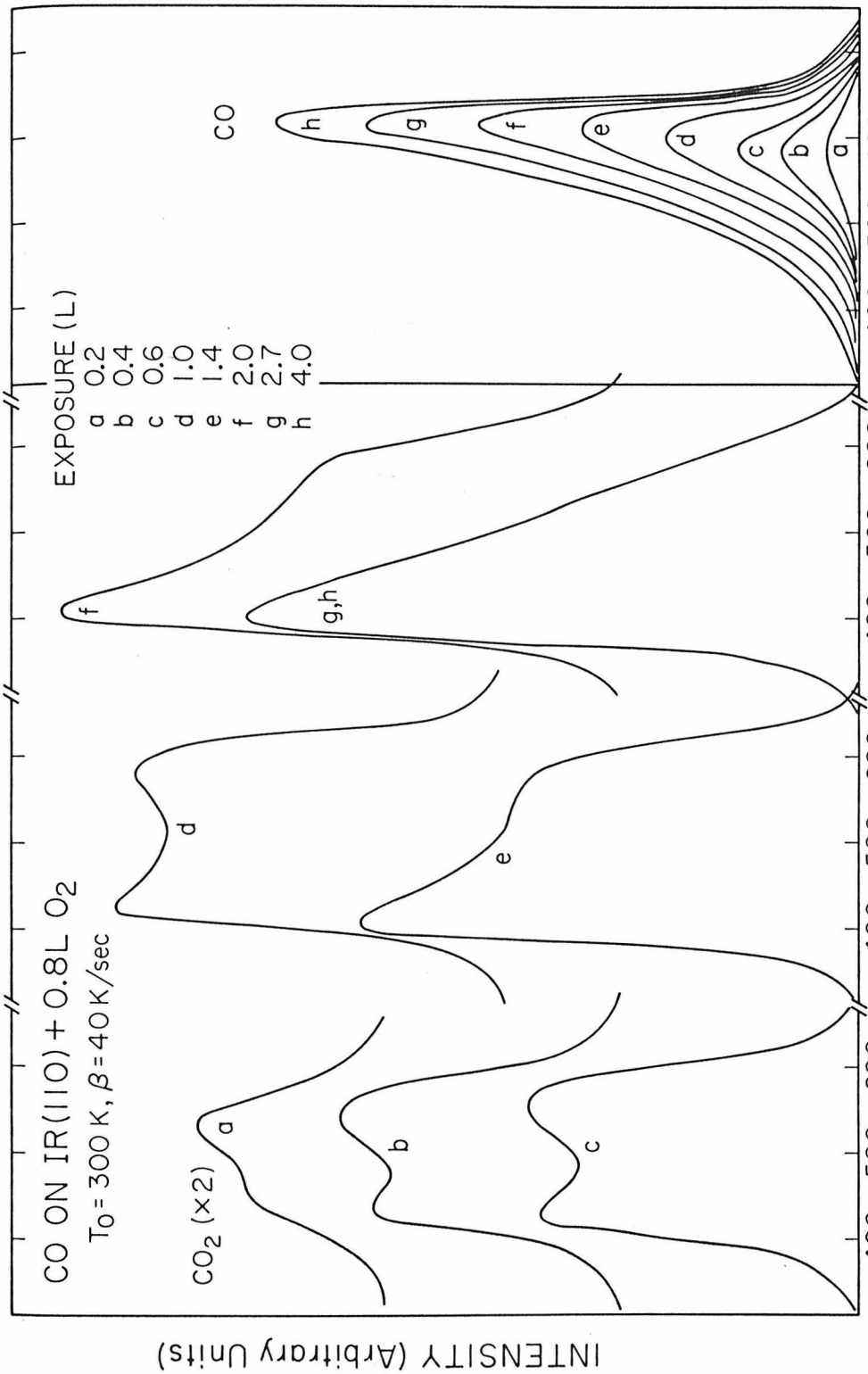

CO ON IR(110) + 0.8L O2
T0 = 300 K, β = 40 K/sec
CO2 (x2)
CO
EXPOSURE (L)
a 0.2
b 0.4
c 0.6
d 1.0
e 1.4
f 2.0
g 2.7
h 4.0
INTENSITY (Arbitrary Units)
TEMPERATURE (K)
400
500
600
700

Fig. 2

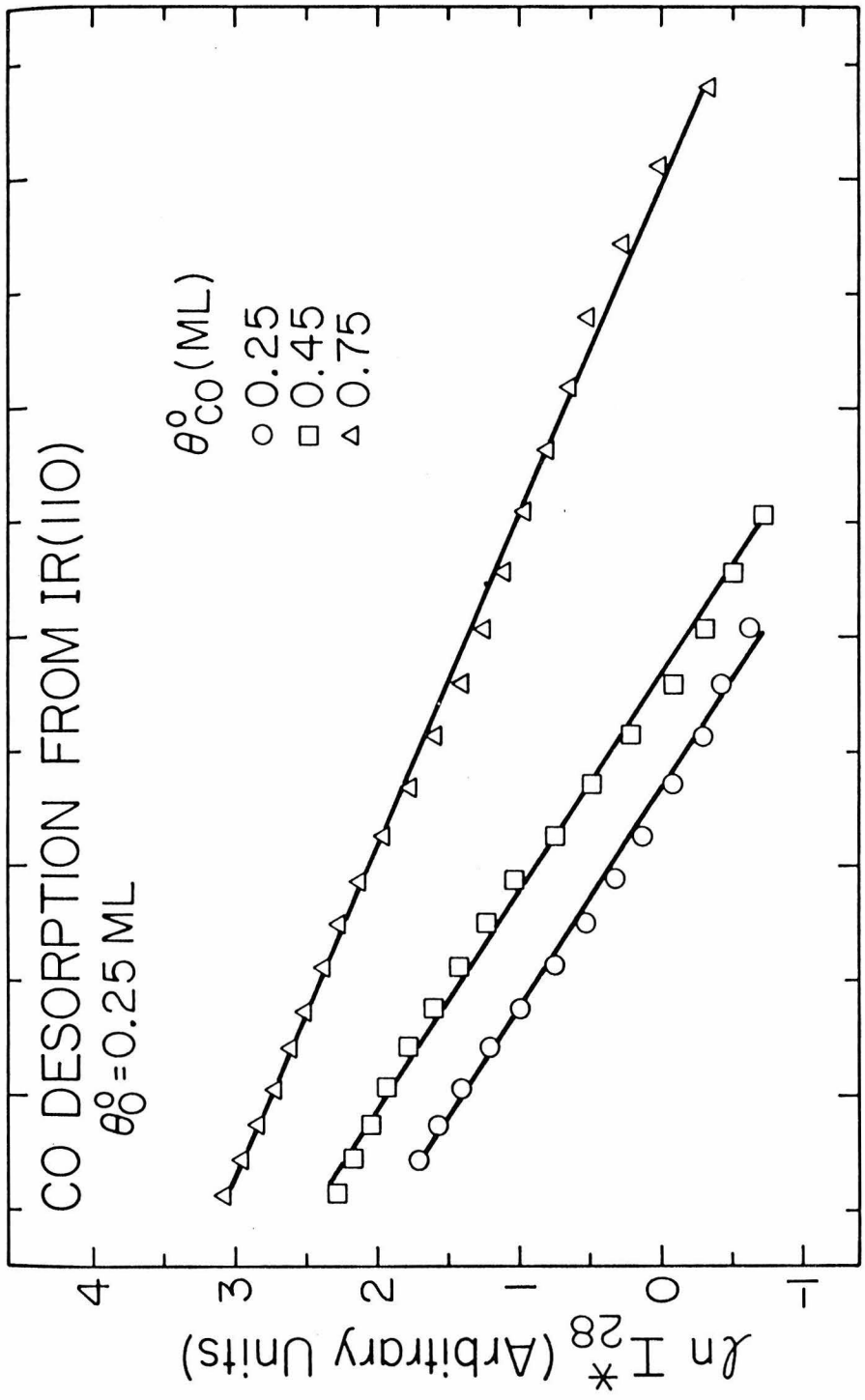

CO DESORPTION FROM IR(110)
θ_O^o=0.25 ML
θ_{CO}^o(ML)
○ 0.25
□ 0.45
△ 0.75
ln I_{28}^* (Arbitrary Units)
$10^3/T(K^{-1})$
4
3
2
1
0
-1
1.8
2.0
2.2
2.4
2.6

Fig. 3

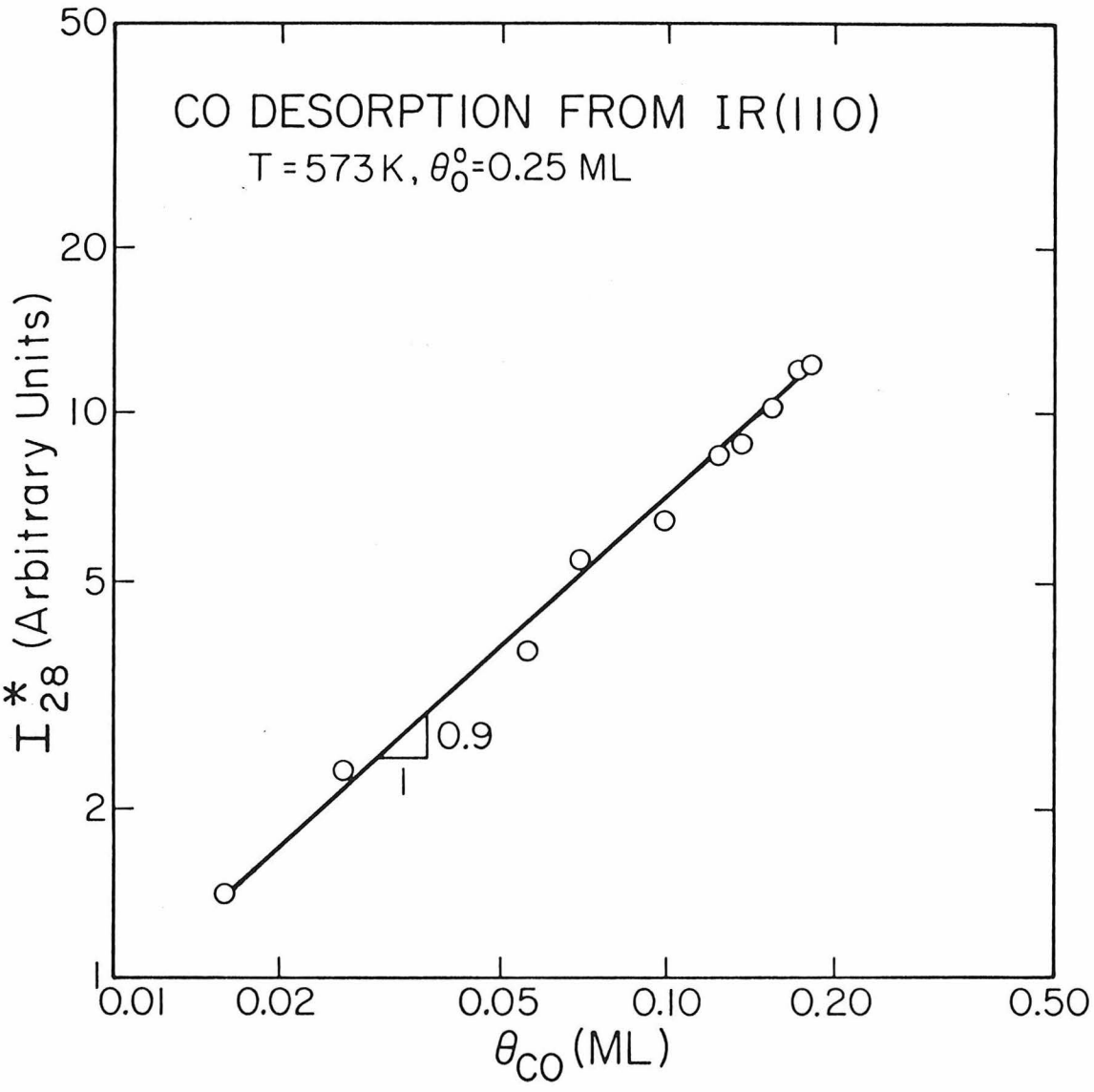

CO DESORPTION FROM IR(110)
T = 573 K, θ_O^0 = 0.25 ML
I_{28}^* (Arbitrary Units)
θ_{CO} (ML)
50
20
10
5
2
1
0.01
0.02
0.05
0.10
0.20
0.50
0.9
1

Fig.4

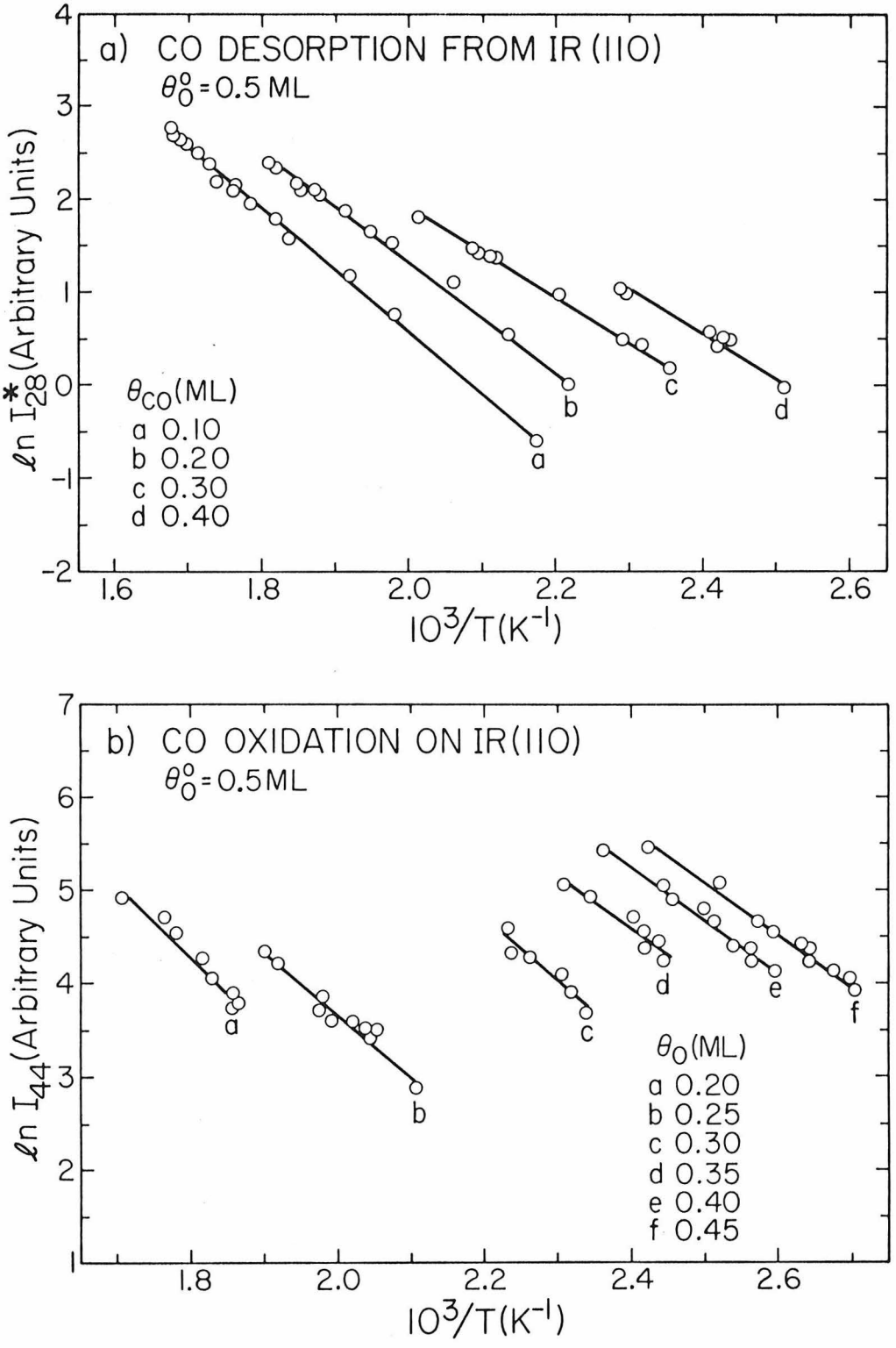

a) CO DESORPTION FROM IR (110)
θ_0^0 = 0.5 ML
$\ell n\, I_{28}^*$ (Arbitrary Units)
θ_{CO}(ML)
a 0.10
b 0.20
c 0.30
d 0.40
$10^3/T(K^{-1})$
b) CO OXIDATION ON IR(110)
θ_0^0 = 0.5ML
$\ell n\, I_{44}$ (Arbitrary Units)
θ_0(ML)
a 0.20
b 0.25
c 0.30
d 0.35
e 0.40
f 0.45
$10^3/T(K^{-1})$

Fig.5

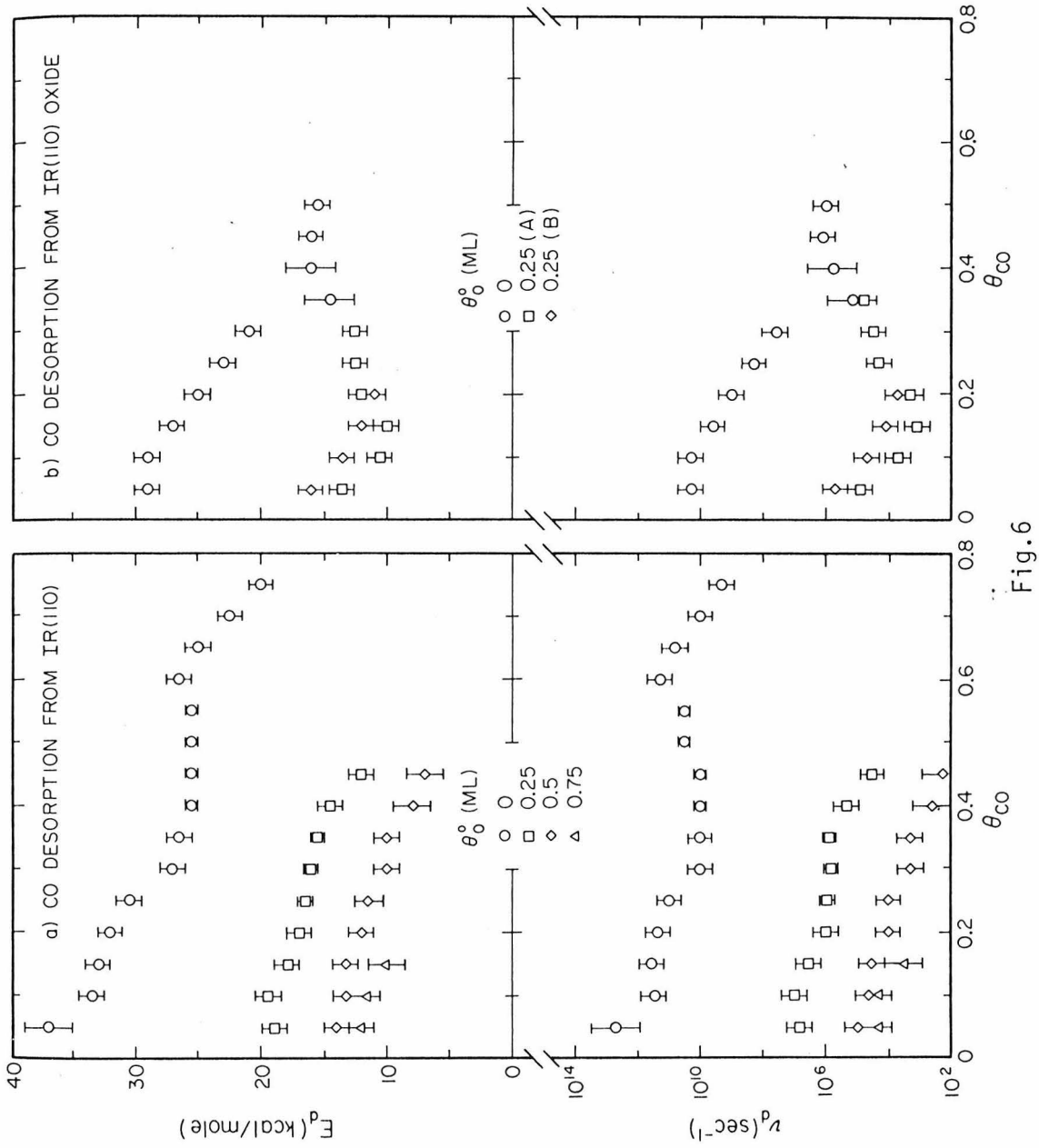

a) CO DESORPTION FROM IR(110)
b) CO DESORPTION FROM IR(110) OXIDE
E_d(kcal/mole)
ν_d(sec^{-1})
θ_{CO}
θ_0^o (ML)
0
0.25
0.5
0.75
0.25 (A)
0.25 (B)

Fig.6

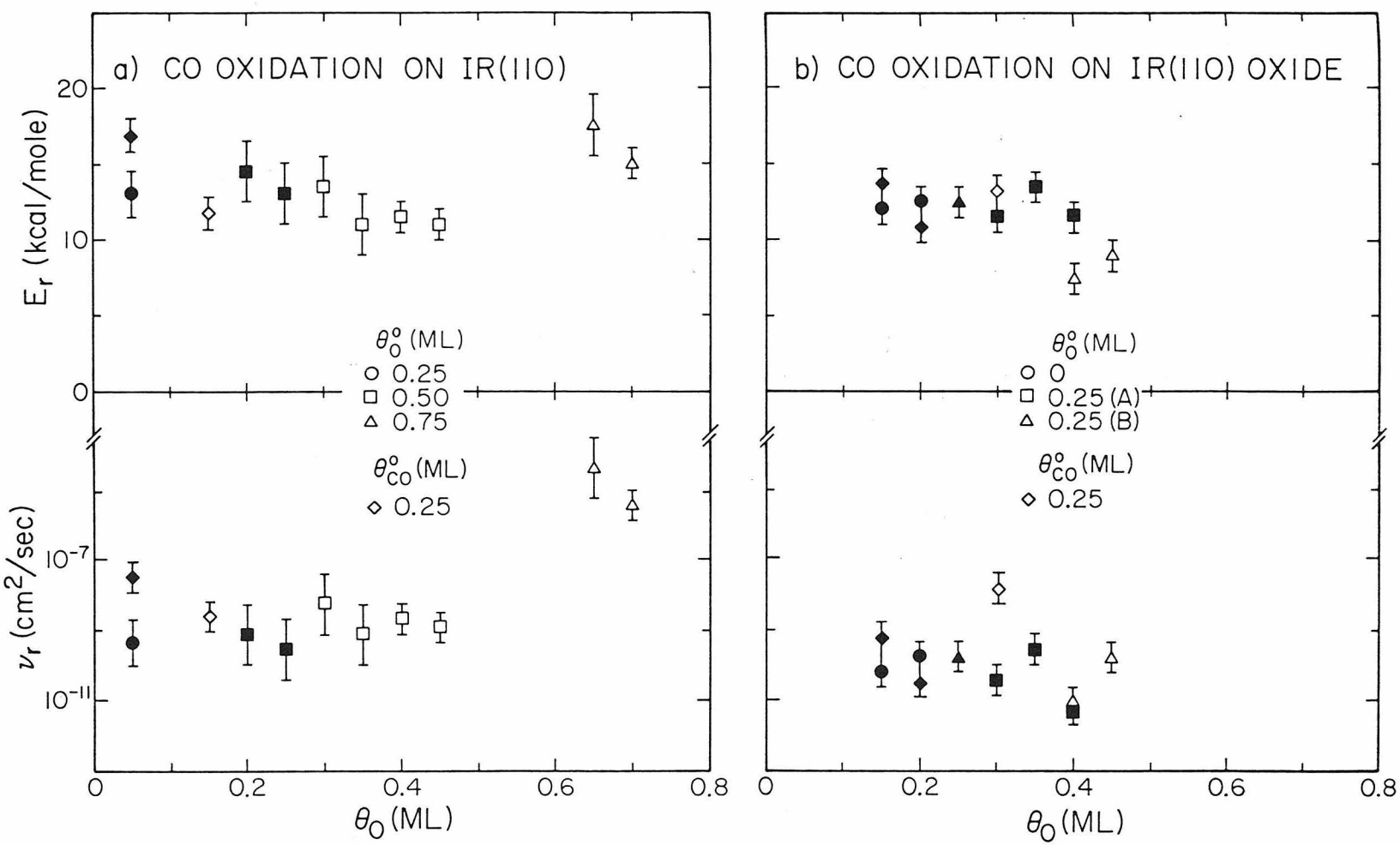

a) CO OXIDATION ON IR(110)
b) CO OXIDATION ON IR(110) OXIDE
E_r (kcal/mole)
ν_r (cm^2/sec)
θ_O^o (ML)
0.25
0.50
0.75
θ_{CO}^o (ML)
0.25
θ_O^o (ML)
0
0.25 (A)
0.25 (B)
θ_{CO}^o (ML)
0.25
θ_O (ML)
θ_O (ML)

Fig.7

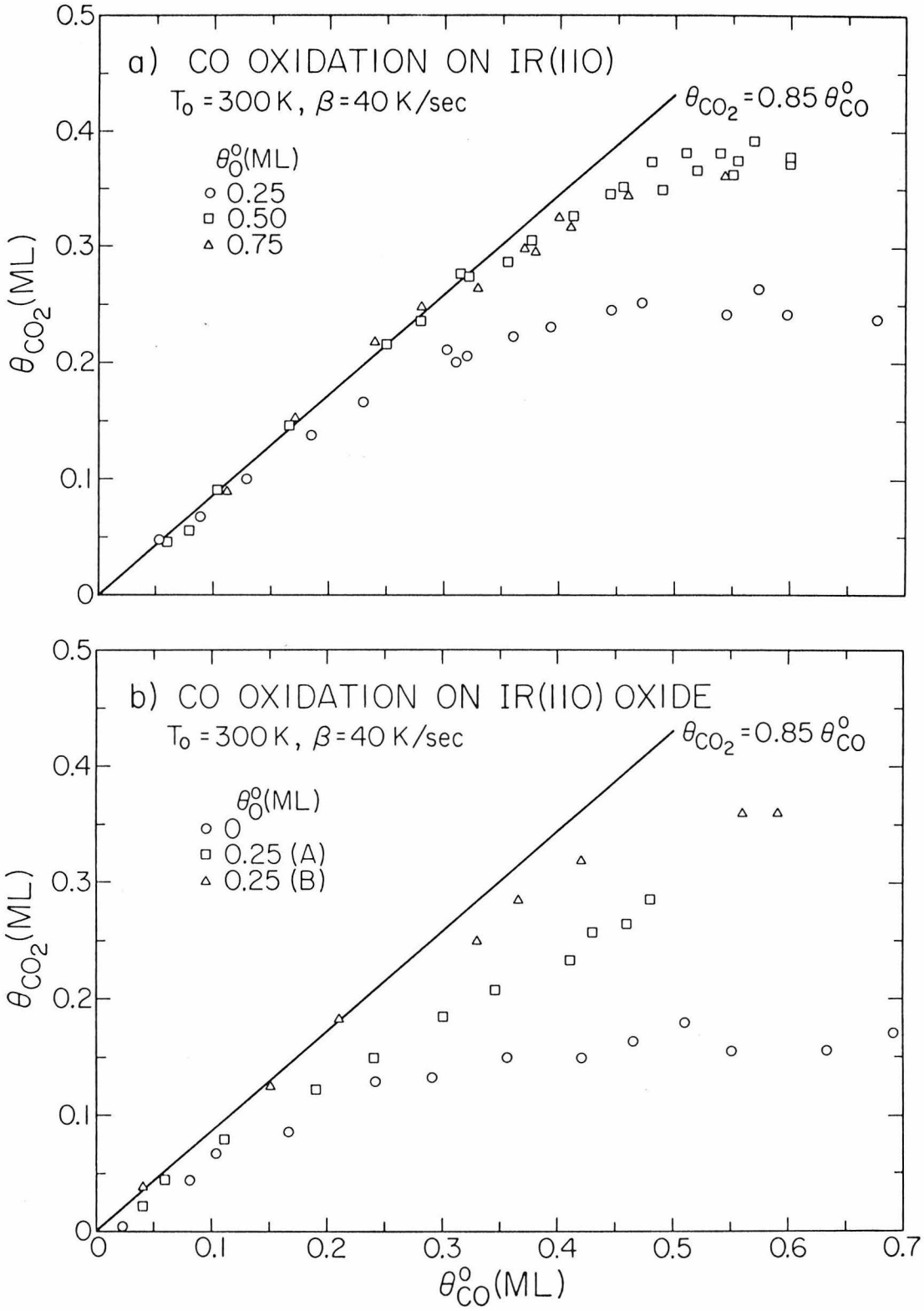

a) CO OXIDATION ON IR(110)
T_0 = 300 K, β = 40 K/sec
$\theta_{CO_2} = 0.85\,\theta^0_{CO}$
θ^0_O(ML)
0.25
0.50
0.75
θ_{CO_2}(ML)
b) CO OXIDATION ON IR(110) OXIDE
T_0 = 300 K, β = 40 K/sec
$\theta_{CO_2} = 0.85\,\theta^0_{CO}$
θ^0_O(ML)
0
0.25 (A)
0.25 (B)
θ_{CO_2}(ML)
θ^0_{CO}(ML)

Fig.8

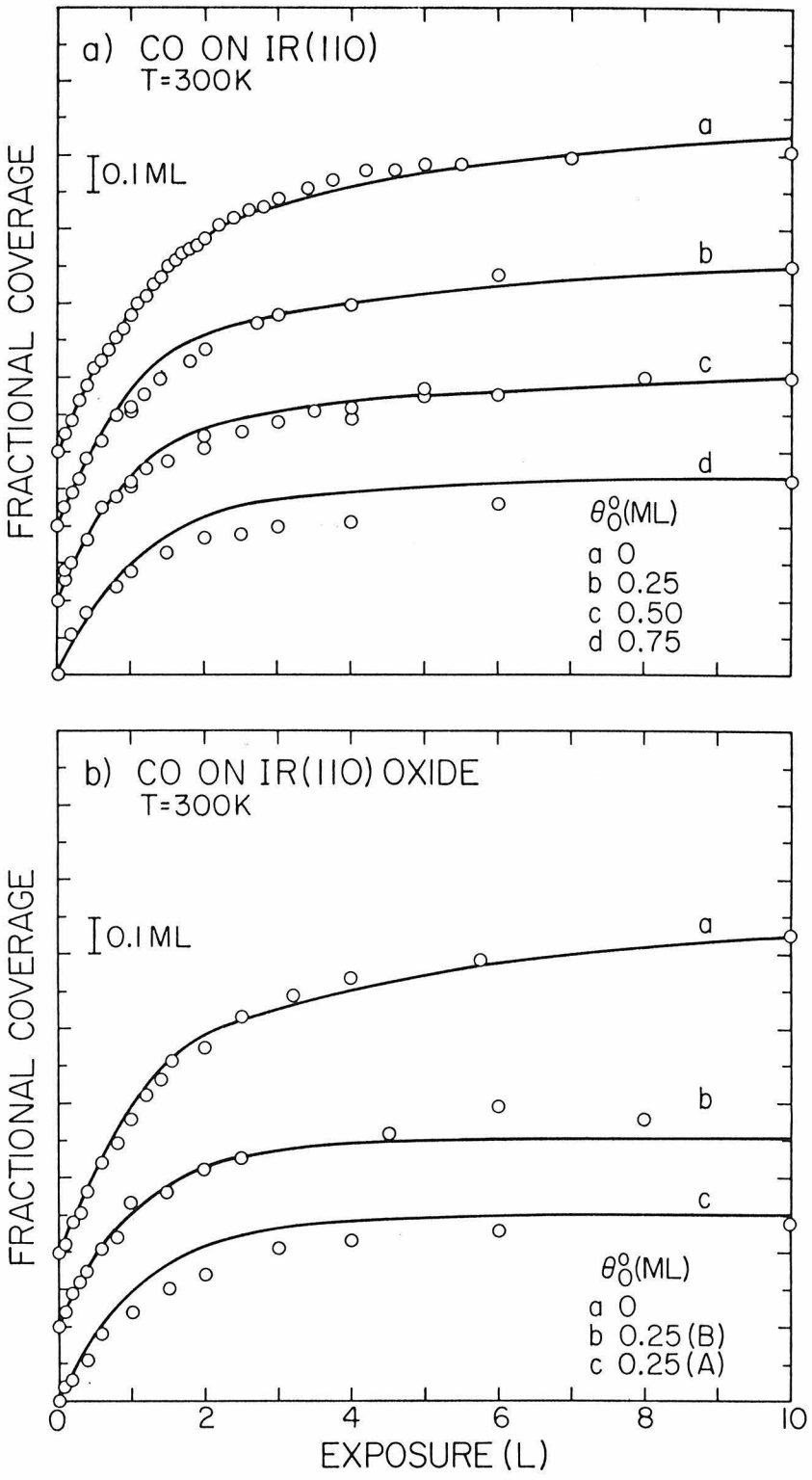

a) CO ON IR(110)
T=300K
0.1 ML
FRACTIONAL COVERAGE
a
b
c
d
θ_O^o(ML)
a 0
b 0.25
c 0.50
d 0.75
b) CO ON IR(110) OXIDE
T=300K
0.1 ML
FRACTIONAL COVERAGE
a
b
c
θ_O^o(ML)
a 0
b 0.25(B)
c 0.25(A)
0
2
4
6
8
10
EXPOSURE (L)

Fig.9

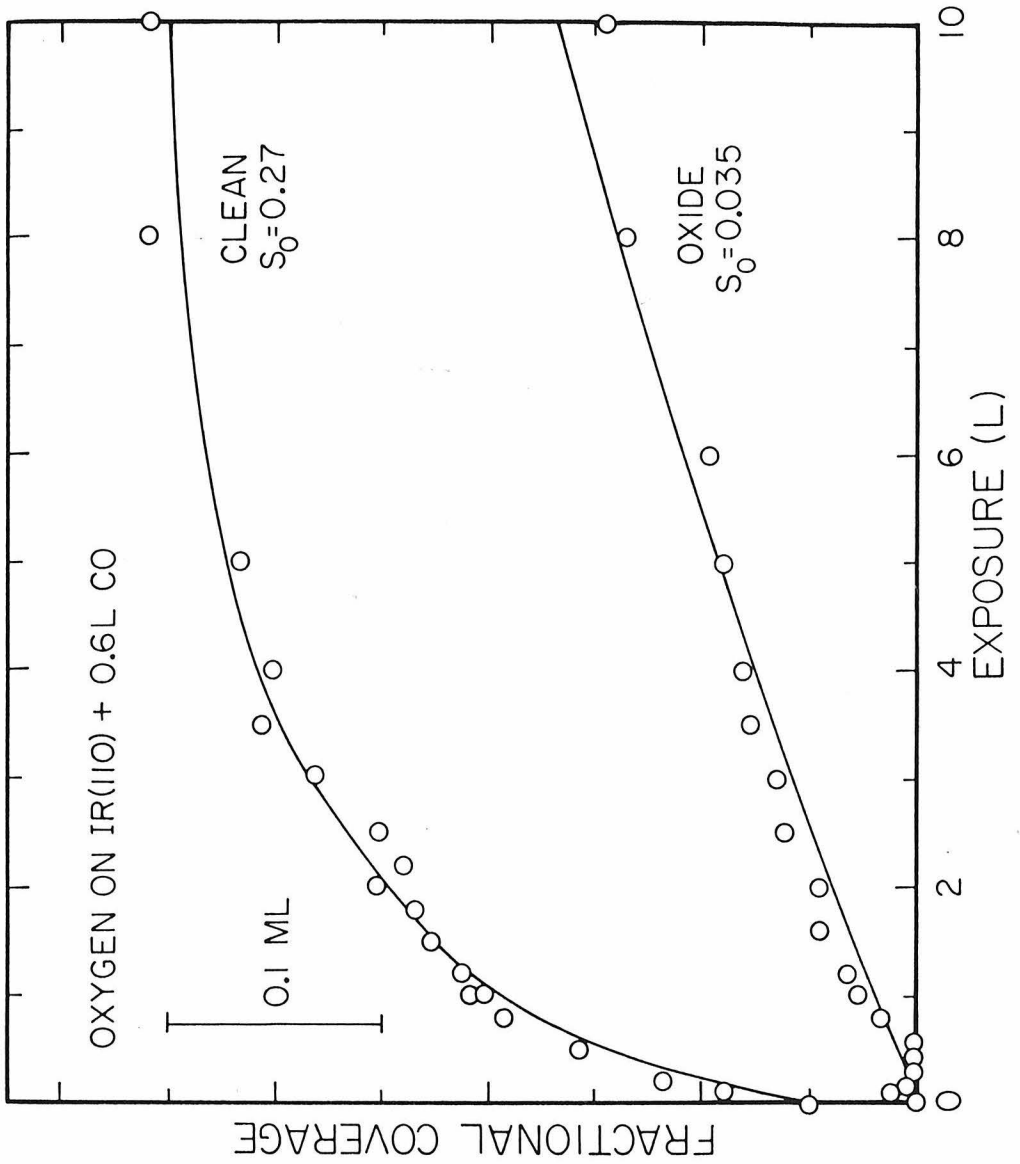

OXYGEN ON IR(110) + 0.6L CO
CLEAN
S_0=0.27
OXIDE
S_0=0.035
0.1 ML
FRACTIONAL COVERAGE
EXPOSURE (L)
0
2
4
6
8
10

Fig.10

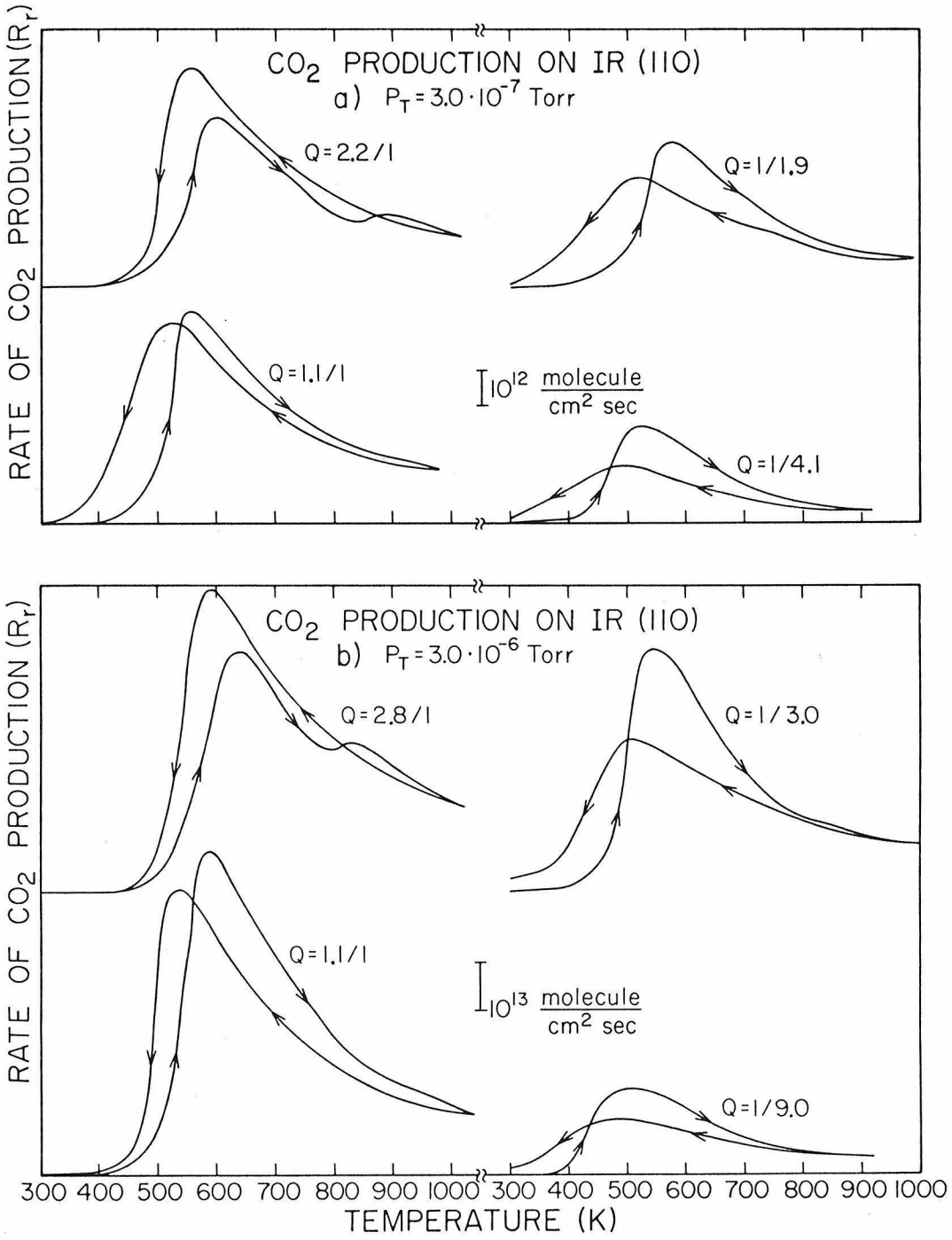

CO2 PRODUCTION ON IR (110)
a) PT = 3.0·10-7 Torr
Q = 2.2/1
Q = 1/1.9
Q = 1.1/1
10^12 molecule / cm^2 sec
Q = 1/4.1
CO2 PRODUCTION ON IR (110)
b) PT = 3.0·10-6 Torr
Q = 2.8/1
Q = 1/3.0
Q = 1.1/1
10^13 molecule / cm^2 sec
Q = 1/9.0
RATE OF CO2 PRODUCTION (Rr)
300 400 500 600 700 800 900 1000 300 400 500 600 700 800 900 1000
TEMPERATURE (K)

Fig.11

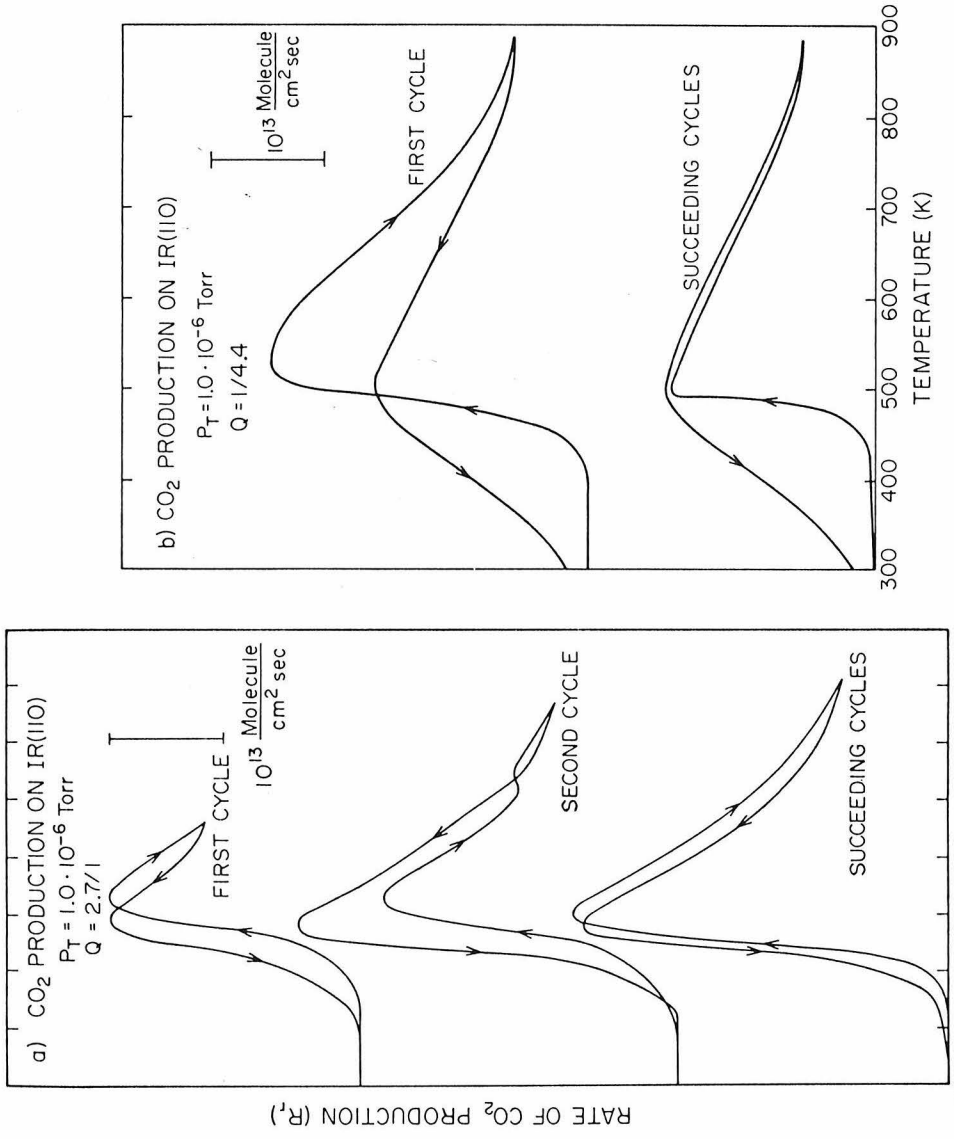

a) CO_2 PRODUCTION ON IR(110)
$P_T = 1.0 \cdot 10^{-6}$ Torr
Q = 2.7/1
FIRST CYCLE
10^{13} Molecule / cm² sec
SECOND CYCLE
SUCCEEDING CYCLES
RATE OF CO_2 PRODUCTION (R_j)
300
500
700
900
TEMPERATURE (K)
b) CO_2 PRODUCTION ON IR(110)
$P_T = 1.0 \cdot 10^{-6}$ Torr
Q = 1/4.4
10^{13} Molecule / cm² sec
FIRST CYCLE
SUCCEEDING CYCLES
300
400
500
600
700
800
900
TEMPERATURE (K)

Fig.12

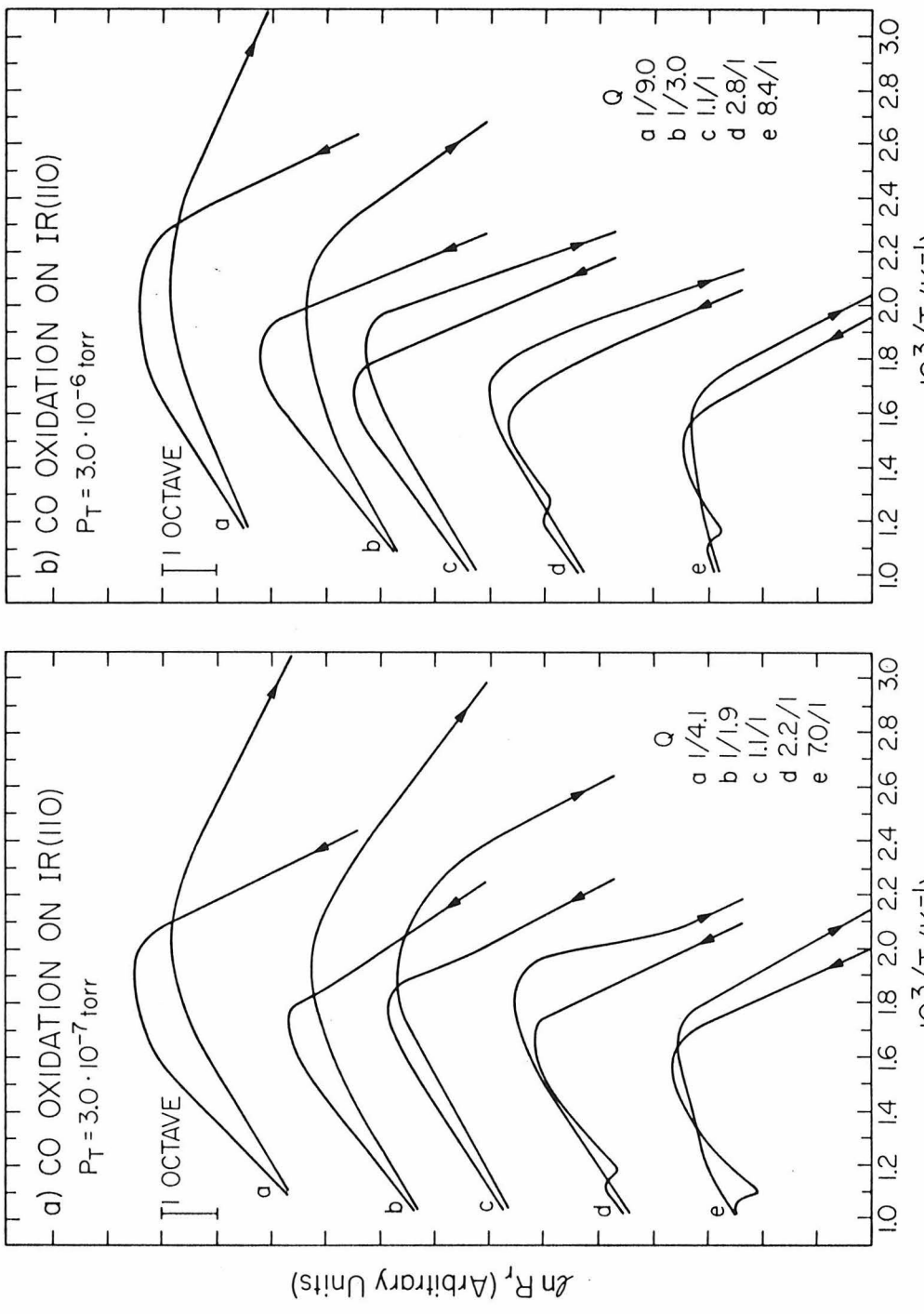

a) CO OXIDATION ON IR(110)
$P_T = 3.0 \cdot 10^{-7}$ torr
b) CO OXIDATION ON IR(110)
$P_T = 3.0 \cdot 10^{-6}$ torr
1 OCTAVE
a
b
c
d
e
Q
a 1/4.1
b 1/1.9
c 1.1/1
d 2.2/1
e 7.0/1
Q
a 1/9.0
b 1/3.0
c 1.1/1
d 2.8/1
e 8.4/1
1.0 1.2 1.4 1.6 1.8 2.0 2.2 2.4 2.6 2.8 3.0
$10^3/T$ (K^{-1})
$\mathcal{R}n\ R_r$ (Arbitrary Units)

Fig.13

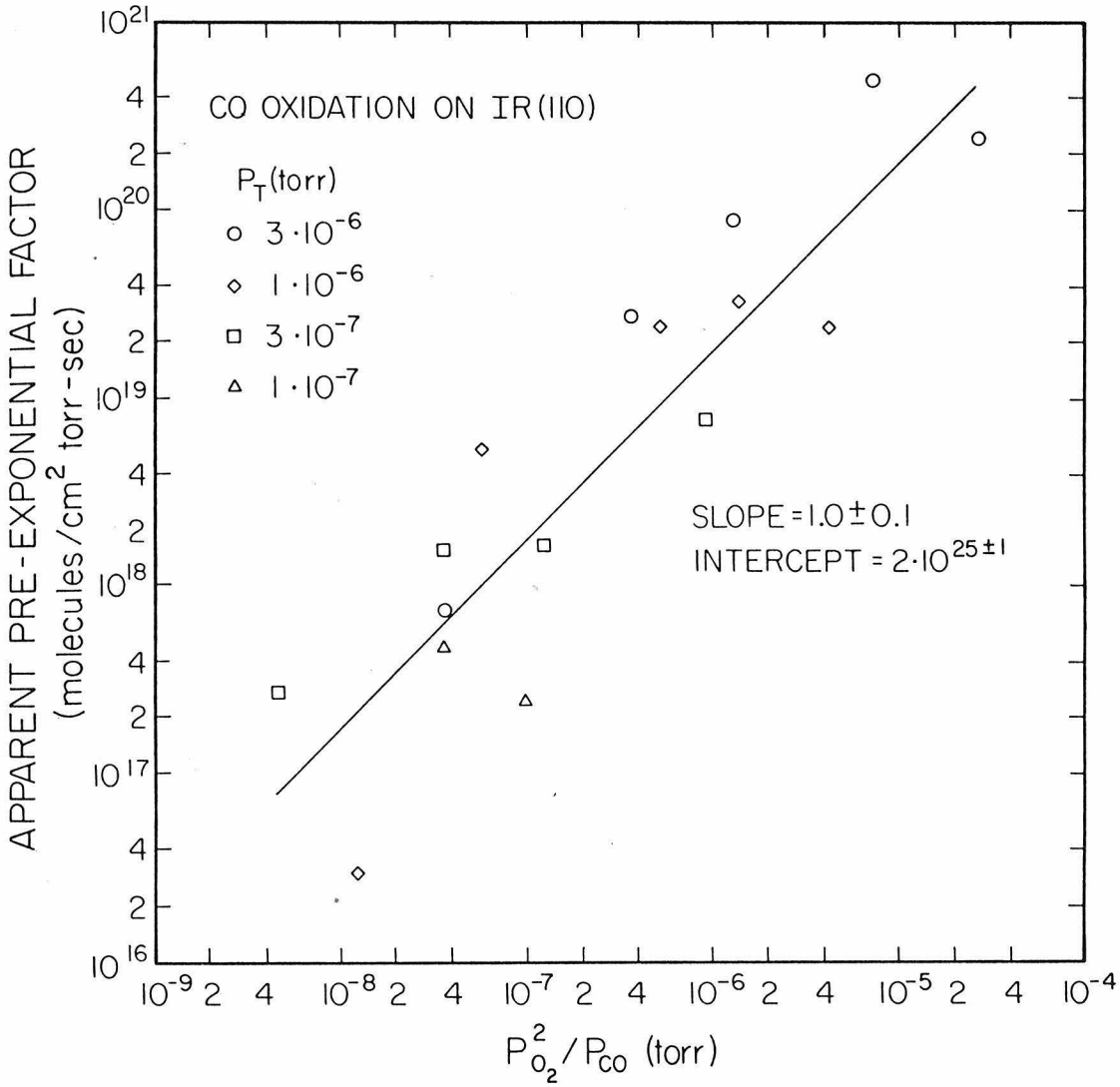

CO OXIDATION ON IR(110)
P_T(torr)
○ $3\cdot10^{-6}$
◇ $1\cdot10^{-6}$
□ $3\cdot10^{-7}$
△ $1\cdot10^{-7}$
SLOPE = 1.0 ± 0.1
INTERCEPT = $2\cdot10^{25\pm1}$
APPARENT PRE-EXPONENTIAL FACTOR
(molecules/cm^2 torr-sec)
$P_{O_2}^2/P_{CO}$ (torr)

Fig.14

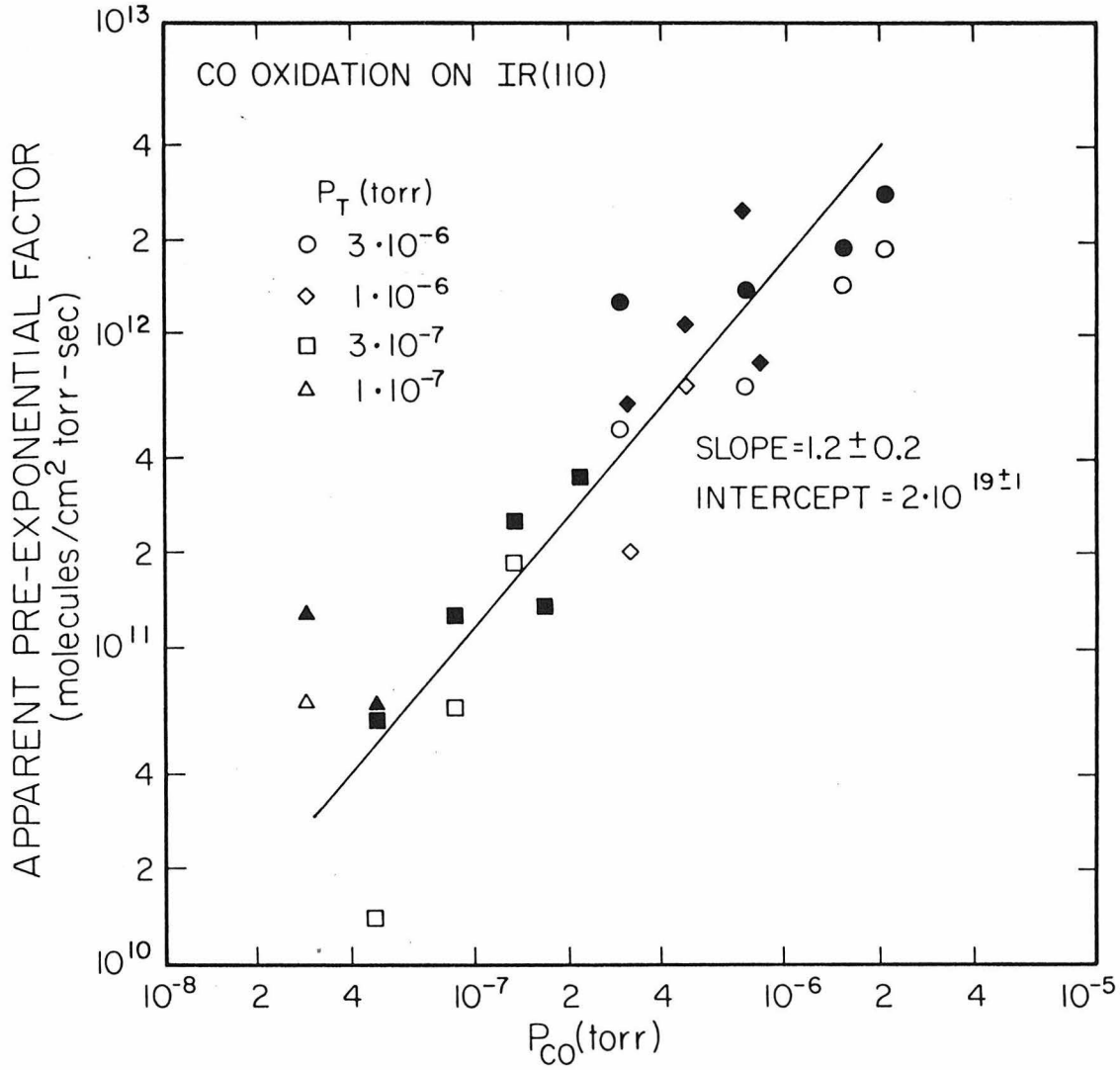

CO OXIDATION ON IR(110)
P_T (torr)
○ $3 \cdot 10^{-6}$
◇ $1 \cdot 10^{-6}$
□ $3 \cdot 10^{-7}$
△ $1 \cdot 10^{-7}$
SLOPE = 1.2 ± 0.2
INTERCEPT = $2 \cdot 10^{19 \pm 1}$
APPARENT PRE-EXPONENTIAL FACTOR
(molecules/cm^2 torr-sec)
P_{CO}(torr)
10^{13}
10^{12}
10^{11}
10^{10}
10^{-8}
10^{-7}
10^{-6}
10^{-5}
2
4

Fig.15

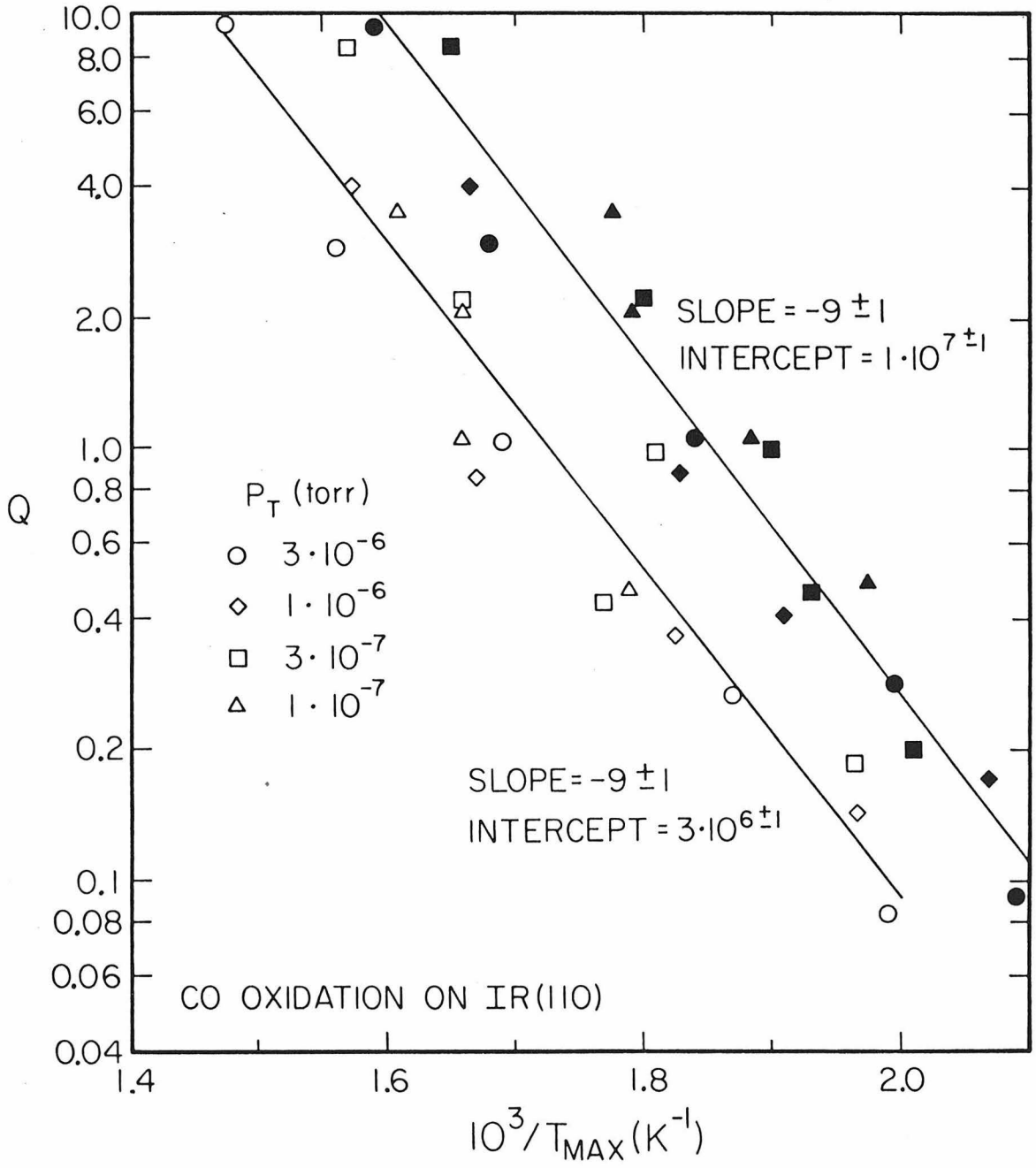

10.0
8.0
6.0
4.0
2.0
1.0
0.8
0.6
0.4
0.2
0.1
0.08
0.06
0.04
Q
1.4
1.6
1.8
2.0
$10^3/T_{MAX}(K^{-1})$
P_T (torr)
○ $3\cdot10^{-6}$
◇ $1\cdot10^{-6}$
□ $3\cdot10^{-7}$
△ $1\cdot10^{-7}$
SLOPE = -9 ± 1
INTERCEPT = $1\cdot10^{7\pm1}$
SLOPE = -9 ± 1
INTERCEPT = $3\cdot10^{6\pm1}$
CO OXIDATION ON IR(110)

Fig.16

CHAPTER 6

CONCLUSIONS

The heterogeneously catalyzed reaction of gaseous CO and O_2 to form CO_2 over Ir(110), as well as the chemisorption of the reactants, has been studied at low pressures ($<10^{-5}$ torr). These experiments were performed with four surface sensitive probes - thermal desorption mass spectrometry (TDS), contact potential difference (CPD) measurements, Auger electron spectroscopy (AES) and LEED. Oxidation of the surface complicated several aspects of this study; that is, a second, tightly bound, dissociately chemisorbed state of oxygen forms irreversibly on Ir(110) near 700 K. Hence, several of the experiments were performed for both clean and oxidized surfaces of Ir(110). For the conclusions presented below, oxidized Ir(110) will refer to a surface having a saturated oxide layer whereas clean Ir(110) will refer to a surface having no oxide layer. The conclusions below apply to both clean and oxidized Ir(110), except as noted otherwise.

The following conclusions pertain to CO chemisorption on Ir(110).

(1) For pressures below 10^{-6} torr and for temperatures below 1300 K, CO adsorbs molecularly, rather than dissociatively, on Ir(110). The saturation coverage of CO is 1 ML ($=10^{15}cm^{-2}$) at 90 and 300 K.

(2) For surface temperatures above 90 K, the initial probability of adsorption, i.e. the initial sticking probability, is unity. At larger coverages, the adsorption kinetics are described by a second-order precursor model that accounts for ordering of the adlayer by intermolecular repulsion. The kinetics are second-order because two empty sites are needed for a CO molecule to adsorb, not because adsorption is dissociative.

(3) The CO overlayer on clean Ir(110) appears to compress continuously as the p(2x2) LEED pattern for 0.25 ML coverage is transformed into the (4x2) pattern for 0.75 ML coverage. However, since the (2x1)p1g1 pattern is observed for all coverages above 0.05 ML, the overlayer on oxidized Ir(110) does not compress; instead, the adsorbed CO molecules coalesce into islands.

(4) The compensation effect occurs for CO desorption; that is, at any coverage, the activation energy is proportional to the logarithm of the pre-exponential factor. The activation energy decreases slowly up to 0.25 ML coverage, drops precipitously there, is nearly constant up to 0.6 ML coverage and then decreases rapidly once again. In the limit of small coverage, the activation energy and pre-exponential factor for CO desorption are 37 kcal/mole and $10^{13}sec^{-1}$ on clean Ir(110), and are 31 kcal/mole and $10^{11}sec^{-1}$ on oxidized Ir(110).

(5) The coverage dependence of the CPD upon CO chemisorption is fit by a modified Topping model. That is, the CPD is proportional to the coverage multiplied by the average component of the dipole moment of the CO that is normal to the surface, but the average normal component of the dipole moment decreases with increasing coverage because dipole-dipole interactions depolarize the adlayer. The CPD for any coverage of CO changes with temperature because the adsorbates are energetically partitioned between binding sites of different polarizabilities and dipole moments. The decrease in the dipole moment of CO with increasing temperature indicates that sites between the topmost rows of surface atoms are favored energetically whereas sites above the rows are disfavored energetically.

(6) A significant error may be introduced in the isosteric heat of adsorption if the Clausius-Clapeyron equation is evaluated for points of constant CPD, rather than points of constant coverage.

The following conclusions pertain to oxygen chemisorption on Ir(110).

(1) Oxygen may adsorb into three distinct chemical states: a molecularly chemisorbed species, a dissociatively chemisorbed species, and a surface oxide.

(2) Above 200 K, oxygen adsorption is dissociative since the activation energy for dissociating molecularly chemisorbed oxygen is roughly 8 kcal/mole. As indicated by CPD measurements, the molecularly chemisorbed species binds with the interatomic axis parallel to the surface and fills two sites above the topmost rows of surface atoms.

(3) The surface oxide forms rapidly above 700 K and saturates at a coverage of 0.25 ML. Although the oxide layer may be ordered, giving a (1x4) LEED pattern, or amorphous, giving a (1x1) LEED pattern with streaks between the beams in the [001] azimuth, the chemical properties for these two forms of oxide are otherwise identical. The surface oxide, compared to chemisorbed oxygen is less reactive to reduction by CO.

(4) The saturation coverage of chemisorbed oxygen is 1 ML for clean or oxidized Ir(110) at 300 K, but is reduced to 0.5 ML for clean Ir(110) near 600 K.

(5) Oxygen chemisorption is not activated. Regardless of the surface temperature, the initial sticking probability is 0.28 on clean Ir(110) and is 0.4 on oxidized Ir(110). For larger coverages, the adsorption kinetics for the clean surface are fit consistently by a

second-order precursor model; that is, the logarithm of the measured lifetime for the precursor is inversely proportional to the surface temperature. From this relation, the difference in the activation energies for the desorption and the diffusion of the precursor is 2.1 kcal/mole.

(6) Oxygen desorbs molecularly with an activation energy between 45 and 70 kcal/mole, which decreases continuously with increasing coverage. The pre-exponential factor for desorption is 3.5×10^{-4} cm^2/sec at 0.35 ML coverage. The oxide layer forms rapidly as chemisorbed oxygen desorbs. The desorption kinetics are second-order in the oxygen coverage above 0.1 ML, but are first-order at lower coverages since the decomposition of the oxide, rather than the union of chemisorbed atoms, is rate limiting.

(7) The activation energy for diffusion of dissociatively chemisorbed oxygen is about 30 kcal/mole for diffusion in the [001] azimuth and around 1 kcal/mole for diffusion in the [110] azimuth.

(8) The variation in the CPD with coverage and surface temperature upon oxygen chemisorption may be modeled like the variation upon CO chemisorption. CPD measurements indicate that, as coverage increases, an overlayer of chemisorbed oxygen undergoes a structural transformation at a fixed coverage that does not vary measurably with the surface temperature. During this transformation, the oxygen atoms move from sites above the topmost rows of surface atoms to sites between the rows. On clean Ir(110), this transformation occurs near 0.25 ML coverage where the p(2x2) LEED pattern appears with maximum intensity; on oxidized

Ir(110), this transformation occurs near 0.50 ML coverage where the c(2x2) LEED pattern appears with maximum intensity.

The following conclusions pertain to the heterogeneously catalyzed reaction of gaseous CO and O_2 to form CO_2 over Ir(110).

(1) For surface temperatures from 300 to 1000 K and for partial pressures of the reactants from $1x10^{-8}$ to $3x10^{-6}$ torr, four elementary reactions - the desorption of CO, the oxidation of CO via the Langmuir-Hinshelwood mechanism, the adsorption of CO and the adsorption of oxygen - may limit the rate of CO_2 production under equilibrium conditions. Three reactions - the oxidation of CO via the Eley-Rideal mechanism, the desorption of oxygen, and the desorption of CO_2 - do not affect the rate perceptibly.

(2) The kinetics of CO desorption and CO oxidation via the Langmuir-Hinshelwood mechanism may be modeled accurately by Arrhenius expressions. However, the rate parameters in these expressions - the pre-exponential factor and the activation energy - vary markedly with the composition of the adlayer. The dependence in the rate resulting from the coverage dependence of the rate parameters may overshadow completely the dependence resulting from leading coefficients that vary as coverages to integral exponents. Moreover, the rate of an elementary reaction may depend on the coverage of a species even though the species does appear in the chemical equation for the elementary reaction.

(3) Although the pre-exponential factor and the activation energy for CO desorption and CO oxidation vary markedly with the composition of the adlayer, average values for these rate parameters that pertain to the oxidation of CO under equilibrium conditions may be estimated.

The average activation energy and pre-exponential factor for CO desorption are 17 kcal/mole and $10^6 sec^{-1}$ on clean Ir(110), and are 14 kcal/mole and $10^5 sec^{-1}$ on oxidized Ir(110). These values differ markedly from those obtained when chemisorbed oxygen is absent from the surface. The average activation energy and pre-exponential factor for CO oxidation are 12 kcal/mole and $3x10^{-10} cm^2/sec$ on either surface.

(4) The relatively small values for the rate parameters noted above, as well as the O-p(2x2) and O-c(2x2) LEED patterns observed as the CO oxidation reaction occurs under equilibrium conditions, indicate that the rates of diffusion for chemisorbed CO and oxygen may limit the equilibrium rate of CO_2 production. These rates were not measured directly in this study, but were included empirically in the Arrhenius expressions for CO desorption and CO oxidation.

(5) The co-adsorption of CO and oxygen is not, in the strictest sense, competitive; that is, chemisorbed CO blocks sites for oxygen adsorption, but chemisorbed oxygen does not block sites for CO chemisorption. When co-adsorption is occurring, the adsorption rates for both CO and oxygen are fit by Langmuir-like expressions. The rate of CO adsorption is second-order in the coverage of CO whereas the rate of oxygen adsorption is second-order in the sum of the coverages of CO and oxygen and depends on the order in the adlayer. Regardless of the coverage of oxygen, the initial sticking probability for CO is unity. If the sites blocked by CO are accounted for, the initial sticking probability for oxygen is about 0.1.

(6) Models for the rate of CO_2 production under equilibrium conditions, based on the kinetic expressions for the four elementary

reactions that may limit the rate, give a consistent interpretation of qualititive and quantitative trends in the equilibrium rate. These trends include the variation in the equilibrium rate when the surface temperature is varied between 300 and 1000 K and when the partial pressures of the reactants are varied between $1x10^{-8}$ and $3x10^{-6}$ torr.

(7) Below 600 K, where the equilibrium rate increases with the surface temperature, the rate is limited by the desorption of CO and the subsequent competitive co-adsorption of CO and oxygen. If the equilibrium rate is measured over clean Ir(110) as the surface temperature is increased, the rate is proportional to the partial pressure of O_2 squared, but is inversely proportional to the partial pressure of CO. Moreover, the apparent activation energy and pre-exponential factor for the equilibrium rate are 14 kcal/mole and $2x10^{25}$ molecules/cm^2torr-sec. If the equilibrium rate is measured as the surface temperature is decreased, an expression for the rate is difficult to obtain since hysteresis occurs in the "equilibrium"rate. Because the kinetic expressions for the elementary reactions that may limit the rate may be non-linear functions of the coverages of oxygen and CO, the Gibbs free energy for the oxidation reaction may have several minima for any surface temperature so that hysteresis may occur.

(8) Above 600 K, where the equilibrium rate decreases with the surface temperature, the processes of CO oxidation and CO desorption compete in removing chemisorbed CO from the surface. Regardless of the temperature history of the surface and of whether the surface is clean or oxidized, the apparent activation energy and pre-exponential

factor for the equilibrium reaction are -4 kcal/mole and $2x10^{19}$ molecules/cm^2torr-sec. Moreover, the rate is proportional to the partial pressure of CO, but does not depend on the partial pressure of O_2.

(9) The temperature associated with the maximum rate of CO_2 production under equilibrium conditions is inversely proportional to the logarithm of the ratio of the CO partial pressure to the O_2 partial pressure. When the partial pressure of CO exceeds twice the partial pressure of O_2, the maximum in the equilibrium rate is larger for oxidized Ir(110). When the partial pressure of O_2 exceeds the partial pressure of CO, the maximum in the equilibrium rate is larger for clean Ir(110).

APPENDIX A

A METHOD OF CONTINUOUSLY MEASURING WORK FUNCTION CHANGES

(The text of Appendix A consists of an article coauthored with W. H. Weinberg that has been submitted to Journal of Vacuum Science and Technology for publication.)

Abstract

A method has been developed for measuring work function changes continuously. It is an ac variation of the diode technique and can be implemented easily in a standard retarding potential LEED/Auger system with derivative detection. Measurements for oxygen adsorption on an Ir single crystal illustrate that this method resolves work function changes within 5 meV.

Introduction

Various surface phenomena, especially transient ones, may be studied both qualitatively and quantitatively through work function changes, or contact potential difference (CPD) measurements (1). Measuring schemes are diverse. In studying gas-surface interactions, the Kelvin (2) and diode (3,4) techniques are used most frequently since these are less destructive to the interface. Although both techniques yield comparable results with similar precision (4), one irrefutable reason exists for preferring the diode method. It can be implemented more easily. A vacuum compatible probe must be constructed in implementing the Kelvin technique whereas an electron gun, available in commercial LEED/Auger systems, is the probe in the diode method.

The theory of the diode method has been treated thoroughly (1,3,4) but will be reviewed here to reiterate its limitations. The system consisting of an emitter, an electron gun, and a collector, the solid surface on which the measurements are made, constitute an energy analyzer. When a fixed current (I_0) impinges on the surface that is biased negatively, the current collected at the solid (I) varies with the retarding potential (E_r) as shown in the diode curve of Fig. 1a. Modifications of the surface potential shift the abscissa by an amount equal to the work function change. The first derivative of the diode curve, shown in Fig. 1b, approximates the energy distribution of the impinging electrons. Since electrons scatter from the band gaps of the solid, additional structure may appear (5). This structure does not influence the CPD measurement if it is a few volts distant from the crossover point (E_c shown in Fig. 1). The shape of the diode curve also varies with the angle of incidence of the incident electron beam onto the surface since the reflectivity of the sample varies with this angle (4). These artifacts that distort the diode curve must be circumvented in implementing a diode method.

Techniques which continuously monitor the CPD are useful for examining non-equilibrium surface phenomena (4). In continuous diode measurements, the collected current or a derivative of it is sampled to provide an error signal, which is used for modifying the gun potential or the retarding potential. The change in either potential reflects the CPD. Since dc techiques are plagued with noise and drift problems (6), ac methods with derivative detection are preferred in continuous measurements. A sinusoidal voltage ($V_m \sin\omega t$) is superimposed on the gun potential, and, typically, the second harmonic of the gun current (i_2) is collected. If the width of the electron energy distribution (E_w) exceeds the sinusioidal modulation, the second harmonic is approximated closely by (7)

$$i_2(E_r) = (V_m^2/2) \left. \frac{d^2I}{dE^2} \right|_{E_r} \tag{1}$$

Fig. 1c displays this function for a fixed modulation potential. Near the crossover point (E_c), the second harmonic is proportional to the retarding potential (E_r),

$$i_2(E_r) = g\,(E_r + \phi - E_g) \tag{2}$$

where g is the proportionality constant, E_g is the gun potential, and ϕ is the work function of the surface. To reflect work function changes, either the gun potential or the retarding potential is modified so that the second harmonic remains constant.

Using these principles, Nathan and Hopkins (4) developed an ac method to monitor work function changes continuously. The retarding potential is varied to reflect the CPD ($\Delta\phi$), and, if the experimental conditions are

chosen carefully, the change in the retarding potential equals the CPD. In the present work, the gun potential, rather than the retarding potential, is varied to reflect the CPD, for this scheme could be implemented more easily with the existing experimental equipment. Moreover, the gun potential is varied so that the CPD equals the change in gun potential. This scheme and the method of Nathan and Hopkins are equivalent procedures having the same advantages and problems (4).

Experimental Configuration

Fig. 2 diagrams this continuous diode method. Throughout an experiment, the change in gun potential equals the work function change. Several components in the diagram - an electron gun with a control unit, a phase sensitive detector, an audio oscillator, and a digital multimeter - are used routinely in commercial LEED/Auger systems with derivative detection. The filament and high voltage power supplies are components of the gun control unit (Varian #981-2145), which has two noteworthy features. First, the gun current is regulated by maintaining a small constant potential (E_p) between point P and ground. Second, the high voltage supply is programmable. A 0-10 V programming signal (V_p) produces a proportional change in the potential E_g'. The gun potential is the sum of E_p and E_g'. Since this method measures differences in the potential, E_p may be

ignored. The most severe requirement of this method is that the gun potential must be programmable.

The details of the measuring scheme are the following. Through the network of L and C, a sinusoidal voltage is superimposed on the output of the high voltage supply (8). This modulates the gun potential as well as the potentials of the lens elements in the gun. In principle, this technique introduces less harmonic distortion to the gun current than that produced by modulating the gun potential only, although both techniques yield comparable experimental results. A fraction of the current is reflected to ground by the retarding potential, while the remainder arrives at the sample. Since the dc current would overload the phase sensitive detector, this unwanted signal is removed by the audio transformer. The second harmonic is selectively amplified by the detector. The detector output serves as an error signal to adjust the gun potential through the programming voltage (V_p). The CPD equals the change in gun potential compared to its initial value. A computer monitors this potential through a digital multimeter (9).

Since the programming potential is referenced to point C that is several volts above ground, an isolation amplifier tranduces the error signal between the reference points. This isolation amplifier is the only special component which has been added to the conventional LEED/Auger system. It is not used if the gun voltage is programmable from ground. The integrator constructed from an FET operational amplifier, has two functions - filtering noise and providing high gain at low frequencies. This component is crucial in reducing tracking errors in the gun potential. Both the integrator and the isolation amplifier are powered by the gun control unit.

Various parameters affect the measured CPD through the open-loop gain

(A). During closed-loop operation, the gun potential tracks the sum of the retarding potential and the work function of the sample according to the relation

$$E_g = \frac{A}{A + 1} (E_r + \phi) \tag{3}$$

The open-loop gain is the product of the following factors: the slope of the second harmonic near the crossover point (g), the transimpedance of the audio transformer (Z), the gain of the phase sensitive detector at resonance (G_p), the pass function of the integrator ($1/\tau s$), and the gain for programming the high voltage supply (G_g)

$$A = {}_gZG_pG_g/\tau s \tag{4}$$

where s is the complex frequency in rad/sec.

Two parasitic effects, not represented in Eq. (4), limit the practical values for the open-loop gain. The integrator has a finite gain at the low frequencies. To insure that the CPD is measured accurately, other gain parameters should be increased. However, these cannot be increased indefinitely without producing an instability, for poles associated with programming the high voltage supply introduce phase shifts. Open-loop gains from 10/s to 1000/s were used in this work. The time constant of the integrator (τ) was fixed at one sec and delimited the range of useful gains that could be selected on the phase sensitive detector. Since the response time of a digital multimeter is on the order of 0.1 sec, the open-loop gain does not limit the rate of data acquisition.

In setting up this procedure, the retarding potential is fixed so that the diode curve does not change shape if the retarding potential is varied by several volts. The device which provides the retarding potential

should be selected carefully. Instabilities in this potential appear directly in the measured CPD. When a carbon dry cell battery supplies the retarding potential, measurements are reproducible within 10 meV, and noise margins are 5 meV. Noise is minimized best by maximizing the value of g in Eq. (2). Hence, the gun is sharply focused, and both the modulation voltage and the gun current are large. With the loop severed at the output of the integrator, the phase and frequency of the phase sensitive detector are tuned to the second harmonic. When the detector is truly aligned, its maximum output is proportional to the square of the modulation voltage (V_m) for small voltages. This voltage should be set so that the necessary proportionality is maintained. Otherwise, the second harmonic will not reflect the second derivative of the diode curve accurately. If the phase sensitive detector can filter the dc output, the filter is not used because it introduces phase shifts that induce instability. The integrator alone provides adequate filtering. The zero adjustment on the detector nulls dc offsets in the measuring loop. When the offsets are nulled, the loop is closed, and the gun is switched on; its potential rises rapidly to that of the crossover point. During a measurement, the sample can be heated if the double modulation technique of Tracy and Rowe (3) is implemented.

Performance Evaluation

Work function changes for saturation coverages of oxygen and CO on the (110) surface of Ir at 300 K were measured both with the continuous method and with a static dc diode technique. The results were identical. The CPD was 0.44 eV for oxygen and 0.23 eV for CO. Although CO has a large cross section for electron beam dissociation, no dissociation occurred since the beam potential is retarded to a small value before it strikes the surface. The measured CPD did not vary with either the modulation potential

or the beam current because the open loop gain (A) was chosen to be suitably large. The CPD for a saturation coverage of oxygen on Ir(110) was measured for small deviations of the angle between the beam and the surface from normal incidence. When the angle changed by $\pm 5^{o}$, the measured CPD varied by approximately 10 meV, which is within the experimental error of the technique. If the open loop gain is large, no variation is expected since the image charge of the dipole in the metal should cancel exactly any component of the dipole moment parallel to the surface.

This continuous diode technique was used to study the adsorption dynamics of oxygen on Ir(110) at 300 K. The CPD and fractional surface coverage are plotted against oxygen exposure in Fig. 3. Coverages were measured by thermal desorption mass spectrometry. Surface coverage and CPD measurements are related through a Topping plot (10) in order to determine the normal components of both the dipole moment and the polarizability of the adsorbate for a fixed substrate temperature. In this Topping construction, the surface coverage is plotted against the quotient of surface coverage and work function change. Results for oxygen adsorption at 300 K are shown in Fig. 4. For a surface coverage less than 0.25 ML, the dipole moment is 1.0 ± 0.1 D (1D = 10^{-18} esu-cm) and the polarizability is 63 ± 3 A^{o3}. For higher surface coverages, the dipole moment is 0.25 ± 0.02 debye, and the polarizability is 3.7 ± 0.6 A^{o3}. The discontinuity in slope of the Topping plot at 0.25 ML is attributed to an alteration of the overlayer structure. At that point, an oxygen p(2x2) LEED structure appears with maximum intensity, but vanishes at higher surface coverages. The details of the interaction of oxygen with Ir(110) are discussed elsewhere (11).

Acknowledgments

The authors are grateful to the AROD (Grant No. DAHC04-75-0170) for support of this research and to Dr. B. F. Lewis for his helpful advice.

References

1. G. Ertl and J. Küppers, Low Energy Electrons and Surface Chemistry, Verlag Chemie, Weinheim, Chapter 8, 1974.
2. N. A. Surplice and R. J. D'Arcy, J. Phys. E3, 477 (1970).
3. J. C. Tracy and J. E. Rowe, Appl. Phys. Letters 19, 388 (1971).
4. R. Nathan and B. J. Hopkins, J. Phys. E7, 851 (1974).
5. R. E. Thomas, J.Appl.Phys. 41, 5330 (1971).
6. A continuous dc method was implemented in this work. Although the method produced consistent and accurate CPD measurements, a thermocouple on the sample could not be read during an experiment since noise from the digital multimeter altered the reading.
7. N. J. Taylor, Rev. Sci. Instrum. 40, 792 (1969).
8. A transformer with high voltage isolation can replace the LC network. The output impedance of the high voltage supply fixes the pass characteristic of this network. The signal at the output of the high voltage supply is equal to the reference signal up to one Hz. For higher frequencies, a single pole attenuates the output signal by six dB per octave.
9. This computer, a PDP-11/10, performs several functions. The interfacing is detailed in: J. L. Taylor, Ph.D. Thesis, California Institute of Technology, 1978.
10. J. T. Yates, Jr., R. Klein, and T. E. Madey, Surface Sci. 58, 469 (1976).
11. J. L. Taylor and W. H. Weinberg (in preparation).

Figure Captions

Figure 1: The diode curve and its derivatives.

Figure 2: Experimental configuration.

Figure 3: Coverage and CPD as a function of oxygen exposure on Ir(110) at 300 K.

Figure 4: Topping plot for oxygen adsorption on Ir(110) at 300 K.

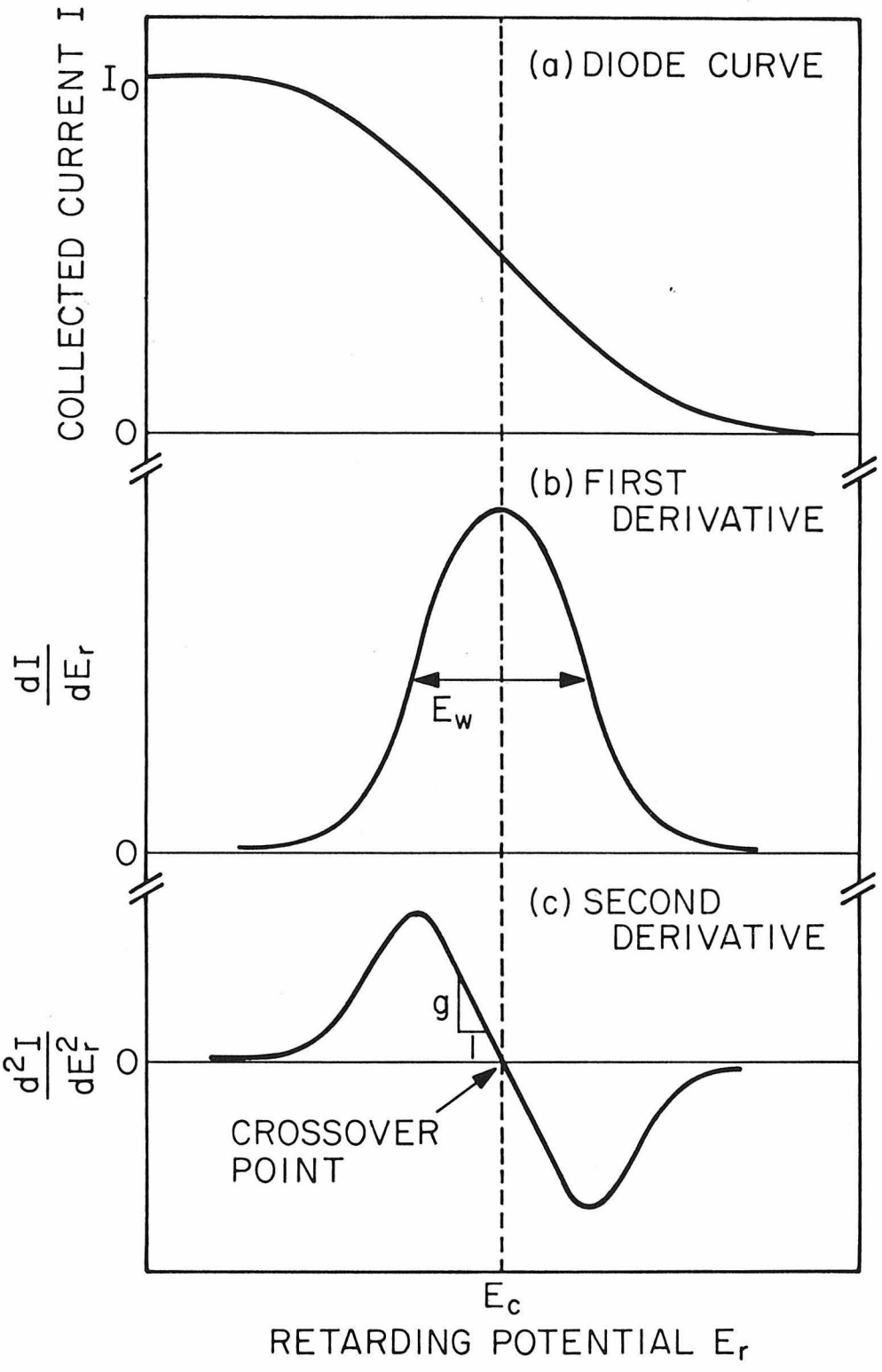

COLLECTED CURRENT I
I_o
0
(a) DIODE CURVE
(b) FIRST DERIVATIVE
$\frac{dI}{dE_r}$
E_w
0
(c) SECOND DERIVATIVE
$\frac{d^2I}{dE_r^2}$
g
1
0
CROSSOVER POINT
E_c
RETARDING POTENTIAL E_r

Fig. 1

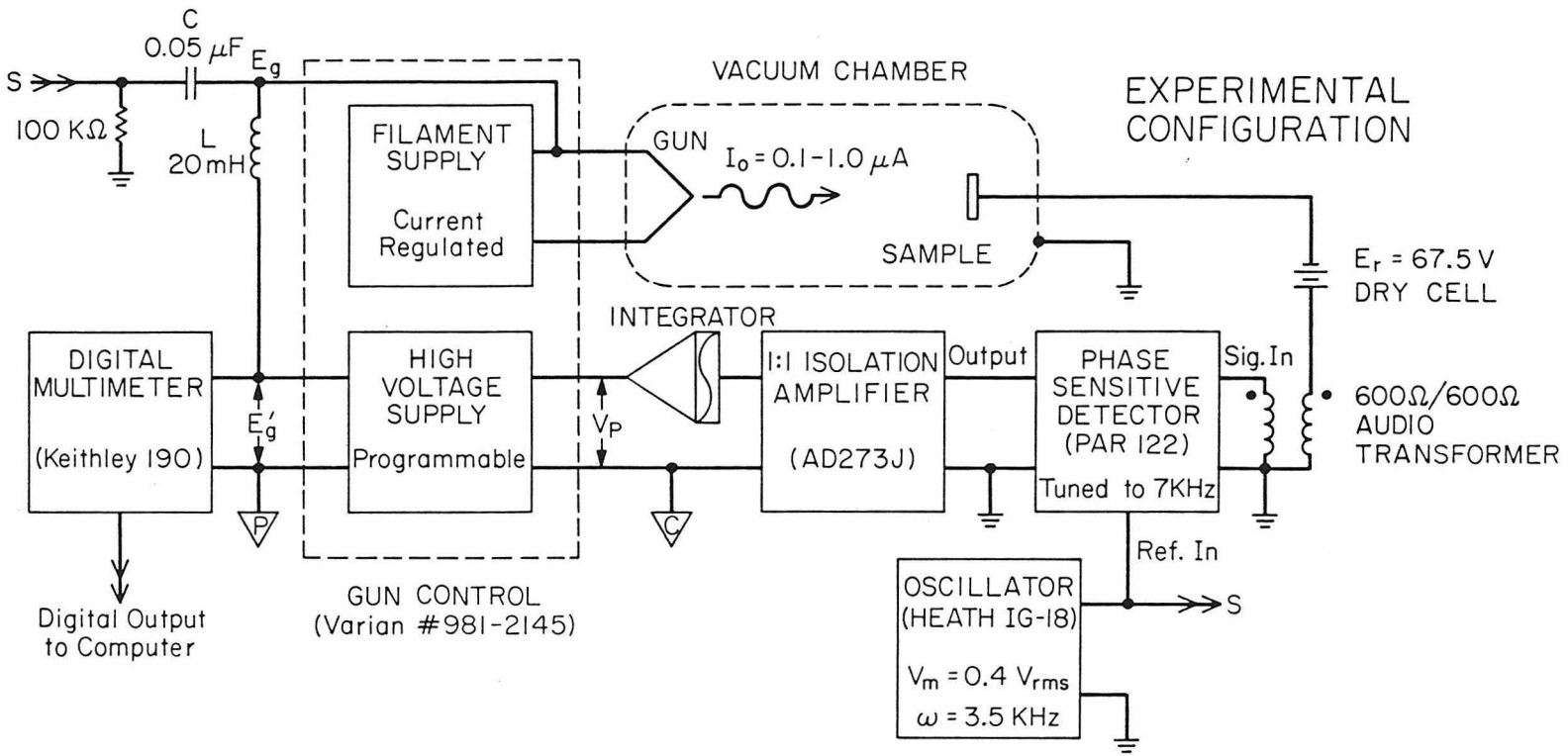

C
0.05 μF
E_g
S
100 KΩ
L
20 mH
FILAMENT SUPPLY
Current Regulated
VACUUM CHAMBER
GUN
I_o = 0.1-1.0 μA
SAMPLE
EXPERIMENTAL CONFIGURATION
E_r = 67.5 V
DRY CELL
INTEGRATOR
DIGITAL MULTIMETER
(Keithley 190)
E'_g
HIGH VOLTAGE SUPPLY
Programmable
V_P
1:1 ISOLATION AMPLIFIER
(AD273J)
Output
PHASE SENSITIVE DETECTOR
(PAR 122)
Tuned to 7KHz
Sig. In
600Ω/600Ω
AUDIO
TRANSFORMER
Ref. In
OSCILLATOR
(HEATH IG-18)
V_m = 0.4 V_{rms}
ω = 3.5 KHz
S
P
C
Digital Output
to Computer
GUN CONTROL
(Varian #981-2145)

Fig. 2

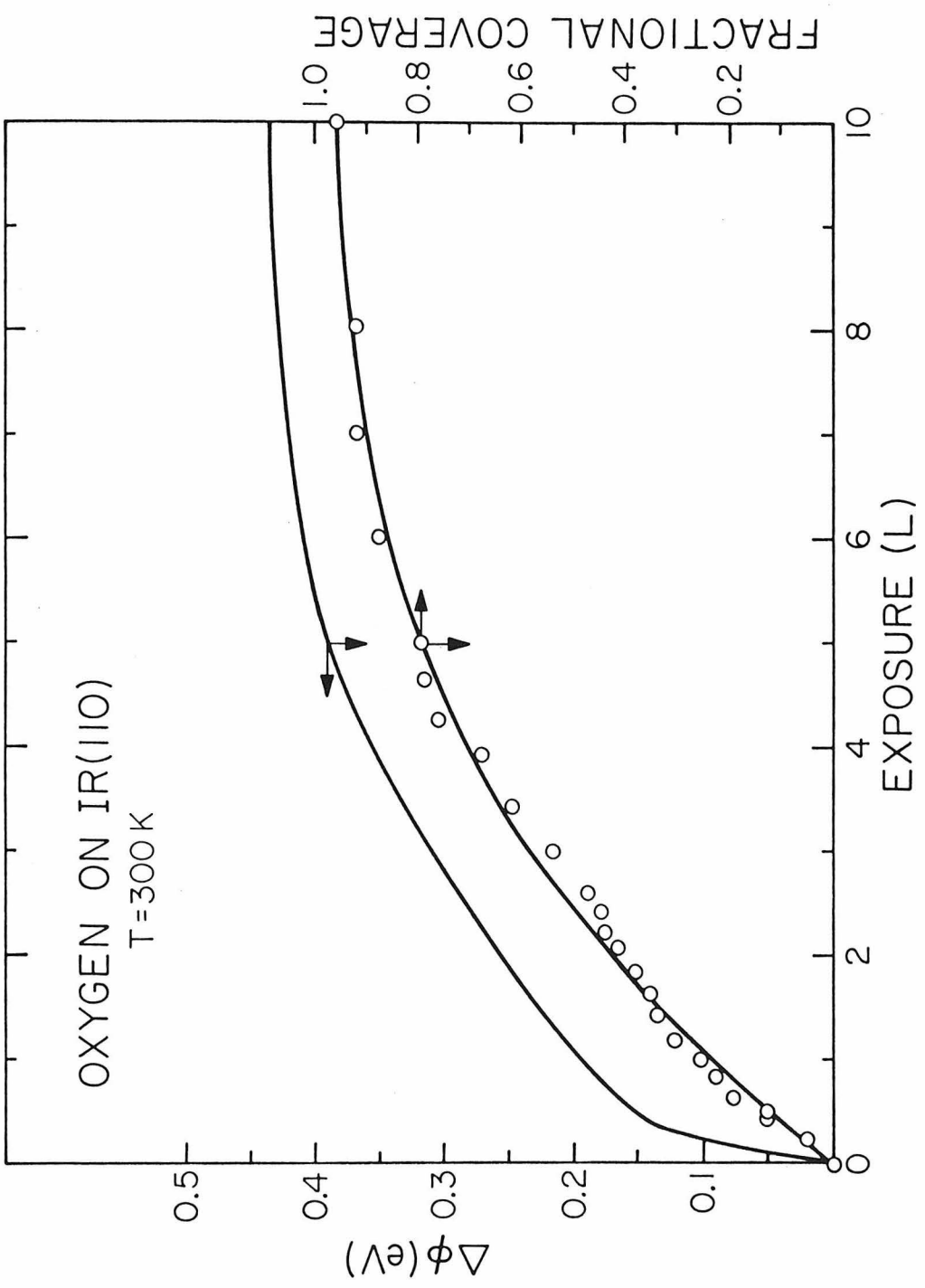

OXYGEN ON IR(110)
T=300 K
Δφ (eV)
0.1
0.2
0.3
0.4
0.5
EXPOSURE (L)
0
2
4
6
8
10
FRACTIONAL COVERAGE
0.2
0.4
0.6
0.8
1.0

Fig. 3

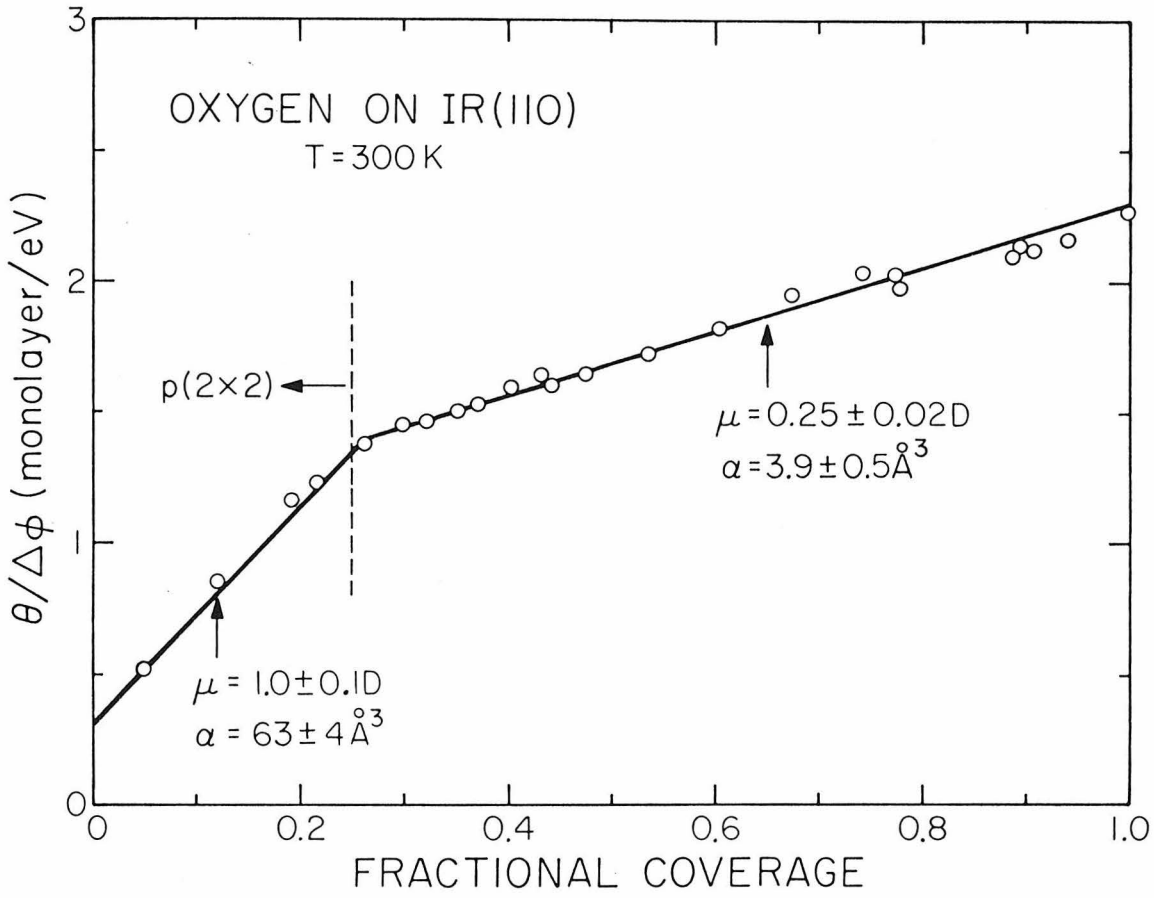

OXYGEN ON IR(110)
T = 300 K
θ/Δφ (monolayer/eV)
FRACTIONAL COVERAGE
p(2×2)
μ = 1.0 ± 0.1D
α = 63 ± 4 Å³
μ = 0.25 ± 0.02D
α = 3.9 ± 0.5 Å³
0
1
2
3
0
0.2
0.4
0.6
0.8
1.0

Fig. 4

APPENDIX B

POL-AN INTERACTIVE PROGRAMMING LANGUAGE FOR EXPERIMENTAL RESEARCH

POL-

an Interactive Programming Language for Experimental Research

by James L. Taylor

Part I: POL Users' Manual

Easy ys myne boke to rede and telleth of moche fyte, but then your easy rede is damned hard to wryte. - Napier

Table of Contents Page

Part I: POL Users' Manual 283

Preface

1. Features of the POL Operating System 286
   1.1 Functions Performed by POL 286
   1.2 Interrupts and Programming Modes 291
   1.3 Error Detection 293
   1.4 Structure of Commands 297
   1.5 Storing Data – The Filing System 299
   1.6 Protection of Resources 303
   1.7 Summary 305

2. The Storage of Data and Programs 306
   2.1 Allocating Memory for Files 306
   2.2 Disc I/O Operations Involving Files 309
   2.3 Allocating Memory for Programs 313
   2.4 Disc I/O Operations Involving Programs 315
   2.5 Summary 317

3. The Assembly and Execution of Programs 320
   3.1 Assembly of Programs 320
   3.2 Execution of Programs 325
   3.3 Iterative Execution in Programs 327
   3.4 File-Referencing Statements 330
   3.5 Summary 333

4. Monitoring Experiments 336
   4.1 Commands that Read Data 337
   4.2 Controlling the Storage of Data 347
   4.3 Controlling Experimental Parameters with DAC Channels 350
   4.4 Programmed Delays 360
   4.5 Summary 363

5. Displaying Data 366
   5.1 The Output Stack 366
   5.2 Plotting Data 367
   5.3 Writing Data 376
   5.4 Summary 378

6. Manipulating Data 381
   6.1 Calculator Commands 381
   6.2 Spectrum Manipulations 382
   6.3 Channel Manipulations 385
   6.4 Channel to-Channel Manipulations 390
   6.5 Summary 397

7. Summary of the POL Operating System 401

Part II: POL Software Manual 413

8. Introduction 414
8.1 Programming Philosophy 414
8.2 Terminology 416

9. Peripheral Addresses
9.1 The Structure of Peripheral Addresses 418
9.2 Peripherals that are Synchronously Controlled by POL 420
9.3 Peripherals that are Asynchronously Controlled by POL 422
9.4 Interrupt Vectors 424

10. A Description of Reserved Addresses 426
10.1 Fixed Reserved Addresses 427
10.2 Variable Reserved Addresses 432

11. Routines in the POL Operating System 444
11.1 Command Subroutines 444
11.2 Management Routines 457
11.3 Function Subroutines 463

12. The POL Listing and Flow Charts 483
12.1 Terminology Used in the Flow Charts 483
12.2 Finding Routines in the POL Listing 488
12.3 The POL Program 490

Appendices

A. Starting Up the POL Operating System 547
B. Glossary 548
C. The Construction of Data Files 556
D. How the Second Operands of Some Commands are Modified in Assembly 559

List of Tables Page

Part I: POL Users' Manual

2.1 Commands that Manage Storage Media 319

3.1 Commands that Manage the Assembly and Execution of Programs 335

4.1 How the Application Modes of DVMs are Associated with Experimental Variables 340
4.2 The Assignment of ADC Channels to Experimental Variables 346
4.3 The Assignment of DAC Channels to Experimental Parameters 352
4.4 Foreground Commands 365

5.1 Plotting Modes 368
5.2 Correlation between Experimental Options and Display Parameters 370
5.3 Correlation between DAC Channels and Display Parameters 371
5.4 Commands Involving the Display of Data 380

6.1 Data Analysis Commands 398

7.1 POL Commands 404
7.2 Conditions that Flag Error Messages 408
7.3 Circumstances that Suspend the Execution of a Foreground Program For One Interrupt Period 410
7.4 Assignments of Experimental Options and DAC Channels 411

Part II: POL Software Manual

11.1 Listing of Command Subroutines 446
11.2 Reserved Addresses, Peripheral Addresses, and Routines Referenced in Command Subroutines 451
11.3 Reserved Addresses, Peripheral Addresses, and Routines Referenced by Assembly Subroutines 454
11.4 Miscellaneous Entry Points 456
11.5 Listing of Management Routines 460
11.6 Reserved Addresses, Peripheral Addresses and Routines Referenced by Management Routines 460
11.7 Listing of Function Subroutines 464
11.8 Registers, Reserved Addresses, Peripheral Addresses, and Routines Referenced by Function Subroutines 466

12.1 How the Operand of the Trap Statement Indicates the Type of Error being Flagged 486

Appendices

C. Correlation between Application Modes of DVMs and the Correction Factor for Translating Binary Readings 558

D. How the Second Operands of Some Commands are Assembled 561

List of Examples Page

Part I: POL Users' Manual

3.1 The Sequence of Program Construction 322
3.2 Nested Loops 329
3.3 An Iterative Program 332

4.1 Reading the System Clock 340
4.2 Monitoring LEED Intensities with a DVM and as a Function of Time 343
4.3 Monitoring LEED Intensities with an ADC Channel and as a Function of Time 343
4.4 An ADC Channel used as an Accumulator 349
4.5 Collecting an Auger Spectrum 355
4.6 Collecting an Auger Spectrum with Reduced Base-Line Drift 357
4.7 Monitoring Three Thermal Desorption Mass Peaks Simultaneously 359
4.8 Employing Switches to Delay Data Acquisition 362

5.1 Manipulating the Display on the Oscilloscope

6.1 Determining the Area under a Peak in an Auger Spectrum 386
6.2 Re-Establishing the Zero Point of a File Containing Time 388
6.3 Summing Auger Spectra to Enhance the Signal-to-Noise Ratio 392
6.4 Forming a Difference Spectrum in UPS 392

List of Figures Page

Part I: POL Users' Manual

1.1 Flow Diagram of the POL Operating System 290

5.1 An Oscilloscope Display Before and After Manipulation 374

6.1 Auger Spectrum with the Background Removed 386
6.2 An XPS Spectrum Before and After the Al-K_α Line is Removed 396

Part II: POL Software Manual

11.1 How Control is Passed between Management Routines 461

Preface

To interpret and utilize the contents of this manual successfully, the reader, an eventual user of POL, should possess certain knowledge. He must understand the care, maintenance, and operation of all experimental equipment and peripherals which he will employ in his research. Equipment manuals are the authoritative source, not one's peers. Secondly, the user should comprehend digital-to-analog (D/A) and analog-to-digital (A/D) conversions to the extent that he has a rudimentary knowledge of the interaction between a computer and an experiment. The seventh chapter of Malmstadt's book, Digital Electronics for Scientists, contains this material. Finally, the reader must understand fundamentally a high-level programming language, preferably FORTRAN. A number of excellent books documenting these languages exist. Before consulting the systems programmer on any point contained in this manual, the user should acquire this knowledge.

Since POL is a self-contained operating system, no conversance of the RT-11 operation system of the PDP-11 computer is necessary to utilize the full capabilities of POL. However, this system stores experimental data in a format that is compatible with the RT-11 filing scheme. This feature allows the user to perform virtually limitless manipulations of experimental data in the RT-11 environment.

Unlike many operating systems that monitor experiments, POL is a low-level language, as assembly language is to FORTRAN. Routines that acquire data are not canned (i.e., pre-programmed). The user must attend to several bookkeeping functions, such as memory management. However, these apparent disadvantages make POL very powerful.

POL was designed to be employed in a variety of original experiments, not just routine spectral analysis. Because the users control several bookkeeping operations cooperatively, time sharing on a mini-computer is practical.

The author has endeavored to organize this manual in a form that is most digestable to the reader. Contrary to the pattern of most computer manuals, this one does not describe individual commands in brutal detail. Instead, a group of commands is discussed in a section so that features which are common to the group may be revealed.

Fundamental concepts of the POL operating system are treated in the first chapter. The succeeding five chapters discuss the structure and functions of individual statements, which are the building blocks of programs. Fine points concerning commands are elucidated through programming examples. In the last chapter, the salient features of POL are summarized in tabular form. Many tables occur throughout the text to eliminate lengthy explanations.

Four appendices contain material that is peripheral to the description of instructions. The most valuable of these, Appendix A, describes the start-up procedure to run the POL operating system. Appendix B is a glossary of terminology used throughout this manual. To avoid confusing the reader, several words which have two meanings appear either capitalized or uncapitalized in the text. The glossary defines the word in both cases. Note that all numbers appearing in the text are decimal numbers, unless another radix is defined.

The author is indebted to Dr. Blair F. Lewis for unselfishly furnishing advice and designing much of the hardware manipulated by POL.

- James L. Taylor
November, 1976

1. Features of the POL Operating System

1.1 Functions Performed by POL

The POL (Peripheral Oriented Language) operating system employs a PDP-11 computer to control experiments, and to read and store data. In addition, facilities are provided for the display and on-line analysis of data. By entering instructions on a LA36 Decwriter, a programmer directs POL to execute these functions.

A multiplexed DAC with eight Channels controls experimental parameters. The analog signal from any Channel is actually the sum of the outputs of two 12-bit D/A converters. Because one of these converters, the narrow converter, has one tenth the output range of the other, the wide converter, the ultimate resolution of an analog signal is one part in 40,960 over the full range of the experimental parameter. A user need not be concerned with operating the wide and narrow converters independently. POL commands controlling the multiplexed DAC are implemented so that he uses this device as if it contained a single converter with 15 1/2 bits of resolution. When the command executes, the POL operating system, not the user, determines the digital signals that are entered into the wide and narrow converters.

Each DAC Channel is assigned to a single experimental device. This scheme was implemented for two reasons. The analog signal of each channel can be adjusted to the range accepted by a particular device. To control the device, the user enters DAC settings in the

units of the controlled parameter (see § 4.3). The decimal-to-binary translation is performed by the POL monitor so that the maximum resolution of the wide and narrow converters is utilized.

Data are read from a multiplexed ADC with eight Channels, digital voltmeters (DVMs), pulse counters and an internal clock. Before experimental data are plotted or printed, they are converted to a form most accessible to the user. This necessitates assigning an ADC Channel to a particular experiment. Data are read by one of two different schemes, depending on the Peripheral. (If a Peripheral is associated with an experiment, the word Peripheral is capitalized.) When data are acquired from the clock or a DVM, the Peripheral is immediately read and the data stored. Then execution of the user's program continues. However, data garnered from the pulse counters and ADC Channels may be summed over a period of time before the program continues processing. Data accumulated from ADC Channels are averaged by the number of readings.

The timing and the control of Programs that amass data, Foreground Programs, are accomplished through a Hardware Interrupt system. Every sixteen milliseconds, the computer temporarily abandons Background Functions (mainly, the analysis and display of data) so that it can service Foreground Programs. After the latter task has been accomplished, the computer attends to Background Functions. Little time is required for the servicing of Foreground Programs. Certain statements in these Programs terminate their execution for an Interrupt Period. (When POL Files or Programs are referenced, these words appear capitalized to distinguish between POL Files and Programs, and those

associated with other computer languages.)

Immediately after data are read, they are stored in the memory of the computer, called core. To obtain a permanent record, the data are transferred to a mass storage device, an RX01 floppy disc. In POL, a filing system facilitates the maintenance of data records in core and on disc. To perform data manipulations or display with the contents of a File, it must reside in core. Like data Files, Programs are stored either in core or on disc, but only those in core execute.

Data manipulation routines permit on-line analysis. These routines have purposely been limited to brief programs whose results dictate the short-term course of a user's experiments. Since all the POL operating system concurrently resides in core and the PDP-11 has a limited storage capacity, longer routines are excluded from the set of analysis programs.

Raw data, or data made more accessible through manipulation, can be displayed in various ways. Individual data points or entire data Files may be printed out on the LA36 Decwriter, the computer terminal. The contents of a File are printed in units that are most convenient to a programmer.

Often, a plot of a data File is much more useful than a list of data points. Data are graphed on a display scope (a storage oscilloscope), or a hard copy is available from a point-plotter. An axis of either device corresponds to a preset range which varies with the type of data displayed. For example, data acquired with a cylindrical mirror electron energy analyzer (CMA) have the abscissa varying from 0 to 2000 eV. while the ordinate ranges from 0 to 10^8 counts. Because

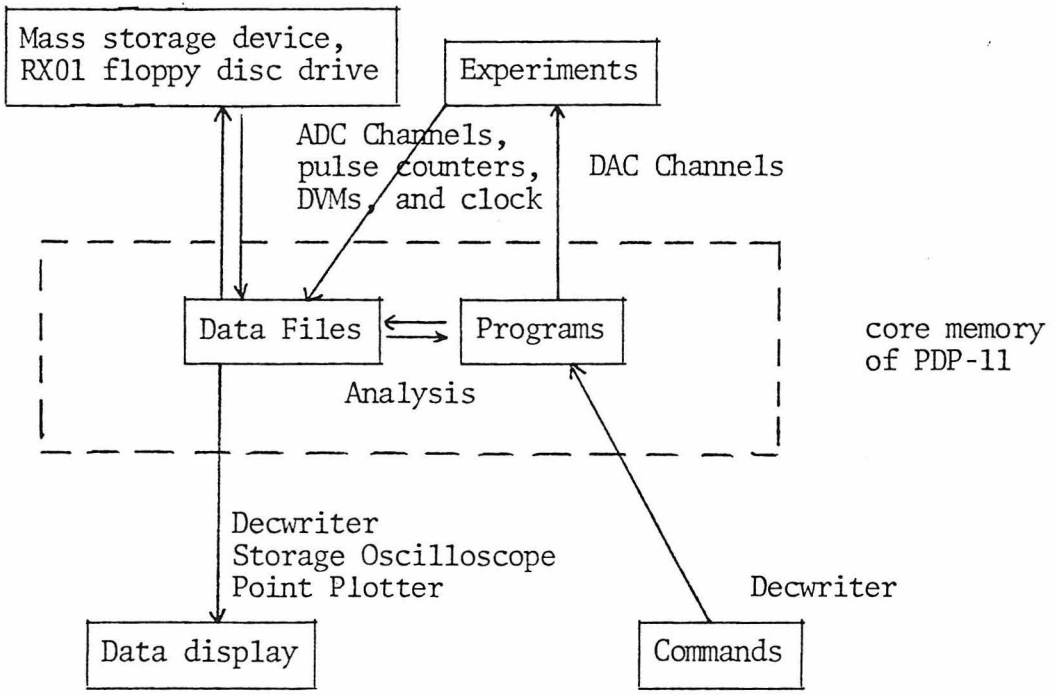
only a fraction of any range is normally used in an experiment, certain commands translate and expand each axis spatially. If a programmer makes intelligent use of these instructions, he can easily label the axes of an x-y plot.

A Background Program or Immediate Instructions, conduct the analysis and display of data. These directives are serviced at a lower priority than Foreground Programs. When Foreground Programs are not running, Background Programs and Immediate Instructions execute.

Some minor, yet essential, features of the POL system have not been mentioned. These include a group of statements which allow the conditional execution of a Program. An example is the LOOP instruction that directs the iterative processing of a section in a program. Instructions which control the assembly and execution of Programs have also been neglected.

The functions of the POL operating system are summarized in the flow diagram of Figure 1.1. This diagram emphasizes the role of peripherals in executing these tasks. On the Decwriter, the user enters a series of commands into the PDP-11 computer to form a Program. Foreground Programs monitor experiments. While an experiment is controlled by DAC Channel(s), data may be accumulated through ADC Channel(s), a pulse counter, DVM(s), and a system clock. These data are stored in Files that reside in core. Files and Programs are preserved on a mass storage device, a floppy disc. It has numerous storage locations that supplement the limited capacity of core. Information that is placed on disc may be retrieved by the POL monitor. During or after an experiment, Background Programs or Immediate

Figure 1.1: Flow Diagram of the POL Operating System



Instructions analyze and display data. Peripheral devices employed in the latter function are a display scope, a point-plotter, and the Decwriter. A Hardware Interrupt system schedules the many tasks performed by the POL operating system.

1.2 Interrupts and Programming Modes

POL is a time-sharing system; that is, up to four Programs which monitor experiments can execute concurrently. Usually, only one or two of these Foreground Programs are present in core. From hardware limitations and constraints on processing time, statements which analyze data, display data, or perform disc I/O transfers may not be entered into Foreground Programs. Therefore, a fifth Program, a Background Program, which accomplishes these tasks runs concurrently with the four Foreground Programs. The scheduling problem of concurrently running five Programs is compounded. Often a Foreground Program must be assembled while others are executing. By employing Hardware Interrupts, the POL operating system can rise to this Herculian task.

A Hardware Interrupt is generated every sixteen milliseconds. At this time, the POL system temporarily suspends the execution of Background processes. The system switches from the Background to the Foreground mode. (These modes should not be confused with the foreground and background modes of the RT-11 operating system. However, the POL modes are similar to those of RT-11 in that the foreground task executes with highest priority.)

Foreground Programs are processed in sequence. When certain flags are encountered in a Foreground Program, execution is discontinued for an Interrupt Period. For example, the second attempt to read data from a single Peripheral in one Period suspends processing of a Program during that Period. ADC Channels are considered to be separate Peripherals. In the next interrupt period, processing of the Program continues from the point of previous termination. No statements are neglected. Because Foreground instructions are processed quickly and terminating conditions are numerous, four Foreground Programs execute in less than sixteen milliseconds. After these Programs are serviced, the POL system reverts to the Background mode.

An example exposes three distinct tasks that are performed in the Background mode. A user writes and executes a Background Program as follows. Since only one Background Program may reside in core at any instant, this user erases another's Background Program by entering a single instruction on the Decwriter. Another single instruction allocates space in core for the new Background Program. Yet another signals the system to assemble all succeeding instructions into this space until a statement flags that all instructions of the Program are entered. Then the Program is typed on the Decwriter. After the terminating statement is issued, the user executes the Program by issuing a single command. When processing has ended, a message, printed on the Decwriter, informs the user of the termination. Finally, a single command erases the Program. The space in core that was allocated for the Background Program is then available to other users.

As the example demonstrates, three tasks are performed in the

Background mode: (1) the processing of Immediate Instructions, (2) the assembly of Programs, and (3) the execution of Background Programs. No two of these can occur concurrently.

The POL system receives single instructions and executes them instantly, barring interrupts. These directives are called Immediate Instructions. Many display and data analysis statements can be issued either in Background Programs or as Immediate Instructions.

In the Background mode, a sequence of commands can be assembled into a Program. As statements are received by the POL monitor, they are assembled, that is, converted in a form that is suitable for execution. Although a Background Program was the subject of the example, Foreground Programs are compiled in the same manner. The execution of these Programs is initiated by an immediate instruction.

A Background Program begins processing in the same manner. By printing a message on the Decwriter, the POL operating system informs the user that execution has terminated. Then Immediate Instructions are accepted in the Background mode.

No errors were flagged in the example. Since a flawless program is seldom written, an operating system must take action to protect it and other programs from errors in any one program. The POL operating system reveals mistakes and allows rapid recovery from them.

1.3 Error Detection

To prevent mistakes in one Program from interfering with other

Programs, an error detection scheme has been incorporated into the POL operating system. Most errors are revealed during assembly; others are discovered during execution of a Program. Because the latter type of error terminates execution of the infringing Program, all errors that can be revealed during assembly are exposed at that time. Whenever an error is detected, one of several messages is printed on the Decwriter.

Only one error terminates the execution of a Foreground Program. A Program attempts to store experimental data in a non-existent File. This error is flagged by the same message that signals a normal (flawless) suspension of execution:

STOP n

The number n corresponds to a Foreground Program; which are numbered 1 through 4. By inspecting a Peripheral that a Program manipulates, a user distinguishes between a normal termination and this Filing error. Since every conceivable effort is made to detect this error in assembly, it should not be flagged during execution. However, this feature provides additional protection for the POL operating system.

In Background Programs, this Filing error is also flagged. When a non-existent data file is referenced, processing ceases and the STOP message is printed

STOP 0

Two other classes of errors are revealed during the execution of a Background Program: manipulation errors and disc I/O errors. A

manipulation error occurs when a Program tries to perform an inappropriate analysis operation on a data file. As an example of this error, an attempt to deconvolute the Al K_a line from LEED I-V data generates the message

ERROR MANIP

Then the Background Program stops executing.

A transfer error results when a user endeavors to retrieve a non-existent file from disc.

ERROR TRANS

If a parity check fails during a disc I/O transfer, a hardware error is flagged

ERROR HRDWR

More types of errors are revealed in assembly than those detected in execution. Some commands are unacceptable in certain Programming environments. Commands that display data may not be entered in Foreground Programs. A violation of this constraint flags a usage error.

ERROR USAGE

Once resources, such as Peripherals, are allotted to a Program, these resources are protected from interference by other Programs. When infringement occurs, an allocation error results.

ERROR ALLOC

The Key system, which protects resources, is discussed in § 1.6.

The ordering of statements is crucial for the successful execution of a Program. For instance, improperly nested loops are the bane of many programmers. In addition to noting these mistakes with an ordering error,

ERROR ORDER

the POL system allows the user to eliminate segments of his program to recover from these mistakes. During assembly, the user may choose to eradicate portions of his program whenever he thinks he has erred.

An attempt to erase past the beginning of a Program flags an end-of-file (EOF) error:

ERROR EOF.

The same message is printed when a user tries to enter more statements into a Program than he has reserved space for.

Not only must all commands be ones recognized by the POL monitor, but the operands associated with these must be in a prescribed range. The next section describes error messages that result from violations of these constraints.

Normally, the POL system assembles statements into Programs as soon as they are entered on the Decwriter. When an error is detected, the command that was entered last is completely ignored. The system treats this command as if it were never issued and retains no memory of it. This action is appropriate because the error usually results solely from the latest command. Errors in Immediate Instructions are

handled as those resulting from individual statements of an assembling Program. The user is instantly informed; execution of the command is suspended.

1.4 Structure of Commands

This document exposes the functions of commands in the POL system so that the user can construct Programs successfully. Since a multitude of tasks is directed by the system, the functions of instructions vary markedly. However, all commands have essentially the same format:

```
NAME      OPER1; OPER2
```

Any number of spaces may precede the name or mnemonic of the command, NAME, and may separate it from the operand field, OPER1; OPER2. Although no space is required between the name and operand field, some should be inserted to enhance legibility. An instruction must be completed on a single line of the Decwriter. No more than one command should appear on any line. Sometimes, violations of these two restrictions are not flagged. Even though the erring Program does not assemble and execute as intended, other Programs will not be harmed. Any number of blank lines may intervene between statements.

The command name is an alphabetic mnemonic which describes a task to be performed by the computer. The operand field that succeeds the name may contain up to two operands, OPER1 and OPER2. These operands delimit the task to insure its successful completion.

The first three characters of a mnemonic must correspond to a legal POL instruction, or else an instruction error results.

ERROR INSTR

Although the user is urged to type the complete mnemonic for a given command, the number and order of characters succeeding the first three is arbitrary.

Certain instructions require only one operand, and a few need none. Commands with two operands have a semicolon separating these. Otherwise, an error is flagged. If one or no operand is associated with a statement, no semicolon should be typed or an error message ensues. Either of these mistakes produces the following message:

ERROR OPER1

Each operand is limited in range. This range depends on the command and whether the operand is the first or second in the field. Unless a limit is described with digits to the right of the decimal point, these digits are truncated in assembly. The number of fractional decimal places that are recognized by the POL system is given by the number in the range statement. If the maximum value of an operand listed is 200.000, three digits to the right of the decimal point are employed in assembly of the operand. Other fractional digits are truncated. If an operand varies over three orders of magnitude or more, the stated limits are approximate in the second significant digit.

When the first operand exceeds its range, an error message,

identical to the one above, appears on the Decwriter. If the second operand does likewise, the message

ERROR OPER2

is printed. A single operand associated with a statement is considered to be OPER2. That is, commands having fewer than two operands produce the latter message when the range limitation is violated. If an instruction requires no operands, the operand field must be left blank.

The POL system flags only one error before the assembly of a statement is terminated. Although a hierarchy system exists for detecting errors, this ordering is relatively unimportant to the user. Errors are detected in the sequence that they are described in this section. After these errors are sought, those of the previous section are flagged.

1.5 Storing Data - The Filing System

The Filing system of POL provides a convenient method for storing experimental data. Data from separate experiments are stored in different Files. Each type of data read by the computer is stored in a File. For instance, the beam voltage and beam intensity readings of a LEED I-V spectrum are stored in separate Files.

Often data are accumulated with a DAC Channel controlling an experimental parameter such as the beam voltage. Information

concerning the DAC signal is stored in the beginning of the File. When results are graphed, a File of intensities is plotted against a File of beam voltages. Alternatively, the DAC information at the beginning of the intensity File can serve as the abscissa.

To facilitate the manipulation of Files, those stored in core and on disc are maintained in the same format. A File consists of two distinct parts, a Label followed by a string of Slots. One data point, coded in binary, resides in the two words of each Slot. (A word is sixteen bits long.) Slots are also referred to as "points" or "channels." To describe the data contained in the channels, a Label is appended to the beginning of a File. The Label and Slots of a File occupy consecutive locations in core or on disc. The Label is comprised of five words. Along with various storage formats for data, the Label is described copiously in Appendix C, the contents of which users must understand fully.

In the first word, the number of Slots is delimited. The second word marks the Slot where a piece of datum is to be stored. This storage pointer is automatically manipulated by a Foreground Program. The first datum is placed in the first channel, and successive readings are deposited sequentially. At his option, the user may manage the storage order with POL commands that deposit data in other series. The storage pointer is the only location in the Label that may be manipulated by the user. Others are automatically loaded by the POL system.

Each half, or byte, of the third word is dedicated to a separate function. The experimental option is stored in the first, or lower

order, byte. The option describes what type of data is contained in the File and which Peripheral was used to acquire the data. For example, option fourteen notes that the data points are thermocouple EMFs read with a DVM. If a DAC Channel were used to control an experimental parameter, the second half of the third word contains the Channel number. Otherwise, this byte is not loaded.

The final two words of the Label describe how a DAC was programmed to control an experimental parameter. Like the third word, these locations are loaded without user intervention during the execution of a Foreground Program. For an accurate loading and interpretation of these locations by the POL system, a parameter is monitored as follows. The user sets the output of a DAC Channel so that an experimental parameter, such as the pass energy of the CMA, is fixed to the beginning (or end) of the range of interest. Then the Channel is incremented (or decremented) uniformly. Hence, the data points of a File are sampled at equal spacings of the parameter.

The initial value (s) of the DAC Channel is stored in the fourth word of the Label and the increment (I) is placed in the fifth. Slots are numbered from 0 through (n-1) where n is the number of channels in a File. The algorithm to generate the abscissa for any Slot m is

$$X(m) = S + I \cdot m$$

If more than one range of an experimental parameter is examined, data obtained in each range are placed in separate Files. Two Files store the intensity data from two Auger peaks that are distant in energy.

At first glance, the reader may think that this scheme of

controlling experiments is too restrictive. However, virtually all the theory for the filtering and deconvolution of data has been developed for data equally spaced in the abscissa. The smoothing, deconvolution, integration, and differentiation routines of POL are intended for such data.

The Label is crucial in the display and analysis of data, particularly the former function. By employing information in the Label, the POL system can graph and print data in a form most accessible to the user. Under POL, the Label is invisible to the user. When a precocious programmer writes his own RT-11 programs for data analysis, these should employ the valuable information contained in the Label.

One hundred ten (110) different data Files may reside in core at any instant. These are numbered 1 through 110. A dummy File, numbered 0, is employed in display operations. It signifies that the DAC information in the Label of the File, not another File, will generate the abscissa. When any Files are not used by other programmers, the user may acquire core memory for the unused Files. To maximize the utilization of memory, the POL monitor maintains a list of the core locations of Files and Programs.

Many POL commands operate on parts of a data File. This File is usually not delimited by being an operand of a command. Instead, the file is numbered by a File-referencing statement which precedes the command in a Program. If an instruction that should be prefaced by a File-reference is not, an order error is flagged. File-references need not immediately prefix the command which operates on the File. With this scheme, complex manipulations on a single File may be

programmed with a minimum of effort.

A protection system is included in POL so that only one Foreground Program can reference a particular File. This Protection system, which uses Keys, is the subject of the next section. In the Background mode, Files are also protected.

When a File is no longer needed in core, it is transferred to disc for storage. Then the File is erased so that the core memory that it occupied is available for other uses. After erasure, the File number returns to the lot of free Files. Disc files may return to core or may be permanently deleted from disc. Hundreds of files can be stored on a single disc. These files retain the format that they had in core and are named

$\alpha\beta\gamma XXX$

Note, XXX is a number from 001 through 110, and α, β, and γ are alphabetic characters. Each user is expected to maintain a log of his disc files. This log is a listing of file names with the experimental conditions under which the data were obtained.

1.6 Protection of Resources

In time-sharing environments, a common, yet heinous, type of programming error occurs: one user errs, while another suffers the consequences. To eradicate this mistake, the POL system includes a means of protecting each user's resources from the interference of other

programmers.

One operand of the command that allots core memory to Files is a Key. This Key is the number of the Foreground Program, 1 through 4, that amasses data into the File. No other Foreground Program may reference the File without flagging an allocation error in assembly. When a Program might reference a File which has been assigned no space in memory, the same error results. This mistake is flagged in both the Foreground and Background modes.

Peripheral devices (DVMs, pulse counters, A/D Channels, D/A Channels, but not the system clock) are also protected. The first Foreground Program which contains a valid instruction referencing a device is the sole Program that may manipulate that Peripheral. If another Program might attempt to, an allocation error is produced in assembly. Keys that protect Peripheral devices are automatically assigned by the POL monitor as a Program compiles. When a Program is erased, the Keys are removed so that other users may manipulate the relinquished devices. File Keys are not released until a File is erased.

Files referenced by the Background Program and Immediate Instructions are also protected. Using a POL command, the user defines a Key which all Files referenced in the Background mode must possess. If a File does not have this Key and is referenced in the Background Program or by an Immediate Instruction, an allocation error is flagged in assembly.

1.7 Summary

This chapter exposes the salient features of the POL operating system. In the Foreground and Background modes, separate functions of POL are realized. Foreground Programs control experiments and read data. Data analysis and display are accomplished in Background Programs and with Immediate Instructions. Storage facilities are also manipulated in the Background mode. Regardless of their function, commands have the same structure: a name followed by a field of up to two operands. Error detection, data filing, and protection expedite the construction of Programs from statements.

Future chapters, which describe the functions of individual instructions, are based on the foundations laid here. In Chapter 2, commands which manipulate storage facilities, core and disc, are introduced. Chapter 3 deals with the assembly and execution of Programs. Methods for monitoring experiments are developed in Chapter 4. Then Chapters 5 and 6 discuss the display and manipulation of data, respectively. Before a user can implement any command, he must start-up the POL operating system as described in Appendix A.

2. The Storage of Data and Programs

Unlike other operating systems that monitor experiments, the POL system does not manage the storage media without user intervention. However, time sharing on a mini-computer is practical when the users manipulate these media. All memory and disc management commands are entered only as Immediate Instructions. Otherwise, a usage error is flagged.

2.1 Allocating Memory for Files

Except commands that perform disc I/O operations, POL instructions operate only on those Files stored in core memory. Before any Program that references a File is assembled, space in core must be reserved for the File. A series of two Immediate Instructions allots memory to a File or sequence of Files. This two step process is inconvenient but reduces the chance that a programmer will carelessly destroy data Files.

To assign 100 channels to Files 26 through 30 that will be loaded by Foreground Program 1, the following statements are entered on the Decwriter:

```
F:OPEN    1;26
F:ASSIGN  5;100
```

The first command delimits the Key and the first File of the sequence;

the second assigns 100 channels to Files 26 through 30. After the second line-feed is entered, memory is allotted to the Files. Then Keys for each File are deposited in the POL system.

Every Slot requires two words of core, and a Label of five words prefixes each File. Hence, each of the Files would occupy 205 words in core. About 14,000 words of memory are available to store Files and Programs so that about 7,000 data points may reside in core concurrently.

After space is allotted to a File, its contents are cleared or initialized. The first word of each Label stores the number of Slots, 100. The second is set to the address of the first Slot. Until the experimental option and DAC Channel are loaded, the bytes of the third word contain negative one. The fourth and fifth words are zeroed as are all Slots.

In general, the series that assigns core memory to Files is

```
F:OPEN    Key; first File
F:ASSIGN  number of Files; number of Slots
```

(Command names are always written in capital letters while lower case characters describe the operands.) The Key may assume values 0 through 4; the first File, 0 through 110; the number of Files, 1 through 50; and the number of Slots, 1 through 2000. (See § 3.4 to determine why the operands of the F:OPEN command may be zero.) After the F:OPEN instruction is issued, the Key and first File are stored in the POL monitor. These are erased when the F:ASSIGN statement, or any memory or disc management operation involving Files, executes. To allot

space to Files, these two statements should be entered in the given sequence.

When the F:ASSIGN directive is issued, four mistakes suspend its execution and flag an allocation error. One or more Files in the sequence has previously been assigned space in core or indexes a File out of the acceptable range of 1 through 110. Enough memory is not available to accommodate the request. An F:OPEN statement must preface the F:ASSIGN directive but need not be contiguous to it. When an error is flagged, no File is assigned space.

Under the POL system, experiments are easily repeated. After data Files from the first run are transferred onto disc, the run is repeated by clearing these Files and re-executing the Foreground Program that filled them in the first run. To clear Files, a series similar to the assignment series is issued:

```
F:OPEN  Key; first File
F:CLEAR number of Files
```

The F:CLEAR directive zeroes all Slots of each File in the sequence. All Labels are also reset as described for the F:ASSIGN command. To clear Files 26 through 28, which were assigned key 1, the user types:

```
F:OPEN  1;26
F:CLEAR   3
```

An allocation error is flagged when the Key of one of the Files in the sequence does not match the Key of the F:OPEN statement. No File in the sequence may be free, i.e., assigned no memory. In either case, an allocation error results. If one or more File numbers

in the sequence is not in the range of 1 through 110, or the F:OPEN command does not prefix the F:CLEAR instruction, an allocation error is flagged. All errors suspend execution of the F:CLEAR directive.

When Files are no longer needed in core, they are transferred to disc and erased so that their space is available to other programmers. Files are erased with the following series of Immediate Instructions:

```
F:OPEN  Key; first File
F:NULL  number of Files
```

For both the F:NULL and F:CLEAR instructions, the number of Files ranges from 1 through 50. After the second statement is completed with a line-feed, all Files in the sequence numbered by the operands are returned to the pool of free Files and relinquish their space in core. The errors flagged by the F:NULL command are identical to those of the F:CLEAR statement.

2.2 Disc I/O Operations Involving Files

Three disc I/O operations involving Files are supported under POL. Files may be: transferred from core to disc, imaged from disc into core, and erased from disc. Like memory management operations involving Files, disc I/O operations involving Files are realized through a sequence of two Immediate Instructions. The first instruction is the F:OPEN command. The operand fields of the second instructions are identical.

number of Files; alphabetic name

The number of Files involved in the operation is given by the first operand of the second instruction, while the number of the first File in the sequence is given by the second operand of the F:OPEN command. An alphabetic name is a string of three alphabetic characters, A through Z. Exactly three alphabetic characters must be entered. Else an operand error is flagged.

If a user wants to transfer File 27 with Key 1 (assigned by the F:ASSIGN command) from core to disc, the following sequence of Immediate Instructions is issued:

```
F:OPEN  1;27
F:PUT   1;ABC
```

The File in memory is unchanged following the execution of this instruction. A name of a disc record is the concatenation of the second operand of the F:PUT statement, followed by a number of a File in the sequence, 001 through 110. Although the extension LDA is appended to all disc records created under POL, it is invisible to the user. In this example, the disc record is named ABC027.LDA.

Once on disc, a record can be manipulated by RT-11 programs. All Programs and Files that are transferred onto disc are stored on drive 1 of the RX01 floppy disc unit. Drive 0 is reserved for the POL and RT-11 operating systems.

The general format of the F:PUT command is

F:PUT number of Files; alphabetic name

If the Key of the F:OPEN command does not match the Key of any File in the sequence, an allocation error is flagged. The number of Files in the sequence may vary from 1 through 50. As mentioned previously, an alphabetic name must consist of exactly three alphabetic characters. To make the POL and the RT-11 systems compatible, alphabetic names are assembled into RADIX 50. (See Appendix D.)

Several errors, unique to disc I/O operations, are flagged during the execution of these routines. Until an error is generated, the operation proceeds normally. When the disc is filled with records, or the F:PUT command might create two disc records with the same name on the same disc, a transfer error is generated. Then execution of the F:PUT command ceases. A disc I/O operation never writes over records that are already on disc. To simulate this action, the user must erase a disc record, then create another record with the same name.

A hardware error does not suspend execution of any disc I/O operations involving Files. These errors are unique, for they result from the faulty operation of the computer, not the blundering of a user. When the computer errs in a disc I/O operation (a parity check fails), it informs the POL system. The system will attempt three transfers from core to disc before a hardware error is signalled. Although some data in the record made in the third attempt is incorrect, the record is left on disc. Since partially correct data are far better than none, the tainted data remain on disc.

Files 27 through 29 with Key 1 are transferred from core onto disc with the instructions

```
F:OPEN  1;27
F:PUT   3;ABC
```

The disc records are named ABC027.LDA,ABC028.LDA, and ABC029.LDA. If a record named ABC029.LDA already resides on disc, the first two Files in the sequence are transferred without error, but the third flags a transfer error. The user must consult his notebook, which lists the names of disc records, to determine which record might be duplicated.

Files are copied from disc to core by the F:GET statement. It is implemented like the F:PUT command. The formats of these two commands are identical as are the ranges of their operands.

```
F:OPEN  1;26
F:GET   1;ABC
```

With these Immediate Instructions, disc record ABC026.LDA is transferred into File 26 in core but remains unchanged. Space was allotted to File 26 under Key 1 prior to the transfer.

If no space were reserved or if File 26 were assigned a different Key, an allocation error is flagged. No error results when the amount of reserved core is too large or small to accommodate the disc File. The Label of the disc File is duplicated in core except that the first word of the Label, the number of Slots in the File, remains. Then data succeeding the Label are deposited sequentially in Slots until none are available. If an excess of Slots exists, the excess is zeroed.

A transfer error results when a non-existent disc record is referenced. Before a hardware error is flagged, the POL system attempts three disc to core imagings so that the record is correctly copied

from disc. A hardware error does not terminate execution, but the File in core is not correctly copied from disc.

A sequence of disc files is eliminated by an F:RID command. It is implemented like the F:GET and F:PUT directives. The operand fields and their ranges for the three commands are identical. When the F:RID command executes, the Key of the F:OPEN command is ignored, for no Key is associated with disc records. To erase disc files ABC027.LDA, ABC028.LDA, and ABC029.LDA, the user types

```
F:OPEN  0;27
F:RID   3;ABC
```

The reason for allowing the F:OPEN statement to have a Key of zero is now apparent. After any disc I/O operation involving Files has executed, the first and second operands of the F:OPEN command are erased (zeroed) from the POL system. As before, the alphabetic name and a File number are concatenated to reference a disc record. When this command references a non-existent record, a transfer error is flagged.

2.3 Allocating Memory for Programs

Before any Program is assembled and executed by the POL operating system, space in memory must be allotted to the Program. Space is not reserved for Immediate Instructions, for these are instantly executed following their assembly.

Two words of memory store each POL command of a Program. The lower order byte of the first word contains the command number. Each of the 63 POL commands is assigned a number from 0 through 62. The higher order byte contains the first operand of an instruction, while the second word holds the second operand. If a command has a single operand, it is placed in the second word. All numeric operands are coded in binary. During assembly, operands may be modified. If an operand is an alphabetic name, it is converted into Radix 50. (See Appendix D.)

The P:ASSIGN instruction allocates memory to a Program. Twenty instructions are allotted to Foreground Program 1 with the command

```
P:ASSIGN  1;20
```

The first operand is the Program number; the second is the maximum number of statements which can be entered into the Program. During assembly, fewer than the maximum number may be entered. One through one hundred statements may be reserved for a single Program, but the second operand cannot be zero or less. Program numbers vary from 0 through 4. Numbers 1 through 4 reference Foreground Programs, while number 0 is associated with the Background Program. At most, a single Background Program resides in core at any instant.

Like any command which manages the storage of Programs, the P:ASSIGN statement is issued only as an Immediate Instruction. Else, a usage error is signalled on the Decwriter. If insufficient memory is available to accommodate a P:ASSIGN request, an allocation error is flagged. This error is also produced when the following constraint

is violated. Until a Program is erased, only one P:ASSIGN request reserves space for a particular Program. The P:ASSIGN command does not execute if any of these errors are flagged.

Programs are erased by the P:NULL command. Its format is

P:NULL Program number

The single operand is a Program number in the range 0 through 4. When this statement executes, all space that belonged to the Program returns to the pool of unused core. An allocation error is flagged following an attempt to erase a program that is executing or is assigned no memory.

2.4 Disc I/O Operations Involving Programs

Three instructions perform disc I/O operations, so that Programs are maintained as permanent records. One statement transfers a Program from core to disc, another images a Program from disc to core, and yet another eliminates Programs on disc. All three statements are issued only as Immediate Instructions. The operand fields for these commands are identical:

Program number; alphabetic name

To reference a disc file, the POL system concatenates the alphabetic name with a "P", followed by the Program number, 00 through 04.

P:PUT 1;ABC

For example, this statement transfers Foreground Program 1 from memory onto disc and names the disc listing ABCP01.LDA. The Program in core is unchanged. Note that the statements recorded on each media are in assembled form. (See Appendix D.) An allocation error is flagged if the indicated Program does not exist in core. Hardware and transfer errors are generated like those of the F:PUT instruction.

The P:GET command images disc programs into memory. Prior to the transfer, core for the Program must be reserved through the P:ASSIGN directive. Otherwise an allocation error results. The command

```
P:GET  1;ABC
```

moves disc record ABCP01.LDA into the memory reserved for Foreground Program 1, but does not modify the disc listing. Following the transfer, the POL monitor scans the core listing to check Keys. All Files referenced in the Program must be allocated to it, and the Peripherals that it references cannot be used in other Foreground Programs. Otherwise, an allocation error is flagged.

This error is also flagged if the disc record might write over a Program that has previously been entered. (With the REENTER and IGNORE commands, a disc program can be modified once it is transferred into core. See § 3.1.) Hardware and transfer errors are generated by the same mistakes noted for the F:GET statement. If any mistake is flagged by the P:GET statement, the Program marked by the first operand is cleared but remains in memory. The memory remains assigned to the Program. This drastic action is taken to prevent one user's errors from destroying others' efforts.

The P:RID statement eliminates records of programs that are on disc. When the following statement is entered, disc record ABCP 1.LDA is erased.

```
P:RID  1;ABC
```

A transfer error is flagged if the indicated disc record does not exist.

2.5 Summary

This chapter describes how the programmer and the POL system manage the storage media, i.e., disc and core. Although different instructions manipulate data Files and Programs, the user should recognize the similarities between the operand fields and the action of these different instructions. Programs are manipulated by commands whose mnemonics begin with "P:". File management is directed by statements whose mnemonics begin with "F:".

A sequence of two Immediate Instructions handles Files in core and on disc. Although instructions create, erase, and clear Files in core, only the first two operations are performed on Programs. These are managed with a single instruction. Disc records of Programs or Files may be created from records in core, imaged to core, or deleted.

As in later chapters, this one ends with a tabular summary of newly introduced commands. Their mnemonics describe their function. Along with other information associated with the commands, operands and their legal ranges are shown in Table 2.1. Since all instructions

flag operand and usage errors, these are not listed in the error column. However, legal programming environments are noted for each instruction so that the user can decipher usage errors.

Table 2.1: Commands that Manage Storage Media

Mnemonic	First Operand	Range	Second Operand	Range	Legal Environments	Errors Flagged
Commands that Manage Storage in Core						
F:OPEN	Key	0-4	first File	0-110	I	ALLOC
F:ASSIGN	number of Files	1-50	number of Slots	1-2000	I	ALLOC
F:CLEAR			number of Files	1-50	I	ALLOC
F:NULL			number of Files	1-50	I	ALLOC
P:ASSIGN	Program number[3]	0-4	number of statements	1-100	I	ALLOC
P:NULL			Program number[3]	0-4	I	ALLOC
Commands that Manage Storage on Disc						
F:PUT	number of Files	1-50	alphabetic name[2]	αβγ	I	ALLOC,TRANS,HRDWR
F:GET	number of Files	1-50	alphabetic name[2]	αβγ	I	ALLOC, TRANS,HRDWR
F:RID	number of Files	1-50	alphabetic name[2]	αβγ	I	TRANS
P:PUT	Program number[3]	0-4	alphabetic name[2]	αβγ	I	ALLOC,TRANS,HRDWR
P:GET	Program number[3]	0-4	alphabetic name[2]	αβγ	I	ALLOC,TRANS,HRDWR
P:RID	Program number[3]	0-4	alphabetic name[2]	αβγ	I	TRANS

Notes: 1. Programming environments are: I = Immediate Instruction, B = Background Program, and F = Foreground Programs.
2. Alphabetic names are composed of exactly three characters, A through Z.
3. Numbers 1 through 4 reference Foreground Programs; 0 denotes the Background Program.

3. The Assembly and Execution of Programs

Once space in core is allotted to a Program, two further steps are required to realize the programmer's intentions. The Program must be loaded and interpreted, or assembled, into the space. Then the Program may execute.

3.1 Assembly of Programs

The assembly of a Program is initiated by the START command, which is used only as an Immediate Instruction. After all statements have been entered into a Program, the END command signals the POL system that assembly is complete. Immediate Instructions may be entered on the Decwriter again. Remember, statements in a Program are preceded by any number of spaces or are separated by any number of lines. The user is responsible for making his Programs legible.

The START instruction has the format:

START Program number

Numbers 0 through 4 constitute the legal range of the operand. As always, 1 through 4 reference Foreground Programs while 0 denotes the Background Program. If no core is reserved for the Program or it has already assembled, an allocation error is flagged.

Although the END command requires no operands, the Program number may be typed in the operand field. The END command is entered only

as the final statement in a Background or Foreground Program.

Example 3.1 illustrates the complete sequence for the construction of a Program -- allocating space in core, starting assembly, entering statements, and ending compilation. A statement in a Program must be allocated for the END command but is not needed for the START command. This command is not compiled into a Program and simply delineates the start of assembly. If more statements than space permits might be entered into a Program, an EOF error succeeds the violating statement. It is completely ignored by the POL system.

When a string of commands produces a logic error in assembly, the IGNORE directive aids in recovery. A logic error occurs in a Program that the POL system recognizes as entirely correct but would not execute as the user intends. These errors should not be confused with those flagged on the Decwriter.

The single operand of the IGNORE command delimits the number of statements preceding it which are to be eliminated.

IGNORE number of statements

After the complete IGNORE command is typed, the space which was assigned to the erased statements becomes free and may be written over with other instructions. The IGNORE command is not compiled; it occupies no space in the assembling Program. When an IGNORE instruction eliminates a statement that reads data or controls an experiment, the Peripheral allocated by this statement is released from the Program. That is, the Key of the Peripheral is erased. The Peripheral is reassigned by referencing it with a statement later in the Program.

Example 3.1: The Sequence of Program Construction

```
P:ASSIGN          0;50

START             0
   :
   :

Statements of the
Program
   :
   :

END
```

The number of instructions that are erased may range from 0 through 100. If the operand is zero (blank) or one, the command prefixing the IGNORE directive is nulled. An EOF error is flagged when this directive might eliminate more statements than are in a Program. In this case, the entire Program is cleared, but the space that was allotted to it remains. The IGNORE command is issued only during assembly. If this statement is used as an Immediate Instruction, a usage error results.

Suppose a user made a mistake in a statement of a Program, entered two more statements, and then realized that he had erred. Rather than erasing the entire Program and re-entering it, he issues the command

IGNORE 3

This erases the previous two statements and the incorrect statement. Starting with the statement that was in error, he now can enter the rest of his Program.

To correct statements in, or append statements to a Program following assembly, the REENTER command removes the END directive from a particular Program. Assembly is then re-initiated at the end of the program.

REENTER Program number

As usual, the Program number may vary from 0 through 4. The REENTER command, like the START directive, is used only as an Immediate Instruction. If this command references a Program that is allotted no core, no Program is assembled in the allocated space, or the

Program is currently executing; an allocation error is flagged.

The REENTER and IGNORE statements are programmed in concert to recover from logic errors discovered following assembly. Together, these statements are also used to clear the contents of a Program but retain the space in core that was assigned to it. Following assembly, the sample Program of this section is cleared by the sequence:

```
REENTER   0
IGNORE   10
ERROR   EOF
END
```

(Messages produced by the POL system are underlined.) Because the number of statements to be erased exceeds the number used in the Program, an EOF error is flagged. Despite this, the Program is cleared. Program 0 is in the same state as when the P:ASSIGN directive was given, except assembly must be initiated by a REENTER instruction.

When the LIST command is issued, the contents of any Program in core are printed on the Decwriter in assembled form.

LIST Program number

The Program number may range from 0 through 4. Statements are listed in the sequence that they were entered. Only the first three characters of the mnemonic are written. In general, operands appear as they were entered. However, a few operands are modified in assembly, but all are coded in decimal. (See Appendix D.)

If the operand is the number of a Foreground Program, 1 through 4,

then all Files which have been assigned to that Program are displayed following the Program Listing. Unused Files appear after the listing of a Background Program. This feature exposes unused Files for a programmer.

The LIST command is used only as an Immediate Instruction. If the Program referenced by the operand is allotted no core, and no Files are assigned to the Program, then nothing, not even an error message, is printed.

3.2 Execution of Programs

Once a Program is assembled, it may be run any number of times. The EXECUTE command initiates execution.

EXECUTE Program number

Again, 0 through 4 are legal operands. If this statement is not issued as an Immediate Instruction, a usage error is flagged. An allocation error results when the directive might disturb a Program that is currently executing or might attempt to run a Program that has not been assembled.

A Foreground Program is executed periodically through the Interrupt System. When the Background Program, number 0, is processing, Immediate Instructions are not recognized by the POL monitor until processing ends. An error message, except a hardware error, signals the abnormal suspension of execution for a Background Program. A

STOP message,

STOP Program number

usually indicates the normal (error-free) termination of processing. The sole exception arises when a Program references a File that is allotted no core.

Only the execution of Foreground Programs, numbers 1 through 4, may be prematurely suspended by an ABORT directive. When an experiment appears to be failing, the user may issue this statement, but only as an Immediate Instruction. The Program number, the second operand, must match the Key (first operand) of the last F:OPEN statement entered in the Background mode. Else an allocation error is flagged. This feature prevents one user from interfering with another's efforts.

Because ABORT Immediate Instructions are not recognized during the execution of another Immediate Instruction or a Background Program, the ABORT command does not terminate these processes. Moreover, an entire Background Program can often execute faster than a statement can be typed. Instructions and programs involving long I/O operations can be prematurely terminated with the "Panic" button. This switch is located, and clearly marked on the upper left-front of the computer rack. Only operations involving the Decwriter or the point-plotter may be terminated. To suspend these processes, the "Panic" button is depressed until the operation has visibly halted. No memory of the stopping point is retained. Once the "Panic" button is pushed, the user may not resume the operation, other than by re-executing the instruction or Program.

The PROGRESS command ascertains the status of an executing Foreground Program, numbered 1 through 4.

PROGRESS Program number

If the indicated Program is not executing, then no action is performed by this statement. Otherwise, a listing of Loop numbers versus remaining iterations is printed on the Decwriter. The extent to which the Program has executed is determined from the listing. This command is used only as an Immediate Instruction.

3.3 Iterative Execution in Programs

This section and the next discuss two groups of statements most commonly entered in Programs. One group executes sections of a Program iteratively; the other group numbers the File that is manipulated by a Program.

Perhaps the most powerful asset of a computer is its ability to perform the same function over and over again. The LOOP and CONT statement of POL direct the computer to process a segment of a Program, either Background or Foreground, iteratively. These statements are always used in pairs. When the first operand of the LOOP command matches the single operand of the CONT statement, the sequence of instructions between them will execute a number of times given by the second operand of the LOOP statement.

LOOP Loop number; number of iterations
CONT Loop number

The first operand of the LOOP command and the single operand of the CONT command, which serve as identification numbers, assume values from zero to three, inclusive. Hence, Loops may be nested up to four times in any section of a Program. However, a pair of these statements may occur any number of times in a Program as long as the Loops are nested properly. Otherwise, an order error is flagged in assembly. The reader should be acquainted with nesting, but Example 3.2 demonstrates correct and erroneous techniques. A subtle form of nesting error is illustrated by the second Program in the example. Two LOOP statements with the same first operands are not separated by a CONT instruction with an identical operand. An ordering error is also generated if: (1) some command does not separate LOOP and CONT instructions having the same Loop number, or (2) a File-referencing statement does not precede a LOOP command. (File-referencing statements are the subject of the next section.)

Under POL, iterative execution is never conditional. When a Program runs, the segment of POL code between the LOOP and CONT statements is always executed the number of times given by the second operand of the LOOP command. This process is never ended prematurely unless an error occurs.

Example 3.2: Nested Loops

Incorrect

```
LOOP  0;4
  :
LOOP  1;10
  :
CONT  0
  :
CONT  1
```

Incorrect

```
LOOP 0;10
  :
LOOP 0;4
  :
CONT 0
  :
CONT 0
```

Correct

```
LOOP 0;10
  :
LOOP 1;4
  :
CONT 1
  :
LOOP 1;4
  :
CONT 1
CONT 0
```

3.4 File-Referencing Statements

A File number is seldom an operand of an instruction. Instead, a File-referencing statement directs all succeeding instructions in the Program to operate on the File numbered by the statement. Until another File-referencing statement is issued, the File numbered by the single operand of the S:FILE statement is manipulated.

S:FILE File number

Suppose the File number were 10, and this statement were entered in Program 1. Then until another S:FILE or I:FILE directive is issued in Program 1, succeeding instructions that operate on a File manipulate File 10.

The S:FILE statement is used in all environments – in Foreground Programs, in the Background Program, or as an Immediate Instruction. The operand of the S:FILE statement pertains only to instructions of the Program in which it is issued. When this statement is an Immediate Instruction, all succeeding Immediate Instruction manipulate the File numbered by the single operand. The POL system stores this operand until a Background Program executes. Then the File number is erased.

If an instruction that should be preceded by an S:FILE statement is not, an order error is flagged in assembly. All errors of the S:FILE and I:FILE statements are generated in assembly. The single operand of the S:FILE statement must be a legal File number, 1 through 110.

When a free File is referenced, an allocation error is flagged. The same error results if: (1) the Key of the File and the number of the Foreground Program referencing the File differ, or (2) the Key of a File referenced by a Background Program or an Immediate Instruction does not match the Key (first operand) of the last F:OPEN statement issued in the Background mode. Remember, the Key of the F:OPEN statement is erased following the execution of a memory or disc management operation involving Files.

When a user begins employing data analysis or display commands in POL, he should issue the F:OPEN command with the Key equaling the number of his Foreground Program and the File number equaling zero.

```
F:OPEN  Key;0
```

If he executes a disc or memory management command by mistake, the command does not execute. An error is flagged instead. After he finishes manipulating his Files in the Background mode, he should enter the command

```
F:OPEN  0;0
```

This action protects the user's Files if another programmer forgets to deposit his Key.

The I:FILE statement also delineates which File is manipulated by a Foreground or Background Program. When this instruction executes, the operand is added to the File number which was referenced last in a Program. The File numbered by the result is operated on until another I:FILE or S:FILE statement is issued. To program a series of complex

Example 3.3: An Iterative Program

```
START      0
  :
S:FILE     10
  :
LOOP       1;5
  :
string of complex manipulations
  :
I:FILE     2
CONT       1
  :
END        0
```

manipulations on a sequence of Files easily, the I:FILE command is placed in a Loop. The Program in Example 3.3 operates on Files 10, 12, 14, 16, and 18. Note how the START and END commands initiate and end assembly.

An increment, the single operand of the I:FILE directive, may range from -5 through 5 Files, but it cannot be zero. If an S:FILE statement does not prefix the I:FILE statement, an order error results. Unlike the S:FILE statement, the I:FILE statement cannot be entered as an Immediate Instruction without flagging a usage error. All Files generated by an I:FILE statement must have: (1) space in core reserved for them, and (2) Keys corresponding to the number of the Foreground Program referencing them, or (3) Keys corresponding to that of the current F:OPEN statement issued in the Background mode. Otherwise, an allocation error is flagged. The third constraint applies only to Background Programs in assembly. In Example 3.3, Files 10, 12, 14, 16, 18, and 20 must meet these conditions.

3.5 Summary

Whereas the previous chapter was concerned with the management of storage media, this chapter describes how the assembly and execution of Programs is managed. Four statements that are most commonly entered in Programs were also discussed.

The START and END directives initiate and terminate assembly. During assembly, the POL system temporarily stops acknowledging

Immediate Instructions so that statements may be entered into a Program. The IGNORE and REENTER commands allow the user to recover from errors during or following assembly. The EXECUTE instruction initiates the processing of any assembled Program; the ABORT command prematurely terminates the execution of a Foreground Program. The "Panic" switch aborts long I/O operations in the Background mode. The LIST and PROGRESS statements print listings on the Decwriter so that the user may obtain information concerning his Programs.

The LOOP, CONT, S:FILE, and I:FILE instructions are the most common statements entered in Programs. When used in concert, the LOOP and CONT statements iteratively execute a segment of a Program. The number of iterations is never conditional but always is given by the second operand of the LOOP command. The S:FILE and I:FILE statements number a File that is to be manipulated by a Program or an Immediate Instruction. To prevent one user from destroying another's Files a protection system is implemented in both the Foreground and Background modes of POL.

Table 3.1: Commands that Manage the Assembly and Execution of Programs

Mnemonic	First Operand	Range	Second Operand	Range	Legal Environments[1]	Errors Flagged
Commands that Manage Assembly of Programs						
START			Program number[2,3]	0-4	I	ALLOC
END			Program number[2,3]	(0-4)	B,F	
IGNORE			number of statements	0-100	B,F	EOF
REENTER			Program number[2]	0-4	I	ALLOC
LIST			Program number[2]	0-4	I	
Commands that Manage Execution of Programs						
EXECUTE			Program number[2]	0-4	I	ALLOC
ABORT			Program number[2]	1-4	I	
PROGRESS			Program number[2]	1-4	I	
Commands Most Commonly Entered in Programs						
LOOP	Loop number	0-3	iterations	1-2000	B,F	
CONT			Loop number	0-3	B,F	ORDER
S:FILE			File number	1-110	I,B,F	ALLOC
I:FILE			increment	±5,≠0	B,F	ORDER,ALLOC

Notes:
1. Programming environments are: I = Immediate Instructions, B = Background Program, and F = Foreground Programs.
2. Numbers 1 through 4 reference Foreground Programs; 0 denotes the Background Program.
3. This operand is optional.

4. Monitoring Experiments

This chapter introduces all statements employed in Foreground Programs which monitor experiments. If these commands are entered in other Programming environments, a usage error is flagged in assembly. All statements described in this chapter, except for programmed delays, must be prefaced by a File-referencing statement. Otherwise, an order error is generated during assembly. At that time, allocation errors are flagged to prevent two Programs from manipulating a particular Peripheral resource or File. Peripheral resources are: DVMs, pulse counters, ADC Channels, or DAC Channels. All Foreground Programs may read the system clock.

Except when a File-referencing statement numbers a File that is allotted no space in memory, Foreground Programs produce no errors during execution. In this case and following the normal end of processing, the STOP message is printed on the Decwriter.

Other commands that are employed legally in Foreground Programs, in addition to those described in this chapter, are: END, S:FILE, I:FILE, LOOP, and CONT. In general, these five commands do not flag errors as noted above. The reader should review these commands at this time. (See Table 3.1.)

Most Foreground commands load information into either the Label or Slots of Files. Filling a location in the Label is seldom the sole action of a command but is implemented as a secondary function. Once this feature is mastered by the user, writing Foreground Programs becomes fast and facile. Although Slots may be loaded in any order,

the sequence should be dictated by the way the data are to be displayed and analyzed. Remember: a Foreground Program puts data into Files; the user directs the computer to collect data intelligibly.

4.1 Commands that Read Data

Four classes of Peripheral devices read data: a system or computer clock, digital voltmeters (DVMs), ADC Channels, and pulse counters. The former two devices are read asynchronously. As soon as the read request is encountered in processing, the data are collected. However, the latter two devices are serviced synchronously. This type of read request is queued during the Interrupt Period in which it is encountered. Throughout the next Period, the datum is read. At the beginning of the following Period, the data point is stored. The POL system automatically performs the three steps of a synchronous read request after a single instruction is recognized in execution. In summary, an asynchronous request obtains data during the Interrupt Period in which the request is issued, while a synchronous request collects data during the succeeding Interrupt Period.

Each data point is stored in the File numbered by the most recent File-referencing statement. The storage pointer, the second word of the Label, dictates which channel will store the information. A F:ASSIGN or F:CLEAR command sets the storage pointer to the first Slot of a File. The pointer is incremented by one Slot after a read request is processed completely. If incrementing might cause the pointer to

reference a location past the last (first) Slot of a File, the pointer is reset to the first (last) Slot of the File. Storage is controlled automatically, as described; or by commands, discussed in the next section.

Each File should store only one experimental variable. When a read statement first deposits a data point in a File, the type of read request is noted in the option byte of the Label. The option denotes the Peripheral which was read and the type of data which were collected. This option will pertain to all data stored in the File.

All read requests have two operands, except the command that reads the system clock. The latter has no operands. The first operand is always a device number. All DVMs, pulse counters, and ADC Channels are numbered.

Since Interrupts occur over a short time scale, 16 msec., compared to that of an experiment, data are often accumulated or averaged over many Interrupt Periods. The second operand of synchronous read requests is the length of time over which an observation occurs. Because DVMs average over tenths of seconds by virtue of their electronic operation, this operand is not associated with these devices. From a single synchronous request, data are collected for an interval of up to 10 sec, but at least for one Interrupt Period, 16 msec. During assembly, operands which denote a length of time are always converted to the nearest number of Interrupt Periods.

When a Foreground Program might read a particular Peripheral more than once during an Interrupt Period, processing of that Program ceases for that Interrupt Period. (ADC Channels are separate

Peripherals.) This is one of several conditions that causes a Foreground Program to cease processing during a Period. By coordinating these conditions, a user times the execution of his Programs.

If time is an operand of a read statement, the POL system processes only that command for several Interrupt Periods. For example, if a Peripheral is to be read for three Periods, 0.048 sec, the read request is processed as follows. During the first Period in which the command is encountered, the read request is queued. Execution of that Program is suspended until the next Period. Then the request is queued again and execution is suspended. In the third Period, the read request is queued once more, and then the next statement in the Program is processed.

Statements in programs are processed sequentially. As is the case with most computer languages, no "look ahead" capability has been incorporated into POL. That is, no statement partially executes because it was anticipated by the POL system.

The R:CLOCK command stores the current reading of the system clock in the File numbered by the last File-referencing statement. Time is resolved into milliseconds. Each reading is an absolute time. The clock is zeroed only when POL starts up, not when a Foreground Program begins executing. After the clock completes an entire cycle, which requires about two months, the time is again zeroed. The OFFSET command, a data analysis directive, sets the relative zero point in time for any File. (See § 6.3.) No operands are associated with the R:CLOCK statement.

Example 4.1: Reading the System Clock

```
S:FILE   10
LOOP     0;50
R:CLOCK
CONT     0
```

Table 4.1: How Application Modes of DVMs are Associated with Experimental Variables

Application Mode	Experimental Variable	Range Setting
0	CMA:pass energy, LEED:gun voltage	10
1	IET:junction voltage	1
2	CPD:gun voltage	100
3	Thermocouple:EMF	1
4	Picoammeter:intensity	10
5	Mass Spectrometer:intensity	10
6	Lock-in Amplifier:intensity	10

The Foreground Program in Example 4.1 samples the system clock fifty times and stores the observations in sequential Slots of File 10. When a Program might read a device more than once during a single Interrupt Period, the execution of that Program is suspended for that Period. Hence, the contents of adjacent Slots of File 10 will differ by 16 msec, the length of an Interrupt Period. Note, a R:CLOCK directive samples the time at which an Interrupt Period begins. Time readings always differ by multiples of 16 msec.

As with the synchronous read statements, the request to read a DVM has two operands:

R:VM device number; application mode

The two voltmeters in the lab are numbered 0 and 1. With a range from 0 through 6, the application mode describes the type of experimental data which are collected by the voltmeter. The application modes are correlated with experimental variables in Table 4.1. To insure that data are correctly displayed, the range setting of a DVM is selected as shown in this table. Numbers of experimental options are obtained from application modes by adding eleven to the number corresponding to the latter.

The Program of Example 4.2 monitors the intensity of a LEED beam as a function of time. Voltmeter 1 samples the output of the pico-ammeter. Since only one type of variable should be stored in each file, intensities and times are stored in Files 10 and 11, respectively. These assignments are noted by the experimental option which is contained in the Label of each File. In each Interrupt Period, the

the voltmeter and system clock are read. When the R:VM statement is encountered for the second time, processing ceases until the next Period. Before execution is completed, this cycle is iterated 1000 times. The experiment occurs over 1000 x 16 msec = 16 sec.

Data points are deposited in the Slots of each File in the order that they are collected. If Files 10 and 11 are cleared prior to execution, the first observations are contained in the first Slots and the last readings are in the last Slots.

Voltmeters 0 and 1 read at a maximum rate of one point per 500 msec. At most, 33 voltmeter readings are unique. This example illustrates that the POL system does not always prevent the user from performing ludicrous experiments. He must understand the limitations of all experimental equipment that he employs.

If both Files 10 and 11 were allotted only 999 data slots, the final readings of the DVM and clock are written over the initial observations in the first Slot. The storage pointer is reset to the first Slot of a File whenever the pointer might mark a location past the end of the File. Synchronous read directives do not write data over readings that were previously acquired. Instead, the latest observation is added to the contents of the Slot. For this reason and to set up the Label, Files should be cleared prior to execution. Remember, either a F:ASSIGN or a F:CLEAR statement clears a File.

Pulse counters are sampled with the R:PC command.

R:PC device number; time

The first operand identifies which pulse counter is to be read. The

Example 4.2: Monitoring LEED Intensities with a DVM and as a Function of Time

```
LOOP          0;1000
S:FILE        10
R:VM          1;4
S:FILE        11
R:CLOCK
CONT          0
```

Example 4.3: Monitoring LEED Intensities with an ADC Channel and as a Function of Time

```
LOOP          0;100
S:FILE        21
R:AD          3;1.00
S:FILE        20
R:CLOCK
CONT          0
```

pulse counter on the CMA is numbered 0 while the one for IESS is numbered 1. To produce an acceptable signal-to-noise ratio, a pulse counter is read for about a second under each set of experimental conditions, but the observation time varies markedly with the particular conditions. The second operand denotes the sampling time for each data point. Time is entered in seconds and may vary from 0.001 to 10.000 sec. During assembly, this operand is converted into Interrupt Periods by conventional rounding procedures.

With the exception of the first and last Periods that the pulse counter is read, the R:PC command is the only one serviced for the number of interrupt periods noted. Each observation queued by a R:PC statement is added to, not written over, the previous value in the Slot marked by the storage pointer. If a single R:PC command executes over several Interrupt Periods, the storage pointer is not incremented until all readings have been added to the current Slot. Since a pulse counter and a DAC Channel usually sample an experimental spectrum in concert, examples utilizing the R:PC command are postponed to § 4.3 where statements that drive DAC Channels are discussed.

The R:AD command collects data from ADC Channels.

R:AD device number; time

Each ADC Channel, or device, is numbered 0 through 7 and is assigned by the POL system, not the user, to a particular experimental variable. Along with the sampling ranges, Channel assignments are listed in Table 4.2. Because signals are often preconditioned before entering a Channel, actual input ranges are also noted. All Channels are

linear to one part in 4096, but their absolute accuracy is 0.1%. Only two separate ADC Channels may be read in a single Interrupt Period. A third request is deferred until the next Period. The reading time may vary from 0.001 to 10.000 sec and is assembled into Interrupt Periods.

When an ADC Channel is sampled for several Interrupt Periods under one request, the storage pointer is incremented following the last Period. Consecutive observations are summed in a single storage location. Following the last sampling, the contents of the Slot are averaged by the number of observations. No averaging occurs if the Channel is read for 0.023 sec or less. Hence, the clever programmer can use this single command to average readings or to accumulate observations like a pulse counter.

The Program in Example 4.3 performs an intensity versus time measurement similar to that described in Example 4.2. The picoammeter signal is fed into ADC Channel 3 rather than voltmeter 1. Each picoammeter sample is averaged for 1.00 sec or 63 Interrupt Periods (rounded). Clock readings are stored in File 21, while ADC data reside in File 20. During the Period that the clock is read, the last of 63 ADC observations is added to the Channel accumulating these readings. Afterwards, the sum is averaged and the storage pointer incremented. The experiment lasts for:

$$100 \text{ Loops} \times 63 \frac{\text{Periods}}{\text{Loop}} \times .016 \frac{\text{sec}}{\text{Period}} = 108 \text{ sec}$$

Table 4.2: The Assignment of ADC Channels to Experimental Variables

Channel Number	Experimental Variable	Sampling Range	Actual Input Range
0	LEED: gun voltage	0 to 1500 eV.	0 - 1.5 V.
1	unassigned	-	0 - 10 V.
2	Thermocouple:EMF	0 to 10.00 mV.	0 - 10 V.
3	Picoammeter:intensity	full scale[1]	0 - 3 V.
4	Mass Spect.:intensity	full scale[1]	0 - 10 V.
5	Lock-in Amp.:intensity	full scale[1]	±10 V.
6	Lock-in Amp.:intensity	full scale[1]	±10 V.
7	unassigned	-	0 - 10 V.

Note: 1. "Full scale" indicates that the range selection switch on a device is employed. Although the device may be read over the full range at any setting, no record of the setting is maintained by the POL system. Instead, it is recorded by the user in his lab notebook.

4.2 Controlling the Storage of Data

The S:STORE and I:STORE commands manipulate the storage pointer of Files. Both instructions operate on the File numbered by the last File-referencing statement. Although storage pointers are automatically managed by the POL system, commands that move the pointer allow greater flexibility in storing data.

The S:STORE command positions the storage pointer of a File to the Slot numbered by the single operand.

```
S:STORE   Slot number
```

Slots of a File are numbered 0,1,2,...,(n - 1) where n is the number of Slots in the File. This operand can be as large as 2000. If the Slot number exceeds the range of the File, nothing happens when the S:STORE instruction processes.

At the start of execution, a careful user zeroes the storage pointers of all Files in a Foreground Program. This action is accomplished in example 4.3 by prefixing this Program with the statements;

```
S:FILE    20
S:STORE    0
S:FILE    21
S:STORE    0
```

After a data reading operation, the storage pointer is automatically incremented. With the I:STORE command, the pointer may be incremented or decremented, or its position may remain unchanged. This statement should always immediately succeed a request to read data. Otherwise an order error is flagged in assembly.

I:STORE increment

The single operand assumes values of -1,0, and 1. When the operand is one, the storage pointer of the File numbered by the last File-referencing statement is incremented by one Slot. This action is identical to the normal incrementing process that follows a read statement.

When the operand of the I:STORE command is zero, the storage pointer does not change after a read request. With this operand, the I:STORE and R:AD statements are programmed in concert to free an ADC Channel to act as an accumulator rather than as an averager. Thus, successive readings are added and stored rather than added, averaged, and stored.

The Program in Example 4.4 illustrates how the experiment of Example 4.3 is performed using ADC Channel 3 as an accumulator. Remember, a R:AD instruction does no averaging when the time operand is 0.023 sec or less. After the storage pointer is kept stationary for many iterations of Loop 1, the R:AD command outside the Loop increments the storage pointer automatically. An

I:STORE 1

directive may succeed the second R:AD statement, but the execution of the Program would not change.

If the operand of the I:STORE command is -1, the storage pointer is decremented following a read request. When used in the same program, the I:STORE and S:STORE commands can store data from the end to the

Example 4.4: An ADC Channel used as an Accumulator

```
LOOP            0;100
S:FILE          21

LOOP            1;62
R:AD            3;0.016
I-:STORE        0
CONT            1
R:AD            3;0.016

S:FILE          20
R:CLOCK
CONT            0
```

beginning of a file. This technique is described in the next section. Whenever the storage pointer might be incremented (decremented) beyond the end (beginning) of a file, the pointer is set to the first (last) Slot instead.

4.3 Controlling Experimental Parameters with DAC Channels

An 8-Channel DAC unit controls experimental parameters. As shown in Table 4.3, each Channel is assigned permanently to a single parameter. Each parameter can be resolved to one part in 40,960 of the full-scale range, but hardware limitations reduce the absolute accuracy of any setting to 0.1%. However, each Channel is linear to 0.0025%. After the output of any Channel is set, it may drift by as much as 0.005% per Interrupt Period. The output of a Channel settles exponentially with a time constant of 400 μsec but changes over only 2 msec per 16 msec Interrupt Period. (See Chapter 9.) That is, for a step change in the full-scale output, the output settles within 1% of the true output in one Period. After two Periods, the output is accurate to within 0.01% of its final value.

When a DAC Channel monitors an experimental parameter, data are sampled at equal spacings of the parameter. For example, the counts from an Auger spectrum are taken at energies of 100.0, 100.5, 101.0,... eV, not 100.0, 101.2, 101.5... eV. Many data analysis routines in POL, such as the smoothing procedure, are applicable only to spectra that are sampled at equal spacings of the abscissa.

Two commands control the outputs of each Channel. One command sets the Channel to an initial value, while the second increments the output. Although two 12-bit D/A converters set the output of each Channel, the user does not manipulate the converters separately. The POL system does this automatically so that data points are sampled at equal spacings of the abscissa. Each Channel appears to have 15 1/2 bits of resolution to the user.

The first operand of the two commands that control the 8-Channel DAC unit is the Channel number, 0 through 7, which is manipulated. The second gives a starting value or increment of the output for the Channel. With the S:DA command, the output is set to an initial value. The I:DA statement increases the output signal from the Channel by an amount given by the second operand.

S:DA	Channel number; initial value
I:DA	Channel number; increment

Channel numbers vary from 0 through 7. The second operands are entered in units appropriate to the experimental parameter controlled by the Channel. These units are noted in Table 4.3. During assembly, the second operand is converted into a 16-bit number in twos complement code. (See Appendix D.) The POL system insures only that the initial value of the S:DA command is between 0.00 and 2000.00 and that the increment of the I:DA command is between -20.00 and 20.00. For the accurate control of an experimental parameter, the output of any DAC Channel must confine the parameter within the range listed for the Channel in Table 4.3. The user must insure that this constraint is

Table 4.3: The Assignment of DAC Channels to Experimental Parameters

Channel Number	Controlled Parameter	Range of the Parameter	True Output Range of the Channel
0	CMA: pass energy	0-2000.0 eV	0-5 V
1	LEED: gun voltage	0-1500.0 eV	0-10 V
2	Mass Spect.: mass select	0-300.0 AMU	0-10 V
3	IET: junction potential	0-1000.0 mV	0-10 V
4	unassigned[1]	--	0-10 V
5	unassigned[1]	--	0-10 V
6	unused[2]	--	--
7	unused[2]	--	--

1. Unassigned Channels have their true output range set but currently are not associated with an experimental parameter.

2. Unused Channels do not have their true output range set and are not associated with an experimental parameter.

observed.

When a S:DA command executes, the second operand is modified slightly (see Chapter 10) and placed in the Label of the File numbered by the last File-referencing statement. This operand is deposited in the fourth word of the Label, and the Channel number, the first operand, is placed in the higher order byte of the third word. Once the former location assumes a non-zero value, a S:DA instruction will not load the starting value or Channel number into the Label. The second operand of the I:DA command is also placed in the Label of the File numbered by the last File-referencing statement. This operand also is modified before it is deposited in the fifth word of the Label. After this location assumes a non-zero value, it will not change.

When an output of a DAC Channel might change more than once in a single Interrupt Period, the Foreground Program that manipulates the Channel ceases to execute for that Period. Only two different Channels may be monitored by a single Program. If a third Channel is referenced in a Foreground Program, an allocation error is flagged in assembly. However, both Channels may be manipulated in a single Interrupt Period.

To insure that a Program will execute with maximum accuracy, an experimental variable associated with a controlled parameter should be read immediately after the value of the parameter is fixed. Otherwise, hardware limitations garble the data. For example, a Foreground Program samples both Auger and mass spectra in the sequence

```
S:DA   0;100.0
R:PC   0;1.0
S:DA   2;18.20
```

not

```
S:DA   0;100.0
S:DA   2;18.20
R:PC   0;1.0
```

or

```
R:PC   0;1.0
S:DA   0;100.0
S:DA   2;18.20
```

In the first example, the pulse counter is sampled after the pass energy of the CMA is set. If this correct sequence is reversed, or other variables are set or read in between, the data are corrupted.

A R:AD or R:PC request is not serviced in the same Interrupt Period that a S:DA command is issued. Instead, reading is delayed for a Period. That is, the Foreground Program suspends its execution for a Period before a synchronous read request is acknowledged. This feature allows the output of the DAC Channel to settle before the reading is made.

A surface is examined for cleanliness with an Auger spectrum from 10.0 to 600.0 eV. Readings are made at intervals of 1.0 eV and for 0.4 sec. The Program of Example 4.5 acquires the spectrum and stores it in File 5. DAC Channel 0 controls the pass energy of the CMA, and pulse counter 0 accumulates pulses from the CMA. Although all spectra from the CMA should be obtained in the "Retard 1" mode, the pass energy is selected at the user's discretion. The Channel number of the DAC,0; the initial value, 10.0 eV; and the increment, 1.0 eV, are stored in the Label of File 5. Remember, the first Channel

Example 4.5: Collecting an Auger Spectrum

```
S:FILE     5
S:STORE    0
S:DA       0;10.0

LOOP       0;590
R:PC       0;0.4
I:DA       0;1.0
CONT       0
```

number, initial value, and increment following a File-referencing statement are placed in the Label of that File. These values generate the abscissa when the spectrum is displayed. Note the first reading of the pulse counter is delayed for an Interrupt Period when the output of DAC Channel 0 is first set. This allows the output to settle.

Although this example illustrates an acceptable method for sampling the spectrum, the method is far from optimum. The experiment lasts for approximately 4 min. On this time scale, low frequency noise, also known as base-line drift, may cause a slight degradation in the quality of the data. This noise is eliminated by sweeping the pass energy back and forth, and sampling several times at each energy. Example 4.6 illustrates this technique. Systematic noise is best eliminated by a clever measuring scheme. Random, Gaussian or white noise is filtered from spectra with a smoothing routine. (See § 6.2.)

The R:PC statement accumulates readings as shown in this example. Rather than sampling each point once for 0.4 sec, a point is observed at eight different times for 0.05 sec each time. Data acquired at one pass energy are summed in a common Slot of File 60. Although the first I:STORE statement is not needed, it makes the Program more interpretable. During the first iteration, the first S:STORE statement is unnecessary. However, it is essential in resetting the storage pointer in later iterations. When the second operand of an I:DA command is negative, an experimental parameter is ramped in a decreasing rather than increasing manner. The second I:STORE command steps the storage pointer downward so that each Slot samples counts at the

Example 4.6: Collecting an Auger Spectrum with Reduced Base-Line Drift

```
S:FILE      60
LOOP        0;4
S:DA        0;10.0

S:STORE     0
LOOP        1;590
R:PC        0;0.05
I:STORE     1
I:DA        0;1.0
CONT        1

S:STORE     589
LOOP        1;590
I:DA        0;-1.0
R:PC        0;0.05
I:STORE     -1
CONT        1

CONT        0
```

same energy while it is increasing and decreasing. S:STORE and I:STORE instructions should be inserted liberally in a Program to insure that it executes as the user intends.

The S:DA command can tune a DAC Channel onto a peak in a spectrum. For example, a quadrupole mass spectrometer samples the CO, O_2, and CO_2 peaks of a thermal desorption spectrum. Previous data indicate that the maxima of these peaks occur at DAC settings of 28.22, 32.88, and 44.34 AMU, respectively, when the sample heating rate is $10^{o}K$/sec. Suppose, the sample is initially at room temperature and all products desorb from the surface at $1000^{o}K$. The experiment is performed over about 70 sec, or 4300 Interrupt Periods. A Program that monitors the three mass peaks simultaneously is shown in Example 4.7.

DAC Channel 2 selects the mass peak, while ADC Channel 4 reads the intensity of the peak. The CO, O_2, and CO_2 intensities are stored in Files 21, 22, and 23, respectively. One hundred measurements are deposited in each File. ADC measurements are the average of samples obtained over 0.16 sec, or 10 Interrupt Periods. Employing application mode 3, voltmeter 1 reads the thermocouple. These observations are stored in File 20. After each read statement, the storage pointer of a File is automatically incremented. This File is numbered by the S:FILE statement preceding the read request. Note that the ADC Channel is not read in the same Interrupt Period that the output of the DAC Channel is set.

The first of the ten observations of each ADC reading is slightly inaccurate since a DAC Channel settles to within 99% of the full scale change in a single Period. A better scheme for sampling the mass

Example 4.7: Monitoring Three Thermal Desorption Mass Peaks Simultaneously

```
LOOP        0;140

S:FILE      20
R:VM        1;3

S:FILE      21
S:DA        2;28.22
R:AD        4;0.16

S:FILE      22
S:DA        2;32.88
R:AD        4;0.16

S:FILE      23
S:DA        2;44.34
R:AD        4;0.16

CONT        0
```

peaks is to delay the first observation for an additional Interrupt Period. Execution of a Foreground Program is deferred with a programmed delay.

4.4 Programmed Delays

Programmed delays are statements that postpone the execution of a Foreground Program for one or more Interrupt Periods. These statements allow experimental devices to settle or experimental parameters to be modified manually before an observation is sampled.

The W:CLOCK instruction suspends the execution of a Program for an interval of time specified by the single operand.

```
W:CLOCK     time
```

The time is entered in seconds and may vary between 0.001 and 10.000 sec. During assembly, the time is converted into the nearest number of Interrupt Periods. When a W:CLOCK statement is encountered in execution, processing of the Program ceases immediately, and is not resumed until this number of Periods has passed. This command always postpones execution for at least one Interrupt Period.

The W:CLOCK directive is often issued to delay processing of a Program until a DAC Channel has settled completely. In Example 4.7, the settling problem is remedied by replacing each R:AD command with two statements:

```
W:CLOCK     0.032
R:AD        4;0.16
```

Note, however, the Program requires more time to execute fully. No delay is necessary when a DAC Channel is stepped by a small amount, 0.5% of full-scale or less. The output of each DAC Channel attains 99% of the programmed increment during the Interrupt Period that the Channel is stepped.

Since voltmeters 0 and 1, which are Keithley 190 DVMs, require 0.5 sec to complete a reading, the W:CLOCK command often precedes a R:VM statement in a Program. Sometimes a signal is observed with a sufficient signal-to-noise ratio in a time that is short compared to the time scale of the experiment. In these cases, W:CLOCK directives separate commands that read data so that the computer is not glutted with data.

The W:SWITCH statement delays processing of a Program until one of several switches is depressed. Two switches are assigned to each of four experimental stations. (Each of these is monitored with a Foreground Program.) These pairs of switches share a common housing, and each switch is clearly numbered.

```
W:SWITCH     switch number
```

When a W:SWITCH command is processing and the button indicated by the operand is depressed, the next statement in the Program is executed. Otherwise, execution is suspended temporarily with the W:SWITCH instruction.

To prevent several iterations of a Program from processing during

Example 4.8: Employing Switches to Delay Data Acquisition

```
LOOP          0;500
W:SWITCH      0
S:FILE        40
R:AD          3;1.0
S:FILE        41
R:AD          0;1.0
W:SWITCH      1
CONT          0
```

one depression of a switch, two switches are employed in the Program. A LEED I-V spectrum containing 500 points is acquired with the Program in Example 4.8. The intensity is sampled by ADC Channel 3, and ADC Channel 0 reads the gun voltage. These quantities are averaged for 1.0 sec. at each point and are stored in Files 40 and 41, respectively. During the experiment, the gun voltage and Faraday cup position are set manually. Until these conditions are adjusted for an observation, switches 0 and 1 delay data acquisition.

The experimenter manipulates the switches as follows. When the beam voltage and Faraday cup positions are set, switch 0 is depressed to sample the beam voltage and beam intensity. Once switch 0 is released, switch 1 is depressed. The experimental conditions are again adjusted, and the process is repeated. If the second W:SWITCH statement were not present, and the user depressed switch 0 for more than two seconds, the same data point would be sampled and stored twice.

4.6 Summary

Commands used exclusively in Foreground Programs have been described in this chapter. These commands read data, monitor the storage of data, control experimental conditions, and allow programmed delays. These statements, except programmed delays, must be preceded by a File-referencing statement. Like Files, all devices that control experiments or read data, except the system clock, are protected by a

Key system to prevent interference among Foreground Programs. The experimenter should take special care in ordering statements in a Foreground program so that the Label and Slots of a File are loaded correctly.

In addition, the order of commands determines how a Foreground Program executes. In particular, the order dictates which Peripherals are read or controlled in each Interrupt Period. When the following conditions are violated during a Period, processing of the transgressing Foreground Program is terminated for that Period.

(1) A single Peripheral that reads data may be sampled only once. (ADC Channels are separate Peripherals.)

(2) No more than two separate ADC Channels may be read.

(3) The output of each DAC Channel may be modified only once.

(4) No more than two separate DAC Channels may be set.

(5) A R:PC or R:AD command does not execute in the same Period that a S:DA command is issued.

Execution is also suspended with programmed delays.

(6) Until the indicated switch is depressed, a W:SWITCH command causes processing to cease.

(7) A W:CLOCK statement delays execution for an interval of time given by the single operand.

The formats of the Foreground or experimental commands are summarized in Table 4.4. Data accumulated with these statements are of limited value unless the data are accessible through analysis and display. These subjects are discussed in the next two chapters.

Table 4.4: Foreground Commands

Mnemonic	First Operand	Range	Second Operand	Range	Legal Environments[1]	Errors Flagged
Commands that Read Data						
R:CL					F	ORDER
R:VM	device number	0,1	application mode	0-6	F	ORDER,ALLOC
R:PC	device number	0,1	time	0.001-10.000[2]	F	ORDER,ALLOC
R:AD	Channel number	0-7	time	0.001-10.000[2]	F	ORDER,ALLOC
Commands that Direct the Storage of Data						
S:STORE			Slot number	0-2000	F	ORDER
I:STORE			increment	-1,0,1	F	ORDER
Commands that Control Experiments						
S:DA	Channel number	0-7	initial value	0-2000.00[3]	F	ORDER,ALLOC
I:DA	Channel number	0-7	increment	20.00, 0[3]	F	ORDER,ALLOC
Programmed Delays						
W:CLOCK			time	0.001-10.000[2]	F	
W:SWITCH			switch number	0-7	F	

Notes:
1. Programming environments are: I = Immediate Instruction, B = Background Program, and F = Foreground Programs.
2. Times are entered in seconds.
3. These operands are entered in units of the controlled parameter. See Table 4.3. Additional constraints are placed on the range of each Channel but flag no errors.

5. Displaying Data

5.1 The Output Stack

Three devices display data – a display scope, a point plotter, and the LA36 Decwriter. The scope only plots, but does not print out data. Since the simultaneous display of many records is often necessary, an output stack is available to manage these operations.

Up to eight pairs of Files concurrently reside on the stack. Whenever a File on the output stack is erased from memory, the entire stack is nulled. One File of a pair furnishes the y-axis, while the other supplies the x-axis. Alternatively, information in the Label of the ordinate File, the initial setting S and the increment I of a DAC Channel, generates the abscissa. Like most stacks, the output stack operates through a "last-in, first-out" hierarchy. That is, commands that manipulate the stack operate on the last or latter entries.

The PUSH command places an x-y pair of Files on the output stack. The ordinate File is numbered by the last File-referencing statement, while the abscissa File is numbered by the second operand of the PUSH instruction. This operand assumes values from 0 through 110. If the second operand is zero, the DAC information in the Label of the ordinate File generates the abscissa. When no DAC information is provided, points are graphed at equal spacings of the abscissa, but the abscissa will not be printed out. The DAC information creates the abscissa with the algorithm:

$$x(m) = S + m \cdot I$$

where m refers to the number of a Slot. These are successively enumerated m = 0,1,2....

The first operand of the PUSH statement is the display mode, which ranges from 0 through 5. As dictated in Table 5.1, this operand determines how a pair of Files will be plotted.

PUSH mode;abscissa File

Both the point-plotter and display scope are divided into four quadrants having the conventional designations. The grid of the display scope clearly delineates the quadrants. Although a pair is not graphed if the display mode is zero, an entry may always be printed on the Decwriter.

Files that were last entered on the output stack are removed with the POP command. The operand gives the number of Files that are eliminated.

POP number of Files

This operand can range from 0 through 8. Both 0 and 1 remove the last entry. If the operand exceeds the number of entries on the stack, the stack is cleared, but no error is flagged.

Both the PUSH and POP commands are issued in Background Programs or as Immediate Instructions. Otherwise, a usage error is generated. An order error is produced when the PUSH command is not preceded by a File-referencing statement. If more than eight File pairs might be pushed on the stack, an end-of-file error is flagged in execution. The contents of the stack do not change.

Table 5.1: Plotting Modes

Mode	Plotting Position
0	not plotted
1	first quadrant
2	second quadrant
3	third quadrant
4	fourth quadrant
5	entire display surface

5.2 Plotting Data

The display scope continually graphs stack entries with a non-zero display mode. Every 5 sec the trace is refreshed. After the output stack is modified by any command, the trace is cleared. Then a new graph is plotted employing the new parameters on the stack.

Four commands manipulate the axes of plotted records so that they may be inspected more easily. Two statements expand the x- and y-axes; the other two spatially translate the x- and y-axes to establish the zero point. Although data presented in graphical form appear to be modified by the manipulation of axes, the contents of a File are never changed with these commands.

Prior to manipulation of the axes, each delimits a range prescribed by the experimental option. The option gives information about what variable was observed and which peripheral read the data. When DAC information generates the x-axis, the unmodified full-scale range of the abscissa is the full extent of the associated experimental parameter. Table 5.3 shows the full-scale ranges of the DAC Channels, while Table 5.2 correlates the option number with the limits of an axis. The ordinate or abscissa may be generated by a File with any option.

When a File is plotted in a quadrant, the full-scale is contained in the half of the axis that lies in that quadrant. Otherwise, the entire vertical or horizontal range corresponds to full-scale.

The 12-bit range of an ADC Channel is always graphed over the full extent of an axis. If the ADC Channel samples data as an

Table 5.2: Correlation between Experimental Options and Display Parameters

Experimental Option	Associated Device(s)	Input Parameter	Input Range[1]	Plotted Range	Units
0	A/D channel 0	LEED: Gun Voltage	0-1500.	0-1500.	EV
1	A/D channel 1	unassigned	—	—	—
2	A/D channel 2	Thermocouple: EMF	0-10.00	0-10.00	MV.
3	A/D channel 3	Picoammeter: Intensity	full scale[2]	full scale[2]	% F.S.
4	A/D channel 4	Mass Spect.: Intensity	full scale[2]	full scale[2]	% F.S.
5	A/D channel 5	Lock-in Amp.: Intensity	full scale[2]	full scale[2]	% F.S.
6	A/D channel 6	Lock-in Amp.: Intensity	full scale[2]	full scale[2]	% F.S.
7	A/D channel 7	unassigned	—	—	—
8	Pulse Counters	Number of Counts	0-4 x 10^9	0-1 x 10^8	COUNTS
9	System Clock	Absolute Time	0-4 x 10^6	0-1 x 10^3	SEC.
10	———	Temperature	0-2000.	0-2000.	DEG. K
11	DVMs (Mode 0)	CMA: Pass Energy/LEED: Gun Voltage	±19999.9	0-2000.	EV
12	DVMs (Mode 1)	IET: Junction Potential	±1.99999	0-1000.0	MV.
13	DVMs (Mode 2)	CPD: Gun Voltage	±199.999	0-1.000	EV
14	DVMs (Mode 3)	Thermocouple: EMF	±1999.99	0-20.00	MV.
15	DVMs (Mode 4)	Picoammeter: Intensity	full scale[2]	full scale[2]	% F.S.
16	DVMs (Mode 5)	Mass Spect.: Intensity	full scale[2]	full scale[2]	% F.S.
17	DVMs (Mode 6)	Lock-in Amp.: Intensity	full scale[2]	full scale[2]	% F.S.

Notes:
1. Data are printed out over the entire input range.
2. "Full scale" indicates that the range selection switch on a device is employed. Although the device may be read over the full range at any setting, no record of the setting is maintained by the POL monitor. This setting is entered in a lab notebook.

Table 5.3: Correlation between DAC Channels and Display Parameters

D/A Channel	Controlled Parameter	Display Range[1]	Units
0	CMA: Pass Energy	0-2000.00	EV
1	LEED: Gun Voltage	0-1500.00	EV
2	Mass Spect.: Mass Select	0-300.00	AMU
3	IET: Junction Potential	0-1000.00	MV
4	unassigned	—	—
5	unassigned	—	—
6	unused	—	—
7	unused	—	—

Note: 1. Data are printed out and plotted over this range.

accumulator rather than an averager, a data analysis command, EXPAND, can normalize the data before they are graphed. (See § 6.3.)

Often data are observed over a narrow portion of the full-scale range. Hence, commands are available to expand and set the zero points of plots. By manipulating axes intelligently, the user can calibrate the grid marks of graphs easily. He may check these calibrations with commands that print out data points on the Decwriter.

The zero point, or lower-left corner, of a graph is positioned with the X:MOVE and Y:MOVE instructions. The operand of each command is the per cent of the full-scale range by which the axis is shifted. X:MOVE modifies the x-axis; Y:MOVE, the y-axis. Grids on the display scope aid the programmer in using this command.

```
X:MOVE   % of full-scale
Y:MOVE   % of full-scale
```

The operand of each statement may vary from -100.0 to +100.0%. If the sign is minus (plus), the plot is shifted to the left (right). A plot reverts to its original zero point when the following commands are issued:

```
X:MOVE   0
Y:MOVE   0
```

After the zero point is set, the axes of a plot are expanded with the X:EXPAND and Y:EXPAND directives. With these commands, a plot is expanded to cover a large fraction of the graphics surface. Each statement has a single operand which is the expansion factor.

```
X:EXPAND   expansion factor
Y:EXPAND   expansion factor
```

This operand is an integer in the range -25,000 through 25,000, excluding zero. A negative operand causes a record to be plotted up-side down. This feature is useful if data from a DVM are sampled with the inputs reversed. When the following two commands are issued, a plot returns to its original dimensions.

```
X:EXPAND   1
Y:EXPAND   1
```

Effects of commands that manipulate axes are not cumulative. The position or size of a plot is delimited by the operand of the last move or expand command. Before axes are expanded, the zero point of the graph should be set. Otherwise, the plot may "jump off" the graphics surface.

An Auger spectrum observed over the range of 230.0 to 300.0 eV is plotted on the scope. File 20 stores the number of counts, which is a maximum near 10^6. DAC information contained in the label generates the abscissa. The sequence of Immediate Instructions in Example 5.1 plots File 20 over an optimum area of the scope.

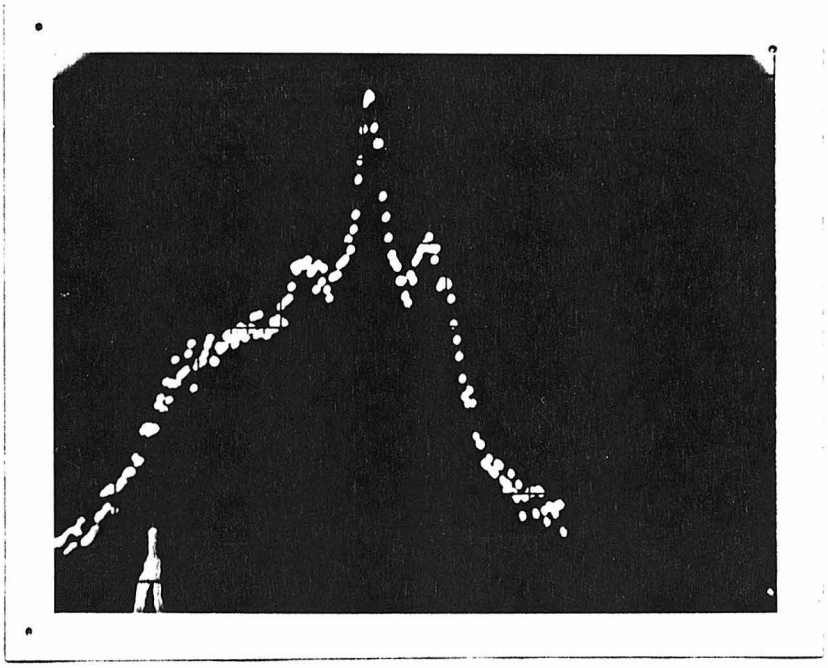
Because the minimum number of counts is about an order of magnitude less than the maximum, no Y:MOVE command was necessary. Figure 5.1 shows the scope display before and after the command sequence is issued. All commands that manipulate axes refer only to the File pair that was last entered on the stack.

Each entry on the output stack requires six consecutive words of memory. In the first work, the lower order byte numbers the ordinate File, while the higher order byte numbers the abscissa File. The next

Example 5.1: Manipulating the Display on the Oscilloscope

```
POP         8
S:FILE      20
PUSH        5;0
X:MOVE      -11.5
X:EXPAND    20
Y:EXPAND    40
```

Figure 5.1: An Oscilloscope Display Before and After Manipulation



four hold the y-expansion factor, y-zero point, x-expansion factor, and x-zero point, respectively. The display mode is stored in the last word. When a PUSH instruction is issued, zero points are nulled, and expansion factors are set to one.

All commands that modify axes are used only as Immediate Instructions. If these commands are not prefaced by a File-referenceing statement, an order error is flagged. As usual, this statement need not be contiguous to the command that manipulates the axis. These commands may be issued when the output stack is empty. Although no error is flagged, these commands are ignored by the POL monitor.

A cursor is available to mark a particular point in a File or pair of Files. On the scope, the cursor appears as a vertical line through the indicated point. Only points of the File pair that was last entered are marked. For their successful execution, some display and analysis commands require that the cursor delimits a point. (See § 5.3 and Chapter 6.)

The S:CURSOR instruction selects a point or Slot. The number of the Slot is the single operand.

S:CURSOR Slot number

As previously noted, Slots are numbered 0,1,2,... but do not exceed 2000. To remove the cursor from the display scope, the user issues a S:CURSOR instruction which references a Channel which exceeds the range of the File. The S:CURSOR command is entered only as an Immediate Instruction or in a Background Program. This command must be preceded by a File-referencing statement.

The oscilloscope continually displays the contents of the output stack. To obtain a hard copy of the trace on the point-plotter, a PLOT statement is used. It must be issued as an Immediate Instruction and prefaced by a File-referencing statement.

The scope trace is imaged on the entire area (25 x 18 cm) of a piece of graph paper. The top 25% of the scope display is lost. Both axes of the plotter are set to the 0.4 V/cm scales. When the inputs to the plotter are zeroed, the pen should be 12.5 cm up and 12.5 cm over from the lower left-hand corner of the graph.

5.3 Writing Data

The PRINT command writes out 1 to 20 data points on the Decwriter. A single operand gives the number of Slots. The first point is numbered by the cursor, and consecutive points with higher Slot numbers provide the remainder. Both operands 0 and 1 cause a single point to be printed. Remember, the cursor references only the last File pair entered on the stack. If the abscissa File is zero, the DAC information in the label of the ordinate file generates the abscissa. As indicated in Tables 5.2 and 5.3, points are printed in units that are convenient for each option or DAC Channel.

Suppose two Slots of the Auger spectrum of Example 5.1 are printed. The command and response appear on the Decwriter as

```
PRINT     2
            EV          COUNTS
        200.00          48.097
        201.00          47.421
```

If the PRINT command references channels past the end of a File, only channels within the range of both Files in a pair are listed. No error is flagged.

Both the PRINT and WRITE commands are issued as Immediate Instructions or in Background Programs. These commands must be preceded by a File-referencing statement.

The WRITE command lists entire Files in the same units as the PRINT statement. If a WRITE statement is issued with an operand of zero, the last File pair entered on the output stack is printed out in five pairs of columns. The first column in the pair gives the abscissa, while the second lists the ordinate. When a record is printed out in this format, the abscissa is listed so that it increases downward then to the next column to the right.

Labels listed above the first pair of columns denote the units of the coordinates. As noted in Table 5.2, the units are determined by the experimental options of the File pair. When a DAC Channel generates the abscissa, it is labeled as noted in Table 5.3.

The WRITE command may have an operand in the range of 1 through 8, so that data in different Files on the OUTPUT stack may be compared. All Files are printed in a single column. The abscissa of the File last entered on the stack creates the first column. Successive columns are generated from ordinate Files, with the last entry being

listed first.

All ordinate Files should have identical abscissae and should share the same experimental option. The number of Files given by the operand of the WRITE statement delimits the number of ordinate columns. If a user attempts to write out more File pairs than are on the stack, no error results. All entries are listed.

WRITE number of Files

Only the first two columns, the abscissa and the first ordinate Files, are labeled with the units in Tables 5.1 and 5.2. If one ordinate File has fewer channels than the first File, the overflow of spaces is blanked. However, no more channels are printed than the number in the first ordinate File.

As described in Chapter 3, the "Panic" button aborts the PLOT, WRITE, and PRINT operations. This switch is depressed until the operation is visibly terminated. The "Panic" button is clearly labeled on the upper-left section of the computer rack.

5.4 Summary

The subject of this chapter has been display operations. The output stack allows several pairs of Files to be displayed concurrently on the scope, point plotter, or Decwriter. Up to eight pairs of Files may reside on the stack simultaneously. Table 5.4 summarizes commands that manipulate the output stack and display data.

The PUSH and POP statements enter and remove File pairs on the stack. Transfers occur with a "last-in, first-out" hierarchy. That is, all commands that modify the stack refer to the pair that was entered last. Four of these commands move and expand the x- and y-axes of graphs. Another moves the cursor which marks a Slot for some display and analysis routines.

A display scope continually graphs the contents of the output stack. Although no command is necessary to initiate output on the scope, the PLOT instruction makes a hard copy of the scope trace on the point plotter. The PRINT directive writes out a few points of a File pair on the Decwriter. When the WRITE instruction is issued, the contents of one or more File pairs are listed. All written output is in units that are convenient for the user.

Table 5.4: Commands Involving the Display of Data

Nnemonic	First Operand	Range	Second Operand	Range	Legal Environments[1]	Errors Flagged
Commands that Manipulate the Output Stack						
PUSH	display mode	0-5	abscissa File	0-110	I,B	ORDER,EOF
POP			number of entries	0-8	I,B	
Y:EXPAND			expansion factor	±25000,≠0	I	ORDER
X:EXPAND			expansion factor	±25000,≠0	I	ORDER
Y:MOVE			% of full scale	±100.0	I	ORDER
X:MOVE			% of full scale	±100.0	I	ORDER
S:CURSOR			Slot number	0-2000	I,B	ORDER
Commands that Display Data						
PLOT					I	ORDER
PRINT			number of points	0-20	I,B	ORDER
WRITE			number of Files	0-8	I,B	ORDER

Notes: 1. Programming environments are: I = Immediate Instruction, B = Background Program, and F = Foreground Programs.

6. Manipulating Data

6.1 Calculator Commands

Common calculator functions are implemented under POL. Numbers are manipulated in the reverse Polish notation that is used by many calculators. As their names imply, the LOAD, ADD, SUB, MUL, and DIV instructions are employed to Load, add, subtract, multiply, and divide numbers. The result of any calculation is printed on the Decwriter. All calculator commands are used only as Immediate Instructions.

The two operands of each of the five calculator commands are identical. Both describe a single number in scientific notation.

```
LOAD    exponent;mantissa
```

The first operand, the exponent, varies from 0 through 7 while the second, the mantissa, ranges between -10.000 and 10.000. Only three digits to the right of the decimal point are recognized in the mantissa. Although the operands of the calculator commands reverse the conventional ordering of the mantissa and exponent, the printed answers do not! There, the mantissa precedes the exponent, and "E" separates them.

To perform a calculation, one number is loaded into the POL system, and then it is operated on by another. The second number is the addend, subtrahend, multiplier, or divisor. For example, $1.000 \cdot 10^4$ is divided by $5.000 \cdot 10^3$. The following sequence appears on the Decwriter:

```
LOAD   4;1.000
DIV    3;5.000
  2.000  E   0
```

If the divisor were zero, the quotient is assigned value zero.

A load operation is performed on the result of each manipulation. To calculate $[(1.000 \cdot 10^4 / 5.000 \cdot 10^3) - 1.500]$, the previous sequence is succeeded by

```
SUB    0;1.500
  5.000   E-1
```

Negative exponents are displayed but are never entered as an operand.

6.2 Spectrum Manipulations

Commands discussed in the remainder of this chapter perform data analysis operations on Files. The particular File is numbered by the operand of the last File-referencing statement. If no such statement prefaces a File-oriented manipulation, an order error is flagged. The indicated File must be loaded with data; else a manipulation error is generated. The same error is produced when a command may not operate on Files with certain experimental options. If a user attempts to manipulate Slots while a Foreground Program is loading these with data, a manipulation error is also flagged. Unlike order errors, all manipulation errors are flagged during execution. Because the contents of a File are usually permanently modified by commands that perform data analysis, a File might be duplicated in core or on disc prior to its

manipulation. Data analysis commands are entered as Immediate Instructions or in Background Programs.

This section describes statements that operate on a series of adjacent channels to modify a single channel. No limitation is placed upon the experimental option of the File that is manipulated.

The PEAK instruction integrates the area under a peak in a spectrum. Data points comprising the peak are summed and averaged. On the Decwriter, the result is printed in units that are accessible to the user as given in Table 5.2. The File that contains the spectrum is not changed.

The single operand of the PEAK command, which varies from 1 through 2000, is the number of Slots that comprise peak.

PEAK number of Slots

The first point of a peak is numbered by the cursor, while the others are provided by consecutive channels with lower numbers (to the left in a display). Because the high-energy side of a spectral feature (to the right in a display) is more sharply defined, a right to left labeling is associated with the PEAK instruction. If any channel referenced by the instruction is outside the range of the File, no error is flagged, but no integration is performed.

Like all data analysis commands, this statement operates on the File numbered by the last File-referencing statement, not the ordinate File that was last entered on the output stack. However, the user may manipulate the File and display it on the scope simultaneously. A common starting point and number of Slots for integration should be chosen for comparing peak areas among spectra.

The SMOOTH command filters Gaussian, random, or white noise from a data File.(1) Noise from a pulse counter is not random but varies as $N^{1/2}$, where N is the number of counts in a channel.(1) Often the magnitude of the base line is comparable to the peak heights in a spectrum of pulse counts. Then the noise appears to be random, and the smooth routine may be employed successfully.

Smoothing involves the fitting of five adjacent data Slots with a cubic polynomial. The result is placed in the center channel. Savitsky and Golay have shown that this fit is equivalent to a convolution of five channels with constant weighting factors.(2) The single operand of the SMOOTH directive fixes whether 5, 9, 13 or 17 adjacent channels will be convolved in the smoothing procedure. The operands for the respective cases are 0, 1, 2 and 3.

SMOOTH iterations

Commands that differentiate and integrate entire Files, DIFF and INTE, allow the background to be subtracted from spectra. After n differentiations, followed by the same number of integrations, the

first n components in the Taylor series expansion of the background are removed.[3] Both the DIFF and INTE commands have no operands.

Differentiation occurs by subtracting adjacent channels and storing the result in the first of the two channels. To fill the last Slot of a File, the result of the last subtraction is duplicated. Because the high-energy end of a kinetic energy spectrum usually varies more slowly than the low-energy side, this algorithm is preferred over duplicating the first channel. Integration is performed by adding adjacent channels of a File. The cursor marks the zero point of the integration.

Suppose that the background of an Auger spectrum in File 20 is to be removed from the high-energy side of a peak whose area is to be determined. With the aid of the scope display, the cursor is set to Slot 43. First-order background subtraction and a 30 point integration of the peak area are performed as illustrated in Example 6.1. After a sequence of Immediate Instructions sets the position of the cursor, a Background Program calculates the peak area. The average number of counts in the feature is 126,964. Figure 6.1 shows the data before and after the background is subtracted.

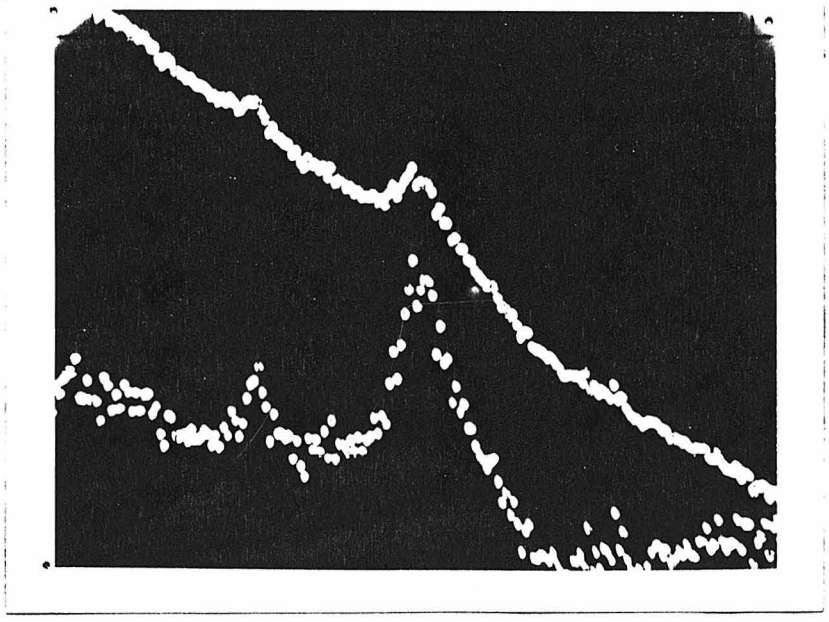
6.3 Channel Manipulations

Commands described in this section operate on single channels of a spectrum. The result of each manipulation is placed in the original data Slot.

Example 6.1: Determining the Area under a Peak in an Auger Spectrum

POP	8
S:FILE	20
PUSH	5;0
X:MOVE	-22.5
X:EXPAND	20
Y:EXPAND	25
S:CURSOR	43
P:ASSIGN	0;10
START	0
S:FILE	20
DIFF	
INTE	
PEAK	30
END	0
EXECUTE	0
A=	126.964
STOP	0

igure 6.1: Auger Spectrum with the Background Removed



EMFs of Pt;Pt - 10% Rh thermocouples are

transformed into degrees Kelvin with the VTOT command, which has no operands. Conversion is accomplished through a 10-piece linear fit of the known EMF versus temperature relation. Results in the range of 170^{o}K to 1800^{o}K are accurate to within 1^{o}K of the true temperature.

To prevent unnecessary destruction of data, the VTOT directive only operates on Files with experimental options of 2 and 14. That is, the File must contain thermocouple data. After all channels in a File are transposed, the experimental option in the Label is changed to 10. Consequently, the user cannot destroy a File by attempting two conversions. The File will then be displayed in the units of temperature, ^{o}K.

The OFFSET statement adds or subtracts a constant to each data point of a File. The operands are identical to those of the calculator commands.

OFFSET exponent;mantissa

The exponent ranges from 0 through 7 while the mantissa varies from -10.000 to 10.000, but cannot be zero. Units of the operands should be the same as those used for displaying data of the File as given in Table 5.2.

When applied in succession, the PRINT and OFFSET commands re-establish the zero of a File containing time. Remember, the system clock is never zeroed when a Foreground Program executes. Times are

Example 6.2: Re-Establishing the Zero Point of a File Containing Time

```
S:FILE          86
S:CURSOR        0
PRINT
  1843.680
OFFSET          3;1.843
PRINT
  0.680
OFFSET          0;0.680
PRINT
  0.000
```

stored in chronological order in File 86. Example 6.2 illustrates how the first Slot of a File is zeroed. Succeeding channels are then referenced to this point. Since the accuracy of the mantissa is limited to three digits to the right of the decimal point, two OFFSET instructions are needed to zero the first Slot.

The EXPAND instruction normalizes Files for comparison. Each channel is multiplied by the operand, an expansion factor which ranges from -25.000 to 25.000 but cannot be zero.

EXPAND factor

This command operates on Files which contain spectral intensities. That is, the experimental option must range from 3 through 8 or 15 through 17. In general, Files should be contracted rather than expanded when they are normalized. Expanding the data tends to accentuate noise and falsifies the true signal intensity.

For lack of a better classification, the DAOFF and DAEXP instructions are called channel manipulations. Often DAC Channels have an offset error; the zero point drifts slowly with time. To avoid recalibrating a DAC Channel every few days, a DVM samples an experimental parameter associated with the initial setting of the Channel. With this reading, the setting stored in the File Label is modified through the DAOFF command. In displays, the numbers generated from this information will coincide closely with the true values of the experimental parameters. Increments of DAC Channels can be modified with the DAEXP command.

The DAOFF and DAEXP commands have two operands. The first is

the number of a DAC Channel; the second is the true initial setting or increment of the parameter associated with the Channel. Units of this operand are those noted in Table 5.3.

```
DAOFF   Channel number; initial value
DAEXP   Channel number; increment
```

If the Channel number in the Label differs from the first operand, a manipulation error is flagged.

Suppose the first abscissa of an Auger spectrum in File 20 was set by DAC Channel 0 to 100.00 eV, but the measured value was 100.76 eV. To place the true initial value of the DAC Channel into the Label of File 20, the following Immediate Instructions are issued:

```
S:FILE  20
DAOFF   0;100.76
```

The DAOFF and DAEXP instructions correctly set the initial value or increment to within 1 part in 40,960, the ultimate resolution of the experimental parameter controlled by the D/A unit.

6.4 Channel-to-Channel Manipulations

Two files are required for channel-to-channel manipulations. The first is numbered by a File-referencing statement, and the second, by an operand of the manipulative command. As usual, the second varies from 1 through 110. These commands perform operations between parallel channels of the two Files. Parallel channels are identified

by the same Slot number. Although the first File is not modified, the second File stores the result of each operation.

When the experimental options of the two Files do not coincide, a manipulation error is flagged in execution. However, the second File may be empty, i.e., contain no data. Except for the number of points and the storage pointer, the Label of the first File is then copied into the second. If the lengths of the Files are not identical, no error is flagged. Instead, only the first m channels of the Files are operated on where m is the number of Slots in the shorter File. The abscissae of parallel channels should always coincide.

The C:COPY command copies a File numbered by the last File-referencing statement into the File numbered by the single operand. Parallel channels are duplicated. If File 20 is to be reproduced in File 21, the following Immediate Instructions are issued:

```
S:FILE   20
C:COPY   21
```

The C:ADD (C:SUB) instruction adds (subtracts) parallel channels of two Files. The last File-reference provides the addehend (subtrahend), and the result is stored in the File noted by the single operand. Both the C:ADD and C:SUB statements operate only on Files containing intensity data. That is, the experimental option may range from 3 through 8 or 15 through 17.

To enhance the signal-to-noise ratio of a feature, several Auger spectra with identical abscissae are summed in an empty File, File 30. Files 20 through 29 contain the spectra. The Background Program in Example 6.3 performs this operation.

Example 6.3: Summing Auger Spectra to Enhance the Signal-to-Noise Ratio

```
S:FILE    20
LOOP      0;10
C:ADD     30
I:FILE    1
CONT      0
```

Example 6.4: Forming a Difference Spectrum in UPS

```
S:FILE    44
PEAK      100
A = 394.277
S:FILE    91
PEAK      100
A = 576.342

LOAD      2;3.943
DIV       2;5.763
  6.849   E-1

C:COPY    102
S:FILE    102
EXPAND    0.685
S:FILE    44
C:SUB     102
```

The C:SUB statement forms difference spectra from UPS data. Intensity data in File 91 are subtracted from File 44. Each File holds 100 points. Because several experimental controls were not duplicated for each spectrum, these are normalized to contain the same number of counts. To preserve Files 44 and 91, File 102 is used as a "scratch pad" and stores the difference spectrum. All manipulations are performed with the sequence of Immediate Instructions listed in Example 6.4.

Here, the PEAK command averages the intensities of entire spectra. Along with the EXPAND instruction, these averages are employed in normalization. The user chooses to shrink the File with the larger number of counts so that the noise in the difference spectrum is not accentuated. (Note that this technique may not be the optimum method for taking difference spectra in UPS but shows how several data analysis commands are used.)

When many spectra with identical abscissae are acquired with one Foreground Program, the first point of each spectrum may be stored in one File. Then, second points are placed in another File and so on. This technique is useful when mass peaks, rather than points, are monitored in thermal desorption experiments. If the intensities at 20 mass points are sampled at 100 temperatures, 20 Files store the intensities, and one holds the temperatures. All 21 Files are 100 points in length. With the integration and differentiation of § 6.2, and the C:ADD and C:SUB commands, background subtraction is performed on each peak. Peak areas are determined by summing the intensity Files with the C:ADD statement. To show the evolution of peak area

with temperature, the File containing the sum is plotted against the temperature File.

The C:DIV statement divides the data points of the last File-reference into parallel channels of the File noted by the single operand. The latter File stores the dividend. Since the magnitude of the result is severely diminished, all dividends are multiplied by 4096 prior to their storage. This command only operates on Files that store spectral intensities.

With the C:DIV instruction, LEED I-V curves are corrected for variations of the gun current with the gun voltage. Beam intensities are stored in File 50, while the gun current is contained in File 51. Parallel channels correspond to the same gun voltages. The contents of File 50 are corrected with the Immediate Instructions:

```
S:FILE   51
C:DIV    50
```

The "black sheep" of channel-to-channel manipulations is the DECON command. Using point successive over-relaxation,[4] it deconvolutes the Al-K_α line from x-ray spectra. (The relaxation parameter is approximately 0.1.) Only the peaks resulting from the two p-orbital spin states and no satellites comprise the aluminum excitation line. The DECON command operates only on Files containing pulse counts. Points must be sampled at intervals of 0.1 eV.

The measured (raw) spectrum is duplicated prior to execution of this routine. One copy serves as a reference spectrum and is given by the second operand of the DECON instruction. When it executes, this File is not modified. Another File, marked by the last File-reference,

stores the result of the deconvolution. The first operand, which ranges from 1 to 20, delimits the iterations of the point successive over-relaxation routine.

DECON number of iterations; File number

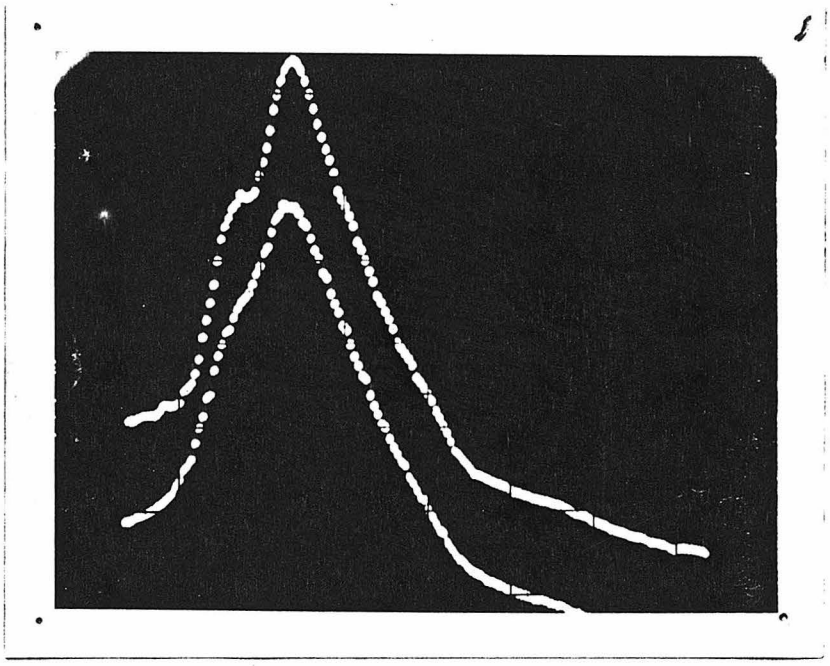
Because this algorithm performs a partial deconvolution, it is very stable. A large number of iterations may clearly resolve a feature but might introduce oscillations of a large magnitude into the spectrum. These artifacts should not be confused with spectral features. Since data are sampled at intervals of 0.1 eV, the Nyquist Sampling Theorem[1] dictates that X-ray features are ultimately resolved to a FWHM of 0.2 eV.

The Al-K_α line is filtered from an x-ray spectrum in File 75. Before two iterations of point successive over-relaxation are performed, the spectrum is duplicated in File 76.

```
S:File    75
C:COPY    76
DECON     2;76
SMOOTH    1
```

If further iterations are required, another DECON instruction may simply succeed the first so long as the File-reference is not changed. Figure 6.2 shows the Al-K_α line before and after it is deconvolved from itself. As one would expect, the deconvoluted spectrum resembles a delta function. The features on either side of the main peak are <u>not</u> true peaks but are Gibbs oscillations.

Figure 6.2: An XPS Spectrum Before and After the Al-K_2 Line is Removed.



6.5 Summary

Before Foreground Programs are written, the user must carefully scrutinize the sequence of accumulation, analysis, and display of data. This chapter has discussed the second step, which is occasionally absent from the sequence. POL implements several common analysis routines. When incorporated into Programs, these routines can perform many powerful manipulations. Complex or specialized routines are effected by RT-11 programs. To this end, Appendix C describes File structure in great detail. All analysis commands in POL can be entered as Immediate Instructions. Hence, Files may be displayed while they are manipulated.

As shown in Table 6.1, four types of analysis commands are supported by POL. Calculator commands employ reverse Polish notation. These are the only analysis instructions which cannot be entered in Background Programs and do not operate on Files. Spectrum manipulations modify channels of a spectrum by operating on several adjacent channels of a single File. The PEAK statement is atypical in that it does not change the contents of a File. Channel manipulations are directed to individual Slots of a single spectrum. These commands allow spectra to be normalized and the zero point to be re-established. In contrast, instructions that modify display axes do not alter the channels of a File. Only the visual trace is varied. Channel-to-channel manipulations operate between parallel Slots of two Files. The most powerful of these statements, C:COPY, duplicates Files in memory.

Table 6.1: Data Analysis Commands

Mnemonic	Operand 1	Range 1	Operand 2	Range 2	Legal Environments[1]	Errors Flagged
Calculator Commands						
LOAD	exponent	0-7	mantissa	±10.000	I	
ADD	exponent	0-7	mantissa	±10.000	I	
SUB	exponent	0-7	mantissa	±10.000	I	
MUL	exponent	0-7	mantissa	±10.000	I	
DIV	exponent	0-7	mantissa	±10.000	I	
Spectrum Manipulations						
PEAK			number of Slots	1-2000	I,B	ORDER,MANIP
SMOOTH			iterations	0-3	I,B	ORDER,MANIP
DIFF					I,B	ORDER,MANIP
INTE					I,B	ORDER,MANIP
Channel Manipulations						
VTOT					I,B	ORDER,MANIP
OFFSET	exponent	0-7	mantissa[2]	±10.000,≠0	I,B	ORDER,MANIP
EXPAND			factor	±25.000,≠0	I,B	ORDER,MANIP
DAOFF	Channel number	0-7	initial value[3]	0-2000.000	I,B	ORDER,MANIP
DAEXP	Channel number	0-7	increment[3]	±200.000	I,B	ORDER,MANIP
Channel-to-Channel Manipulations						
C:COPY			File number	1-110	I,B	ORDER,ALLOC,MANIP
C:ADD			File number	1-110	I,B	ORDER,ALLOC,MANIP
C:SUB			File number	1-110	I,B	ORDER,ALLOC,MANIP
C:DIV			File number	1-110	I,B	ORDER,ALLOC,MANIP
DECON	iterations	1 to 20	File number	1-110	I,B	ORDER,ALLOC,MANIP

Notes: 1. Programming environments are: I = Immediate Instruction, B = Background Program, and F = Foreground Program.
2. Units depend on the experimental option. See Table 5.2.
3. Units depend on the Channel number. See Table 5.3. The range is further restricted for a particular Channel, but no errors are flagged.

References

1. See any elementary book on statistics such as:

   R. B. Blackman and J. W. Tukey, The Measurement of Power Spectra, Dover, New York, 1958.

2. A. Savitzky and M. J. E. Golay, Anal. Chem. 36, 1627 (1964).

3. For a detailed discussion of the mathematics see:

   J. E. Houston, Rev. Sci. Instrum. 45, 897 (1974).

4. P. A. Jansson, J. Opt. Soc. Am. 60, 184 (1970).

7. Summary of the POL Operating System

Peripheral Oriented Language, POL, employs the powerful features of a PDP-11 computer to monitor experiments. Among these features are: (1) the definitive control of peripherals, (2) the accurate timing of events, and (3) the concurrent sampling of several sources of data. In addition to monitoring experiments, POL provides facilities to store, analyze, and display data. To meet the needs of virtually any user, data files on disc, that were generated by the POL operating system, can be manipulated by RT-11 programs.

The instructions of POL are executed in three Programming environments. When an Immediate Instruction is issued, the statement is assembled and executed instantly. Bookkeeping functions, such as the management of storage space in core and on disc, are performed by Immediate Instructions. A sequence of statements is assembled into a Foreground or Background Program. Foreground Programs monitor experiments; Background Programs analyze and display data.

Statements in POL have either none, one, or two operands. Each must be contained within a prescribed range, or an error message is generated. In Table 7.1, all legal POL statements are listed. Note that most of these statements operate on data Files. Commands may only be issued in certain Programming environments without flagging a usage error. A list of these environments and a compendium of errors flagged by each instruction are included in Table 7.1.

Conditions that generate error messages are summarized in Table 7.2. When a programmer uses this table in conjunction with Table 7.1, he quickly recovers from mistakes. Conditions marked with an

asterisk (*) are flagged during execution. With the exception of the hardware error, these errors terminate execution of a Background Program following their detection. Other errors are flagged in assembly. The POL system ignores these transgressing instructions as if they were not entered.

POL is a time-sharing system; up to four Foreground Programs may execute simultaneously. A Hardware Interrupt, which suspends the execution of Background processes is generated every 16 msec. Then the four Foreground Programs are processed in succession. When any of the circumstances in Table 7.3 is encountered in the processing of a Foreground Program, its execution is suspended until the next Interrupt Period. After all Foreground Programs have executed, Background processes are serviced. These include: (1) the handling of Immediate Instructions, 2) the assembly of Programs, and (3) the execution of Background Programs. No two of these three tasks are processed concurrently.

Individual Channels of the ADC and DAC are assigned to specified experiments. These assignments, together with those of experimental options, are listed in Table 7.4. The input and output ranges of all Peripherals are shown. Data records are printed over this range in units noted in the table. However, data are graphed over different ranges. To insure the correct display of his data, the user must set the range selection switch of a DVM as shown in Table 7.4. Since analog signals are often preconditioned before transmission to (from) an experiment from (to) the computer, the true analog limits of DAC and ADC Channels are listed to aid the precocious experimenter.

Tables 7.1 through 7.4 and this summary are in abbreviated manual for the experienced programmer. Prior to an experiment, he plans the acquisition, analysis, and accessing of data carefully. Since he understands the limitations both of software and hardware, he can employ the advantageous features of POL optimumly.

Table 7.1: POL Commands

Mnemonic	First Operand	Range	Second Operand	Range	Legal Environments[8]	Errors Flagged
File-Referencing Statements						
S:FILE			File number	1-110	I,B,F	ALLOC
I:FILE			increment	±5,≠0	B,F	ORDER,ALLOC
Commands that Manage Storage in Core						
F:OPEN	Key	0-4	first File	0-110	I	ALLOC
F:ASSIGN	number of Files	1-50	number of Slots	1-2000	I	ALLOC
F:CLEAR			number of Files	1-50	I	ALLOC
F:NULL			number of Files	1-50	I	ALLOC
P:ASSIGN	Program number[9]	0-4	number of statements	1-100	I	ALLOC
P:NULL			Program number[9]			ALLOC
Commands that Manage Storage on Disc						
F:PUT	number of Files	1-50	alphabetic name[3]	αβγ	I	ALLOC TRANS,HRDWR
F:GET	number of Files	1-50	alphabetic name[3]	αβγ	I	ALLOC TRANS,HRDWR
F:RID	number of Files	1-50	alphabetic name[3]	αβγ	I	TRANS
P:PUT	Program number[9]	0-4	alphabetic name[3]	αβγ	I	ALLOC,TRANS,HRDWR
P:GET	Program number[9]	0-4	alphabetic name[3]	αβγ	I	ALLOC,TRANS,HRDWR
P:RID	Program number[9]	0-4	alphabetic name[3]	αβγ	I	TRANS
Commands that Manage Assembly of Programs						
START			Program number[9]	0-4	I	ALLOC
END			(Program number)[4,9]	(0-4)	B,F	
IGNORE			number of statements	0-100	B,F	EOF
REENTER			Program number[9]	0-4	I	ALLOC
LIST			Program number[9]	0-4	I	

Commands that Manage Execution of Programs

EXECUTE			Program number[9]	0-4	I	ALLOC
ABORT			Program number[9]	1-4	I	
PROGRESS			Program number[9]			

Conditional Statements

LOOP	Loop number	0-3	number of iterations	1-2000	B,F	
CONT			Loop number	0-3	B,F	ORDER
W:CLOCK			time[5]	0.001-10.000	F	
W:SWITCH			switch number	0-7	F	

Commands that Read Data

R:CL					F	ORDER
R:VM	device number	0,1	application mode	0-6	F	ORDER,ALLOC
R:PC	device number	0,1	time[5]	0.001-10.000	F	ORDER,ALLOC
R:AD	Channel number	0-7	time[5]	0.001-10.000	F	ORDER,ALLOC

Commands that Direct the Storage of Data

S:STORE			Slot number	0-2000	F	ORDER
I:STORE			increment	-1,0,1	F	ORDER

Commands that Control Experiments

S:DA	Channel number	0-7	initial value[6]	0-2000.0	F	ORDER,ALLOC

Commands that Manipulate the Output Stack

PUSH	display mode	0-5	abscissa File	0-110	I,B	ORDER,ALLOC,EOF
POP			number of entries	0-8	I,B	

Y:EXPAND			expansion factor	±25000,≠0	I	ORDER
X:EXPAND			expansion factor	±25000,≠0	I	ORDER
Y:MOVE			% of full scale	±100.00	I	ORDER
X:MOVE			% of full scale	±100.00	I	ORDER
S:CURSOR			Slot number	0-2000	I,B	ORDER
Commands that Display Data						
PLOT					I	ORDER
PRINT			number of Slots	0-20	I,B	ORDER
WRITE			number of entries	0-8	I,B	ORDER
Spectrum Manipulations						
PEAK			number of Slots	0-2000	I,B	ORDER,MANIP
SMOOTH			iterations	0-3	I,B	ORDER,MANIP
DIFF					I,B	ORDER,MANIP
INTE					I,B	ORDER,MANIP
Channel Manipulations						
VTOT					I,B	ORDER,MANIP
OFFSET	exponent	0-7	mantissa[7]	±10.000,≠0	I,B	ORDER,MANIP
EXPAND			factor	±25.000,≠0	I,B	ORDER,MANIP
DAOFF	Channel number	0-7	initial value[6]	0-2000.000	I,B	ORDER,MANIP
DAEXP	Channel number	0-7	increment[6]	±200.000	I,B	ORDER,MANIP
Channel-to-Channel Manipulations						
C:COPY			File number	1-110	I,B	ORDER,ALLOC,MANIP
C:ADD			File number	1-110	I,B	ORDER,ALLOC,MANIP
C:SUB			File number	1-110	I,B	ORDER,ALLOC,MANIP
C:DIV			File number	1-110	I,B	ORDER,ALLOC,MANIP
DECON	iterations	1-10	File number	1-110	I,B	ORDER,ALLOC,MANIP

Calculator Commands

LOAD	exponent	0-7	mantissa	±10.000	I
ADD	exponent	0-7	mantissa	±10.000	I
SUB	exponent	0-7	mantissa	±10.000	I
MUL	exponent	0-7	mantissa	±10.000	I
DIV	exponent	0-7	mantissa	±10.000	I

Notes:

1. Ranges which vary over three orders of magnitude or more are approximate.
2. Digits to the right of a decimal point are not recognized unless the range is defined with a decimal point. The number of these digits that are assembled is indicated by the decimal places in the range definition.
3. Alphabetic names must consist of exactly three alphabetic characters. Characters may be repeated.
4. This operand is optional.
5. All times are given in seconds.
6. Units depend on the Channel number. Ranges may be further restricted for a particular Channel, but no errors are flagged. See Table 7.3.
7. Units depend on the experimental option. See Table 7.3.
8. Immediate Instruction Background statement, and Foreground statement are the Programming environments denoted by I,B, and F, respectively.
9. Numbers 1 through 4 reference the Foreground Programs; 0 denotes the Background Program.

Table 7.2: Conditions that Flag Error Messages

1. Instruction Error: INSTR
   (a) The name of the command is not a legal POL instruction.

2. First Operand Error: OPER1
   (a) Two operands are given when one is required for the command. One operand is supplied but two are necessary. Note that two operands must be separated by a semicolon (;). Else no semicolon is used.
   (b) The first operand is out of the legal range that is particular to the instruction.

3. Second Operand Error: OPER2
   (a) The second or single operand is out of the limits prescribed by the instruction.
   (b) An operand is associated with a command that needs none.

4. Usage Error: USAGE
   (a) A statement appears in an unacceptable Programming environment.

5. Ordering Error: ORDER
   (a) Loops are improperly nested.
   (b) A File-referencing statement, S:FILE or I:FILE, should occur prior to the command but does not. The File reference need not immediately proceed the command.
   (c) A R:CL, R:VM, R:PC, or R:AD command should precede the I:STORE command, but does not.

6. Allocation Error: ALLOC
   (a) A file number indicated by a command does not have a Key identical to the number of the Foreground Program where the statement is employed. The file is not assigned space in core.
   (b) The Peripheral device, which is numbered by an operand of the command, is being used in another Program. ADC or DAC Channels are separate devices.
   (c)* The Program number given by the instruction is already in use or is not assigned space in core.
   (d)* The Key given by the operand of disc I/O command does not match the Key of the File to be transferred.

7. End of File Error: EOF
   (a) All locations of the Program that is assembling are full.
   (b) The user has attempted to erase past the beginning of his Program.
   (c)* The output stack is already full, another entry cannot be made.

8. Transfer Error: EOF
   (a)* A disc I/O routine has attempted to access a mon-existent disc file or duplicate a file name.
   (b)* The disc is full and is not accepting more data.

9. Hardware Error: HRDWR
   (a)* A hardware failure occurred during a disc I/O transfer. Part of the transmitted data is unreliable.

10. Manipulation Error: MANIP
   (a)* The user is attempting to manipulate an empty File.
   (b)* A channel-to-channel operation is requested for dissimilar Files.
   (c)* Execution of the command would cause needless destruction of data.
   (d)* A user attempts to manipulate Slots while a Foreground Program is filling them concurrently.

Notes:

*: These errors are flagged during execution. All other errors are produced in assembly.

Table 7.3: Circumstances that Suspend the Execution of a Foreground Program for One Interrupt Period.

(1) A single device that reads data might be sampled twice in one Interrupt Period. ADC Channels are separate devices.

(2) A third distinct ADC Channel might be sampled in one Interrupt Period.

(3) The output of a DAC Channel might be modified twice in one Interrupt Period.

(4) The output of a third distinct DAC Channel might be modified in one Interrupt Period.

(5) A pulse counter or ADC Channel might be read in the same Interrupt Period that a S:DA command is issued.

(6) A W:SWITCH command is processed, but the switch numbered by the single operand is not depressed.

(7) A W:CLOCK command is processed, but the time noted by the single operand has not elapsed.

Table 7.4: Assignments of Experimental Options and DAC Channels

Experimental Option	Associated Device(s)	Input Parameter	Input Range[2]	Plotted Range	Units	DVM Range Select[4]	True Input of ADC (v)[5]
0	A/D channel 0	LEED:Gun Voltage	0-1500.	0-1500	EV		0-1.500
1	A/D channel 1	Unassigned[1]	—	—	—		0-10.00
2	A/D channel 2	Thermocouple:EMF	0-10.00	0-10.00	MV		0-10.00
3	A/D channel 3	Pico meter:Intensity	Full Scale[3]	full scale[3]	% F.S.		0-3.00
4	A/D channel 4	Mass Spect.:Intensity	Full Scale[3]	full scale[3]	% F.S.		0-10.00
5	A/D channel 5	Lock-in Amp.:Intensity	Full Scale[3]	full scale[3]	% F.S.		±10.00
6	A/D channel 6	Lock-in Amp.:Intensity	Full Scale[3]	full scale[3]	% F.S.		±10.00
7	A/D channel 7	Unassigned[1]	—	—	—		
8	Pulse Counters	Number of Counts	$0\text{-}4\cdot10^9$	$0\text{-}1\text{x}10^8$	COUNTS		
9	System Clock	Absolute Time	$0\text{-}4\cdot10^6$	$0\text{-}1\text{x}10^3$	SEC		
10	——	Temperature	0-2000.	0-1000.	DEG K		
11	DVMs (Mode 0)	CMA:Pass Energy/LEED:Gun Voltage	±15000.0	0-2000.	EV	10	
12	DVMs (Mode 1)	IET:Junction Potential	±1.99999	0-2.000	EV	1	
13	DVMs (Mode 2)	CPD:Gun Voltage	±199.999	0-2.000	EV	100	
14	DVMs (Mode 3)	Thermocouple:EMF	±1999.99	0-20.00	MV	1	
15	DVMs (Mode 4)	Picoammeter:Intensity	full scale[3]	full scale[3]	% F.S.	10	
16	DVMs((Mode 5)	Mass Spect.:Intensity	full scale[3]	full scale[3]	% F.S.	10	
17	DVMs (Mode 6)	Lock-in Amp.:Intensity	full scale[3]	full scale[3]	% F.S.	10	

Notes:

1. Unassigned ADC Channel should not be employed in experiments.
2. Data are printed out over the entire input range.
3. "Full scale" indicates that the range selection switch on a device is employed. Although the device may be read over the full range at any setting, no record of the setting is maintained by the POL system.
4. For the correct display of data, the DVM range should be affixed as indicated.
5. Signals are often preconditioned before they are sent to the ADC Channels.

DAC Channel	Controlled Parameter	Range of the Channel[3]	Units	True Output of the Channel (V)[2]
0	CMA:Pass Energy	0-2000.0	EV	0-5.000
1	LEED:Gun Voltage	0-1500.0	EV	0-3.000
2	Mass Spect.:Mass Select	0-300.0	AMU	0-10.00
3	IET:Junction Potential	0-1000.0	MV	0-10.00
4	unassigned[1]	—	—	0-10.00
5	unassigned[1]	—	—	0-10.00
6	unused[1]	—	—	—
7	unused[1]	—	—	—

Notes:

1. Unassigned or unused Channels should not be employed in experiments.
2. Typically, DAC signals are preconditioned by other electronics to produce the full range of the output.
3. Data are printed out and plotted over this range.

POL-
an Interactive Programming Language
for Experimental Research

by James L. Taylor

Part II: POL Software Manual

The principal load is standing disappointment and upsetting of plans. Everything conspires against him--dumb execution, weather, breakdowns, misunderstandings, deliberate obstructions, jealousies, etc. Must be prepared to accept fifty per cent results in twice the time calculated.

- General Joseph W. Stilwell

8. Introduction

8.1 Programming Philosophy

The POL operating system is best analyzed as a collection of routines and subroutines. In turn, these are constructed around Peripheral and Reserved Addresses, which are locations in core manipulated by routines to perform their functions. In this manual, POL is described in an ascending hierarchy. Peripheral and Reserved Addresses are described in Chapters 9 and 10. In Chapter 11, the functions of the routines that access these variables are briefly outlined. The POL listing and flow charts of Chapter 12 detail the functions of the routines. When studying this manual, the reader may wish to refer to Appendix B of Part I, which is a glossary of POL terminology.

The PDP-11 computer communicates with experiments through Peripherals (ADCs, DACs, etc.) which are manipulated through Peripheral Addresses. To monitor Peripherals, a routine loads information into, or accesses data from these addresses. Two types of addresses, Buffer and Status Registers, are associated with each Peripheral. The Buffer Register holds the information passed between the routine and Peripheral. The Status Register dictates when the transfer will occur.

Routines communicate among themselves through General Programming Registers, or simply Registers, and Reserved Addresses. Long-term communication can only be affected through Reserved Addresses, for the contents of Registers change frequently to expedite execution. Each

POL instruction is serviced by a separate subroutine. Hence, two instructions interact through Reserved Addresses. The reader must appreciate the role of Peripheral and Reserved Addresses as communication paths. If he understands what is stored in these addresses and how they are modified, he can determine how POL implements several complex interactions between routines, and between routines and experiments. Subsequently, he can appreciate how each routine in POL performs certain tasks.

POL is a compendium of routines which are classified into three types--Command Subroutines, Function Subroutines, and Management Routines. POL instructions are implemented by Command Subroutines. All POL instructions with the same mnemonic are serviced by a unique Command Subroutine. Function Subroutines perform functions common to several Command Subroutines. With Function Subroutines, repetition is eliminated at a slight cost in execution time. Management Routines schedule the assembly and execution of Command Subroutines.

Once the reader understands what tasks a routine accomplishes and how it implements Peripheral and Reserved Addresses in realizing these tasks, he can easily decipher the assembly language code of a routine. The Table of Contents preceding the POL Listing is useful for locating the various routines in the listing. To the left of each routine is a flow chart that explains the code. The flow charts extend the descriptions of the routines that are outlined in Chapter 11. These descriptions do not elucidate the state of the CPU or mimic the assembly language code, but detail, in words, the actions performed by the routine. Several of these actions are visible to the user through the

operations of Peripherals. When a Reserved or Peripheral Address is manipulated, the flow chart describes how and why. However, the reader should be familiar with the addresses to understand the flow charts.

8.2 Terminology

The assembly language programmer should be familiar with the jargon contained in this manual. To avoid possible confusion, the jargon is described in this section.

General Programming Registers 0 through 5 of the CPU are referred to separately as R0 through R5 or collectively as Registers. If a double length number is held in two consecutive Registers, the Register pair is enclosed in angle brackets, e.g., <R1,R2>. The more significant word appears in the first Register. Whenever "the stack" is mentioned, the area in core marked by R6, the stack pointer, is referenced unless noted otherwise.

If A calls B, then routine A temporarily passes control to subroutine B through a JSR statement. Assembly language commands are always underlined and appear in capital letters. A jump or branch from A to B indicates that control of the CPU permanently passes from routine A to routine B. After an interrupt is generated, the service routine in the associated interrupt vector executes. The Hardware Interrupt, generated so that Foreground Programs can be serviced, always appears capitalized.

If the number symbol, #, precedes the label of an entry point in a routine or of a Reserved Address, the address of the Label is alluded to. The terms "@", "marked by", and "at" mean "at the address in". For example, "the address marked by R1" references the location whose address is contained in R1. When an address or Register "points to" information or is "a pointer", it holds the address of the location that contains the information.

The term "#n", where n is a number, alludes to the number n, not an absolute address in core. The term #3 refers to the number three while 165000 is an absolute address in core. All numbers are in the octal radix, base eight, unless they contain a decimal point. Then the number is enumerated in base ten.

9. Peripheral Addresses

The POL operating system interacts with many peripherals, including those which monitor experiments, by accessing or loading the contents of certain locations in memory. These locations are called Peripheral Addresses. Through Program Requests, POL manipulates addresses assigned to commercial peripherals. Their operation will not be discussed here, for it is the subject of several DEC publications. This chapter describes the operation of peripherals that are manipulated through the interface board, which is located below the point-plotter.

9.1 The Structure of Peripheral Addresses

Each peripheral is assigned two addresses: a Status Register, from 164000 through 164776, and a Buffer Register, from 165000 through 165776. A total of 256 Status and Buffer Registers are assigned to peripherals. All registers are one word in length. Several registers are not used.

Peripherals are assigned parallel Status and Buffer Registers. Parallel addresses are indexed by the same number. For example, 164230 is the Status Register associated with Buffer Register 165230. The Buffer Register holds information that is read from, or written into a peripheral. The Status Register dictates <u>when</u> the transfer between the peripheral and Buffer Address occurs. Although Status

Registers can be read and loaded by a program, Buffer Registers can only be read or loaded, depending on whether the associated peripheral reads or controls an external parameter.

Peripherals are manipulated either synchronously or asynchronously. If a peripheral is accessed asynchronously, the Status Register is zeroed. Then information is read directly from, or written into the Buffer Register. When bit 3 of the Status Register is set, the peripheral operates synchronously. The buffer is accessed or loaded periodically as the Status Register dictates. Bit 4 of the Status Register is zero (one) if a peripheral is read (loaded) synchronously.

The three least significant bits select a Subinterval, 0 through 7, for synchronous operation. Every Interrupt Period of 16 msec is divided into eight Subintervals of 2 msec each. Subinterval 0 occurs from 0 to 2 msec after the Period starts; Subinterval 1 occurs from 2 to 4 msec, etc. When a Buffer Register synchronously writes to a peripheral, the contents of the buffer are loaded into the peripheral during the Subinterval of the next Interrupt Period. If the Buffer Resister synchronously reads from a peripheral, the contents of the buffer were read from the peripheral in the Subinterval of the previous period. The Subinterval scheme is implemented so that several of the eight Channels of the ADC and/or DAC can be manipulated in a single Interrupt Period. Note that bits 5 through 16 of Status Registers are not used.

The POL operating system allows each of the four Foreground Programs to manipulate peripherals synchronously during two

Subintervals of each Interrupt Period. Foreground Program 1 is assigned Subintervals 0 and 4; Foreground Program 2 is assigned Subintervals 1 and 5, etc. With this system, the user can time the execution of his Foreground Program easily. However, only two Channels from both the 8-Channel ADC and 8-Channel DAC can be manipulated by a Foreground Program in an Interrupt Period. The first ADC Channel encountered in execution is manipulated in the first Subinterval, but the first DAC Channel encountered in assembly is manipulated in the first Subinterval. Although a Foreground Program may only load two DAC Channels, it may read any number of ADC Channels. However, only two are read in a single Interrupt Period.

9.2 Peripherals that are Synchronously Controlled by POL

The 8-Channel DAC, the 8-Channel ADC, and pulse counters are operated synchronously. Other peripherals are manipulated asynchronously.

The output from any Channel of the DAC is the sum of the outputs of two 12-bit D/A converters, the wide and narrow converters. Because the output range of the narrow converter is one-tenth the range of the wide converter, the output from any Channel can be resolved to within one part in 40960. The decimal input, entered on the Decwriter as the second operand of the S:DA and I:DA commands, is translated into two words of 12-bit complementary code (12-bit CBIN). These words will be loaded into the wide and narrow converters.

Translations occur so that the full resolution of Channels is utilized.

Buffer Registers 165000 through 165016 are assigned to the wide converter while registers 165020 through 165036 are assigned to the narrow converter. The registers are indexed by a Channel number, 0 to 7. For example, the registers for the wide and narrow converters of Channel 3 are 165006 and 165026, respectively. Only the 12 least significant bits of each Buffer Register are used.

Whenever the output of a Channel is manipulated, the Buffer Registers of the wide and narrow converters are modified. The two Status Registers for the Channel are set to the same values. Bits 3 and 4 are set so that the Channel will be loaded synchronously. Bits 0, 1, and 2 select the Subinterval in which the Channel is manipulated.

Each Channel of the 8-Channel ADC is assigned two Buffer and two Status Registers. The Channel number that will be read during each Subinterval is placed in registers 165740 through 165756. With 165740 containing the Channel number that is read in the first Subinterval, these locations are indexed by the number of a Subinterval. Since the Channel is selected asynchronously, Status Registers 164740 through 164756 are zeroed.

Registers 165700 through 165716 hold the data read synchronously during each Subinterval. To index these registers by the number of a Subinterval, the Status Registers 164700 through 164716 are loaded with 10 through 17, respectively. The contents of the Buffer Registers are in 12-bit, offset, complementary, binary code (12-bit OCBIN). The four most significant bits of each register are set.

Suppose Channel 5 is to be read in the third Subinterval, number 2. Sometime during the first Interrupt Period, 5 is loaded into Register

165744. In the third Subinterval of the next Period, Channel 5 is read. The datum is available in Register 165704 throughout the third Period.

Each pulse counter is allotted two Status and one Buffer Registers. Pulse counter 0 uses registers 164040, 164054, and 165054; pulse counter 1 uses registers 164042, 164056, and 165056. The two Status Registers of either pulse counter are loaded identically. Bit 3 is set and bit 4 is clear to denote that the device is read synchronously. Bits 0, 1, and 2 select the Subinterval in which the counter is read and then zeroed. This Subinterval is always the earlier one assigned to the Foreground Program. The Buffer Register contains datum, in 16-bit complementary binary code (16-bit CBIN), that was read from the counter for and summed 16 msec preceding the Subinterval.

9.3 Peripherals that are Asynchronously Controlled by POL

The multiplexed DVMs, numbers 0 and 1, share three Status Registers (164222 through 164226) and three Buffer Registers (165222 through 165226). All Status Registers are cleared for asynchronous operation. The Buffer Registers are loaded or accessed in uncomplemented binary code.

Register 165224 selects the DVM to be sampled. Bit 0 is set to read DVM0 and bit 1 is set to read DVM 1. Twenty-five μsec after one of these bits is set, the DVM holds its datum and will not sample another reading until the bit is clear. While the device is sampled,

the bit remains set so that the datum does not change. Afterwards the bit is cleared; then the DVM may sample new data.

Register 165222 is loaded to select the character that is read from the DVM. After a 25 μsec delay, the character appears in 165226. The most significant thirteen bits of 165222 are not used; the most significant twelve bits of 165226 are always set. When 165222 is loaded as follows, the indicated character appears in 165226.

Value in 165222	Character in 165226
0	10^0_1 digit
1	10^1_2 digit
2	10^2_3 digit
3	10^3_4 digit
4	10^4 digit
5	bit 0: 10^5 digit; bit 1: sign (0 = minus); bits 2 and 3; not used.
6	decimal point (range setting)
7	function setting

The coding of the decimal point and function is described in the manual for the Keithley 190 DVM. POL does not access these characters. As noted in Table 7.4, the second operand of the R:VM command delimits how the function and range switches should be set to read a particular variable.

The graphics media, the display scope and the point-plotter, are manipulated through Buffer Registers 165230, 165234, and 165236. Status Registers that correspond to these locations are zeroed.

The 10-bit, 8-Channel, graphics DAC is loaded from 165234. Inputs are in 10-bit complementary binary code (10-bit CBIN). The most significant 6 bits of this location are not used. Register 165236

controls writing on the graphics media, and register 165230 senses the progress of graphics operations.

Bits 2, 3, and 4 of 165236 select the Channel, both the axis and device, which is manipulated by the graphics DAC. Channels 0 through 3 correspond to the x-axis of the scope, y-axis of the scope, x-axis of the point-plotter, and y-axis of the point-plotter, respectively. Other Channels are not used. A one-to-zero transition in bit 5 enables the multiplexer to load the datum from 165234 into the selected Channel. Similarly, a one-to-zero transition in bits 0, 1, or 6 enables the point-plotting pen, turns on the intensity of the scope to plot a point, and clears the scope, respectively. Other bits are not used.

Bits 0, 1, and 2 of 165230 are clear if the point-plotting pen is down, the pen has slewed to the correct position, and the scope is done erasing, respectively. If the "panic" button is depressed, bit 7 is set. Bits 3 through 6 are not used.

The condition of sense switches 0 through 7, which are referenced by the W:SWITCH command, are indicated by bits 8 through 15 of 165230, respectively. When the switch is depressed, the appropriate bit is set. The bit is cleared as soon as the switch is released.

9.4 Interrupt Vectors

While POL executes, three types of interrupts are generated. Two of these are software Interrupts, triggered by the EMT and TRAP instructions. The third is the Hardware Interrupt which initiates the

the execution of Foreground Programs. The RT-11 monitor maintains the vector and service routines for the EMT interrupt. Other interrupt vectors and service routines are managed by the POL operating system.

The TRAP instruction, whose interrupt vector is at address 34, is used in error detection. In the first word, the #TRAP is stored. This routine services errors. The second word in the interrupt vector, the new processor status word, is cleared, for errors are serviced at the lowest priority, #0. Remember that errors are flagged only in the Background mode.

A Hardware Interrupt is generated every 16 msec as long as bit 0 of register 164776 is set. The vector for this interrupt starts at location 270. The #FMON is contained in the first word. This routine services synchronously controlled Peripherals and Foreground Programs after each Hardware Interrupt is generated. Since data collection occurs with highest precedence in POL, FMON runs at the highest priority, #7. Hence, the second word of the vector, the new processor status word, is #340. (See the RT-11 Manual for a description of the processor status word.)

10. A Description of Reserved Addresses

Information is passed between routines through Registers and Reserved Addresses, which are labeled addresses in core. Since the contents of Registers change frequently to expedite the execution of routines, information is permanently stored in Reserved Addresses. Hence, the programmer who modifies the POL operating system must thoroughly understand the functions of these addresses. This chapter aids him by describing all Reserved Addresses in POL.

For a 4K operating system, POL contains relatively few Reserved Addresses. These are defined on the first four pages of the listing of POL. (See § 12.3.) After the listing of the POL operating system, a cross-reference table notes where Reserved Addresses are referenced by POL routines. Chapter 5 in the RT-11 Manual describes how this listing is interpreted.

Two types of Reserved Addresses are employed in POL. When POL executes, the contents of Fixed Reserved Addresses are not altered, but the contents of Variable Reserved Addresses are. The classification of Reserved Addresses can be further refined. If a name is assigned to a single Reserved Address, it is called a variable. A table is a succession of Reserved Addresses that are allotted a single name. The programmer must understand how a single address is indexed in each table.

10.1 Fixed Reserved Addresses

All Fixed Reserved Addresses appear in tables. These addresses are defined in the first three pages of the POL Listing. Fixed Reserved Addresses store information for command assembly, ASCII messages, data analysis, and experimental Peripherals.

Entries in COMADD, COMNAM, COMERR, LIMIT1, LIMIT2, COMFL1, and COMFL2 are employed in interpretting commands that are entered on the Decwriter. The word (for COMADD, COMNAM, and COMERR) or byte (for LIMIT1, LIMIT2, COMFL1, and COMFL2) of information pertaining to an instruction is indexed by a command number.

Each of the 63. instructions in POL is assigned a command number from 0 through 62. Assignments appear in Table 11.1. To create a new instruction, the programmer must write a subroutine to realize the instruction and append an entry to six of these seven tables (not COMERR). New commands will be numbered 63., 64., etc.

A subroutine services each instruction of POL. COMADD tabulates the first address of each service routine, called a Command Subroutine. Note, the names of the service routines are almost the same as the mnemonics of the instruction. By observing this and the fact that the entries in COMADD are ordered by command number, one can determine the Command Subroutine for any mnemonic without consulting Table 11.1.

COMNAM holds a numeric code of the first three characters of an instruction. If the first three characters appear in ASCII as x_α, x_β, and x_γ, the code c is generated by the algorithm,

$$c = (x_\alpha - 72)\cdot 2^{10.} + (x_\beta - 72)\cdot 2^5 + (x_\gamma - 72)$$

Only ASCII characters from 72 to 131, inclusive, are used in the names of instructions. After an instruction is entered on the Decwriter, the name is coded. To determine the number of the command, which indexes an entry in COMNAM, the code is sought in this table. An error is flagged if the search fails.

Entries in COMERR are Assembly Subroutines which are called during assembly to detect errors and/or assemble the second operands of instructions. These subroutines are unique to an instruction or group of instructions. Unlike the contents of other tables that are referenced in assembly, entries in COMERR are indexed by subtracting 25. from the command number. For example, the first entry in the table, CHMOVE, is called when the X:MOVE instruction is assembled. Commands numbered 0 through 24., and 49. and greater summon no special subroutines in assembly.

A maximum value of the first operand of an instruction, m_1, is given as an entry in LIMIT1. Normally, the first operand varies from 0 to m_1. LIMIT2 establishes a maximum absolute value for the second operand, m_2.

Entries in COMFL1 and COMFL2 are flags that direct the assembly of instructions. The first table dictates how operands are assembled. Often, the ranges which the operands may assume are modified by flags in COMFL1. The latter table facilitates error detection.

The bits of each entry in COMFL1 have the following significance. Bits that are not discussed are not used. If an instruction should have two operands, bit 7 is set. Otherwise, it is clear. The first operand should not be zero when bit 6 is set. Similarly, bit 2 is set

if the second operand should not be zero. Bit 4 is set so that three digits right of the decimal point are assembled into the binary code for the second operand. Otherwise, decimal places are truncated from either operand. When the second operand should not be negative, bit 3 is clear. The range of the second operand is extended by a factor of $2^{16.}$ if bit 1 is set. Likewise, bit 0 is set to extend the range of the second operand by a factor of $2^{8.}$. When bits 0 and 1 are set, the range is extended by $2^{24.}$.

Entries in COMFL2 are used in error detection. Bits 7, 6, or 5 are set if a command can be legally issued as an Immediate Instruction, in a Background Program, or in a Foreground Program, respectively. When bit 0 is set, an instruction must be preceded by a File-referencing statement. Otherwise, an error is flagged.

ERROR contains error messages, coded in ASCII. Messages appear on the Decwriter as

ERROR aaaaa

where aaaaa is one of the last ten entries in this table. Entries are indexed by the last byte of the TRAP instruction, 1044xx, that flags the error. If xx is 01, then aaaaa is "INSTR".

LABTAB lists the headings for columns of experimental data that are printed on the Decwriter. Entries in WRLAB and RDLAB (q.v.) index the heading for a type of data.

The powers of ten from $10^{8.}$ to 10^{0}, required for binary to decimal conversion, are stored in TENS. Each power is contained in two words of memory. As is always the rule in POL, the more significant word

occurs in higher core.

Experimental data are filtered by a convolution scheme. CONFAC holds the factors for fitting five data points with a quadratic or cubic function. The excitation spectrum of the Al-K_α line is contained in DECFAC. Points in the spectrum, i.e., bytes, give intensities at intervals of 0.1 eV. This table is used in deconvoluting the excitation line of the aluminum source from measured XPS data.

EMFs from Pt, Pt-10% Rh thermocouples are converted to temperatures by a piece-wise linear fit of the EMF versus temperature curve. The curve up to 18.60 meV has been divided into ten linear ranges. Three words in TEMP describe each line. The first word stores the maximum EMF, in tens of μeV, over which the line is accurate, and the minimum EMF is given by the same word of the preceding range. For EMFs that are less than 0.05 meV, the first range is assumed to be correct. When the EMF falls in a particular range, the temperature in ^{o}K is calculated by multiplying the EMF by the second word, dividing the product by 1024., then adding the third word to the quotient.

Entries in WRFACT, WRSHFT, and WRLAB are indexed by a channel number, 0 through 7, of the 8-Channel DAC. As listed in Table 7.4, Channels are assigned to <u>one</u> experiment.

Using WRFACT and WRSHFT, the POL operating system translates numbers entered on the Decwriter into a 12-bit code. This code is loaded into the wide and narrow converters of the 8-Channel DAC. If an experimental parameter is to be set to the decimal value v, then the codes for the wide and narrow converters, w and n, are generated by the algorithms:

$$w = -1000. \times 2^{WRSHFT} \times v/WRFACT$$

$$n = \left[(w+1) \times \frac{WRFACT}{2^{WRSHFT}} + 1000. \times v \right] \cdot \frac{10. \times 2^{WRSHFT}}{WRFACT}$$

The output of a Channel is set correctly only when the units of v match those in Table 7.4.

When the initial setting and increment of a DAC Channel generate the abscissa in a listing of data, WRLAB determines which heading will be printed above the abscissa. To mark the address of a heading, an entry in WRLAB is added to #LABTAB. The heading is stored in the five bytes preceding the marker.

Entries in RDLAB, RDDP, and RDFACT are indexed by the number of an experimental option, 0 through 17. In Table 7.4, options are assigned to experiments.

RDLAB indicates which heading is printed above the ordinate in a listing of data. If the data points of a File provide the abscissa, then the option number of the File indexes an entry in RDLAB to provide a heading for the abscissa. To mark a heading, the contents of RDLAB are accessed like those of WRLAB.

When experimental data are printed on the Decwriter, RDDP gives the number of digits that appear to the right of the decimal point. DAC settings are always listed with two decimal places.

The contents of RDFACT adjust experimental data for printing or plotting. Data from the 12-bit, 8-Channel ADC are divided by four before they are plotted by the 10-bit graphics DAC. Prior to their listing on the Decwriter, these data are divided by 4.096, then multiplied by the appropriate entry in RDFACT. Data from other

devices, with experimental options from 8. through 17., are not modified before they are listed except for the insertion of a decimal point. When these data are plotted, they are multiplied by 1.024, then divided by an entry in RDFACT.

10.2 Variable Reserved Addresses

Variable Reserved Addresses are defined on the fourth page of the POL Listing. Since many of these addresses reference Files and Programs, the structures of these records will be discussed first. Programs and Files are stored between #POLEP and the beginning of the RT-11 operating system, 113050. The core locations of these records are variable. When the user requests space in memory for a File or Program, the memory management routines assign the lowest area in core, not occupied by other records, to accommodate the new record.

Space for a File or Program consists of sequential addresses in core. Preceding the record, the number of words in the record is stored. The address of the first word in the record is held in PRBUFF if the record is a Program, or FLBUFF if the record is a File. Entries in PRBUFF are indexed by Program number, 0 through 4; entries in FLBUFF are indexed by File number, 0 through 110.

When space is reserved for a Program, one word is allocated to store the number of words in the record, and two words are allotted for each statement in the Program. The first word of each statement stores the command number in the lower-order byte and the first operand

in the higher-order byte. The END or last statement of an assembled program has a command number #-1. The second operand, which is often modified in assembly, is contained in the second word. (See Appendix D.)

The word preceding a File holds the number of words in the record. The record consists of the next five words, which store the Label, and two words for each Slot. Slots are numbered from 0 through (n-1) where n is the total number of slots. Appendix C describes the format of entries in Slots. The number of Slots is stored in the first word of the Label. The second word contains the lower-order address of the Slot where the next piece of data will be stored. This address is manipulated by Foreground Programs. The experimental option of the data is held in the lower byte of the third word. The next byte hold the Channel number of the 8-Channel DAC that was monitored while the data Slots were loaded. The initial setting and increment of the DAC Channel are held in the fourth and fifth words. The contents of these locations, i, are determined from the algorithm,

$$i = -10 \cdot w - n - 11$$

where w and n are the 12-bit codes of the settings of the wide and narrow converters.

RDKEY, WRKEY, and FLKEY contain the number of a Key from 1 through 4, or zero. If a Peripheral is not being used or the File is not allotted space in core, the entry in these tables, a byte long, is zero. When a Foreground Program, whose number is different from a non-zero Key, attempts to reference the Peripheral or File, an allocation error

is flagged in assembly. The same error is generated if a Background Program or Immediate Instruction references a File whose Key is different from AKEY (q.v.).

Entries in FLKEY are indexed by File number, 0 through 110. The Key is deposited when space is reserved for the File and is zeroed when the File is erased. Locations in WRKEY are indexed by a Channel number, 0 through 7, of the 8-Channel DAC. The first two entries in RDKEY are indexed by a voltmeter number, 0 or 1; the next eight by a Channel number, 0 through 7, of the 8-Channel ADC; and the last two by the number of a pulse counter, 0 or 1. Keys of Peripheral devices are assigned to the first Foreground Program that legally references the Peripheral. When a statement referencing the Peripheral or the Foreground Program is erased, the Key is zeroed.

Entries in STOP, PRFILE, PRITER, and PRCNT are indexed by a Program number, 0 through 4.

When an END statement is processed, the Program ceases to execute, PRCNT is cleared, and the entry in STOP is incremented. The Background monitor, BMON, clears the location in STOP, then informs the user that execution has terminated by printing

STOP x

where x is the Program number.

If a Program has been completely assembled, the higher order byte of PRFILE is one. Otherwise, it is zero. While a Program executes, the lower-order byte stores the File number, given by the last File-referencing statement. The File-reference for Immediate Instructions

is stored at #PRFILE. While any Program assembles, this byte is manipulated to detect errors in referencing Files. Note that this single byte is accessed and modified by Immediate Instructions, Background Programs, and Assembling Programs. No interference occurs because only one of these three Background processes can execute at any time.

Suppose an instruction has time, i.e., Interrupt Periods, as the second operand. During the first Interrupt Period that this instruction is processed, this operand is placed in PRITER. In this and successive Interrupt Periods, the instruction is executed and PRITER decremented. Until the PRITER is decremented to zero, this is the only instruction in that Foreground Program which is processed in that Period. Then the next statement in the Program is processed.

PRCNT is a Program counter that points to the lowest order address of a statement in a Program. During assembly, PRCNT marks the location where the next statement in the Program will be stored. In execution, PRCNT holds the address of the next statement to be processed. When execution of a Program is initiated, PRCNT is set to the first address of the Program. This location is cleared when execution or assembly is completed. Foreground Programs are serviced whenever an interrupt is generated and the contents of PRCNT are not zero. Of course, the Program must be assembled - that is, the higher byte of PRFILE is not zero – if the Program is to execute.

During assembly, an instruction is temporarily stored in INSTR. This variable is loaded to mimic a statement in a Program: the command number is stored in the lowest-order byte, etc. Following assembly,

INSTR is either loaded into a Program or executed if the statement is an Immediate Instruction.

PROGNO holds the number of the Program, 0 through 4, that is executing or assembling. If the POL system is accepting and executing Immediate Instructions, PROGNO is #-1. The START and REENTER instructions initiate assembly when their second operands are placed in PROGNO. Assembly ends when PROGNO is reset to #-1. This variable changes value every time an Interrupt is generated and is restored after all Foreground Programs have been serviced in that Interrupt Period. During the Interrupt Period, PROGNO equals the number of the Foreground Program that is executing.

Each Program is allotted four Loops. Entries in LPITER and LPRET are indexed by multiplying the Program number, 0 through 4, by four and adding the Loop number. For example, Loop 2 in Foreground Program 3 references the fifteenth word in LPITER or LPRET.

When a LOOP command executes the number of iterations, the second operand, is placed in LPITER. The address of the statement following the LOOP command is stored in LPRET. LPITER is decremented when a CONT statement is processed. If LPITER is greater than zero, PRCNT is loaded with LPRET and a section of the Program is re-executed. Otherwise, the instruction following the CONT statement is processed.

RDSTAT and WRSTAT indicate which Peripherals have been read or loaded by a Foreground Program during an Interrupt Period. No Foreground Program may manipulate any Peripheral more than once during one Interrupt Period. (ADC and DAC Channels are separate Peripherals.) Only two DAC and two ADC Channels can be accessed by a Foreground

Program in a single Interrupt Period. To allow for settling of experimental parameters that are controlled by the DAC, synchronous Peripherals cannot be read in the same period that a S:DA command is issued. An additional constraint exists: a Program may sample a single pulse counter during an Interrupt Period. If any of these conditions is violated, the execution of Foreground Program is suspended for one Period.

All bits in RDSTAT and WRSTAT are set before each Foreground Program executes in each Period. The number of the first ADC Channel sampled by a Program is stored in the higher-order byte of RDSTAT. When a second Channel is sampled, bit 4 of RDSTAT is cleared. Similarly, the number of the first DAC Channel loaded by a Foreground Program is placed in the higher-order byte of WRSTAT and bit 4 of WRSTAT is set after the second Channel is set. Bit 7 of WRSTAT is cleared whenever a S:DA command is processed. When voltmeter 0, voltmeter 1, a pulse counter, or the system clock is read, bit 0, 1, 5, or 6 in RDSTAT is cleared, respectively. To determine if a Foreground Program should cease processing, RDSTAT and WRSTAT are examined before a Peripheral is manipulated.

The current reading of the system clock in msec is stored in the two words of TIME. As is the rule in POL, the higher order word occurs in higher memory. TIME is incremented by 16 msec after every Hardware Interrupt.

STORE1 and STORE2 direct the storage of data read from synchronous Peripherals. The first eight entries in each table are indexed by a Subinterval number, 0 through 7, during which an ADC Channel is read.

The last two entries are indexed by the number of a pulse counter, 0 or 1.

STORE1 and STORE2 hold the lower-order addresses of Slots. Data read from synchronous Peripherals are stored in these Slots. If the reading will be available in the next Interrupt Period, STORE1 holds the address where the datum is accumulated. (Remember, datum from a synchronous Peripheral is added to a Slot.) STORE2 marks a SLOT for datum that can be accessed in two Interrupt Periods.

Shortly after an Interrupt is generated, data from synchronous Peripherals are read, then stored, as STORE1 dictates. For example, datum read from the ADC in Subinterval 2 is stored in the Slot marked by the third entry in STORE1; datum from pulse counter 0 is placed in the Slot marked by the ninth entry. No data are stored if the entry is zero. The contents of STORE2 are loaded into STORE1, and STORE2 is zeroed. STORE2 is loaded by Foreground Programs which execute in that Interrupt Period.

Entries in ADITER are indexed by a Subinterval number, 0 through 7, during which an ADC Channel is sampled. Each location numbers the Interrupt Periods over which a single reading is accumulated from an ADC Channel. If parallel entries in STORE1 and STORE2 differ at the beginning of an Interrupt Period, the reading marked by STORE1 is averaged by ADITER after the last datum is accumulated as dictated by STORE1. (Parallel entries have the same index.) Then the entry in ADITER is cleared. Whenever ADITER is zero, or parallel entries in STORE1 and STORE2 are identical, no averaging occurs. After STORE 2 is loaded by the R:AD command, the second operand is stored in ADITER

only if parallel entries in STORE1 and STORE2 are identical.

This scheme has two important features. An ADC Channel can be used either to average or to accumulate data. Although two different Slots may be marked parallel entries of STORE1 and STORE2, only one count of Interrupt Periods, not two, must be stored at any one time to average the readings correctly at STORE1 and STORE2.

Entries in WIDEDA and NARRDA are indexed by a Channel number, 0 through 7, of the DAC. These tables mimic the values that are loaded into the wide and narrow converters for each Channel. When the output from a DAC Channel is to be modified, WIDEDA and NARRDA are modified. Then these values are placed in the Buffer Registers of the wide and narrow converters for a Channel. As discussed in § 9.2, these registers are at addresses 165000 through 165016 for the wide converter and 165020 through 165036 for the narrow converter. These locations, too, are indexed by Channel number. An increment cannot be added directly into the Buffer Registers, for these cannot be read, only loaded.

The output stack, OSTACK, holds up to eight data records so that these can be graphed or listed simultaneously. Each data record requires six words in OSTACK. Records that were entered more recently in OSTACK are stored in higher memory.

The six words of each record contain the following information. The lower-order byte of the first word numbers the File whose data points form the ordinate. The higher order byte numbers the File that produces the abscissa. These bytes are loaded from the operand of the last File-referencing statement and the second operand of the PUSH

command, respectively. When the latter location is zero, the initial value and increment of the DAC Channel, stored in the Label of the ordinate File, generate the abscissa. The expansion factor for the y-axis, the zero-point of the y-axis, the expansion factor for the x-axis, and the zero-point for the x-axis are held in the next four words. After the PUSH command enters a record on OSTACK, these locations are assigned values 1, 0, 1, and 0, respectively. With the following algorithm, zero-points z are assembled from the operand i of the X:MOVE or Y:MOVE command:

$$z = 327.68 \cdot i$$

The last word stores the display mode, whose values are defined in Table 5.1 and entered as the first operand of the PUSH command.

The pointer to the output stack, OSTPNT, marks the lowest-order address of the six words in the latest entry on OSTACK. When a record is pushed or popped off the stack, OSTPNT is incremented or decremented by twelve. If the stack is clear or holds only one record, OSTPNT contains the address of OSTACK.

IOBUF is a buffer that holds 132. ASCII characters for output to or input from the Decwriter. Characters that appear to the left on the Decwriter are stored in lower addresses in core. When IOBUF is used for output, the output pointer, OUTPNT, is incremented as IOBUF is filled. IOBUF is loaded from left to right, i.e., from lower to higher core. The address of IOBUF plus the contents of OUTPNT marks the lowest-order address in IOBUF which has <u>not</u> been filled with characters.

CURSOR numbers the Slot marked by the display cursor.

The variable in GRSTAT mimics the contents of the Buffer Register 165236. (See § 9.3.) To manipulate the plotting media, bits in GRSTAT are set and/or cleared. Then it is loaded into 165236. Bits in this Buffer Register cannot be manipulated directly, for it cannot be read.

YSTART, POINTS, and XSTART are loaded before a record from OSTACK is graphed or a group of records is listed on the Decwriter. These variables facilitate displaying data. YSTART holds the address of the ordinate File while XSTART stores the address of the abscissa File. If the abscissa is generated from DAC information in the Label of the ordinate File, XSTART is zero. POINTS holds the number of data Slots in the ordinate or abscissa File, whichever is fewer.

AFILE, AKEY, LFILE, FINIS, TRNAME, EXAREA, and RQAREA are manipulated by memory and disc management routines.

When the F:OPEN instruction executes, the first two variables are loaded. The first operand, a Key, is placed in AKEY; the second, a File number, is stored in AFILE. LFILE contains the File number succeeding that of the last File in a sequence involved in a memory or disc management operation.

When an EMT interrupt is generated for a disc I/O operation, EXAREA and RQAREA pass information to Program Requests. These areas are filled by POL routines before an EMT trap occurs. The LOOKUP, ENTER, and DELETE requests access RQAREA; the READC and WRITC requests access EXAREA. Although the RT-11 Manual completely describes the information passed through these areas, their contents are described here briefly.

The code for the Program Request is stored in the higher byte of the first word in each area. For example, the code for LOOKUP is #1. The number of the I/O Channel is held in the lower order byte. The POL operating system always uses Channel #1 for transfers to and from the floppy disc. Files and Programs of POL are always stored on, or accessed from, Drive 1. Drive 0 stores the RT-11 and POL operating systems.

The second word in RQAREA points to the name of the disc file referenced by the LOOKUP, ENTER, or DELETE requests. This word always stores #TRNAME. The first and last of the four words in TRNAME are fixed so that a disc file with extension LDA on Drive 1 is referenced. The alphabetic and numeric names are stored in the second and third words. These words are concatenated to name a disc file. The alphabetic name consists of three alphabetic characters. For Files, the numeric name is a File number from 001 to 110, for Programs, it is a Program number, 00 to 04 , prefixed by 'P". All words in TRNAME are in Radix 50 code.

In EXAREA, the third and fourth words hold the Buffer address and word count of a block of memory involved in a READC or WRITC request. The Buffer address is the address of a File or Program. The word count notes the number of successive words, including and following the Buffer address which are transferred to or from disc. Entries in PRBUFF and FLBUFF wlways supply the Buffer address. The word preceding the Buffer address gives the word count, the number of words in the Program or the File.

The address of FINISH, the completion routine for the READC and WRITC requests is contained in the fifth word in EXAREA. The variable FINIS is set to one before these requests execute. If a hardware error occurs during a disc I/O transfer, FINIS is set to #-1 by the completion routine. Otherwise, FINIS is zeroed.

SHIFT, MANT, and EXPON are used by numerical routines.

After a multiplication or division is performed by the MULT or DIVID subroutines, the result is multiplied by a power of two. The exponent is given by SHIFT, which is always zeroed following a multiplication or division. SHIFT is loaded by the routine that calls MULT or DIVID. If SHIFT is not loaded, it defaults to zero. SHIFT is used to expedite binary-to-decimal and decimal-to-binary conversions.

MANT and EXPON are always loaded when a calculator command executes. When the LOAD command processes, the first operand, the exponent, is placed in EXPON. The second, the mantissa, is stored in MANT. The exponent and mantissa of the result from the ADD, SUB, MUL, or DIV commands are loaded in EXPON and MANT.

11. Routines in the POL Operating System

The POL operating system is a compendium of short routines and subroutines. Some of the subroutines, Command Subroutines, service instructions that are entered on the Decwriter. Function Subroutines support Command Subroutines. A Function Subroutine performs an operation necessary in the execution of several Command Subroutines. The assembly and execution of Command Subroutines is directed by Management Routines. Most Management Routines are not subroutines. For example, some are structured as routines that service interrupts.

While POL runs, control of the CPU is frequently passed between Management, Command, and Function Routines. When a routine has control, the CPU executes that routine. A programmer that modifies POL must understand this flow of control thoroughly. This chapter discusses, in general, how control is passed between routines and what the function of each routine is. Chapter 12 details each routine with flow charts.

11.1 Command Subroutines

A Command Subroutine executes all POL instructions with a common mnemonic. When certain of these instructions assemble, Assembly Subroutines are called. These subroutines detect errors that are unique to an instruction and/or modify the second operand of an instruction before it executes. The functions of the Command and Assembly

Subroutines are evident from the description of instructions in Part I. Appendix D describes how the second operands of some instructions are modified in assembly.

Table 11.1 lists the name of the Command and Assembly Subroutines associated with each mnemonic. Note that an Assembly Subroutine is not associated with some instructions. Certain Command Subroutines have entry points, recorded in Table 11.1, that are different from the beginning of the subroutine. Rather than repeat a section of code in several subroutines, these entry points allow the subroutines to execute the same section of code, yet the code is not duplicated. A few command subroutines (END, ISTORE, and ADD) are executed through a call to an entry point in another Command Subroutine.

Only two entry points in the table warrant discussion. The entry point GRAPH in the PLOT subroutine is called to display the File pairs from the output stack on the display scope. After RFRESH is entered, the display scope is cleared and then the trace is refreshed.

Command Subroutines are executed by a two step process. The Foreground and Background monitors, which initiate the execution of instructions in the Foreground and Background modes, enter the Management Routine, NEWCOM, through the statement

```
JSR  PC, —
```

NEWCOM loads arguments of a POL instruction into R0 through R5. Then NEWCOM executes the instruction by branching to its Command Subroutine whose address is an entry in COMADD.

```
JMP  @COMADD(R0)
```

Table 11.1: Listing of Command Subroutines

Command Number	Mnemonic of Command	Name of Command	Page where Listed in POL	Assembly Subroutine	Entry Points
Memory Management Routines					
11	F:NUL	FNUL	16		
10	F:OPEN	FOPEN	16		
8	F:ASSIGN	FASSGN	17		FCLR
9	F:CLEAR	FCLR	17		
6	P:ASSIGN	PASSGN	18		
7	P:NUL	PNUL	18		
Disc Management Routines					
3	P:GET	PGET	19		END
4	P:PUT	PPUT	19		
5	P:RID	PRID	19		
0	F:GET	FGET	20		
1	F:PUT	FPUT	20		
2	F:RID	FRID	20		
Storage Control					
15	END	END	19		
14	START	START	23		
17	REENTER	RENTER	23		
16	IGNORE	IGNORE	24		LASTFL
32	S:FILE	SFILE	24	CHFL2	
33	I:FILE	IFILE	24	CHFL1	
30	S:STORE	SSTORE	24	CHSST	ISTORE
31	I:STORE	ISTORE	24	CHIST	
Program Execution Control					
12	EXECUTE	EXECUT	25		
13	ABORT	ABORT	25		
36	LOOP	LOOP	25		
37	CONT	CONT	25		
Commands that Read Data					
38	R:AD	RDAD	28	CHAD	DECTIM
39	R:PC	RDPC	28	CHPC	
40	R:CLOCK	RDCL	28		
41	R:VM	RDVM	29	CHKEY	

Commands that Control Data Acquisition

34	W:SWITCH	WSWTCH	30		
35	W:CLOCK	WCLOCK	30	CHCL	
42	S:DA	SDA	30	CHDA	
43	I:DA	IDA	30	CHDA	

Graphics Control

28	PUSH	PUSH	32	CHFL3	RFRESH
29	POP	POP	32		
26	X:MOVE	XMOVE	32	CHMOVE	
24	X:EXPAND	XEXP	32		
27	Y:MOVE	YMOVE	32	CHMOVE	
25	Y:EXPAND	YEXP	32		
22	S:CURSOR	MOVCRC	32		
23	PLOT	PLOT	33		GRAPH

Listing Control

21	LIST	LIST	37		
20	PROGRESS	PROG	38		
19	PRINT	PRINT	38		
18	WRITE	WRITE	39		

Numerical Routines

58	LOAD	LOAD	42		
60	SUB	SUB	42		ADD
59	ADD	ADD	42		
61	MUL	MUL	42		
62	DIV	DIV	42		

Data Analysis Routines

44	C:COPY	CCOPY	48	CHFL3	
45	C:ADD	CADD	48	CHFL3	
46	C:SUB	CSUB	48	CHFL3	
47	C:DIV	CDIV	48	CHFL3	
52	EXPAND	EXPAND	49		
51	OFFSET	OFFSET	49		
50	DAEXP	DAEXP	49	DALAB	
49	DAOFF	DAOFF	49	DALAB	
53	VTOT	VTOT	50		
54	INTE	INTE	51		
55	DIFF	DIFF	51		
57	PEAK	PEAK	52		
56	SMOOTH	SMOOTH	53		
48	DECON	DECON	54	CHFL3	

The paths from the Foreground and Background monitors, FMON and BMON, to NEWCOM are somewhat devious. Instructions in Foreground Programs are executed by FMON directly, calling NEWCOM. When an Immediate Instruction is processed, BMON calls the command assembler, COMASS. Following assembly, COMASS branches to the entry point EPIMME in NEWCOM. Background Programs begin processing when the EXECUTE command, an Immediate Instruction, is serviced. Then the Command Subroutine for this instruction calls NEWCOM.

After one instruction in a Background Program completes processing, the instruction returns control to NEWCOM through a JMP statement. The next instruction in the Program will then be serviced. The Program executes until an END instruction is detected. Then control returns to BMON through an RTS PC statement.

Foreground Programs execute similarly, with control returning to FMON after an END instruction is processed. The execution of a Foreground Program is often suspended for an Interrupt Period when one of the conditions in Table 7.3 occurs. Control returns to FMON through an RTS PC statement. In the next Interrupt Period, processing resumes at the point of previous termination.

If a command can be executed either as an Immediate Instruction or in a Background Program, NEWCOM is entered after processing. By examining the variable PROGNO, NEWCOM recognizes when the command was executed as an Immediate Instruction. Then control returns to BMON through an RTS PC statement.

When an error is encountered in the processing of a Background Program or an Immediate Instruction, a software interrupt is generated

with the TRAP statement. The Management Subroutine TRAP services the interrupt by printing an error message on the Decwriter and passing control to BMON. Execution of the Background Program or Immediate Instruction is prematurely suspended through this action.

The TRAP statement flags errors in assembly but never ends assembly. No TRAP statements occur in Command Subroutines called from Foreground Programs. Only when a File-referencing statement numbers a File that is assigned no core is the execution of a Foreground Program prematurely suspended. No error message is generated, but the end of processing is signalled when a "STOP" message appears on the Decwriter.

Before a Command Subroutine is called, NEWCOM loads information into the Registers R0 to R5 as follows. R0 contains twice the command number. As listed in Table 11.1, each POL instruction is associated with a number from 0 through 62. If a Foreground or Background Program is executing, R1 points to the next instruction that will be processed. R2 is loaded with the address of the File numbered by the last File-referencing statement either in the Program or entered as an Immediate Instruction. The File number is obtained from PRFILE and the address is an entry in FLBUFF. R3 and R4 hold the first and second operands of the instruction. Twice the number of the executing Program is placed in R5. Although #-1 is usually associated with an Immediate Instruction, #0 is loaded into R5 for both Background Programs and Immediate Instructions. Foreground Programs are numbered 1 through 4. In general, R0 to R5 are destroyed when a Command Subroutine executes. Occasionally, POL routines other than NEWCOM call Command Subroutines.

These routines set up some or all of the Registers R0 through R5 so that the necessary parameters are passed to the Command Subroutine.

Command Subroutines interact by accessing and modifying Reserved Addresses. To control peripherals, these subroutines manipulate Peripheral Addresses or call Program Requests. Table 11.2 catalogs the Reserved and Peripheral Addresses that are accessed and/or modified by each Command Subroutine. This table also notes the Management Routines, Function Rutroutines, and Program Requests which support each Command Subroutine. Labels for Program Requests are prefixed by a period (.), for this is the format of the Labels in the POL listing. If the Command Subroutine can be entered in a Program, the subroutine branches to NEWCOM after the subroutine runs. Commands that flag errors during execution generate a software interrupt for error detection. This interrupt is serviced by the TRAP routine.

Table 11.3 catalogs the Reserved and Peripheral Addresses that are manipulated by Assembly Subroutines. Routines called from Assembly Subroutines are also listed in this table. R5 is saved by all of these subroutines, but other Registers are typically destroyed.

Adding new Command Subroutines will be the most frequent modification to POL. If the new instruction is used only as an Immediate Instruction, the subroutine should end with an RTS PC statement. Otherwise, it ends with

```
JMP  NEWCOM
```

No TRAP statement should occur in the Command Subroutines called from the Foreground Program. In these cases, all conceivable errors should be detected in assembly. Each new Command Subroutine requires that

Table 11.2: Reserved Addresses, Peripheral Addresses, and Routines Referenced in Command Subroutines

Command Subroutine	Reserved Addresses	Peripheral Addresses	Routines
Memory Management Routines			
FOPEN	AKEY,AFILE,PRFILE		
FASSGN	AKEY,AFILE,LFILE,FLKEY		MANFL,TRAP,GETBLK,FNUL,FCLR
FCLR	FLBUFF,LFILE,AFILE		MANFL,TRAP
FNUL	FLKEY,FLBUFF,LFILE,AFILE,OSTPNT,OSTACK		MANFL,TRAP
PASSGN	PRFILE,PRCNT		MANFL,TRAP,POP
PNUL	PRFILE,PRCNT,PRBUFF		GETBLK,TRAP
			TRAP,DEASS
Disc Management Routines			
PGET	PRFILE,PROGNO,EXAREA,PRCNT		NAME1,TRAP,READC,ERRCHK,END
PPUT	PRFILE		MAME2,TRAP,WRITC
PRID			NAME1,TRAP,DELETE
FGET	EXAREA		MANFL,NAME3,READC,TRAP,ENDFLT
FPUT			MANFL,NAME3,WRITC,TRAP,ENDFLT
FRID			MANFL,NAME3,DELETE,TRAP,ENDFLT
Storage Control			
END	PRFILE,PRCNT		BDSTOP
START	PRFILE,PRCNT,PRBUFF,PROGNO		TRAP
RENTER	PRFILE,PRCNT,PRBUFF,PROGNO		TRAP,LASTFL
IGNORE	PRFILE,PRCNT,PRBUFF		DEASS,TRAP
SFILE	PRFILE		NEWCOM
IFILE	PRFILE		NEWCOM
SSTORE			NEWCOM
ISTORE			NEWCOM

Program Execution Control

LOOP	LPITER,LPRET,PRCNT		NEWCOM
CONT	LPITER,LPRET,PRCNT		NEWCOM
EXECUTE	PRFILE,PRCNT,PRBUFF,PRITER,PROGNO		NEWCOM,BDSTOP,TRAP
ABORT	AKEY,PRCNT		TRAP

Commands that Read Data

RDAD	WRSTAT,RDSTAT,STORE1,STORE2,ADITER PRITER,PRCNT	165740-165756	CHANNL,LOADOP,WCLOCK,NEWCOM
RDPC	WRSTAT,RDSTAT,STORE2,PRITER,PRCNT		LOADOP,DECTIM,NEWCOM
RDCL	RDSTAT,TIME		LOADOP,NEWCOM
RDVM	RDSTAT	165222-165226	LOADOP,TMSTEN,NEGATE,NEWCOM

Commands that Control Peripherals

WSWTCH	PRCNT	165230	
WCLOCK	PRITER,PRCNT		NEWCOM
SDA	WRSTAT,WIDEDA,NARRDA		CHANNL,DALAB,NEWCOM
IDA	WRSTAT,WIDEDA,NARRDA		CHANNL,DALAB,NEWCOM

Graphics Control

PUSH	OSTPNT,OSTACK,PRFILE		TRAP,GRAPH,NEWCOM
POP	OSTPNT,OSTACK		RFRESH,NEWCOM
XMOVE	OSTPNT,OSTACK		RFRESH
XEXP	OSTPNT,OSTACK		RFRESH
YMOVE	OSTPNT,OSTACK		RFRESH
YEXP	OSTPNT,OSTACK		RFRESH
MOVCRS	CURSOR		RFRESH,NEWCOM
PLOT	GRSTAT,OSTPNT,OSTACK,CURSOR,POINTS	165230	GSETUP,DETPLX,DETPLY,PANIC,LOADPT

Listing Control

LIST	PRBUFF,COMNAM,IOBUFF,COMFL1,OUTPNT,FLKEY	MULT,BINDOC,DMPLN
PROG	LPITER,IOBUF	BINDOC,DMPLN
PRINT	CURSOR,POINTS	WSETUP,DMPMES,DETWRX,DETWRY, DMPLN,NEWCOM
WRITE	OSTPNT,FLBUFF,YSTART,OUTPNT,OSTACK, POINTS	WSETUP,DMPMES,DETWRX,DETWRY DMPLN,DIVID,NEWCOM

Numerical Routines

LOAD	MANT,EXPON	
SUB		ADD
ADD	MANT,EXPON	TMSEXP,PRANS
MUL	MANT,EXPON	MULT,PRANS
DIV	MANT,EXPON	MULT,DIVID,PRANS

Data Analysis Routines

CCOPY		NUMTWO,TRAP,NEWCOM
CADD		NUMTWO,TRAP,TESTOP,NEWCOM
CSUB		NUMTWO,TRAP,TESTOP,NEWCOM
CDIV	SHIFT	NUMTWO,TRAP,TESTOP,DIVID,NEWCOM
EXPAND	SHIFT	NUMONE,TRAP,DIVID,MULT,NEWCOM
OFFSET	RDDP,SHIFT,RDFACT	NUMONE,TRAP,TMSEXP,DIVID,NEWCOM
DAEXP		TRAP,NEWCOM
DAOFF		TRAP,NEWCOM
VTOT	SHIFT,TEMP	NUMONE,TRAP,MULT,NEWCOM
INTE	CURSOR	NUMONE,TRAP,NEWCOM
DIFF		NUMONE,TRAP,NEWCOM
PEAK	CURSOR,IOBUF	NUMONE,TRAP,DIVID,DETWR,DMPLN,NEWCOM
SMOOTH	CONFAC	NUMONE,TRAP,CONVOL,NEWCOM
DECON	DECFAC	NUMONE,TRAP,CONVOL,NEWCOM

Table 11.3: Reserved Addresses, Peripheral Addresses, and Routines Referenced by Assembly Subroutines

Assembly Subroutine	Reserved Addresses	Peripheral Addresses	Routines
CHMOVE	SHIFT,INSTR		DIVID
CHFL1	PRFILE		TRAP,CHFL
CHFL2	PRFILE		CHFL,TRAP
CHFL3			CHFL,TRAP
CHIST	PRCNT,INSTR		TRAP
CHSST	INSTR		
CHPC	INSTR	164040,164042, 164054,164056	CHCL,CHKEY,TRAP
CHAD	INSTR		CHCL,CHKEY,TRAP
CHKEY	RDKEY,PROGNO		TRAP
CHCL	INSTR		
CHDA	INSTR	164000 - 164036	CHCODE,CHTRAN,CHLAB,CHKEY,TRAP
CHLAB	INSTR		TMSTEN,CHCODE
CHCONT	PRCNT,PRBUFF,LPITER,LPRET		TRAP,CHFL1,CHFL2

one entry be appended to the tables: COMADD, COMNAM, LIMIT1, LIMIT2, COMFL1, and COMFL2. These addresses, discussed in § 10.1, are accessed when an instruction assembles or executes.

When Command Subroutines are written, the programmer may wish to include calls to existing Function Subroutines from the new subroutine. For example, the subroutines MULT and DIVID are called to multiply and divide two numbers. In §11.3, all Function Subroutines are described with emphasis on the input parameters, the function, and the output parameters of the subroutine. A programmer should be able to use these subroutines by reading their brief description and without having to decipher the code of the subroutine.

Note several unimportant entry points occur in the Command, Function, and Management Routines of POL. Most of these entry points, listed in Table 11.4, allow small sections of code to be performed by several Programs. With these Labels, the length of POL has been reduced by over 250 words. Other entry points in Table 11.4 would have been labeled as local symbols, but could not be since the Local symbol block was interceded and terminated by the definition of a non-local symbol. (See Chapter 5 of the RT-11 Manual for a discussion of local symbols.) These entry points are marked by an asterisk (*). None of the Labels in Table 11.4 will be included in the flow charts of § 12.3. Instead, the function of the brief segment of codes is described repetitively, or these labels are treated as local symbols.

Table 11.4: Miscellaneous Entry Points

Entry Point	Page of Listing
EXITC1	5
NEWSEC*	7
EXITC2	11
E06C	13
E04C	13
DIVOP	13
LOADIN	13
CHEXIT	13
EXITM1	18
E06M	18
GDEND*	19
BDEND*	19
E06T	20
E07T	22
STASS	23
E06P	23
SKIPFL*	24
EXITF1	24
EXITH3	28
EXITH4	29
EXITH1	30
EXITH2	30
EXITG1	34
DETHOP*	35
EXITW2	39
EXITW1	40
MULEP*	42
EXITN1	43
EPHOP*	47
EXITA1	47
EL0A	47
EXITA2	48
DAEP*	49
EXITA3	52

* These symbols are used in place of local symbols.

11.2 Management Routines

Management Routines will seldom be changed. However, the programmer who modifies POL must have a fundamental understanding of these routines, for they schedule the assembly and execution of instructions. Although a programmer should examine the flow charts of Management Routines, their general function is described here to further his understanding. The POL operating system has six Management Routines: POLEP, BMON, FMON, COMASS, NEWCOM, and TRAP. Most were introduced in the previous section.

When the POL operating system begins executing, POLEP is entered to setup certain locations in core and the RT-11 operating system. This routine is executed only once and is written over with Files and Programs. POLEP moves the RT-11 program USR from disc into high core. This program services the Program Requests associated with an EMT interrupt. Then POLEP clears Variable Reserved Addresses from #PRBUFF through #IOBUF and all Status Registers, except 164700 through 164716. These are set up as noted in § 9.2 so that the ADC Channels are read synchronously. Finally, the interrupt vectors for the TRAP statement and Hardware Interrupt are loaded. After POLEP executes, control is passed to BMON.

The most essential function of the Background monitor, BMON, is to schedule the Background processes of: assembling programs, and executing Immediate Instructions and Background Programs. All these processes are accomplished through a call to the command assembler, COMASS. After an instruction is assembled, or an Immediate Instruction

or a Background Program has completed executing, BMON is re-entered. BMON also periodically refreshes the display scope and prints "STOP" messages after programs cease executing. Until a Hardware Interrupt is generated or a Background function requires attention, BMON is processed iteratively, i.e., in a loop. The entry point PANIC is called by Command Subroutines that react to the "Panic" button. When this button is depressed, control passes from the subroutine DMPLN or PLOT to BMON. Otherwise, the subroutine continues to process. BDSTOP is entered from TRAP after an error is flagged on the Decwriter. Execution, but not the assembly, of any Program will cease when an error is detected. If execution is suspended either normally or by an error, control is passed from an Immediate Instruction or a Background Program to BMON. Then or when an END instruction terminates assembly, BMON again accepts Immediate Instructions. Remember, the START and REENTER instructions, both Immediate Instructions, initiate assembly by moving their second operand into PROGNO.

When a Hardware Interrupt is generated, Background processes temporarily cease to execute. The Foreground monitor, FMON, services the Interrupt. R0 through R5, and the variables SHIFT and PROGNO are saved on entry to FMON. These are restored prior to exit. The principal task of FMON is executing Foreground Programs, accomplished by changing the variable PROGNO and calling NEWCOM. When a condition in Table 7.3 is encountered, the execution of a Foreground Program is suspended for an Interrupt Period. FMON also monitors synchronously controlled Peripherals – pulse counters, ADC Channels, and DAC Channels.

Using the Fixed Reserved Addresses (COMNAM, LIMIT1, LIMIT2, COMFL1, and COMFL2) the command assembler, COMASS, assembles instructions entered on the Decwriter. This routine detects errors in the name of instructions, in the range of their operands, in the ordering of statements, and in the usage of statements in different Programming environments. By calling ERRCHK, which branches to an Assembly Subroutine listed in COMERR, COMASS flags order and allocation errors that are unique to an instruction. The second operand also may be modified by an Assembly Subroutine. After an instruction is assembled, it is either stored in a Program, whose number is given by PROGNO, or executed as an Immediate Instruction, if PROGNO is #-1. The latter task is accomplished by branching to NEWCOM through the entry point EPIMME.

The Management Routine NEWCOM loads arguments of a Command Subroutine into R0 through R5 as noted in the previous section. When an END statement is encountered or a File-referencing statement numbers a File that is assigned no core, NEWCOM terminates the processing of a Program. Through the table STOP, BMON later recognizes that execution has ceased. Then a "STOP" message appears on the Decwriter. After a Command Subroutine processes an instruction in a Program, NEWCOM is re-entered to set up the next instruction for execution.

TRAP services errors flagged by the TRAP statement. Errors are detected during both the assembly and execution of instructions, but only Background Programs and Immediate Instructions produce errors in execution. TRAP selects a message from the table ERROR which is printed on the Decwriter. Afterwards, BMON is entered through BDSTOP.

Table 11.5: Listing of Mangement Routines

Name of Management Routine	Page where Listed in POL	Entry Points
BMON	5	PANIC,BDSTOP
COMASS	6	
NEWCOM	11	EPIMME
FMON	26	
TRAP	38	
POL	56	

Table 11.6: Reserved Addresses, Peripheral Addresses, and Routines Referenced by Management Routines

Management Routines	Reserved Addresses	Peripheral Addresses	Routines
BMON	GRSTAT,PROGNO,INSTR, PRFILE,PRCNT,STOP, IOBUF,TIME		CLRBUF,BINDOC,DMPLN, GRAPH
CIA	IOBUF,COMNAM,INSTR, COMFL1,LIMIT1,LIMIT2, COMFL2,PRFILE,PROGNO PRCNT,PRBUFF		CLRBUF,.TTINR,MULT, SEARCH,SPSRCH,AS ASSEM,TRAP,ERRCHK, EPIMME
NEWCOM	PROGNO,PRFILE,PRCNT, FLBUFF,COMADD,STOP		all Command Subroutines
FMON	SHIFT,PROGNO,TIME, STORE1,STORE2, ADITER,RDSTAT, WRSTAT,WIDEDA NARRDA	165040, 165042, 165700-165716, 165000-165036	DIVID,NEWCOM
TRAP	ERROR		DMPMES,BDSTOP
POL	Most Variable Reserved addresses	164000-164776, 34,36, 270,272	BMON,.SRESET,.LOCK

Figure 11.1: How Control is Passed between Management Routines

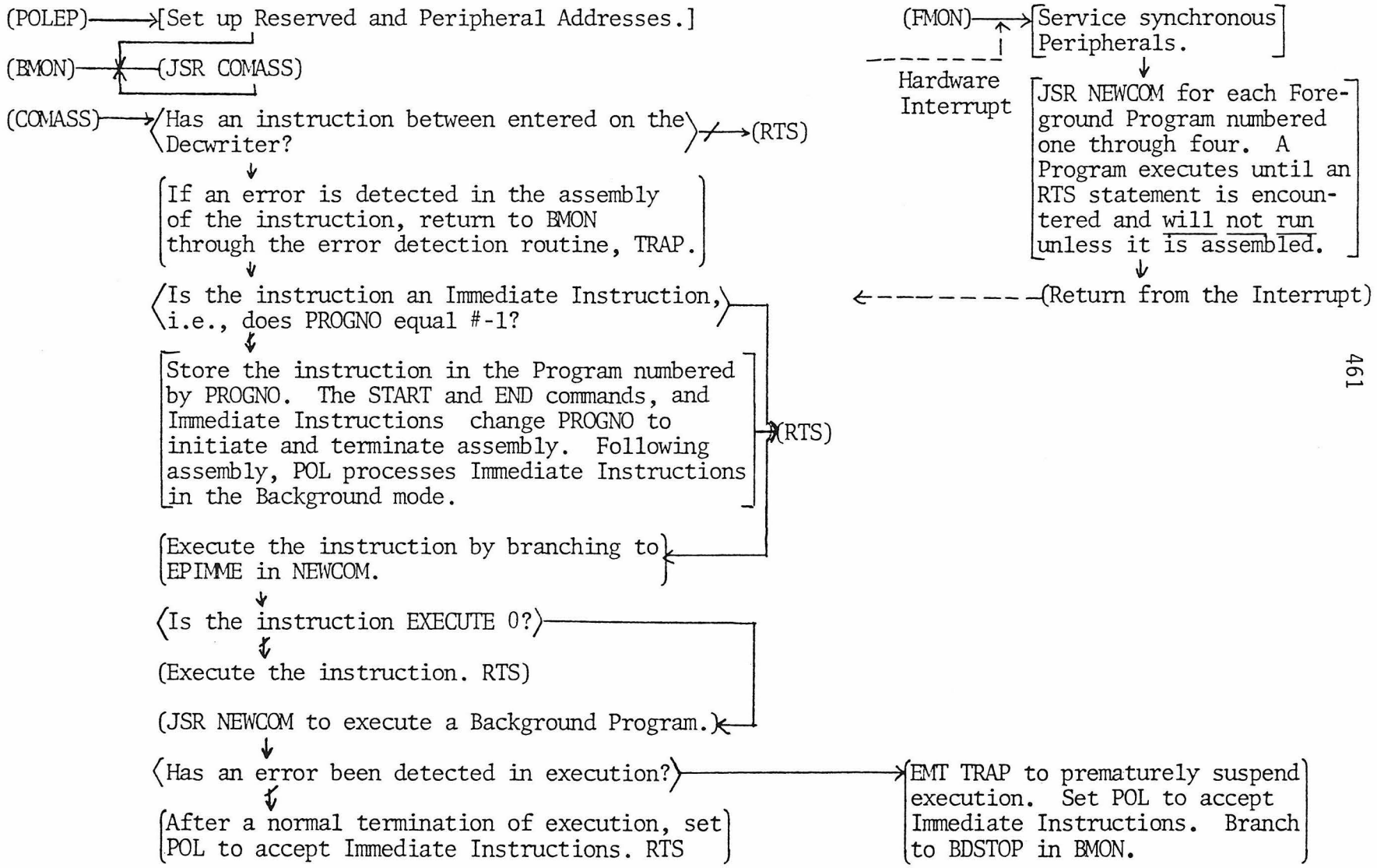


Table 11.5 lists Management Routines and their entry points. Noted in Table 11.6 are the Reserved and Peripheral Addresses manipulated by Management Routines, and routines and Program Requests called by Management Routines. Figure 11.1 summarizes how control passes between Management Routines to schedule Foreground and Background processes.

11.3 Function Subroutines

Function Subroutines perform operations required in the execution of many POL routines. These Subroutines and their entry points are listed in Table 11.7.

Information is passed most commonly between Function Subroutines and the calling routine through Registers R0 to R5. Some Registers are not modified by certain Function Subroutines. Table 11.8 catalogs the Registers that pass information to, pass information from, and are not modified by each subroutine. Note that occasionally Registers pass information to a Function Subroutine, yet their contents are saved. Reserved and Peripheral Addresses are often manipulated by Function Subroutines. These addresses and routines that support each Function Subroutines are cataloged in Table 11.8

When a programmer writes new Command Subroutines for POL, they will often call existing Function Subroutines. For this purpose, the remainder of this section describes what each Function Subroutine does and what addresses and registers it manipulates. From these descriptions, the programmer should be able to use these subroutines without referring to the flow charts in § 12.3. However, he must be familiar with the Reserved Addresses, described in Chapter 10, before he reads this section.

Except for DETSGN (q.v.), all Function Subroutines are called with the statement

```
JSR  PC, name
```

Table 11.7: Listing of Function Subroutines

Name of Function Subroutine	Page of Listing	Entry Points
Assembly Subroutiees		
ERRCHK	9	
ASSEM	10	
SEARCH	11	SPSRCH
CHFL	12	
CHCODE	13	
CHTRAN	13	
Management Subroutines		
GETBLK	16	
MANFL	20	
ENDFL	20	
NAME	21	NAME1,NAME2,NAME3
DELETE	22	
CLOSE	22	
READC	22	
WRITC	22	RWLOOP
FINISH	22	
DEASS	23	
Data Accumulation Subroutines		
LOADOP	27	
DALAB	31	
CHANNL	31	
Data Display Subroutines		
LOADPT	34	
LDAXIS	34	
DETPLX	35	
DETPLY	35	DETPL1,DETPL2
WSETUP	35	GSETUP
XSETUP	40	
YSETUP	40	SETUP
DETWRX	40	
DETWRY	40	DETWR
DMPLN	41	DMPMES,CLRBUF
PRANS	41	

Numerical Subroutines

MULT	43	TWSTWO,NEGATE
DIVID	44	
DETSGN	44	
BINDEC	45	BINDOC
TMSTEN	46	
TMSEXP	46	

Data Analysis Subroutines

NUMONE	47	NUMTWO
TESTOP	47	
CONVOL	55	

Table 11.8: Registers, Reserved Addresses, Peripheral Addresses, and Routines Referenced by Function Subroutines

Function Subroutine	Input Registers	Output Registers	Saved Registers	Reserved Addresses	Peripheral Addresses	Routines
Assembly Subroutines						
ERRCHK	1-4		5	PROGNO		all Assembly Sub-routines
SEARCH	2-4	2	1,3-5	IOBUF		
ASSEM	1,2	1,2	0,5	IOBUF		SEARCH,TMSTEN,NEGATE
CHFL	2		0-3,5	PROGNO,AKEY,AFILE		TRAP
CHCODE	1-3	1,2	3-5	WRSHFT,SHIFT,WRFACT		DIVID
CHTRAN	1,2,4	1,2	3-5	WRSHFT,SHIFT,WRFACT		MULT
Management Subroutines						
GETBLK	3,4	2,3	4,5	FLBUFF,PRBUFF, core from #POLEP to 113050		
MANFL	3,4	0,1,3	2,4,5	TRNAME,AFILE,LFILE,AKEY, PRCNT,FLKEY,AFILE		TRAP
ENDFLT			1-5	AFILE,LFILE,AKEY		
NAME	3	2-5	0,1	PRBUFF,TRNAME,AFILE,FLBUFF, EXAREA,RQAREA		TRAP
DELETE	4		2-5			.DELETE,TRAP
CLOSE		2-5				
READC	3-5		2-4	FINIS		.LOOKUP,.READC,CLOSE TRAP
WRITC	3-5		2-4	FINIS		.LOOKUP,.ENTER,.WRITC, RWLOOP,TRAP
FINISH				FINIS		
DEASS				RDKEY,WRKEY	164040,164042 164000-164036	

Data Accumulation Subroutines

LOADOP	1,2	2	1,3-5			
DALAB	3-5	1,3-5	0,2			
CHANNL	1,3	0	2-5	PROGNO		

Data Display Subroutines

LOADPT	1,2,4		1-5	PROGNO	165234,165236	LDAXIS
LDAXIS	0		0,5	GRSTAT	165236	
DETPLX	3,5	2	3	XSTART,POINTS,SHIFT		XSETUP,MULT,DETPL1, DETPL2
DETPLY	3,5	2	3	SHIFT,RDFACT		YSETUP,MULT,DIVID
WSETUP	5	0,3-5		OSTPNT,YSTART,FLBUFF,POINTS, RDLAB,WRLAB,XSTART,LABTAB		
XSETUP	3	0-2,4	3,5	XSTART,YSTART		MULT
YSETUP	3	1,2,4	3,5	YSTART		
DETWRX	3	1,2,4	3,5	OUTPNT		XSETUP,DETWR,CHTRAN, DIVID
DETWRY	3	1,2,4	3,5	RDFACT,SHIFT,RDDP		YSETUP,MULT,BINDEC
DMPMES	0,4,5		3	OUTPNT,IOBUF		DMPLN
DMPLN			2-5	OUTPNT,IOBUF		PANIC,.TTOUTR,CLRBUF
CLRBUF		1	0,2-5	OUPPNT,IOBUF		
PRANS	1-3			MANT,EXPON,IOBUF		DIVID,BINDEC,BINDOC, DMPLN

Numerical Subroutines

MULT	0-2	0-2	3-5		DETSGN
DIVID	0-2	0-2	3-5		DETSGN,TMSTWO
DETSGN	1-6	1-6	0	SHIFT	
BINDEC	0-2		3-5	OUTPNT,IOBUF,TENS	NEGATE
TMSTEN	1,2	1,2	0,3-5		
TMSEXP	2,3	1,2	4,5		TMSTEN,DIVID

Data Analysis Subroutines

NUMONE	2-4	0,1,3-5		AKEY,PRCNT,FLBUFF	TRAP
TESTOP	0		0-5		TRAP
CONVOL	3-6	3-5		SHIFT	MULT

where name is an entry point to the subroutine. A Function Subroutine does not flag errors unless it can call the TRAP routine. Table 11.8 shows which Function Subroutines call TRAP.

Function Subroutines are described in the order of their usefulness to the programmer. These subroutines are categorized as numerical, data analysis, data display, data accumulation, memory and disc management, and assembly subroutines. Multiplication, division, and other manipulations of numbers are performed by numerical routines. Other types of Function Subroutines support the processes implied by their names. Particularly useful subroutines or entry points are marked with an asterisk (*). Unless a limitation is noted, Registers that pass arguments to Function Subroutines are not restricted in their range.

MULT*,DIVID*,NEGATE*,TMSTWO,DETSGN

MULT (DIVID) multiplies (divides) the number in <R1,R2> by the number in R0. R1 always contains the most significant word of the Register pair <R1,R2>. After the product or quotient is obtained, it may be multiplied by a factor of two. SHIFT, a positive or negative variable loaded by the calling routine, gives the exponent. If SHIFT is not loaded, this variable defaults to zero. The result of all manipulations is passed to the calling routine through <R1,R2>. R0 is cleared after a multiplication but retains the value of the divisor following a division. SHIFT is always cleared upon return.

MULT and DIVID call DETSGN through the statement

```
JSR  R3, DETSGN
```

DETSGN saves registers R3 through R5 and determines the sign of the product or quotient. Multiplications and divisions are performed after R0 and <R1,R2> are made positive. DETSGN also moves SHIFT into R5 and clears SHIFT, R3, and R4. This subroutine should not be used by a programmer unless he inspects its code carefully. After the multiplication or division is performed, TMSTWO is entered to multiply the product or quotient by the power of two given by SHIFT. By branching to NEGATE, TMSTWO negates the number in <R1,R2> if DETSGN indicated that the result should be negative. Any routine may call NEGATE to negate <R1,R2>.

BINDEC*,BINDOC*

BINDEC translates the binary number in <R1,R2> into ASCII digits 60 through 71, then loads the result into IOBUF. When this buffer is full, it is printed on the Decwriter by the calling routine. On entry, R0 contains the number of ASCII digits that will appear to the right of the decimal point. OUTPNT, incremented by 13. at the beginning of BINDEC, marks the address in IOBUF following the location where the least significant digit of the decimal number is placed. At most, BINDEC generates nine digits plus the sign and decimal point. If BINDOC is the entry point, the sign of R2 is extended into R1, R0 is set to zero, and then BINDEC is entered.

TMSTEN*,TMSEXP*

TMSTEN multiplies a number in <R1,R2> by ten. TMSEXP multiplies a number in R2 by a power of ten. On entry, R3 gives the exponent

of ten, which can be positive or negative. The result is passed through <R1,R2>.

NUMONE*,NUMTWO*

NUMONE and NUMTWO set up Registers for, and detect errors in, data analysis commands that operate on data Slots. NUMONE is the entry point for commands that manipulate one File; NUMTWO, for two Files. When either of the following conditions is violated, these Function Subroutines flag manipulation errors. No data analysis operation is performed on a File that is concurrently being loaded with data by a Foreground Program. If two files are used in a data analysis operation, the Files must have the same experimental option, or the second File must be empty. Remember that the first File of a data analysis instruction is numbered by a File-referencing statement; the second is numbered by the second operand of an instruction. When the second File is empty, NUMTWO transfers the experimental option and DAC information from the first File into the Label of the second File.

On entry to NUMONE and NUMTWO, R2, R3, and R4 have the values that were inserted by NEWCOM. A Command Subroutine does not modify these Registers before NUMONE or NUMTWO is called. On exit from these subroutines, R5 and R4 contain the addresses of the first slot in the first and second Files. If NUMONE was called, R4 is zero, but R1 holds the second operand of the data analysis command. R3 stores the number of Slots in the first or second File, whichever number is fewer. The experimental option of the Files is returned in R0.

TESTOP

If the experimental option in R0 does not correspond to intensity data, TESTOP flags a manipulation error. Intensity data have experimental options between three and eight, inclusive, or greater than thirteen.

CONVOL

CONVOL convolutes, or sums, a series of Slots are consecutive locations in a single File. On entry, R5 marks the Slot succeeding the highest order Slot in the convolution. Summing takes place from Slots of higher to lower number so that R5 marks the first Slot in the series on return. The products of each Slot and the weighting factors are added to <R3,R4>. If R5 marks a Slot past the end (beginning) of a File, the last (first) Slot gives the value of the data point.

Three parameters are passed on the stack to CONVOL. They are removed from the stack before CONVOL is exited. The first (first in) is a negative exponent of two, which multiplies each weighting factor so that the sum of the weighting factors is normalized to one. The next is the address succeeding the table of weighting factors. Entries in the table are accessed from higher to lower addresses. The final entry on the stack notes the number of Slots that are convoluted.

YSETUP*,XSETUP*,SETUP

When YSETUP is called, R3 numbers a Slot in the ordinate File. The contents of the Slot are placed in <R1,R2> while the experimental option of the File is moved into R4. Afterwards, control returns to

the calling routine. If an abscissa File exists, XSETUP loads <R1,R2> and R4 from the abscissa File, as YSETUP did for the ordinate File. In this case, XSETUP branches into YSETUP through SETUP. Else the abscissa is produced from DAC information in the label of the ordinate File. The DAC setting for a Slot numbered by R3 is generated from the initial setting s and increment i of the DAC Channel by the algorithm: $s + R3 \cdot i$. This result is placed in <R1,R2> and the Channel number is stored in R4. Only if no abscissa File is available is R0 zeroed when XSETUP is exited.

DETPLY*,DETPLX*,DETPL1,DETPL2

Through DETPLY and DETPLX, data and DAC settings are translated into 10.-bit numbers that will be loaded into the graphics DAC. The expansion and zero-point are accounted for in the 10.-bit code. To calculate the coordinates of a point, DETPLY is called, then DETPLX. On entry to DETPLY, R5 marks the first address of the entry on OSTACK being plotted. Six is added to R5 while DETPLY executes. The contents of this Register should not be changed before DETPLX is called. While either DETPLX or DETPLY executes, R3 numbers the Slot in the ordinate and abscissa Files that is being plotted. Upon return from either subroutine, R2 contains the code for the graphics DAC. If R2 is positive, the ordinate or abscissa exceeds the range of the graphics DAC and must not be plotted.

To calculate the code for the ordinate, DETPLY calls YSETUP. The datum in <R1,R2> is adjusted by RDFACT to scale the ordinate for the particular experimental option (see § 10.1). Then the final value for

R2 is calculated by adjusting <R1,R2> by the zero-point correction and expansion factor. DETPLX determines the code for the abscissa as DETPLY calculated the code for the ordinate. First DETPLX calls XSETUP to load <R1,R2>. If an abscissa File exists, DETPLX branches to DETPL1 to finish coding the abscissa. Otherwise, the DAC setting in <R1,R2> is divided by 40. to scale it for plotting. Then DETPLX branches into DETPLY through DETPL2 to adjust <R1,R2> for the zero-point correction and expansion factor. When no abscissa File or DAC information is available, R4 is negative on return from XSETUP. In this case, data points will be plotted with equal spacings in the abscissa. The full range of the x-axis on the graphics media is used.

DETWRY*,DETWRX*,DETWR

DETWRY calls YSETUP to move the contents of a Slot from the ordinate File into <R1,R2>. With RDFACT, this datum is transformed into units that are accessible to the user (see § 10.1). The number of decimal places of the datum, which depends on the experimental option, is moved into R0 from RDDP. After DETWRY branches to BINDEC to translate <R1,R2> into decimal digits and load them in IOBUF, control returns to the calling routine.

Similarly, DETWRX translates a Slot from the abscissa File into decimal digits and places them in IOBUF. After XSETUP is called, DETWRX branches to DETWR in DETWRY. If no abscissa File exists, the DAC setting in <R1,R2> is converted from the code for loading the DAC into units of the controlled parameter in an experiment. For this purpose, CHTRAN is called. This result always has two decimal places

so that R0 is set to #2. Finally BINDEC is entered. When no DAC information exists in the ordinate File and no abscissa File is given, OUTPNT is incremented by 13. from DETWRX, i.e., the abscissa is left blank.

DMPLN*,CLRBUF*,DMPMES

Through the Program Request .TTOUTR, DMPLN prints the ASCII characters in IOBUF on the Decwriter. This subroutine appends a carriage-return, line-feed to IOBUF. R0 and R1 contain no information on entry, but are destroyed by DMPLN. Before any character is passed to .TTOUTR, this Function Subroutine calls PANIC to check if the "Panic" button has been depressed. After all characters in IOBUF have been printed, this subroutine branches to CLRBUF, which moves blanks, ASCII #40, into IOBUF. R1 contains #IOBUF on exit. CLRBUF modifies R1 only and may be called as a Function Subroutine.

DMPMES is entered to print column headings or error messages on the Decwriter. R4 and R5 point to the addresses following a string of ASCII characters from LABTAB or ERROR that are moved into IOBUF. The string marked by R4 is entered first. Five characters in each string are listed. R0 notes the number of characters that separate the ends of the two strings, and the end of the first string and the left-most character in IOBUF. R0, R4, and R5 are destroyed. DMPMES branches to DMPLN which prints IOBUF on the Decwriter.

PRANS

The mantissa in <R1,R2> and the exponent in R3, resulting from calculator commands, are translated into decimal characters and printed on the Decwriter by PRANS. The mantissa is reduced to four significant digits; the exponent is adjusted accordingly. These values of the mantissa and exponent are stored in MANT and EXPON. Through calls to BINDEC and BINDOC, these numbers are moved into IOBUF. The mantissa prefixes the exponent, and the character "E" separates them. PRANS is exited through a branch to DMPLN, which prints IOBUF on the Decwriter.

LOADPT*,LDAXIS

LOADPT plots a point on the display scope or point-plotter. If bit 3 of GRSTAT is set (clear), the point is graphed on the point-plotter (display scope). When LOADPT is called, R1 and R2 hold the ordinate and abscissa, which are coded for loading into the graphics DAC. R4 has bit 0 or 1 set to enable the point-plotting pen or turn on the intensity of the display scope, respectively. LOADPT calls LDAXIS, which enables the multiplexer on the graphics DAC. On entry to LDAXIS, R0 holds #GRSTAT.

LOADOP

Command Subroutines that read data call LOADOP. On entry, R2 contains the address of the File where the datum will be stored. When the storage pointer in the File label marks a Slot past the end (beginning) of the File, the storage pointer is reset to the address of the first (last) Slot in the File. LOADOP also places the experimental

option, entered in R1, into the label if no option has been loaded previously. On return, R2 marks the address succeeding that of the storage pointer in the Label.

DALAB

DALAB separates the second operand of the S:DA or I:DA instruction, in R4, into two values that will set the wide and narrow converters of the DAC. The value for the narrow converter is returned in R5. It is obtained by moving the 4 least significant bits of R4 into R5 and setting the 12 most significant bits. The value for the wide converter, returned in R4, is calculated by dividing R4 by 16 and ignoring the remainder. If the subroutine IDA called DALAB, the sign of the 11th bit is extended into the 4 most significant bits of R4. Else, the 4 most significant bits are set.

R4 and R5 are combined through the algorithm:

$$-[R4 + 11. + (10.\cdot R5)]$$

and returned in R1. R1 is placed in the Label as the initial setting or increment of the DAC Channel. Just prior to return, the Channel number, in R3, is multiplied by two.

CHANNL

CHANNL checks WRSTAT or RDSTAT to determine if one of the following two conditions has been violated. No more than two DAC or ADC Channels may be loaded or accessed in a single Interrupt Period from a Foreground Program. Any DAC or ADC Channel may be manipulated once in

an Interrupt Period. If either condition is violated, processing of the Foreground Program is terminated for the Period. Otherwise, WRSTAT and RDSTAT are modified to show which, and how many Channels have been used (see § 10.2). R0 returns the number of a Subinterval in which the ADC Channel will be read.

GETBLK*

To allot space for a File or Program, GETBLK searches core from #POLEP through 113050 for unused space. On entry, R4 holds the number of words to be allocated to the record. An additional word is reserved so that the number of words in a record prefixes its contents. R3, which is doubled on exit, contains a File number, or a Program number minus 5. If GETBLK finds space for the record, the address of the record is stored in FLBUFF or PRBUFF, then returned in R2. Otherwise R2 returns with #0. On exit, the Z bit in the Precessor Status Word is one (zero) if the search was (un)successful.

MANFL

Memory and disc management routines involving Files call MANFL to check if all Files in the sequence have the same Key as the variable AKEY. (Remember, these routines operate on a sequence of Files.) If any Key does not, or a File is out of the range of legal File numbers, an allocation error is flagged. Then AFILE and AKEY are cleared. On entry, R3 contains the number of Files in the sequence, while AFILE numbers the first File in the sequence. R3 mimics this variable on exit; R1 returns the sum of AFILE and the number of Files in the

sequence. MANFL loads LFILE with R1 and the alphabetic name of TRNAME, from R4, if necessary.

ENDFL

ENDFL is called by disc management routines involving Files to check if the disc I/O operation has been performed on all Files in the sequence. This Function Subroutine increments AFILE by one upon each entry. When all Files have been manipulated, AKEY and AFILE are cleared. Then control returns to the routine that called the disc management routine, BMON. Otherwise, the disc management routine is re-entered.

NAME,NAME1,NAME2,NAME3

All disc management routines call NAME. On entry, R3 contains the number of the Program that is transferred in the disc I/O operation. If a File is transferred, AFILE holds the File number. NAME calculates the numeric name from R3, or AFILE, then loads it into TRNAME. The alphabetic name of a Program is also stored in TRNAME. If the mangement routine transfers a record to or from core, the buffer address and word count are loaded into EXAREA. On exit, the number of Slots in a File is placed in R2, if necessary. R3, R4, and R5 hold #EXAREA, #RQAREA, and #3, respectively. The entry point for management routines involving Files is NAME3; NAME2 is called by PRID; and PGET and PRID enter NAME1.

DELETE*,CLOSE*,READC*,WRITC*,RWLOOP,FINISH

The Function Subroutines marked by an asterisk setup and execute the Programmed Requests associated with their names. READC also performs the .LOOKUP request so that a record can be read from disc into core. WRITC also executes the .ENTER request to enter a new record on disc. Before any of these routines is called, EXAREA, RQAREA, and TRNAME must be filled. R3, R4, and R5 hold #EXAREA, #RQAREA, and #3 on entry. Only R5 is modified by these subroutines. Depending on the subroutine, transfer and/or hardware errors may be flagged. RWLOOP executes a .READC or .WRITC request three times before a hardware error is flagged. The completion routines for these two requests is FINISH. If no hardware error occurred in the transfer, FINISH clears the variable FINIS. Otherwise, FINIS is set to #-1.

DEASS

DEASS removes Keys in WRKEY and RDKEY that were assigned to Peripherals when a Foreground Program assembled. This subroutine clears Status Registers for a DAC Channel if its Key is removed. On entry, R0 points to a statement in a Program.

ERRCHK

ERRCHK looks up and branches to an Assembly Subroutine whose address is listed in COMERR. These subroutines check for order and allocation errors that are unique to an instruction and/or modify the second operand of an instruction. The result of the modification is stored in INSTR. Assembly subroutines are associated with instructions

numbered 26. through 50., inclusive. For other instructions, ERRCHK immediately returns control to the calling routine by executing RTS PC statement. On entry, R4, R3, and <R1,R2> contain the number, the first operand, and second operand of the instruction.

CHFL

Assembly subroutines call CHFL to determine if a File has the same Key as the number of the Foreground Program which references it. When a Background Program or Immediate Instruction alludes to the File, the Key must match the variable AKEY. (Remember, AKEY is loaded with the first operand of the F:OPEN statement.) Otherwise an allocation error is flagged. On entry, R2 contains the File number.

CHCODE*,CHTRAN*

CHCODE translates the number in <R1,R2> into the 12-bit CBIN code that is loaded into a Buffer Register of the wide or narrow converter. The result is returned in R2. R3 contains the Channel number in which R2 is loaded. CHTRAN reverses the translation performed by CHCODE. The Register pair <R1,R2> holds both the modified and unmodified numbers. R4 holds the Channel number on entry.

SEARCH,SPSRCH

SEARCH searches for one or one of many ASCII characters in IOBUF. The search proceeds from right to left, i.e., from higher to lower addresses. Probing begins at the address entered in R2 and ends either when the character is found, a semicolon (;) is encountered,

or #3+IOBUF is reached. In the first case, R2 returns the address of the character. Otherwise, R2 exits with #0. R3 and R4 delimit the range of ASCII characters that are sought. The first character in the range enters in R3; the last, in R4. When a single character is sought, R3 equals R4. SPSRCH is called to find a decimal number in IOBUF. First, a decimal point is sought. If this search fails, a digit, ASCII 60 through 71, is sought.

ASSEM

A string of decimal digits in IOBUF is assembled into a binary number, returned in <R1,R2> by ASSEM. As with SEARCH, this operation proceeds from right to left. On entry, R1 gives the maximum number of characters to be assembled. R2 holds the address of the rightmost character. Decimal numbers, ASCII 60 through 71, are assigned their usual values of zero through nine. Successive characters to the right are multiplied by the powers of ten: $10.^{0}$, $10.^{1}$, etc. Control characters, ASCII 0 through 40, and characters past the end of IOBUF are treated as zeros, ASCII 60. Sign characters, ASCII 41 to 57, are ignored as if they were absent from IOBUF. When any other character, ASCII 72 through 177, is found, assembly ends. Then a minus sign, ASCII 55, is sought, and <R1,R2> is negated if necessary. This routine does <u>not</u> record the position of a decimal point, since the latter is noted by the calling routine.

12. The POL Listing and Flow Charts

12.1 Terminology Used in the Flow Charts

Several symbols and abbreviations are used in the flow charts to shorten their length. Most of the symbols are standard for flow charts written for any computer language. The majority of the abbreviations are employed in assembly language for the PDP-11 computer. Although the reader should be intimately familiar with this terminology, it is described in this section to avoid confusion.

In the POL Listing, a group of statements with a specific function is separated from other statements by a blank line. Each of these groups usually corresponds to a single block in a flow chart which appears to the right of the listing.

When the description in the block is enclosed in brackets, [–], these statements perform an arithmetic operation on Registers, Peripheral Addresses, and/or Reserved Addresses.

A group of statements that produce branching within the program appear in the angle brackets, ⟨–⟩. These blocks always contain a question that has a yes or no answer. If the answer is no, the flow line from the block is intersected by a slash, $\not\rightarrow$. Otherwise, the flow line is unchanged, $\rightarrow$. Flow lines that pass control forward (backward) within a routine are usually drawn on the right (left) of the flow chart. Looping in routines is always accomplished with a backward branch.

If the symbol # prefixes a string of digits, a number, not an a absolute address in memory, is being referenced. For instance, #3 describes the number three, while 165000 is an absolute address. In the flow charts and POL listing, all numbers are octal (base eight) numbers except those containing a decimal point. Those numbers are decimal (base ten) numbers. When a Register or variable is "incremented" or "decremented", the number one is added or subtracted from its value, unless otherwise noted.

The label at the beginning of a table, e.g., PRCNT, often appears in the flow charts. A single entry in the table is always referenced. To determine how the entry is indexed, the reader should refer to Chapter 10. This rule is sometimes defied when PRFILE is mentioned. The phrase "the variable PRFILE," and only this phrase, references the lowest order byte in the table PRFILE. This location is often modified in all Background processes.

Except when a different description is offered, the phrase "the File" refers to the File numbered by the last File-referencing statement which was issued in that Programming environment. Note that "the Label" refers to the Label of this File. The "first File" of a data analysis command is numbered by a File-referencing statement; the "second File" is numbered by the second operand of the command. Many of these commands use only the first File.

To help the programmer understand data analysis operations, an equation describing the operation appears in the flow chart. The symbol a_i (b_j) refers to the datum in Slot number i(j) in the first (second) File. Values of a_i resulting from the analysis operation or

When a group of statements calls a subroutine, the code is described in parentheses, (—). Often, a single label is enclosed in parentheses. If these appear on the left (right) of a flow chart, these are entry (exit) points to (from) the routine. Exit points are branches from the routine to an entry point in another routine, except when the parentheses contains RTS. Then control returns to the routine that called the subroutine being described. To determine the location of the several entry and exit points in the POL Listing, a programmer can refer to the cross-reference table following the listing. When an entry or exit point represents a branch within a routine, the label is enclosed in a circle. More often, no entry or exit points are listed within a routine, just flow lines.

The abbreviations JSR, RTS, and EMT replace the phrases: call the subroutine, return from the subroutine, and call the Program Request (i.e., generate an EMT trap), respectively. In POL, all returns from subroutines occur through an RTS PC statement. Except for DETSGN, subroutines are called by

```
JSR  PC, name
```

where "name" is the entry point in a subroutine. The abbreviation DRTS describes a double return from a subroutine. In assembly language a DRTS statement is coded as

```
CLR  (SP) +
RTS   PC
```

This sequence is equivalent to executing two RTS PC statements in

immediate succession. That is, control returns to the routine that called a subroutine that in turn called the subroutine being described.

If a routine contains the exit point TRAP, an error can be flagged through that point. The TRAP statement generates a software interrupt to detect errors. The service routine is TRAP and its priority is the lowest possible, zero. As catalogued in Table 12.1, the operand of the TRAP statement indicates the type of error which it flags.

As is consistent with DEC terminology, R0 through R5, SP, PC and PS refer to General Programming Registers 0 through 5, the stack pointer, the program counter for the CPU, and the processor status word, respectively. NEWCOM loads R2 through R5 before a Command Subroutine executes. On entry, R2, R3, R4, and R5 contain the address of the File numbered by the last File-referencing statement, the first operand, the second operand, and twice the number of the executing Program, respectively. Many Function Subroutines employ variables of double length. When these are held in a Register pair, the two Registers appear between angle brackets, e.g., <R1,R2>, in the flow charts. The more significant Register of the pair is listed first.

When the number symbol, #, prefixes a label of an instruction, the address of the label is being described. The phrases "at" and "marked by", and the symbol "@" mean "at the address in". For example, @R1 describes the contents at the address in R1. If a Register or variable "points to" or is a "pointer" to information, the Register or variable contains the address of the information.

Table 12.1: How the Operand of the TRAP Statement Indicates the Type of Error being Flagged

Value of the Operand	The error results from:
1	the name of an instruction.
2	the range of the first operand.
3	the range of the second operand
4	the ordering of statements.
5	the usage of a statement in a programming environment.
6	the allocation of protected resources - files, programs or peripherals.
7	the name of a disc record.
10	the length limitation of a file, program, or other record in core.
11	a hardware error in a disc I/O operation.
12	applying a data analysis routine to an inappropriate type of data.

latest iteration of the operation are marked with an asterisk, *. If two Files are associated with the operation, the variable n is the fewer of the number of Slots in the first or second File. Otherwise, n is the number of Slots in the first File. Slots are numbered 0 to (n-1).

In the POL listing, a flow chart appears opposite the code which it describes. Flow charts appear only on pages 5 to 56 of the listing, since Reserved Addresses are defined on pages 1 to 4. The programmer must have a thorough understanding of Reserved and Peripheral Addresses before he attempts to interpret the charts.

12.2 Finding Routines in the POL Listing

POL is assembled from six disc records, namely POLADD.MAC, POLCIT.MAC, POLMAN.MAC, POLIO1.MAC, POLIO2.MAC, and POLNUM.MAC. In this section, the contents and chapters of each record are described in a single paragraph and in the order that the records are assembled. The names of chapters appear in CAPITAL letters. Using the table of contents which catalogs the chapters, a programmer can easily locate any routine in the POL listing.

FIXED RESERVED ADDRESSES lists tables of locations whose values are fixed while POL executes. These tables include information used for interpreting and assembling POL instructions. Single variables or tables of variables manipulated during execution are defined in VARIABLE RESERVED ADDRESSES.

When operating in the Background mode, POL cycles through BMON, the sole routine in BACKGROUND MONITOR. From this loop, the display scope is refreshed, the command assembler COMASS is called, and messages signaling the end of Program execution are issued. COMMAND ASSEMBLER contains the subroutine COMASS that assembles instructions entered from the Decwriter. This subroutine detects errors and stores assembled instructions into Programs. It also initiates the execution of Immediate Instructions. Some ASSEMBLER SUBROUTINES translate the ASCII input from the Decwriter. Another, NEWCOM, directs the execution of instructions entered in Programs or issued as Immediate Instructions. Assembly Subroutines are also listed in this section.

MEMORY MANAGEMENT ROUTINES manage the memory that stores Files and Program. Similarly, DISC MANAGEMENT ROUTINES maintain the permanent records of Files and Programs on floppy discs. The assembly of Programs and the placing of data into Files is directed by the routines in STORAGE CONTROL. Routines in PROGRAM EXECUTION CONTROL initiate and abort the execution of Programs. Other routines manage looping, the iterative processing of a segment in a Program.

When a Hardware Interrupt is generated, FMON, the sole routine in FOREGROUND MONITOR, services the Interrupt. This routine loads or reads Buffer Registers associated with synchronously controlled Peripherals. FMON also monitors the execution of Foreground Programs. COMMANDS THAT READ DATA set up and/or read the asynchronous and synchronous Peripherals that acquire data. These Peripherals include the 8-Channel ADC, pulse counters, DVMs, and the system clock. Some COMMANDS THAT CONTROL DATA ACQUISITION monitor the 8-Channel DAC.

Others delay the reading of data.

The subroutines in GRAPHICS CONTROL plot data on the display scope and the point-plotter. The Output Stack is also manipulated from this section. Subroutines in LISTING CONTROL print out Files and Programs on the Decwriter. Error messages are issued through TRAP, the interrupt service routine for error detection.

NUMERICAL ROUTINES employ software algorithms to multiply numbers divide numbers, and convert binary numbers into ASCII characters. Calculator commands are serviced in this section. DATA ANALYSIS ROUTINES manipulate the Slots and Labels of Files. The INITIALIZATION ROUTINE, POLEP, zeroes tables of variables, and sets up interrupt vectors and Status Registers. This routine runs only when POL first executes. Afterwards, Files and Programs may be stored in memory from #POLEP through 113050.

12.3 The POL Program

A listing of the assembly language code for POL follows. Flow charts describing the routines appear to the right of the code. Only single routines are described in the flow charts. The scheduling of execution of the routines was described in Chapter 11 and was summarized in Figure 11.1.

```
1                       .TITLE  POL-PERIFERAL ORIENTED LANGUAGE
2
3                       .NLIST  BEX
4                       .MCALL  ..V2..,.TTINR,.TTOUTR,.LOCK,.REGDEF,.SRESET
5 000000                ..V2..
6 000000                .REGDEF
7       000000'         .CSECT
8
9                       .SBTTL  FIXED RESERVED ADDRESSES
10
11
12      000012          .RADIX  10
13 00000 006724'COMADD: .WORD   FGET,FPUT,FRID,PGET,PPUT,PRID
14 00014 006450'        .WORD   FASSGN,FNUL,FASSGN,FCLR,FOPEN,FNUL
15 00030 010114'        .WORD   EXECUT,ABORT,START,END,IGNORE,RENTER
16 00044 013210'        .WORD   WRITE,PRINT,PROG,LIST,MOVCRS,PLOT
17 00060 011636'        .WORD   XEXP,YEXP,XMOVE,YMOVE,PUSH,POP
18 00074 010022'        .WORD   SSTORE,ISTORE,SFILE,IFILE
19 00104 011132'        .WORD   WSWTCH,WCLOCK,LOOP,CONT
20 00114 010554'        .WORD   RDAD,RDPC,RDVM,RDCL,SDA,IDA
21 00130 015262'        .WORD   CCOPY,CADD,CSUB,CDIV,DECON
22 00142 015610'        .WORD   DAOFF,DAEXP,OFFSET,EXPAND,VTOT
23 00154 015760'        .WORD   INTE,DIFF,SMOOTH,PEAK
24 00164 014102'        .WORD   LOAD,ADD,SUB,MUL,DIV
25
26 00176 030015 COMNAM: .WORD   12301,12310,12312,22541,22550,22552
27 00212 054007         .WORD   22535,22548,12295,12297,12309,12308
28 00226 027713         .WORD   12235,7445,26439,11914,15796,24939
29 00242 073417         .WORD   30479,23311,23317,18937,25609,23125
30 00256 074013         .WORD   30731,31755,30739,31763,23417,23222
31 00272 062031         .WORD   25625,15385,25612,15372
32 00302 072031         .WORD   29721,29705,19125,9908
33 00312 060007         .WORD   24583,24598,24604,24585,25610,15370
34 00326 022011         .WORD   9225,9223,9241,9226,10601
35 00340 024365         .WORD   10485,10475,21900,12246,29525
36 00352 037232         .WORD   16026,10732,26229,22887
37 00362 045247         .WORD   19111,7498,26472,20338,10748
38
39 00374 005034'COMERR: .WORD   CHMOVE,CHMOVE,CHFL3,CHEXIT
40 00404 005156'        .WORD   CHSST,CHIST,CHFL2,CHFL1
41 00414 005272'        .WORD   CHEXIT,CHCL,CHEXIT,CHCONT
42 00424 005220'        .WORD   CHAD,CHPC,CHKEY,CHEXIT,CHDA,CHDA
43 00440 005100'        .WORD   CHFL3,CHFL3,CHFL3,CHFL3,CHFL3,CHLAB,CHLAB
```

FIXED RESERVED ADDRESSES

```
1 000456    062 LIMIT1: .BYTE   50,50,50,4,4,4
2 000464    004         .BYTE   4,0,50,0,4,0
3 000472    000         .BYTE   0,0,0,0,0,0
4 000500    000         .BYTE   0,0,0,0,0,0
5 000506    000         .BYTE   0,0,0,0,5,0
6 000514    000         .BYTE   0,0,0,0
7 000520    000         .BYTE   0,0,3,0
8 000524    007         .BYTE   7,0,1,0,4,4
9 000532    000         .BYTE   0,0,0,0,20
10 00537    004         .BYTE   4,4,7,0,0
11 00544    000         .BYTE   0,0,0,0
12 00550    007         .BYTE   7,7,7,7,7
13
14 00555    000 LIMIT2: .BYTE   0,0,0,0,0,0
15 00563    144         .BYTE   100,4,8,50,110,50
16 00571    004         .BYTE   4,4,4,4,100,4
17 00577    010         .BYTE   8,20,4,4,8,0
18 00605    142         .BYTE   98,98,2,2,110,8
19 00613    010         .BYTE   8,1,110,10
20 00617    007         .BYTE   7,40,8,3
21 00623    050         .BYTE   40,40,6,0,40,40
22 00631    156         .BYTE   110,110,110,110,110
23 00636    050         .BYTE   40,40,40,98,0
24 00643    000         .BYTE   0,0,3,8
25 00647    050         .BYTE   40,40,40,40,40
26        000000        .RADIX
27
28 00654    300 COMFL1: .BYTE   300,300,300,200,200,200
29 00662    204         .BYTE   204,0,305,4,200,4
30 00670    000         .BYTE   0,4,0,0,0,0
31 00676    000         .BYTE   0,0,0,0,1,0
32 00704    015         .BYTE   15,15,32,32,200,0
33 00712    001         .BYTE   1,10,4,14
34 00716    000         .BYTE   0,25,205,0
35 00722    225         .BYTE   225,225,200,0,222,235
36 00730    004         .BYTE   4,4,4,4,304
37 00735    222         .BYTE   222,235,235,35,0
38 00742    000         .BYTE   0,0,0,5
39 00746    231         .BYTE   231,231,231,231,231
40
41 00753    200 COMFL2: .BYTE   200,200,200,200,200,200
42 00761    200         .BYTE   200,200,200,200,200,200
43 00767    200         .BYTE   200,200,200,140,140,200
44 00775    301         .BYTE   301,301,200,200,301,201
45 01003    200         .BYTE   200,200,200,200,301,300
46 01011    041         .BYTE   41,41,340,141
47 01015    040         .BYTE   40,40,140,141
48 01021    041         .BYTE   41,41,41,41,41,41
49 01027    301         .BYTE   301,301,301,301,301
50 01034    301         .BYTE   301,301,301,301,301
51 01041    301         .BYTE   301,301,301,301
52 01045    200         .BYTE   200,200,200,200,200
```

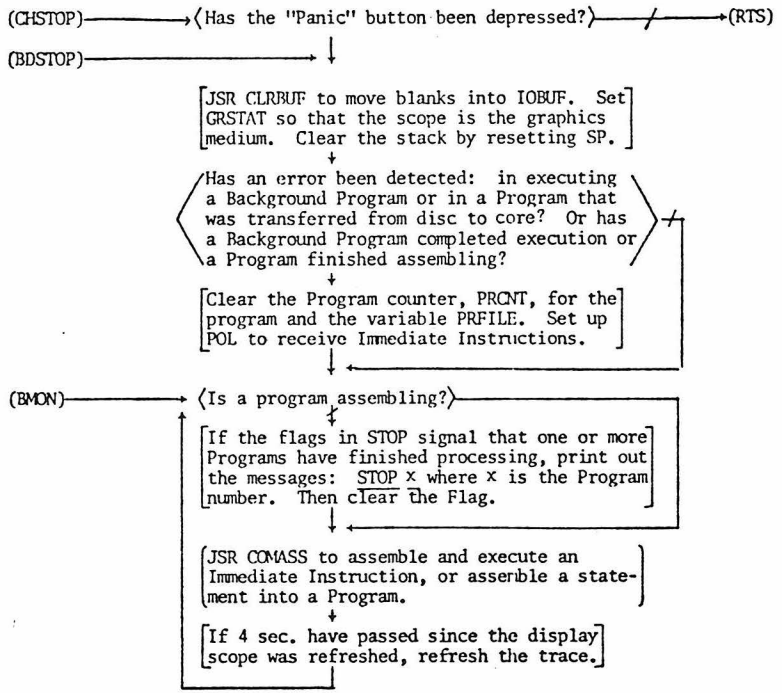
FIXED RESERVED ADDRESSES

```
1 001052     105          .ASCII  /ERROR/
2 001057     111 ERROR:   .ASCII  /INSTR/
3 001064     117          .ASCII  /OPER1/
4 001071     117          .ASCII  /OPER2/
5 001076     117          .ASCII  /ORDER/
6 001103     125          .ASCII  /USAGE/
7 001110     101          .ASCII  /ALLOC/
8 001115     124          .ASCII  /TRANS/
9 001122     105          .ASCII  /EOF  /
10 01127     110          .ASCII  /HRDWR/
11 01134     115          .ASCII  /MANIP/
12
13 01141     040 LABTAB:  .ASCII  /   EV/
14 01146     103          .ASCII  /COUNT/
15 01153     045          .ASCII  /%F.S./
16 01160     104          .ASCII  /DEG K/
17 01165     040          .ASCII  / SEC./
18 01172     040          .ASCII  /  AMU/
19 01177     040          .ASCII  /  MV./
20
21 01204 160400 TENS:     .WORD   160400,2765,113200,230,041100,17,103240,1
22 01224 023420           .WORD   10000.,0,1000.,0,100.,0,10.,0,1.,0
23       000012           .RADIX  10
24
25 01250 001266'SMOLST:   .WORD   SM005,SM009,SM017,SM025
26 01260     365          .BYTE   -11,44,62,44,-11,-7
27 01266 000010 SM005:    .WORD   8
28 01270     351          .BYTE   -23,16,43,60,64,60,43,16,-23,-8
29 01302 000020 SM009:    .WORD   16
30 01304     337          .BYTE   -33,-10,11,29,43,54,62,66,68,66,62,54,43,29,11,-10,-33,-9
31 01326 000040 SM017:    .WORD   32
32 01330     316          .BYTE   -50,-27,-6,13,29,44,57,64,77,84,89,92,92
33 01345     134          .BYTE   92,89,84,77,64,57,44,29,13,-6,-27,-50,-10
34 01362 000060 SM025:    .WORD   48
35
36 01364     377 DECFAC:  .BYTE   -1,-2,-3,-4,-5,-6,-9,-13,-17,-23,-31,-41,-54,-70,-84,-96,-102
37 01405     234          .BYTE   -100,-88,-70,-53,-40,-30,-22,-17,-13,-9,-6,-5,-4,-3,-2,-1,0
38
39 01426 000005 TEMP:     .WORD   5,2143,273
40 01434 000062           .WORD   50,1593,275
41 01442 000177           .WORD   127,1330,288
42 01450 000350           .WORD   232,1170,308
43 01456 000600           .WORD   384,1078,329
44 01464 001110           .WORD   584,1024,349
45 01472 001437           .WORD   799,953,390
46 01500 002003           .WORD   1027,898,432
47 01506 002632           .WORD   1434,855,475
48 01514 003504           .WORD   1860,863,464
49
50 01522 075022 WRFACT:   .WORD   31250,23438,18750,31250,18750
51 01534     006 WRSHFT:  .BYTE   6,6,8,8,8
52 01541     005 WRLAB:   .BYTE   5,5,30,35,30
53
54 01546     005 RDLAB:   .BYTE   5,5,35,15,15,15,15,15,10,25,20,5,35,5,35,15,15,15
55 01570     000 RDDP:    .BYTE   0,0,0,1,1,1,1,1,3,3,0,2,2,3,2,2,3,3
56 01612 002734 RDFACT:   .WORD   1500,1000,1000,1000,1000,1000,1000,1000,10000,1000
57 01636 000002           .WORD   2,200,50,1,2,30,100,100
```

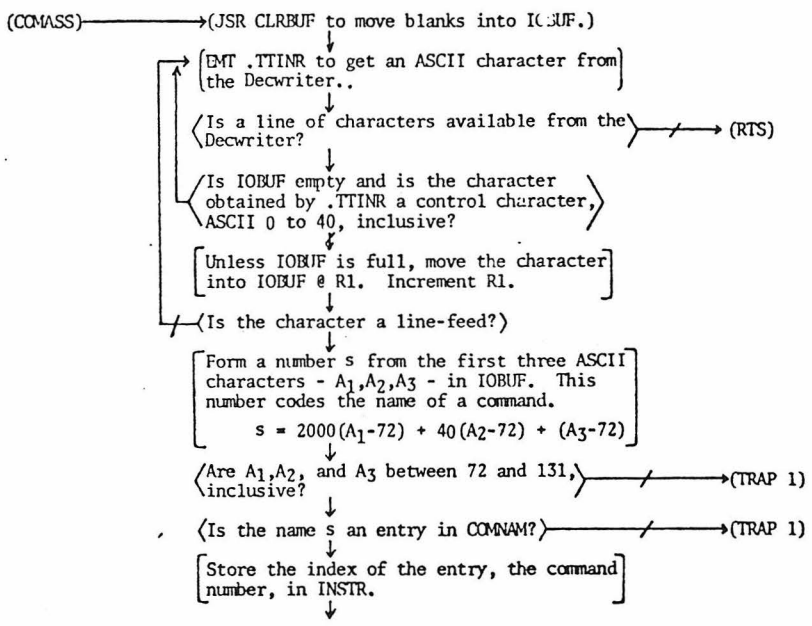
VARIABLE RESERVED ADDRESSES

```
1                       .SBTTL  VARIABLE RESERVED ADDRESSES
2
3
4 001656        PRBUFF: .BLKW   5
5 001670        FLBUFF: .BLKW   111
6
7 002226        RDKEY:  .BLKB   12
8 002242        WRKEY:  .BLKB   8
9 002252        FLKEY:  .BLKB   111
10
11 02431        STOP:   .BLKB   5
12 02436        PRFILE: .BLKW   5
13 02450        PRITER: .BLKW   5
14 02462        PRCNT:  .BLKW   5
15 02474        INSTR:  .BLKW   2
16
17 02500        LPITER: .BLKW   20
18 02550        LPRET:  .BLKW   20
19
20 02620 000000 RDSTAT: .WORD
21 02622 000000 WRSTAT: .WORD
22 02624        TIME:   .BLKW   2
23
24 02630        STORE1: .BLKW   10
25 02654        STORE2: .BLKW   10
26 02700        ADITER: .BLKW   8
27 02720        WIDEDA: .BLKW   8
28 02740        NARRDA: .BLKW   8
29
30 02760        OSTACK: .BLKW   48
31 03120        IOBUF:  .BLKW   66
32 03324 002760'OSTPNT: .WORD   OSTACK
33 03326 000000 OUTPNT: .WORD   0
34 03330 000000 GRSTAT: .WORD   0
35 03332 000000 CURSOR: .WORD   0
36
37 03334 000000 YSTART: .WORD
38 03336 000000 POINTS: .WORD
39 03340 000000 XSTART: .WORD
40
41 03342     000 AFILE:  .BYTE   0
42 03343     000 AKEY:   .BYTE   0
43 03344     000 LFILE:  .BYTE   0
44 03345     000 FINIS:  .BYTE   0
45
46 03346 016337 TRNAME: .RAD50  /DX1XXXXXXLDA/
47 03356 000000 EXAREA: .WORD   0,0,0,0,FINISH
48 03370 000000 RQAREA: .WORD   0,TRNAME,0,0
49
50 03400     000 SHIFT:  .BYTE   0
51 03401     377 PROGNO: .BYTE   -1
52 03402 000000 MANT:   .WORD   0
53 03404 000000 EXPON:  .WORD   0
54       000000         .RADIX
```

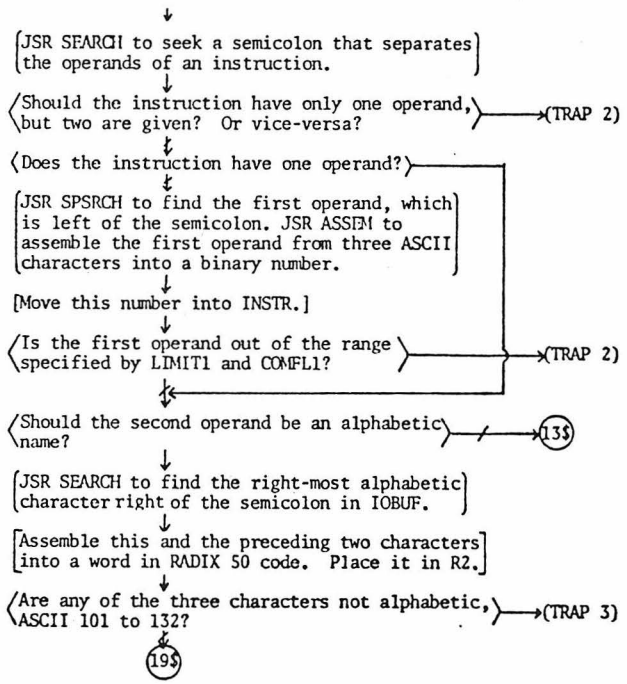
```
2                       .SBTTL  BACKGROUND MONITOR
3
4
5 003406 105737 PANIC:  TSTB    @#165230
6 003412 100401         BMI     BDSTOP
7 003414 000207 EXITC1: RTS     PC
8
9 003416 013700 BDSTOP: MOV     @#54,R0
10 03422 005060         CLR     256(R0)
11 03426 004767         JSR     PC,CLRBUF
12 03432 005067         CLR     GRSTAT
13 03436 012706         MOV     #1000,SP
14
15 03442 116700         MOVB    PROGNO,R0
16 03446 006300         ASL     R0
17 03450 100416         BMI     BMON
18 03452 122767         CMPB    #3.,INSTR
19 03460 001403         BEQ     1$
20 03462 105760         TSTB    PRFILE+1(R0)
21 03466 001407         BEQ     BMON
22
23 03470 105067 1$:     CLRB    PRFILE
24 03474 005060         CLR     PRCNT(R0)
25 03500 112767         MOVB    #-1,PROGNO
26
27 03506 105767 BMON:   TSTB    PROGNO
28 03512 100022         BPL     5$
29
30 03514 012703         MOV     #4,R3
31 03520 106263 3$:     ASRB    STOP(R3)
32 03524 103013         BCC     4$
33 03526 010302         MOV     R3,R2
34 03530 004767         JSR     PC,BINDOC
35 03534 012737         MOV     #'ST,@#IOBUF
36 03542 012737         MOV     #'OP,@#2+IOBUF
37 03550 004767         JSR     PC,DMPLN
38 03554 005303 4$:     DEC     R3
39 03556 100360         BPL     3$
40
41 03560 004767 5$:     JSR     PC,COMASS
42
43 03564 016700         MOV     TIME,R0
44 03570 042700         BIC     #170000,R0
45 03574 022700         CMP     #7700,R0
46 03600 101342         BHI     BMON
47 03602 004767         JSR     PC,GRAPH
48 03606 000737         BR      BMON
```



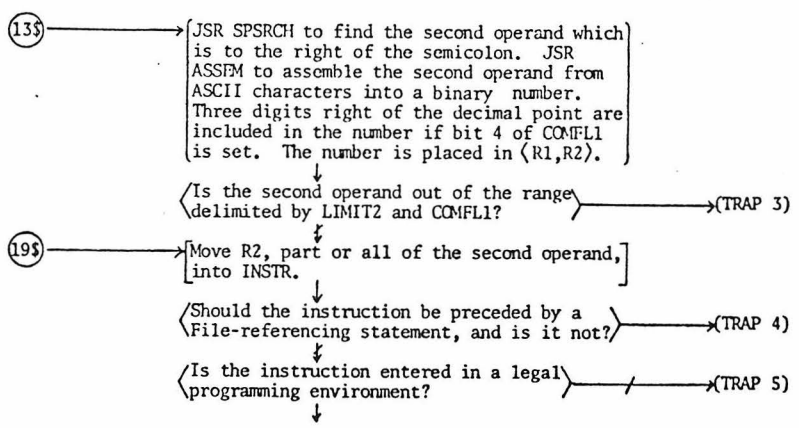
```
                        .SBTTL  COMMAND ASSEMBLER
                
                
003610 004767 COMASS:   JSR     PC,CLRBUF
                
003614         1$:      .TTINR
003616 103676           BCS     EXITC1
                
003620 020127           CMP     R1,#IOBUF
03624 001003            BNE     2$
03626 022700            CMP     #40,R0
03632 002370            BGE     1$
                
03634 022701 2$:        CMP     #132.+IOBUF,R1
03640 101401            BLOS    3$
03642 110021            MOVB    R0,(R1)+
                
03644 122700 3$:        CMPB    #12,R0
03650 001361            BNE     1$
                
                
03652 012703            MOV     #IOBUF,R3
03656 005002            CLR     R2
03660 012704            MOV     #126.+COMNAM,R4
03664 012705            MOV     #62.,R5
                
03670 112301 4$:        MOVB    (R3)+,R1
03672 162701            SUB     #72,R1
03676 100420            BMI     6$
03700 022701            CMP     #40,R1
03704 003415            BLE     6$
03706 060102            ADD     R1,R2
03710 022703            CMP     #3+IOBUF,R3
03714 101405            BLOS    5$
03716 012700            MOV     #32.,R0
03722 004767            JSR     PC,MULT
03726 000760            BR      4$
                
03730 020444 5$:        CMP     R2,-(R4)
03732 001403            BEQ     7$
03734 005305            DEC     R5
03736 100374            BPL     5$
03740 104401 6$:        TRAP    1
03742 010567 7$:        MOV     R5,INSTR
```



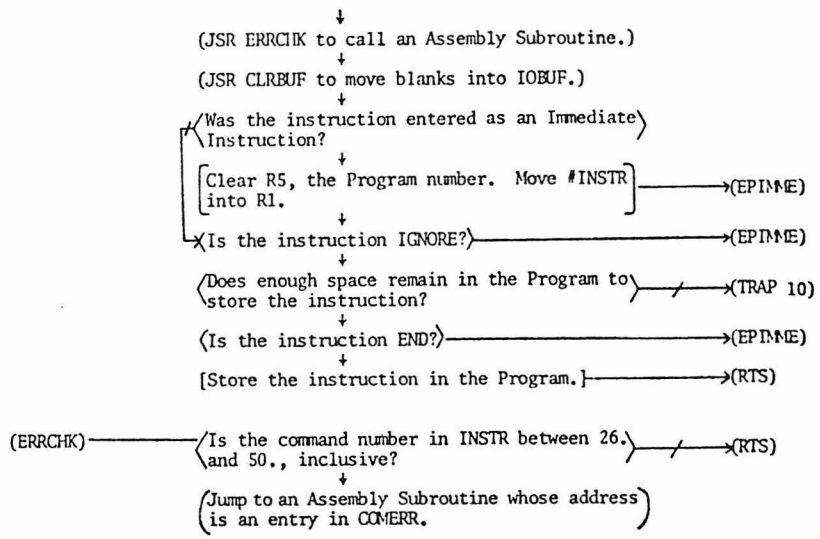
```
1 003746 012702          MOV     #132.+IOBUF,R2
2 003752 012703          MOV     #73,R3
3 003756 010304          MOV     R3,R4
4 003760 004767          JSR     PC,SEARCH
5
6 003764 116500          MOVB    COMFL1(R5),R0
7 003770 100403          BMI     8$
8 003772 005702          TST     R2
9 003774 001023          BNE     10$
10 03776 000423          BR      NEWSEC
11 04000 005702 8$:      TST     R2
12 04002 001420          BEQ     10$
13
14 04004 004767          JSR     PC,SPSRCH
15 04010 012701          MOV     #3,R1
16 04014 004767          JSR     PC,ASSEM
17
18 04020 110237          MOVB    R2,@#1+INSTR
19 04024 100407          BMI     10$
20 04026 001003          BNE     9$
21 04030 032700          BIT     #100,R0
22 04034 001003          BNE     10$
23 04036 126502 9$:      CMPB    LIMIT1(R5),R2
24 04042 002001          BGE     NEWSEC
25 04044 104402 10$:     TRAP    2
26
27
28 04046 012702 NEWSEC:  MOV     #132.+IOBUF,R2
29 04052 022705          CMP     #6,R5
30 04056 003433          BLE     13$
31
32 04060 012703          MOV     #101,R3
33 04064 012704          MOV     #132,R4
34 04070 004767          JSR     PC,SEARCH
35
36 04074 010204          MOV     R2,R4
37 04076 001464          BEQ     17$
38
39 04100 162702          SUB     #2,R2
40 04104 010203          MOV     R2,R3
41 04106 005002          CLR     R2
42
43 04110 112301 12$:     MOVB    (R3)+,R1
44 04112 162701          SUB     #100,R1
45 04116 003454          BLE     17$
46 04120 022701          CMP     #32,R1
47 04124 002451          BLT     17$
48 04126 060102          ADD     R1,R2
49 04130 020304          CMP     R3,R4
50 04132 101056          BHI     19$
51 04134 012700          MOV     #40.,R0
52 04140 004767          JSR     PC,MULT
53 04144 000761          BR      12$
```



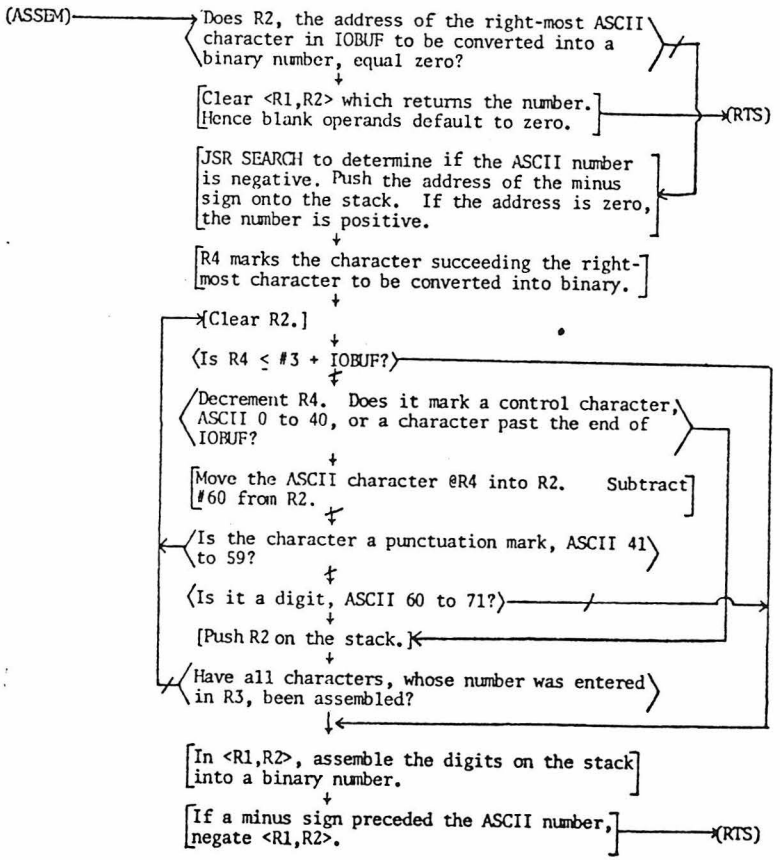
```
1 004146 004767 13$:    JSR     PC,SPSRCH
2 004152 012701         MOV     #5,R1
3 004156 032700         BIT     #20,R0
4 004162 001404         BEQ     14$
5 004164 062701         ADD     #3,R1
6 004170 062702         ADD     #3,R2
7 004174 004767 14$:    JSR     PC,ASSEM
8
9 004200 116504         MOVB    LIMIT2(R5),R4
10 04204 032700         BIT     #1,R0
11 04210 001401         BEQ     15$
12 04212 000304         SWAB    R4
13 04214 010203 15$:    MOV     R2,R3
14 04216 032700         BIT     #2,R0
15 04222 001401         BEQ     16$
16 04224 010103         MOV     R1,R3
17 04226 020403 16$:    CMP     R4,R3
18 04230 002407         BLT     17$
19
20 04232 032700         BIT     #4,R0
21 04236 001405         BEQ     18$
22 04240 005702         TST     R2
23 04242 001003         BNE     18$
24 04244 005701         TST     R1
25 04246 001001         BNE     18$
26 04250 104403 17$:    TRAP    3
27
28 04252 005403 18$:    NEG     R3
29 04254 003405         BLE     19$
30 04256 032700         BIT     #10,R0
31 04262 001772         BEQ     17$
32 04264 020403         CMP     R4,R3
33 04266 002770         BLT     17$
34
35
36 04270 113703 19$:    MOVB    @#1+INSTR,R3
37 04274 010237         MOV     R2,@#2+INSTR
38 04300 010504         MOV     R5,R4
39 04302 116500         MOVB    COMFL2(R5),R0
40
41 04306 105767         TSTB    PRFILE
42 04312 003004         BGT     21$
43 04314 032700         BIT     #1,R0
44 04320 001401         BEQ     21$
45 04322 104404         TRAP    4
46
47 04324 116705 21$:    MOVB    PROGNO,R5
48 04330 001404         BEQ     22$
49 04332 003007         BGT     23$
50 04334 005700         TST     R0
51 04336 100010         BPL     24$
52 04340 000410         BR      25$
```



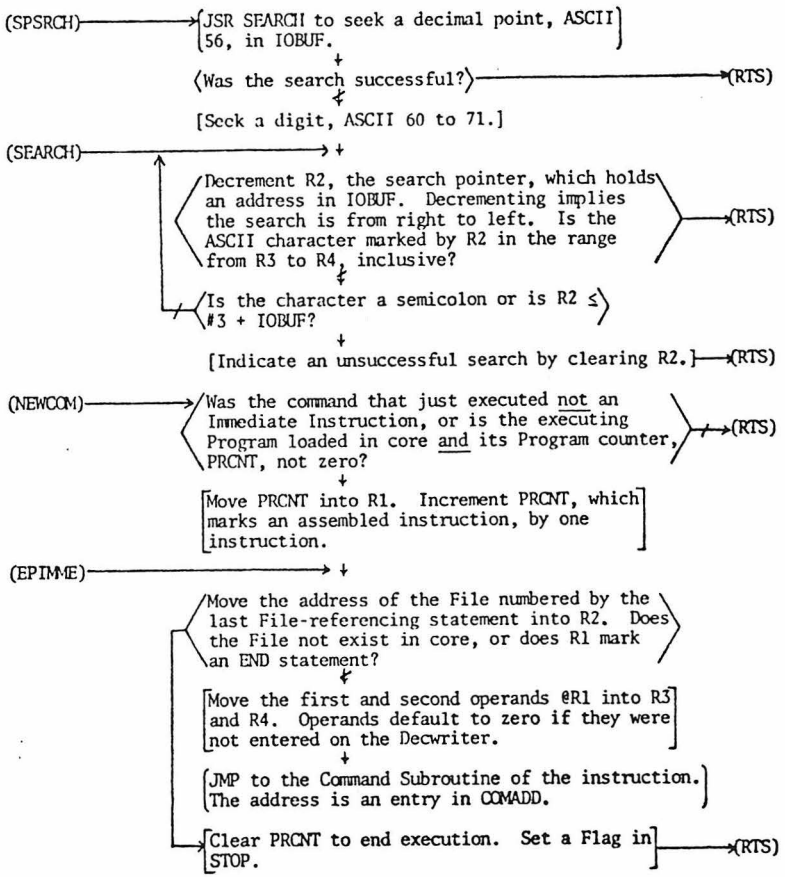
```
1 004342 032700 22$:    BIT     #100,R0
2 004346 001404         BEQ     24$
3 004350 000404         BR      25$
4 004352 032700 23$:    BIT     #40,R0
5 004356 001001         BNE     25$
6 004360 104405 24$:    TRAP    5
7
8 004362 004767 25$:    JSR     PC,ERRCHK
9 004366 004767         JSR     PC,CLRBUF
10
11 04372 012701         MOV     #INSTR,R1
12 04376 006305         ASL     R5
13 04400 100002         BPL     27$
14 04402 005005         CLR     R5
15 04404 000567         BR      EPIMME
16
17 04406 122711 27$:    CMPB    #16.,(R1)
18 04412 001564         BEQ     EPIMME
19
20 04414 016500         MOV     PRCNT(R5),R0
21 04420 016502         MOV     PRBUFF(R5),R2
22 04424 010203         MOV     R2,R3
23 04426 064203         ADD     -(R2),R3
24 04430 061203         ADD     (R2),R3
25 04432 020300         CMP     R3,R0
26 04434 103001         BHIS    28$
27 04436 104410         TRAP    10
28
29 04440 122711 28$:    CMPB    #15.,(R1)
30 04444 001547         BEQ     EPIMME
31
32 04446 012120         MOV     (R1)+,(R0)+
33 04450 012120         MOV     (R1)+,(R0)+
34 04452 010065         MOV     R0,PRCNT(R5)
35 04456 000207         RTS     PC
36
37
38 04460 116700 ERRCHK: MOVB    PROGNO,R0
39 04464 006300         ASL     R0
40 04466 162704         SUB     #26.,R4
41 04472 100516         BMI     EXITC2
42 04474 022704         CMP     #24.,R4
43 04500 002513         BLT     EXITC2
44 04502 006304         ASL     R4
45 04504 000174         JMP     @COMERR(R4)
```



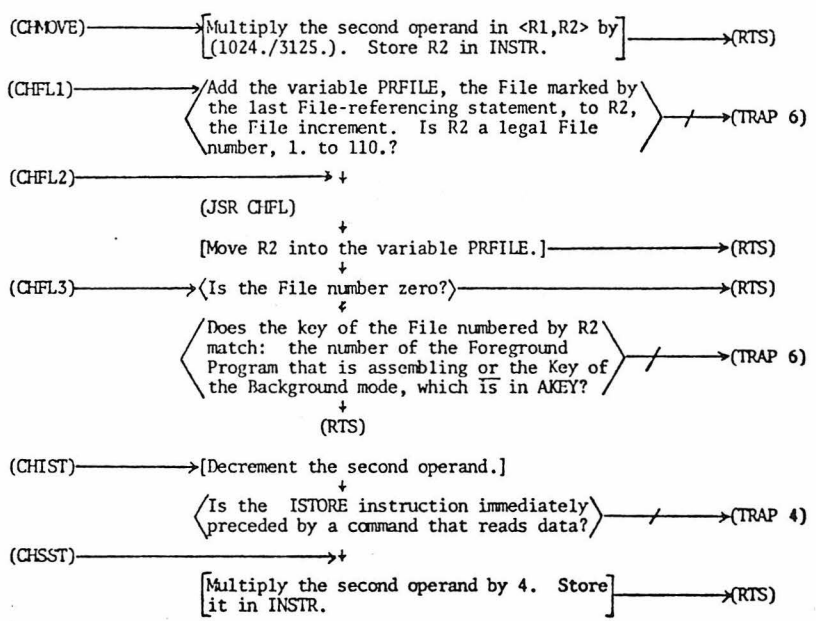
```
1                       .SBTTL  ASSEMBLER SUBROUTINES
2
3
4 004510 005003 ASSEM:  CLR     R3
5 004512 010246         MOV     R2,-(SP)
6 004514 001435         BEQ     3$
7
8 004516 012704         MOV     #55,R4
9 004522 010403         MOV     R4,R3
10 04524 004767         JSR     PC,SEARCH
11
12 04530 012604         MOV     (SP)+,R4
13 04532 005204         INC     R4
14 04534 010246         MOV     R2,-(SP)
15 04536 010103         MOV     R1,R3
16
17 04540 005002 1$:     CLR     R2
18 04542 022704         CMP     #3+IOBUF,R4
19 04546 103020         BHIS    3$
20
21 04550 122744         CMPB    #40,-(R4)
22 04554 002012         BGE     2$
23 04556 022704         CMP     #132.+IOBUF,R4
24 04562 101407         BLOS    2$
25
26 04564 111402         MOVB    (R4),R2
27 04566 162702         SUB     #60,R2
28 04572 100762         BMI     1$
29
30 04574 022702         CMP     #12,R2
31 04600 003403         BLE     3$
32
33 04602 010246 2$:     MOV     R2,-(SP)
34 04604 005301         DEC     R1
35 04606 003354         BGT     1$
36
37 04610 160103 3$:     SUB     R1,R3
38 04612 005001         CLR     R1
39 04614 005002         CLR     R2
40
41 04616 005703         TST     R3
42 04620 003407         BLE     5$
43 04622 062602 4$:     ADD     (SP)+,R2
44 04624 005501         ADC     R1
45 04626 005303         DEC     R3
46 04630 003403         BLE     5$
47 04632 004767         JSR     PC,TMSTEN
48 04636 000771         BR      4$
49
50 04640 005726 5$:     TST     (SP)+
51 04642 001432         BEQ     EXITC2
52 04644 000167         JMP     NEGATE
```



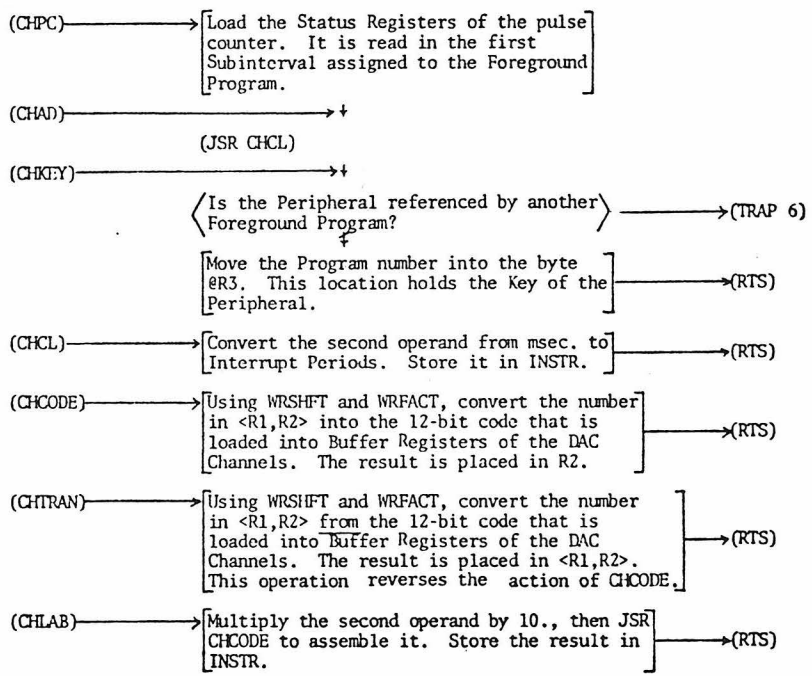
```
1 004650 012703 SPSRCH: MOV     #56,R3
2 004654 010304         MOV     R3,R4
3 004656 010201         MOV     R2,R1
4 004660 004767         JSR     PC,SEARCH
5
6 004664 005702         TST     R2
7 004666 001020         BNE     EXITC2
8 004670 012703         MOV     #60,R3
9 004674 012704         MOV     #71,R4
10 04700 010102         MOV     R1,R2
11
12 04702 124203 SEARCH: CMPB    -(R2),R3
13 04704 002402         BLT     1$
14 04706 121204         CMPB    (R2),R4
15 04710 003407         BLE     EXITC2
16
17 04712 122712 1$:     CMPB    #73,(R2)
18 04716 001403         BEQ     2$
19 04720 022702         CMP     #3+IOBUF,R2
20 04724 103766         BLO     SEARCH
21
22 04726 005002 2$:     CLR     R2
23 04730 000207 EXITC2: RTS     PC
24
25
26 04732 116705 NEWCOM: MOVB    PROGNO,R5
27 04736 006305         ASL     R5
28 04740 100773         BMI     EXITC2
29 04742 105765         TSTB    PRFILE+1(R5)
30 04746 001770         BEQ     EXITC2
31
32 04750 016501         MOV     PRCNT(R5),R1
33 04754 001765         BEQ     EXITC2
34 04756 062765         ADD     #4,PRCNT(R5)
35
36 04764 116502 EPIMME: MOVB    PRFILE(R5),R2
37 04770 001404         BEQ     1$
38 04772 006302         ASL     R2
39 04774 016202         MOV     FLBUFF(R2),R2
40 05000 001407         BEQ     3$
41 05002 112100 1$:     MOVB    (R1)+,R0
42 05004 100405         BMI     3$
43
44 05006 112103         MOVB    (R1)+,R3
45 05010 012104         MOV     (R1)+,R4
46 05012 006300         ASL     R0
47 05014 000170         JMP     @COMADD(R0)
48
49 05020 005065 3$:     CLR     PRCNT(R5)
50 05024 006205         ASR     R5
51 05026 105265         INCB    STOP(R5)
52 05032 000207         RTS     PC
```



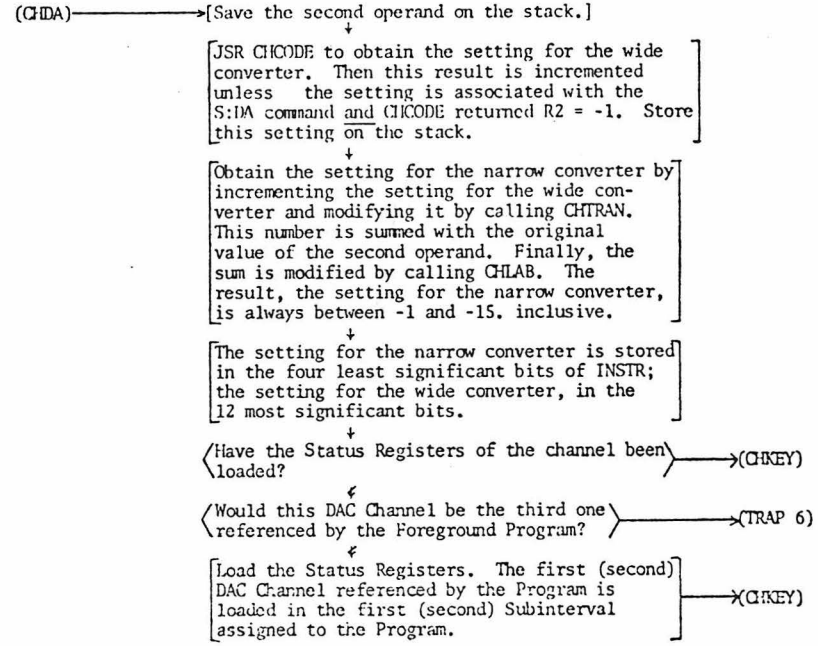
```
1 005034 112767 CHMOVE: MOVB    #10.,SHIFT
2 005042 012700         MOV     #3125.,R0
3 005046 000505         BR      DIVOP
4
5
6 005050 116704 CHFL1:  MOVB    PRFILE,R4
7 005054 060402         ADD     R4,R2
8 005056 003475         BLE     E06C
9 005060 022702         CMP     #110.,R2
10 05064 002472         BLT     E06C
11
12 05066 004767 CHFL2:  JSR     PC,CHFL
13 05072 110267         MOVB    R2,PRFILE
14 05076 000207         RTS     PC
15
16 05100 005702 CHFL3:  TST     R2
17 05102 001473         BEQ     CHEXIT
18
19 05104 116704 CHFL:   MOVB    PROGNO,R4
20 05110 003003         BGT     1$
21 05112 116704         MOVB    AKEY,R4
22 05116 003455         BLE     E06C
23 05120 126204 1$:     CMPB    FLKEY(R2),R4
24 05124 001052         BNE     E06C
25 05126 000207         RTS     PC
26
27
28 05130 005302 CHIST:  DEC     R2
29 05132 016000         MOV     PRCNT(R0),R0
30 05136 122760         CMPB    #38.,-4(R0)
31 05144 003043         BGT     E04C
32 05146 122760         CMPB    #41.,-4(R0)
33 05154 002437         BLT     E04C
34
35 05156 006302 CHSST:  ASL     R2
36 05160 006302         ASL     R2
37 05162 000441         BR      LOADIN
```



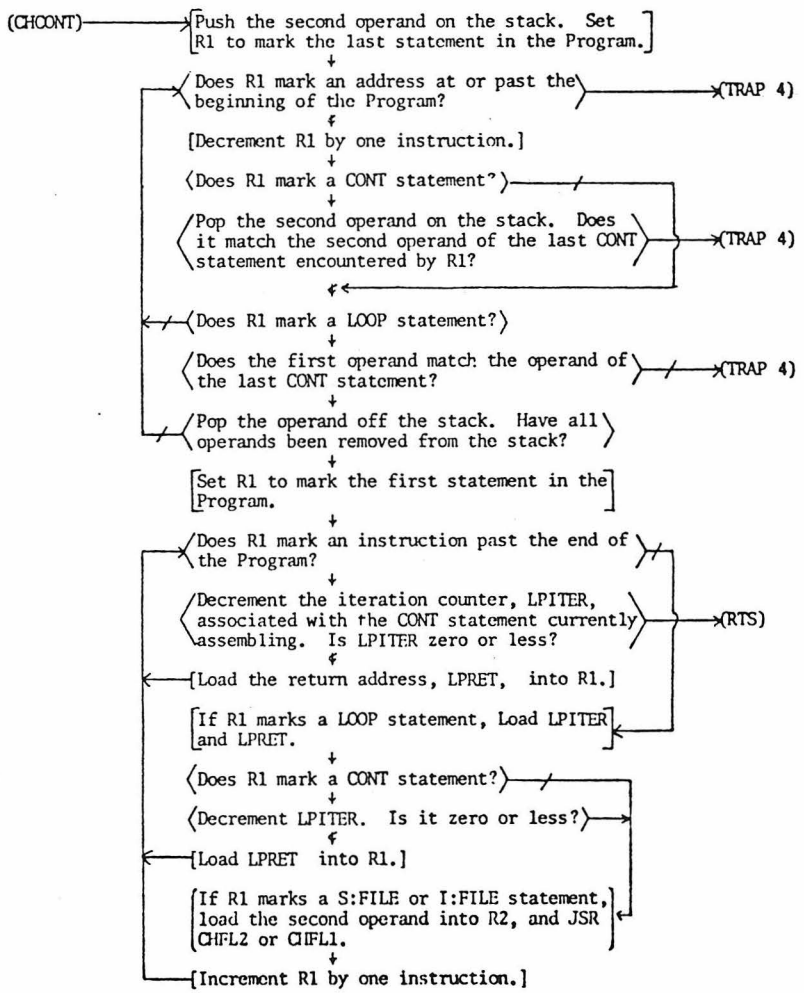
```
1 005164 105763 CHPC:   TSTB    RDKEY+8.(R3)
2 005170 001011         BNE     1$
3
4 005172 006303         ASL     R3
5 005174 006200         ASR     R0
6 005176 062700         ADD     #7,R0
7 005202 010063         MOV     R0,164040(R3)
8 005206 010063         MOV     R0,164054(R3)
9 005212 006203         ASR     R3
10 05214 062703 1$:     ADD     #8.,R3
11
12 05220 062703 CHAD:   ADD     #2.,R3
13 05224 004767         JSR     PC,CHCL
14
15 05230 062703 CHKEY:  ADD     #RDKEY,R3
16 05234 105713         TSTB    (R3)
17 05236 003002         BGT     2$
18 05240 116713         MOVB    PROGNO,(R3)
19 05244 121367 2$:     CMPB    (R3),PROGNO
20 05250 001410         BEQ     CHEXIT
21 05252 104406 E06C:   TRAP    6
22 05254 104404 E04C:   TRAP    4
23
24 05256 012700 CHCL:   MOV     #16.,R0
25 05262 004767 DIVOP:  JSR     PC,DIVID
26 05266 010237 LOADIN: MOV     R2,@#2+INSTR
27 05272 000207 CHEXIT: RTS     PC
28
29
30 05274 116367 CHCODE: MOVB    WRSHFT(R3),SHIFT
31 05302 006303         ASL     R3
32 05304 016300         MOV     WRFACT(R3),R0
33 05310 006203         ASR     R3
34 05312 004767         JSR     PC,DIVID
35 05316 005101         COM     R1
36 05320 005102         COM     R2
37 05322 000207         RTS     PC
38
39 05324 116467 CHTRAN: MOVB    WRSHFT(R4),SHIFT
40 05332 105467         NEGB    SHIFT
41 05336 006304         ASL     R4
42 05340 016400         MOV     WRFACT(R4),R0
43 05344 000167         JMP     MULT
44
45 05350 004767 CHLAB:  JSR     PC,TMSTEN
46 05354 004767         JSR     PC,CHCODE
47 05360 000742         BR      LOADIN
```



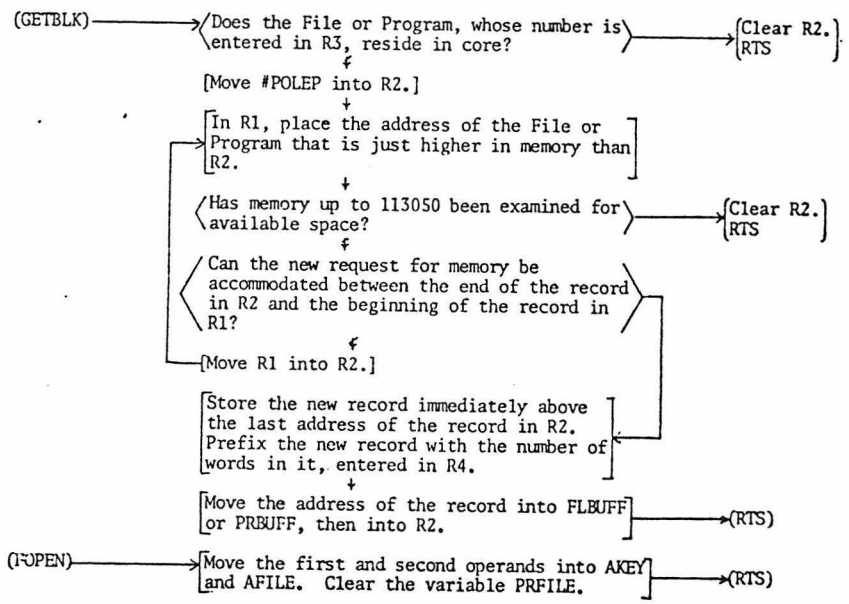
```
1 005362 010246 CHDA:   MOV     R2,-(SP)
2 005364 010146         MOV     R1,-(SP)
3
4 005366 004767         JSR     PC,CHCODE
5 005372 122767         CMPB    #42.,INSTR
6 005400 001003         BNE     6$
7 005402 022702         CMP     #-1,R2
8 005406 001401         BEQ     7$
9 005410 005202 6$:     INC     R2
10 05412 010246 7$:     MOV     R2,-(SP)
11
12 05414 005202         INC     R2
13 05416 010304         MOV     R3,R4
14 05420 004767         JSR     PC,CHTRAN
15 05424 012604         MOV     (SP)+,R4
16 05426 062601         ADD     (SP)+,R1
17 05430 062602         ADD     (SP)+,R2
18 05432 005501         ADC     R1
19 05434 004767         JSR     PC,CHLAB
20
21 05440 006304         ASL     R4
22 05442 006304         ASL     R4
23 05444 006304         ASL     R4
24 05446 006304         ASL     R4
25 05450 042737         BIC     #177760,@#2+INSTR
26 05456 050437         BIS     R4,@#2+INSTR
27
28 05462 105763         TSTB    WRKEY(R3)
29 05466 001033         BNE     5$
30
31 05470 116700         MOVB    PROGNO,R0
32 05474 062700         ADD     #27,R0
33 05500 012704         MOV     #WRKEY,R4
34 05504 126724 1$:     CMPB    PROGNO,(R4)+
35 05510 001404         BEQ     2$
36 05512 022704         CMP     #8.+WRKEY,R4
37 05516 101372         BHI     1$
38 05520 000410         BR      4$
39 05522 062700 2$:     ADD     #4,R0
40 05526 126724 3$:     CMPB    PROGNO,(R4)+
41 05532 001647         BEQ     E06C
42 05534 022704         CMP     #8.+WRKEY,R4
43 05540 101372         BHI     3$
44
45 05542 006303 4$:     ASL     R3
46 05544 010063         MOV     R0,164000(R3)
47 05550 010063         MOV     R0,164020(R3)
48 05554 006203         ASR     R3
49
50 05556 062703 5$:     ADD     #12.,R3
51 05562 000622         BR      CHKEY
```



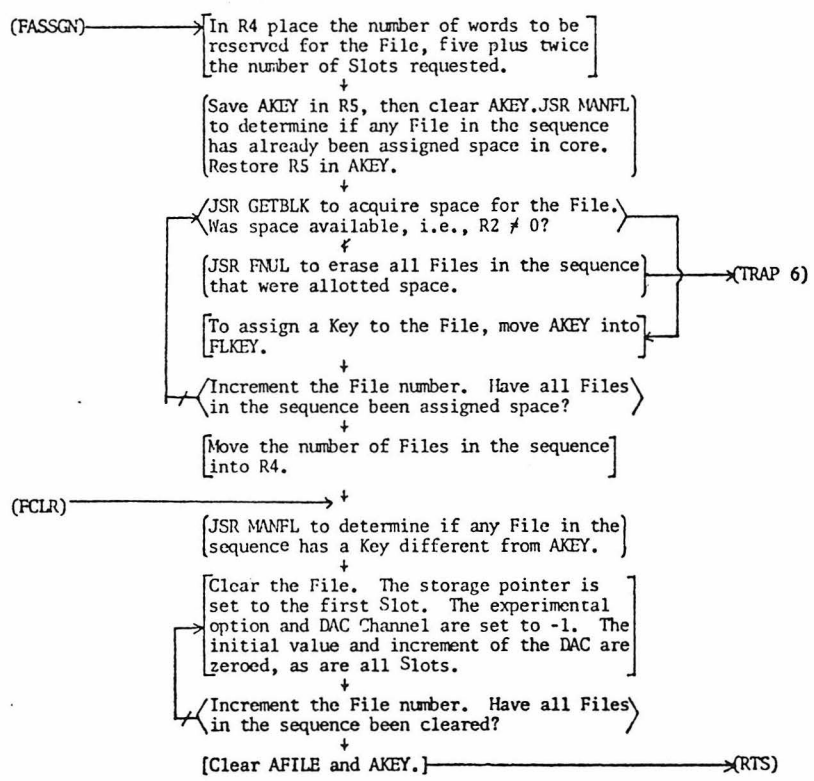
```
1 005564 016001 CHCONT: MOV     PRCNT(R0),R1
2 005570 010604         MOV     SP,R4
3 005572 010246         MOV     R2,-(SP)
4 005574 010203         MOV     R2,R3
5 005576 006303         ASL     R3
6
7 005600 026001 1$:     CMP     PRBUFF(R0),R1
8 005604 103223         BHIS    E04C
9 005606 162701         SUB     #4,R1
10
11 05612 122711         CMPB    #37.,(R1)
12 05616 001005         BNE     4$
13 05620 016146         MOV     2(R1),-(SP)
14 05624 021666         CMP     (SP),2(SP)
15 05630 001611         BEQ     E04C
16
17 05632 122711 4$:     CMPB    #36.,(R1)
18 05636 001360         BNE     1$
19 05640 126126         CMPB    1(R1),(SP)+
20 05644 001203         BNE     E04C
21 05646 020604         CMP     SP,R4
22 05650 001353         BNE     1$
23 05652 016001         MOV     PRBUFF(R0),R1
24
25 05656 026001 5$:     CMP     PRCNT(R0),R1
26 05662 101007         BHI     6$
27 05664 005363         DEC     LPITER(R3)
28 05670 003600         BLE     CHEXIT
29 05672 010304         MOV     R3,R4
30 05674 016401 9$:     MOV     LPRET(R4),R1
31 05700 000766         BR      5$
32
33 05702 122711 6$:     CMPB    #36.,(R1)
34 05706 001007         BNE     7$
35 05710 005201         INC     R1
36 05712 112104         MOVB    (R1)+,R4
37 05714 006304         ASL     R4
38 05716 012164         MOV     (R1)+,LPITER(R4)
39 05722 010164         MOV     R1,LPRET(R4)
40
41 05726 122711 7$:     CMPB    #37.,(R1)
42 05732 001006         BNE     10$
43 05734 016104         MOV     2(R1),R4
44 05740 006304         ASL     R4
45 05742 005364         DEC     LPITER(R4)
46 05746 003352         BGT     9$
47
48 05750 112104 10$:    MOVB    (R1)+,R4
49 05752 005201         INC     R1
50 05754 012102         MOV     (R1)+,R2
51 05756 022704         CMP     #32.,R4
52 05762 001002         BNE     11$
53 05764 004767         JSR     PC,CHFL2
54 05770 022704 11$:    CMP     #33.,R4
55 05774 001330         BNE     5$
56 05776 004767         JSR     PC,CHFL1
57 06002 000725         BR      5$
```



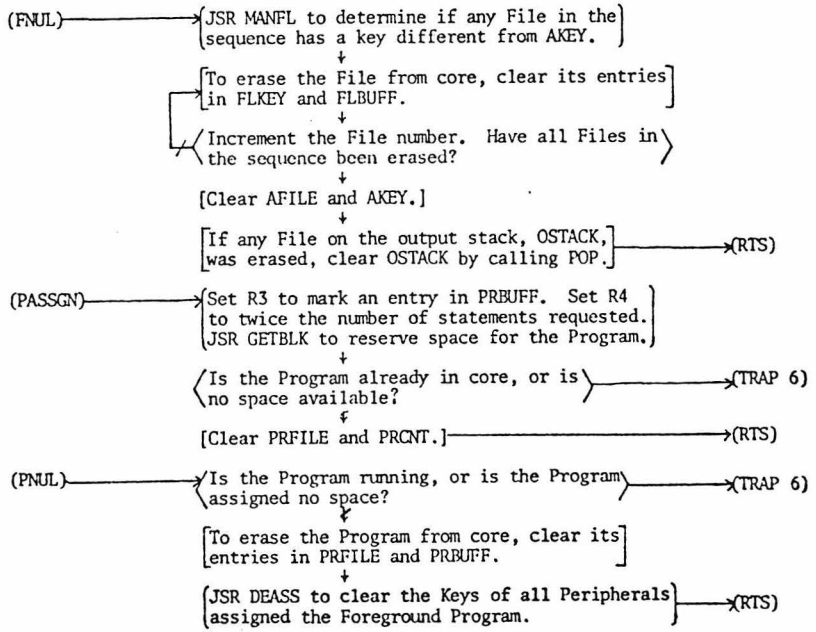
```
2                       .SBTTL  MEMORY MANAGEMENT ROUTINES
3
4
5  006004 006303 GETBLK: ASL     R3
6  006006 005763         TST     FLBUFF(R3)
7  006012 001043         BNE     5$
8  006014 012702         MOV     #POLEP,R2
9
10 06020 012701 1$:      MOV     #113050,R1
11 06024 012700          MOV     #PRBUFF,R0
12 06030 021002 2$:      CMP     (R0),R2
13 06032 101403          BLOS    3$
14 06034 021001          CMP     (R0),R1
15 06036 103001          BHIS    3$
16 06040 011001          MOV     (R0),R1
17 06042 005720 3$:      TST     (R0)+
18 06044 022700          CMP     #220.+FLBUFF,R0
19 06050 101367          BHI     2$
20
21 06052 016200          MOV     -2(R2),R0
22 06056 006300          ASL     R0
23 06060 060002          ADD     R0,R2
24 06062 022702          CMP     #113050,R2
25 06066 101415          BLOS    5$
26
27 06070 010400          MOV     R4,R0
28 06072 005400          NEG     R0
29 06074 005300          DEC     R0
30 06076 006300          ASL     R0
31 06100 060100          ADD     R1,R0
32 06102 020200          CMP     R2,R0
33 06104 101402          BLOS    4$
34
35 06106 010102          MOV     R1,R2
36 06110 000743          BR      1$
37
38 06112 010422 4$:      MOV     R4,(R2)+
39 06114 010263          MOV     R2,FLBUFF(R3)
40 06120 000207          RTS     PC
41
42 06122 005002 5$:      CLR     R2
43 06124 000207          RTS     PC
44
45
46 06126 110367 FOPEN:   MOVB    R3,AKEY
47 06132 110467          MOVB    R4,AFILE
48 06136 105067          CLRB    PRFILE
49 06142 000207          RTS     PC
```



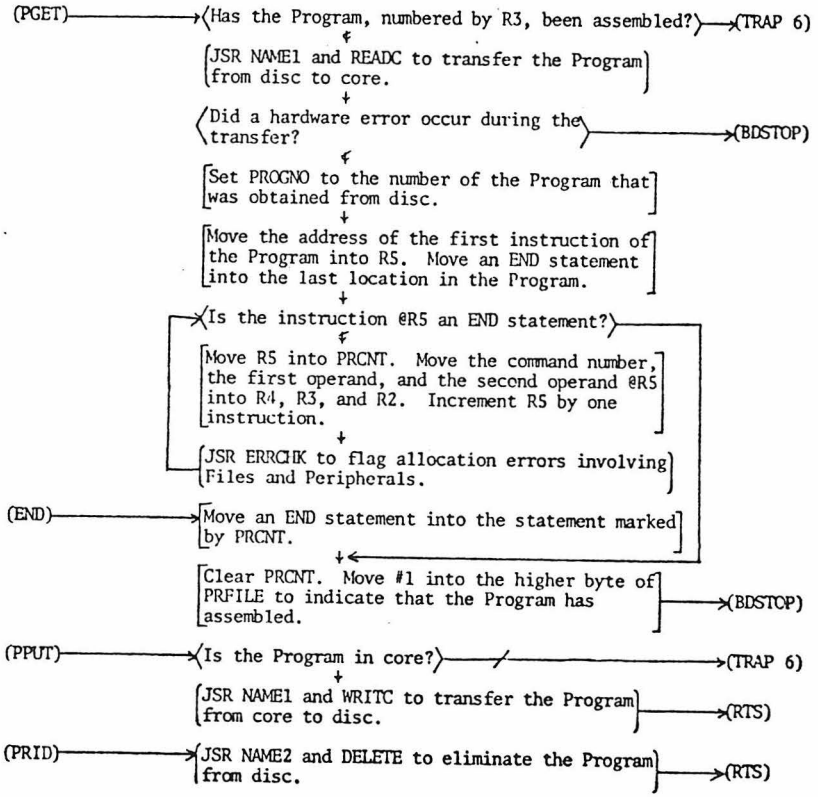
```
1 006144 006304 FASSGN: ASL     R4
2 006146 062704         ADD     #5,R4
3
4 006152 116705         MOVB    AKEY,R5
5 006156 105067         CLRB    AKEY
6 006162 004767         JSR     PC,MANFL
7 006166 110567         MOVB    R5,AKEY
8
9 006172 116705 1$:     MOVB    AFILE,R5
10 06176 005405         NEG     R5
11 06200 060305         ADD     R3,R5
12
13 06202 004767         JSR     PC,GETBLK
14 06206 001005         BNE     2$
15
16 06210 010504         MOV     R5,R4
17 06212 001553         BEQ     E06M
18 06214 004767         JSR     PC,FNUL
19 06220 000550         BR      E06M
20
21 06222 006203 2$:     ASR     R3
22 06224 116763         MOVB    AKEY,FLKEY(R3)
23
24 06232 005203         INC     R3
25 06234 126703         CMPB    LFILE,R3
26 06240 003354         BGT     1$
27
28 06242 010504         MOV     R5,R4
29 06244 005204         INC     R4
30
31 06246 010403 FCLR:   MOV     R4,R3
32 06250 004767         JSR     PC,MANFL
33
34 06254 006303 3$:     ASL     R3
35 06256 016302         MOV     FLBUFF(R3),R2
36 06262 006203         ASR     R3
37
38 06264 016204         MOV     -2(R2),R4
39 06270 162704         SUB     #5,R4
40 06274 010412         MOV     R4,(R2)
41 06276 006222         ASR     (R2)+
42 06300 010212         MOV     R2,(R2)
43 06302 062722         ADD     #8.,(R2)+
44 06306 012722         MOV     #-1,(R2)+
45 06312 005724         TST     (R4)+
46
47 06314 005022 4$:     CLR     (R2)+
48 06316 005304         DEC     R4
49 06320 003375         BGT     4$
50
51 06322 005203         INC     R3
52 06324 126703         CMPB    LFILE,R3
53 06330 003351         BGT     3$
54
55 06332 005067         CLR     AFILE
56 06336 000207         RTS     PC
```



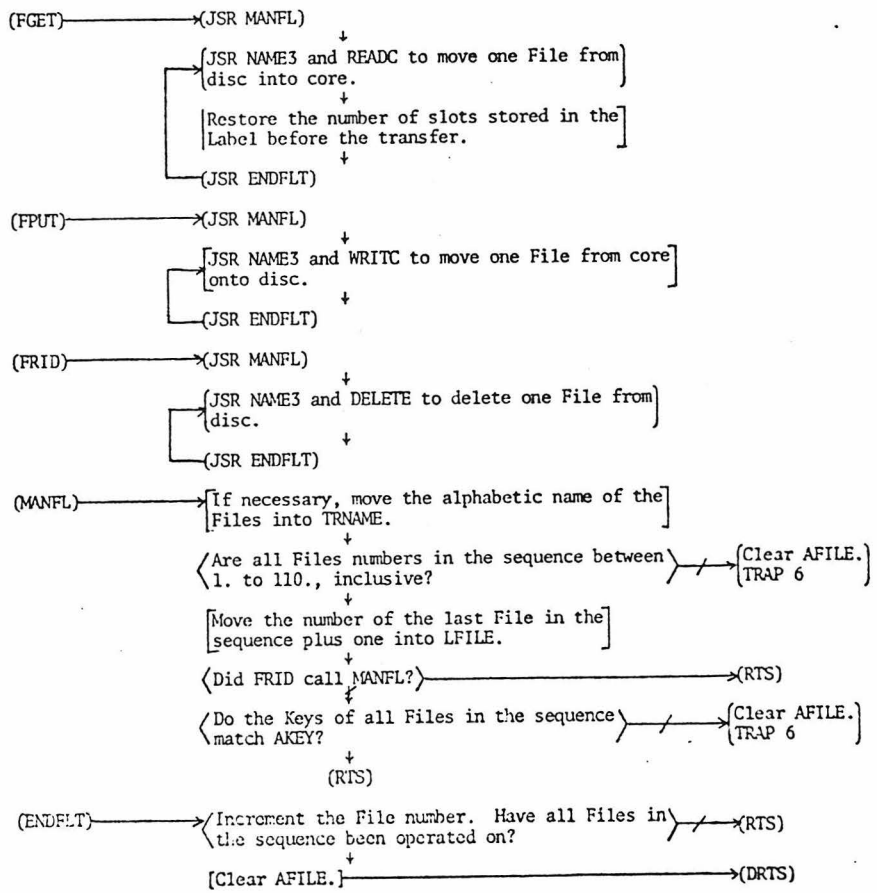
```
1  006340 010403 FNUL:   MOV     R4,R3
2  006342 004767         JSR     PC,MANFL
3
4  006346 105063 1$:     CLRB    FLKEY(R3)
5  006352 006303         ASL     R3
6  006354 005063         CLR     FLBUFF(R3)
7
8  006360 006203         ASR     R3
9  006362 005203         INC     R3
10 06364 126703          CMPB    LFILE,R3
11 06370 003366          BGT     1$
12
13 06372 005067          CLR     AFILE
14
15 06376 016702          MOV     OSTPNT,R2
16 06402 112201 2$:      MOVB    (R2)+,R1
17 06404 105761          TSTB    FLKEY(R1)
18 06410 003413          BLE     4$
19 06412 112201          MOVB    (R2)+,R1
20 06414 001403          BEQ     3$
21 06416 105761          TSTB    FLKEY(R1)
22 06422 003406          BLE     4$
23 06424 162702 3$:      SUB     #14.,R2
24 06430 020227          CMP     R2,#OSTACK
25 06434 101762          BLOS    2$
26 06436 000207          RTS     PC
27 06440 012704 4$:      MOV     #8.,R4
28 06444 000167          JMP     POP
29
30
31 06450 006304 PASSGN: ASL     R4
32 06452 162703          SUB     #5,R3
33 06456 004767          JSR     PC,GETBLK
34 06462 001427          BEQ     E06M
35
36 06464 005063          CLR     PRFILE+10.(R3)
37 06470 005063          CLR     PRCNT+10.(R3)
38 06474 000207 EXITM1: RTS     PC
39
40
41 06476 006304 PNUL:   ASL     R4
42 06500 005764          TST     PRCNT(R4)
43 06504 001016          BNE     E06M
44 06506 016400          MOV     PRBUFF(R4),R0
45 06512 001417          BEQ     E06M
46
47 06514 005064          CLR     PRFILE(R4)
48 06520 005064          CLR     PRBUFF(R4)
49
50 06524 105710 1$:      TSTB    (R0)
51 06526 100762          BMI     EXITM1
52 06530 004767          JSR     PC,DEASS
53 06534 062700          ADD     #4,R0
54 06540 000771          BR      1$
55
56 06542 104406 E06M:    TRAP    6
```



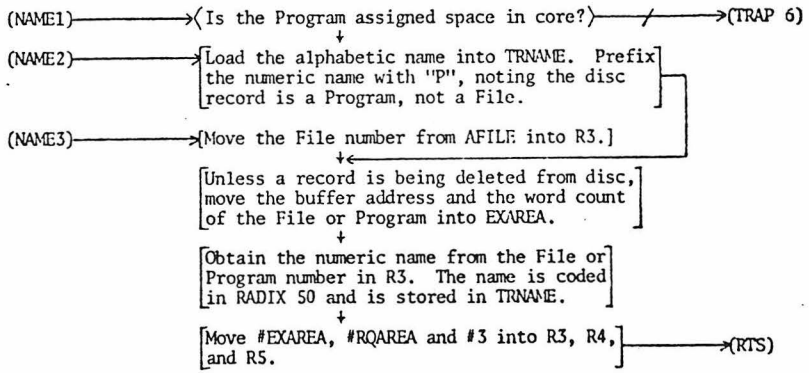
```
1                       .SBTTL  DISC MANAGEMENT ROUTINES
2
3
4 006544 010346 PGET:   MOV     R3,-(SP)
5 006546 006303         ASL     R3
6 006550 105763         TSTB    PRFILE+1(R3)
7 006554 001156         BNE     E06T
8
9 006556 004767         JSR     PC,NAME1
10 06562 004767         JSR     PC,READC
11 06566 103436         BCS     BDEND
12
13 06570 111667         MOVB    (SP),PROGNO
14 06574 006316         ASL     (SP)
15
16 06576 013705         MOV     @#4+EXAREA,R5
17 06602 014500         MOV     -(R5),R0
18 06604 006300         ASL     R0
19 06606 060500         ADD     R5,R0
20 06610 012740         MOV     #-1,-(R0)
21 06614 005725         TST     (R5)+
22
23 06616 011604 1$:     MOV     (SP),R4
24 06620 105715         TSTB    (R5)
25 06622 100414         BMI     GDEND
26
27 06624 010564         MOV     R5,PRCNT(R4)
28 06630 112504         MOVB    (R5)+,R4
29 06632 112503         MOVB    (R5)+,R3
30 06634 012502         MOV     (R5)+,R2
31 06636 004767         JSR     PC,ERRCHK
32 06642 000765         BR      1$
33
34
35 06644 010504 END:    MOV     R5,R4
36 06646 012774         MOV     #-1,@PRCNT(R4)
37
38 06654 005064 GDEND:  CLR     PRCNT(R4)
39 06660 105264         INCB    PRFILE+1(R4)
40 06664 000167 BDEND:  JMP     BDSTOP
41
42
43 06670 006303 PPUT:   ASL     R3
44 06672 105763         TSTB    PRFILE+1(R3)
45 06676 001505         BEQ     E06T
46
47 06700 004767         JSR     PC,NAME1
48 06704 004767         JSR     PC,WRITC
49 06710 000765         BR      BDEND
50
51 06712 004767 PRID:   JSR     PC,NAME2
52 06716 004767         JSR     PC,DELETE
53 06722 000760         BR      BDEND
```



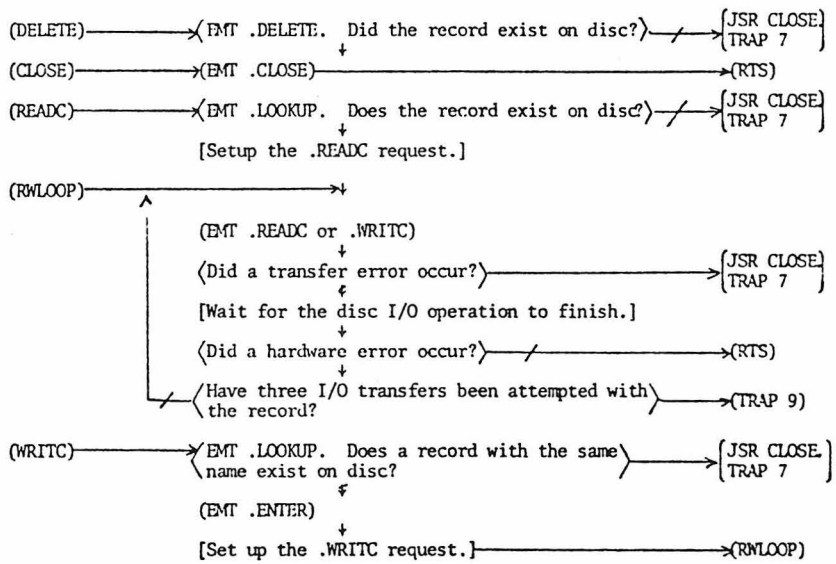
```
1 006724 004767 FGET:   JSR     PC,MANFL
2
3 006730 004767 1$:     JSR     PC,NAME3
4 006734 004767         JSR     PC,READC
5 006740 010277         MOV     R2,@4+EXAREA
6 006744 004767         JSR     PC,ENDFLT
7 006750 000767         BR      1$
8
9 006752 004767 FPUT:   JSR     PC,MANFL
10
11 06756 004767 2$:     JSR     PC,NAME3
12 06762 004767         JSR     PC,WRITC
13 06766 004767         JSR     PC,ENDFLT
14 06772 000771         BR      2$
15
16 06774 004767 FRID:   JSR     PC,MANFL
17
18 07000 004767 3$:     JSR     PC,NAME3
19 07004 004767         JSR     PC,DELETE
20 07010 004767         JSR     PC,ENDFLT
21 07014 000771         BR      3$
22
23
24 07016 010437 MANFL:  MOV     R4,@#2+TRNAME
25
26 07022 116701         MOVB    AFILE,R1
27 07026 001427         BEQ     6$
28 07030 060103         ADD     R1,R3
29 07032 110367         MOVB    R3,LFILE
30 07036 022703         CMP     #111.,R3
31 07042 002421         BLT     6$
32
33 07044 022700         CMP     #4.,R0
34 07050 001415         BEQ     5$
35
36 07052 116700         MOVB    AKEY,R0
37 07056 006300         ASL     R0
38 07060 005760         TST     PRCNT(R0)
39 07064 001010         BNE     6$
40
41 07066 005303 4$:     DEC     R3
42 07070 126763         CMPB    AKEY,FLKEY(R3)
43 07076 001003         BNE     6$
44 07100 020301         CMP     R3,R1
45 07102 003771         BGT     4$
46 07104 000207 5$:     RTS     PC
47
48 07106 005067 6$:     CLR     AFILE
49 07112 104406 E06T:   TRAP    6
50
51 07114 012700 ENDFLT: MOV     #AFILE,R0
52 07120 105210         INCB    (R0)
53 07122 126710         CMPB    LFILE,(R0)
54 07126 003002         BGT     7$
55 07130 005010         CLR     (R0)
56 07132 000654         BR      BDEND
57 07134 000207 7$:     RTS     PC
```



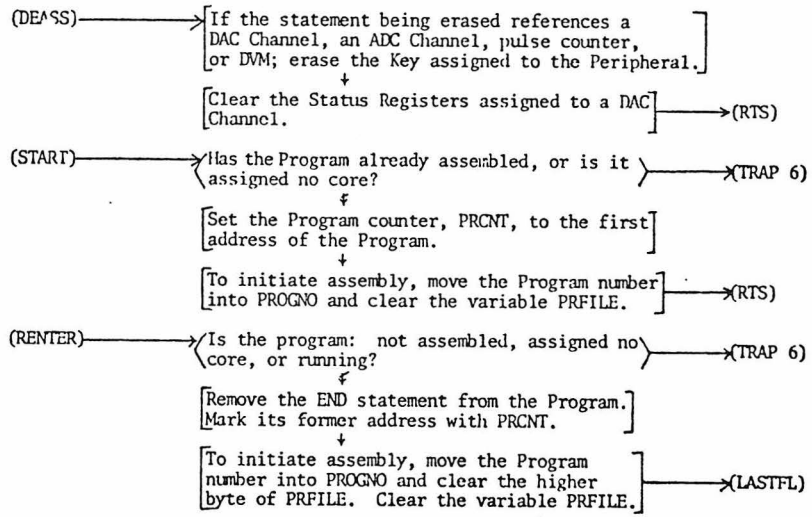
```
1 007136 016302 NAME1:  MOV     PRBUFF(R3),R2
2 007142 001763         BEQ     E06T
3 007144 006203         ASR     R3
4
5 007146 012705 NAME2:  MOV     #-22400.,R5
6 007152 010437         MOV     R4,@#2+TRNAME
7 007156 000407         BR      NAME
8
9 007160 005005 NAME3:  CLR     R5
10
11 07162 116703         MOVB    AFILE,R3
12 07166 010302         MOV     R3,R2
13 07170 006302         ASL     R2
14 07172 016202         MOV     FLBUFF(R2),R2
15
16 07176 010237 NAME:   MOV     R2,@#4+EXAREA
17 07202 001403         BEQ     1$
18 07204 016237         MOV     -2(R2),@#6+EXAREA
19
20 07212 062705 1$:     ADD     #49230.,R5
21 07216 022703         CMP     #100.,R3
22 07222 003004         BGT     2$
23 07224 062705         ADD     #1600.,R5
24 07230 162703         SUB     #100.,R3
25 07234 022703 2$:     CMP     #10.,R3
26 07240 003005         BGT     3$
27 07242 062705         ADD     #40.,R5
28 07246 162703         SUB     #10.,R3
29 07252 000770         BR      2$
30 07254 060305 3$:     ADD     R3,R5
31 07256 010537         MOV     R5,@#4+TRNAME
32
33 07262 112705         MOVB    #3,R5
34 07266 012704         MOV     #RQAREA,R4
35 07272 012703         MOV     #EXAREA,R3
36 07276 011202         MOV     (R2),R2
37 07300 000207         RTS     PC
```



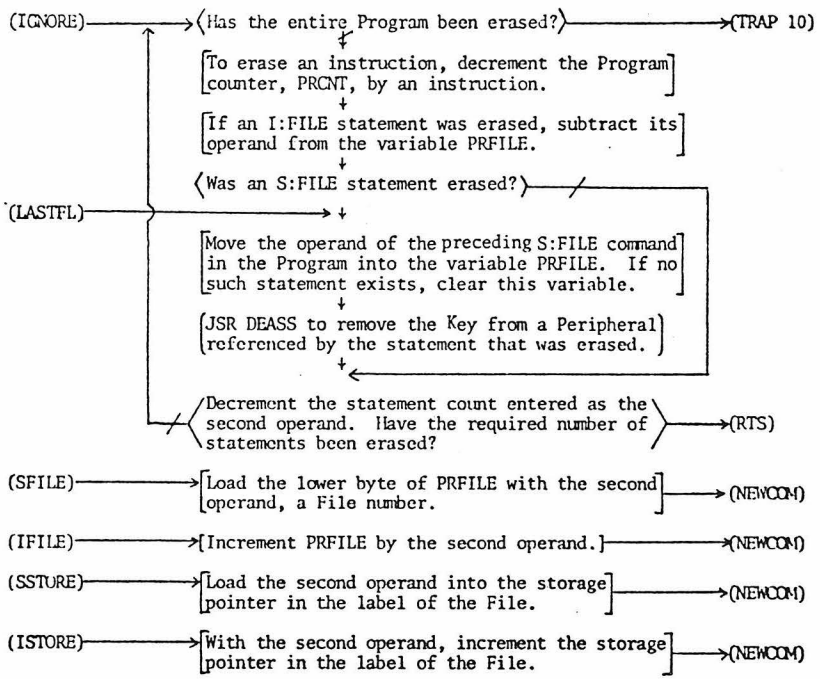
```
1 007302 010400 DELETE: MOV     R4,R0
2 007304 012710         MOV     #1,(R0)
3 007310 104375         EMT     375;    .DELETE
4 007312 103405         BCS     E07T
5
6 007314 012700 CLOSE:  MOV     #3001,R0
7 007320 104374         EMT     374;    .CLOSE
8 007322 000241         CLC
9 007324 000207         RTS     PC
10
11 07326 004767 E07T:   JSR     PC,CLOSE
12 07332 104407         TRAP    7
13
14
15 07334 010400 READC:  MOV     R4,R0
16 07336 012710         MOV     #401,(R0)
17 07342 104375         EMT     375;    .LOOKUP
18 07344 103770         BCS     E07T
19
20 07346 012713         MOV     #4001,(R3)
21
22 07352 010300 RWLOOP: MOV     R3,R0
23 07354 112767         MOVB    #1,FINIS
24 07362 104375         EMT     375;    .READC, .WRITC
25 07364 103760         BCS     E07T
26
27 07366 105767 2$:     TSTB    FINIS
28 07372 001750         BEQ     CLOSE
29 07374 100374         BPL     2$
30
31 07376 105305         DECB    R5
32 07400 003364         BGT     RWLOOP
33 07402 004767         JSR     PC,CLOSE
34 07406 104411         TRAP    11
35
36
37 07410 010400 WRITC:  MOV     R4,R0
38 07412 012710         MOV     #401,(R0)
39 07416 104375         EMT     375;    .LOOKUP
40 07420 103342         BCC     E07T
41 07422 004767         JSR     PC,CLOSE
42
43 07426 010400         MOV     R4,R0
44 07430 012710         MOV     #1001,(R0)
45 07434 104375         EMT     375;    .ENTER
46
47 07436 012713         MOV     #4401,(R3)
48 07442 000743         BR      RWLOOP
49
50
51 07444 105067 FINISH: CLRB    FINIS
52 07450 006000         ROR     R0
53 07452 103002         BCC     1$
54 07454 105367         DECB    FINIS
55 07460 000207 1$:     RTS     PC
```



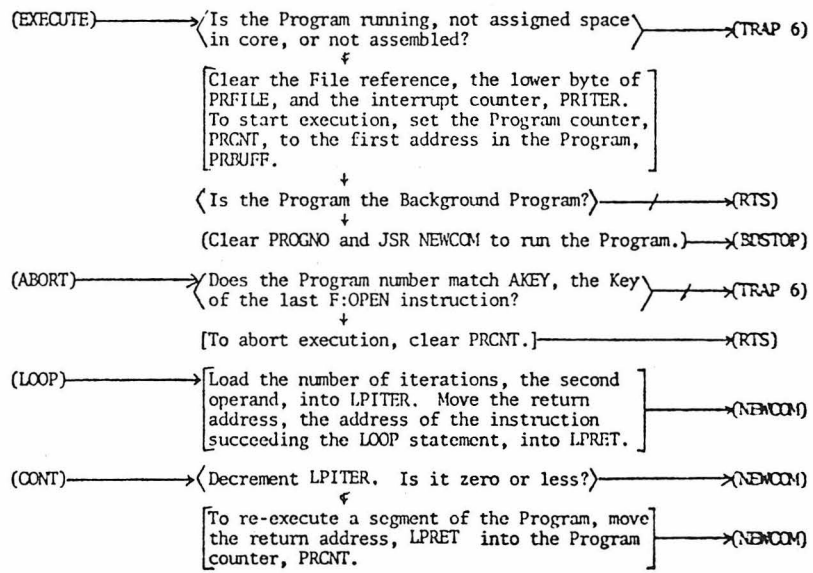
```
1                       .SBTTL  STORAGE CONTROL
2
3
4 007462 116001 DEASS:  MOVB    1(R0),R1
5 007466 122710         CMPB    #38.,(R0)
6 007472 001002         BNE     1$
7 007474 105061         CLRB    RDKEY+2(R1)
8 007500 122710 1$:     CMPB    #39.,(R0)
9 007504 001002         BNE     2$
10 07506 105061         CLRB    RDKEY+10.(R1)
11 07512 122710 2$:     CMPB    #40.,(R0)
12 07516 001002         BNE     3$
13 07520 105061         CLRB    RDKEY(R1)
14 07524 122710 3$:     CMPB    #42.,(R0)
15 07530 003012         BGT     4$
16 07532 122710         CMPB    #43.,(R0)
17 07536 002407         BLT     4$
18 07540 105061         CLRB    WRKEY(R1)
19 07544 006301         ASL     R1
20 07546 005061         CLR     164000(R1)
21 07552 005061         CLR     164020(R1)
22 07556 000207 4$:     RTS     PC
23
24
25 07560 006304 START:  ASL     R4
26 07562 105764         TSTB    PRFILE+1(R4)
27 07566 001041         BNE     E06P
28 07570 016400         MOV     PRBUFF(R4),R0
29 07574 001436         BEQ     E06P
30 07576 010064         MOV     R0,PRCNT(R4)
31
32 07602 006204 STASS:  ASR     R4
33 07604 110467         MOVB    R4,PROGNO
34 07610 105067         CLRB    PRFILE
35 07614 000207         RTS     PC
36
37 07616 006304 RENTER: ASL     R4
38 07620 105764         TSTB    PRFILE+1(R4)
39 07624 001422         BEQ     E06P
40 07626 016400         MOV     PRBUFF(R4),R0
41 07632 001417         BEQ     E06P
42 07634 005764         TST     PRCNT(R4)
43 07640 001014         BNE     E06P
44
45 07642 105710 1$:     TSTB    (R0)
46 07644 100403         BMI     2$
47 07646 062700         ADD     #4,R0
48 07652 000773         BR      1$
49
50 07654 105064 2$:     CLRB    PRFILE+1(R4)
51 07660 010405         MOV     R4,R5
52 07662 004767         JSR     PC,STASS
53 07666 005004         CLR     R4
54 07670 000422         BR      LASTFL
55
56 07672 104406 E06P:   TRAP    6
```



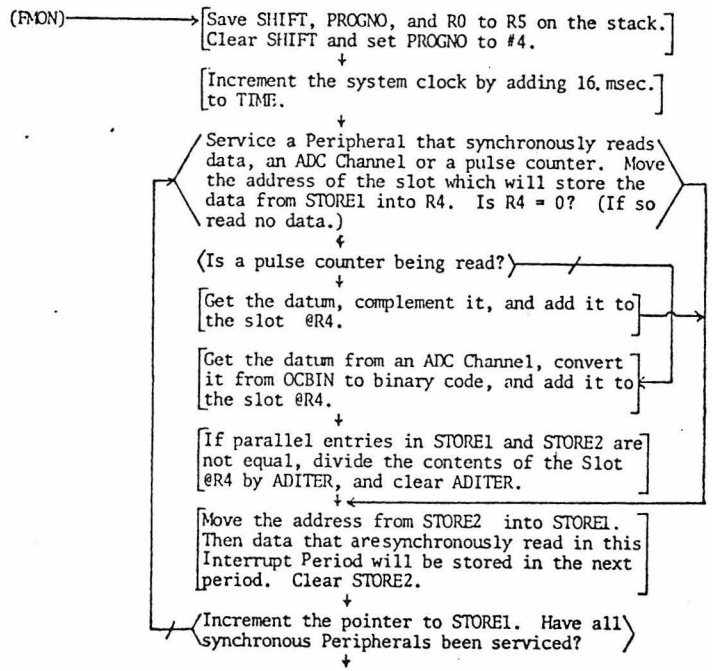
```
1 007674 016500 IGNORE: MOV     PRCNT(R5),R0
2 007700 020065         CMP     R0,PRBUFF(R5)
3 007704 101001         BHI     1$
4 007706 104410         .WORD   104410
5 007710 162700 1$:     SUB     #4,R0
6
7 007714 122710         CMPB    #33.,(R0)
8 007720 001003         BNE     2$
9 007722 166067         SUB     2(R0),PRFILE
10
11 07730 122710 2$:     CMPB    #32.,(R0)
12 07734 001014         BNE     SKIPFL
13
14 07736 010001 LASTFL: MOV     R0,R1
15 07740 105067         CLRB    PRFILE
16 07744 014102 3$:     MOV     -(R1),R2
17 07746 020165         CMP     R1,PRBUFF(R5)
18 07752 103405         BLO     SKIPFL
19 07754 022741         CMP     #32.,-(R1)
20 07760 001371         BNE     3$
21 07762 110267         MOVB    R2,PRFILE
22
23 07766 004767 SKIPFL: JSR     PC,DEASS
24 07772 010065         MOV     R0,PRCNT(R5)
25
26 07776 005304         DEC     R4
27 10000 003335         BGT     IGNORE
28 10002 000207         RTS     PC
29
30
31 10004 110465 SFILE:  MOVB    R4,PRFILE(R5)
32 10010 000167 EXITF1: JMP     NEWCOM
33
34 10014 060465 IFILE:  ADD     R4,PRFILE(R5)
35 10020 000773         BR      EXITF1
36
37
38 10022 010262 SSTORE: MOV     R2,2(R2)
39 10026 062762         ADD     #10.,2(R2)
40
41 10034 060462 ISTORE: ADD     R4,2(R2)
42 10040 000763         BR      EXITF1
```



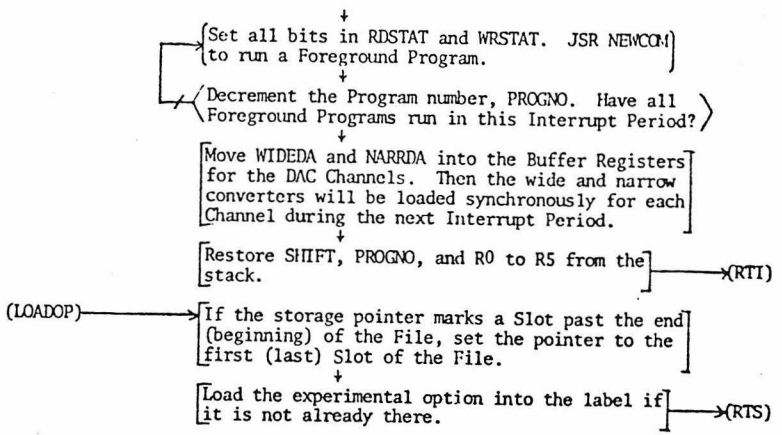
```
1                       .SBTTL  PROGRAM EXECUTION CONTROL
2
3
4 010042 010500 LOOP:   MOV     R5,R0
5 010044 006300         ASL     R0
6 010046 060003         ADD     R0,R3
7 010050 006303         ASL     R3
8 010052 010463         MOV     R4,LPITER(R3)
9 010056 016563         MOV     PRCNT(R5),LPRET(R3)
10 10064 000751         BR      EXITF1
11
12 10066 010500 CONT:   MOV     R5,R0
13 10070 006300         ASL     R0
14 10072 060004         ADD     R0,R4
15 10074 006304         ASL     R4
16 10076 005364         DEC     LPITER(R4)
17 10102 003742         BLE     EXITF1
18
19 10104 016465         MOV     LPRET(R4),PRCNT(R5)
20 10112 000736         BR      EXITF1
21
22
23 10114 006304 EXECUT: ASL     R4
24 10116 105764         TSTB    PRFILE+1(R4)
25 10122 001663         BEQ     E06P
26 10124 005764         TST     PRCNT(R4)
27 10130 001260         BNE     E06P
28
29 10132 105064         CLRB    PRFILE(R4)
30 10136 005064         CLR     PRITER(R4)
31 10142 016464         MOV     PRBUFF(R4),PRCNT(R4)
32 10150 001650         BEQ     E06P
33
34 10152 005704         TST     R4
35 10154 001315         BNE     EXITF1
36 10156 105067         CLRB    PROGNO
37 10162 004767         JSR     PC,NEWCOM
38 10166 000167         JMP     BDSTOP
39
40 10172 120467 ABORT:  CMPB    R4,AKEY
41 10176 001235         BNE     E06P
42
43 10200 006304         ASL     R4
44 10202 005064         CLR     PRCNT(R4)
45 10206 000207         RTS     PC
```



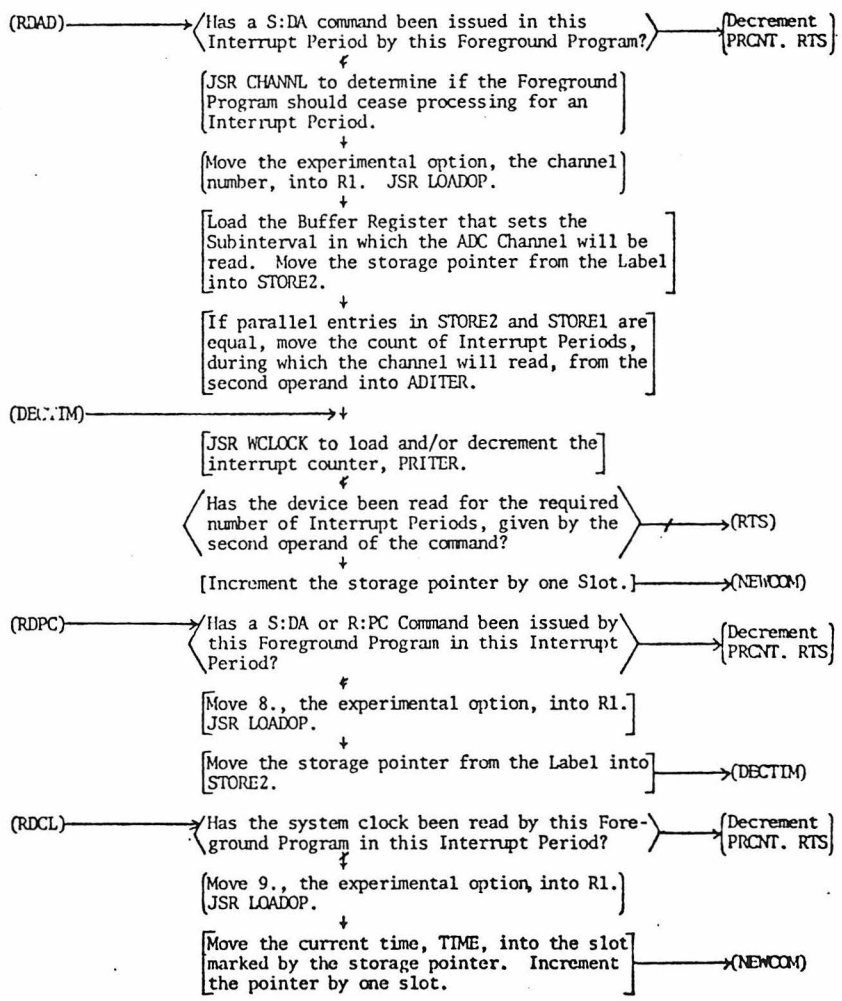
```
2                       .SBTTL  FOREGROUND MONITOR
3
4
5 010210 016746 FMON:   MOV     SHIFT,-(SP)
6 010214 010046         MOV     R0,-(SP)
7 010216 010146         MOV     R1,-(SP)
8 010220 010246         MOV     R2,-(SP)
9 010222 010346         MOV     R3,-(SP)
10 10224 010446         MOV     R4,-(SP)
11 10226 010546         MOV     R5,-(SP)
12 10230 012767         MOV     #2000,SHIFT
13
14 10236 012705         MOV     #TIME,R5
15 10242 062725         ADD     #16.,(R5)+
16 10246 005525         ADC     (R5)+
17 10250 005003         CLR     R3
18
19 10252 011504 1$:     MOV     (R5),R4
20 10254 001442         BEQ     3$
21
22 10256 022703         CMP     #16.,R3
23 10262 003005         BGT     2$
24
25 10264 016302         MOV     165034(R3),R2
26 10270 060224         ADD     R2,(R4)+
27 10272 005514         ADC     (R4)
28 10274 000432         BR      3$
29
30 10276 016302 2$:     MOV     165700(R3),R2
31 10302 005102         COM     R2
32 10304 032702         BIT     #4000,R2
33 10310 001402         BEQ     6$
34 10312 052702         BIS     #170000,R2
35 10316 062702 6$:     ADD     #4000,R2
36 10322 060224         ADD     R2,(R4)+
37 10324 005514         ADC     (R4)
38
39 10326 021565         CMP     (R5),20.(R5)
40 10332 001413         BEQ     3$
41 10334 011401         MOV     (R4),R1
42 10336 014402         MOV     -(R4),R2
43 10340 016300         MOV     ADITER(R3),R0
44 10344 001406         BEQ     3$
45 10346 004767         JSR     PC,DIVID
46 10352 010224         MOV     R2,(R4)+
47 10354 010114         MOV     R1,(R4)
48 10356 005063         CLR     ADITER(R3)
49
50 10362 016525 3$:     MOV     20.(R5),(R5)+
51 10366 005065         CLR     18.(R5)
52
53 10372 005723         TST     (R3)+
54 10374 022703         CMP     #20.,R3
55 10400 003324         BGT     1$
```



```
1 010402 012767 4$:     MOV     #-1,RDSTAT
2 010410 012767         MOV     #-1,WRSTAT
3 010416 004767         JSR     PC,NEWCOM
4
5 010422 105367         DECB    PROGNO
6 010426 003365         BGT     4$
7
8 010430 005003         CLR     R3
9 010432 016363 5$:     MOV     WIDEDA(R3),165000(R3)
10 10440 016363         MOV     NARRDA(R3),165020(R3)
11 10446 005723         TST     (R3)+
12 10450 022703         CMP     #10.,R3
13 10454 003366         BGT     5$
14
15 10456 012605         MOV     (SP)+,R5
16 10460 012604         MOV     (SP)+,R4
17 10462 012603         MOV     (SP)+,R3
18 10464 012602         MOV     (SP)+,R2
19 10466 012601         MOV     (SP)+,R1
20 10470 012600         MOV     (SP)+,R0
21 10472 012667         MOV     (SP)+,SHIFT
22 10476 000002         RTI
23
24
25                      .SBTTL  COMMANDS THAT READ DATA
26
27
28 10500 010246 LOADOP: MOV     R2,-(SP)
29 10502 062716         ADD     #10.,(SP)
30 10506 012246         MOV     (R2)+,-(SP)
31 10510 005316         DEC     (SP)
32 10512 006316         ASL     (SP)
33 10514 006316         ASL     (SP)
34 10516 066616         ADD     2(SP),(SP)
35 10522 026612         CMP     2(SP),(R2)
36 10526 101401         BLOS    3$
37 10530 011602         MOV     (SP),R2
38 10532 022622 3$:     CMP     (SP)+,(R2)+
39 10534 103002         BHIS    4$
40 10536 011662         MOV     (SP),-2(R2)
41 10542 005026 4$:     CLR     (SP)+
42
43 10544 105712         TSTB    (R2)
44 10546 100001         BPL     5$
45 10550 110112         MOVB    R1,(R2)
46 10552 000207 5$:     RTS     PC
```

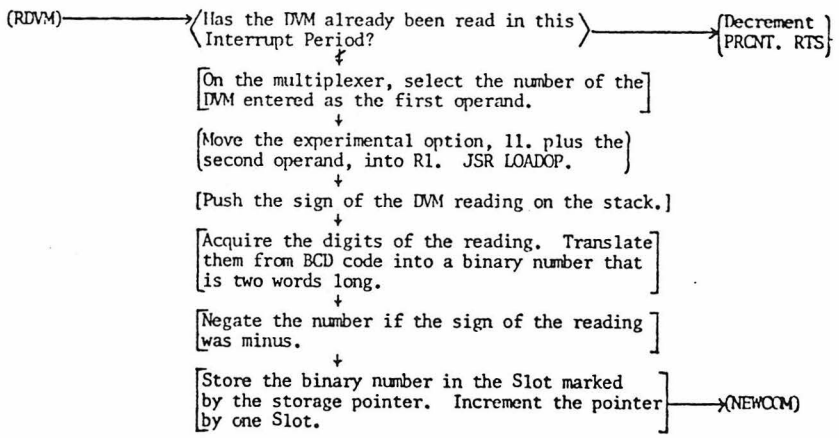


```
1 010554 105767 RDAD:   TSTB    WRSTAT
2 010560 100034         BPL     EXITH3
3
4 010562 012701         MOV     #1+RDSTAT,R1
5 010566 004767         JSR     PC,CHANNL
6
7 010572 010301         MOV     R3,R1
8 010574 004767         JSR     PC,LOADOP
9
10 10600 006300         ASL     R0
11 10602 010360         MOV     R3,165740(R0)
12 10606 014260         MOV     -(R2),STORE2(R0)
13
14 10612 026060         CMP     STORE2(R0),STORE1(R0)
15 10620 001002         BNE     DECTIM
16 10622 010460         MOV     R4,ADITER(R0)
17
18 10626 004767 DECTIM: JSR     PC,WCLOCK
19 10632 005765         TST     PRITER(R5)
20 10636 003161         BGT     EXITH2
21
22 10640 062712         ADD     #4,(R2)
23 10644 000530         BR      EXITH4
24
25 10646 105767 RDPC:   TSTB    WRSTAT
26 10652 100150 EXITH3: BPL     EXITH1
27 10654 032767         BIT     #40,RDSTAT
28 10662 001544         BEQ     EXITH1
29 10664 042767         BIC     #40,RDSTAT
30
31 10672 012701         MOV     #8.,R1
32 10676 004767         JSR     PC,LOADOP
33
34 10702 006303         ASL     R3
35 10704 014263         MOV     -(R2),STORE2+16.(R3)
36 10710 000746         BR      DECTIM
37
38 10712 032767 RDCL:   BIT     #100,RDSTAT
39 10720 001525         BEQ     EXITH1
40 10722 042767         BIC     #100,RDSTAT
41
42 10730 012701         MOV     #9.,R1
43 10734 004767         JSR     PC,LOADOP
44
45 10740 014204         MOV     -(R2),R4
46 10742 013724         MOV     @#TIME,(R4)+
47 10746 013724         MOV     @#2+TIME,(R4)+
48 10752 010412         MOV     R4,(R2)
49 10754 000464         BR      EXITH4
```

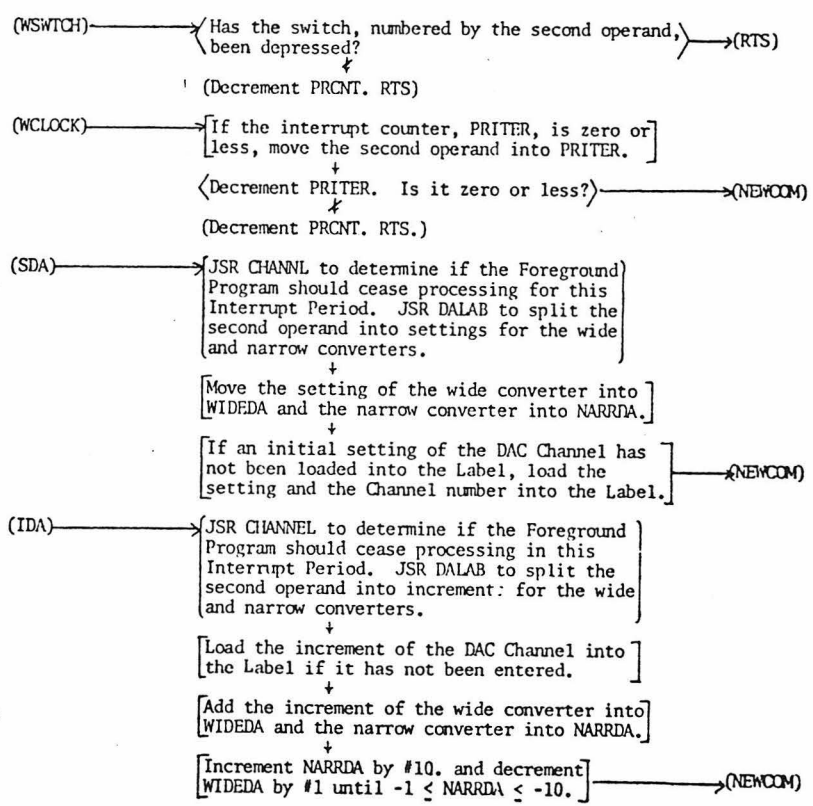


COMMANDS THAT READ DATA

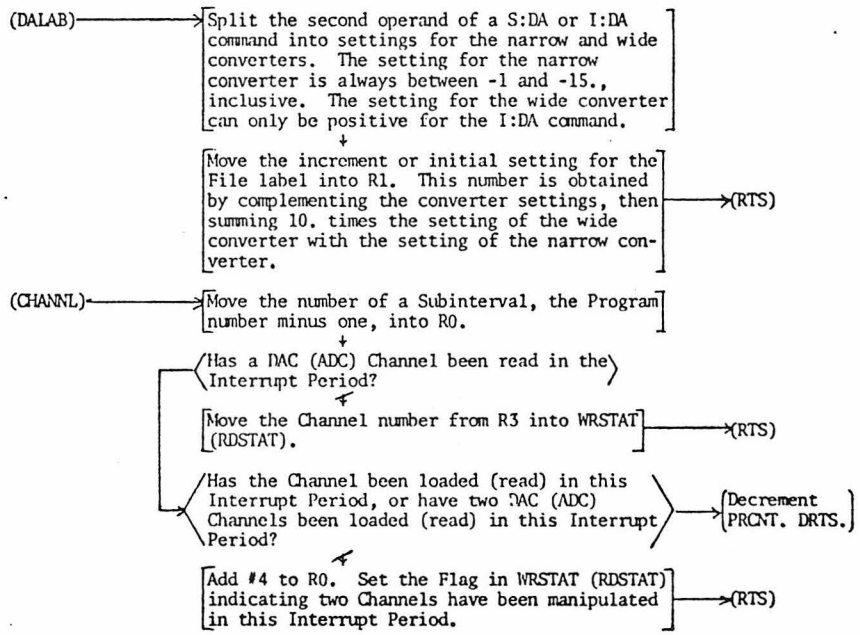
```
1 010756 005000 RDVM:   CLR     R0
2 010760 000261         SEC
3 010762 006100 1$:     ROL     R0
4 010764 005303         DEC     R3
5 010766 002375         BGE     1$
6 010770 030067         BIT     R0,RDSTAT
7 010774 001477         BEQ     EXITH1
8 010776 040067         BIC     R0,RDSTAT
9
10 11002 010037         MOV     R0,@#165224
11 11006 012737         MOV     #5,@#165222
12
13 11014 010401         MOV     R4,R1
14 11016 062701         ADD     #11.,R1
15 11022 004767         JSR     PC,LOADOP
16
17 11026 010205         MOV     R2,R5
18 11030 012703         MOV     #4,R3
19 11034 013746         MOV     @#165226,-(SP)
20 11040 011602         MOV     (SP),R2
21 11042 042702         BIC     #177776,R2
22 11046 005001         CLR     R1
23
24 11050 010337 2$:     MOV     R3,@#165222
25 11054 004767         JSR     PC,TMSTEN
26 11060 013700         MOV     @#165226,R0
27 11064 042700         BIC     #177760,R0
28 11070 060002         ADD     R0,R2
29 11072 005501         ADC     R1
30 11074 005303         DEC     R3
31 11076 100364         BPL     2$
32
33 11100 005037         CLR     @#165224
34 11104 032726         BIT     #2,(SP)+
35 11110 001002         BNE     3$
36 11112 004767         JSR     PC,NEGATE
37
38 11116 014504 3$:     MOV     -(R5),R4
39 11120 010224         MOV     R2,(R4)+
40 11122 010124         MOV     R1,(R4)+
41 11124 010415         MOV     R4,(R5)
42 11126 000167 EXITH4: JMP     NEWCOM
```



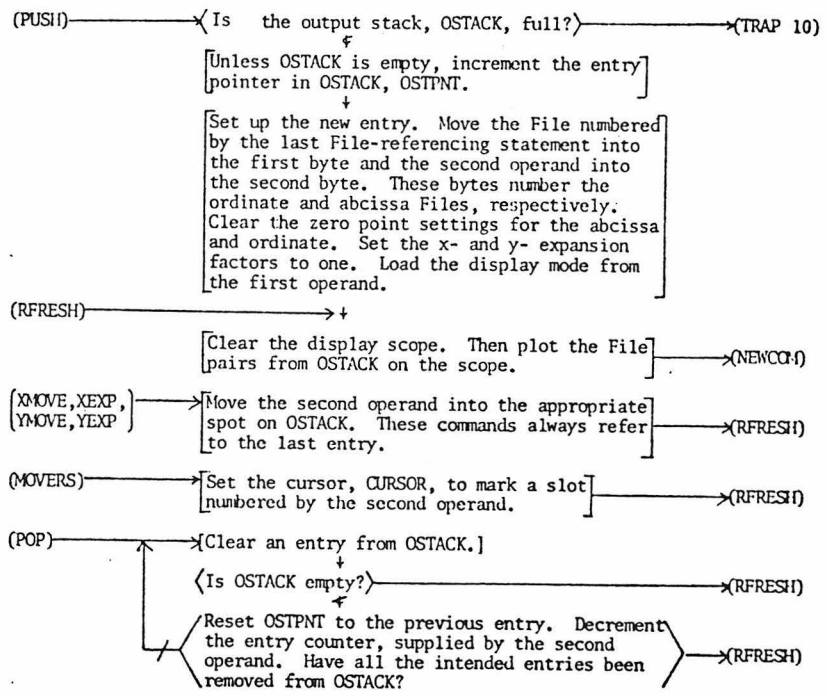
```
1                       .SBTTL  COMMANDS THAT CONTROL PERIPHERALS
2
3
4 011132 000261 WSWTCH: SEC
5 011134 006103 1$:     ROL     R3
6 011136 005304         DEC     R4
7 011140 100375         BPL     1$
8 011142 000303         SWAB    R3
9 011144 030337         BIT     R3,@#165230
10 11150 001411         BEQ     EXITH1
11 11152 000207         RTS     PC
12
13 11154 005765 WCLOCK: TST     PRITER(R5)
14 11160 003002         BGT     2$
15 11162 010465         MOV     R4,PRITER(R5)
16
17 11166 005365 2$:     DEC     PRITER(R5)
18 11172 003403         BLE     EXITH2
19 11174 162765 EXITH1: SUB     #4,PRCNT(R5)
20 11202 000207 EXITH2: RTS     PC
21
22
23 11204 012701 SDA:    MOV     #1+WRSTAT,R1
24 11210 004767         JSR     PC,CHANNL
25 11214 042767         BIC     #200,WRSTAT
26 11222 004767         JSR     PC,DALAB
27
28 11226 010463         MOV     R4,WIDEDA(R3)
29 11232 010563         MOV     R5,NARRDA(R3)
30
31 11236 062702         ADD     #6.,R2
32 11242 005712         TST     (R2)
33 11244 001330         BNE     EXITH4
34 11246 010112         MOV     R1,(R2)
35 11250 110342         MOVB    R3,-(R2)
36 11252 106212         ASRB    (R2)
37 11254 000724         BR      EXITH4
38
39 11256 012701 IDA:    MOV     #1+WRSTAT,R1
40 11262 004767         JSR     PC,CHANNL
41 11266 004767         JSR     PC,DALAB
42
43 11272 062702         ADD     #8.,R2
44 11276 005712         TST     (R2)
45 11300 001001         BNE     1$
46 11302 010112         MOV     R1,(R2)
47
48 11304 005204 1$:     INC     R4
49 11306 005205         INC     R5
50 11310 060463         ADD     R4,WIDEDA(R3)
51 11314 060563         ADD     R5,NARRDA(R3)
52
53 11320 022763 2$:     CMP     #177765,NARRDA(R3)
54 11326 003677         BLE     EXITH4
55 11330 062763         ADD     #12,NARRDA(R3)
56 11336 005363         DEC     WIDEDA(R3)
57 11342 000766         BR      2$
```



```
1 011344 006303 DALAB:  ASL     R3
2 011346 010405         MOV     R4,R5
3 011350 052705         BIS     #177760,R5
4 011354 006204         ASR     R4
5 011356 006204         ASR     R4
6 011360 006204         ASR     R4
7 011362 006204         ASR     R4
8 011364 022716         CMP     #IDA,(SP)
9 011370 103402         BLO     3$
10 11372 052704         BIS     #170000,R4
11
12 11376 005104 3$:     COM     R4
13 11400 005105         COM     R5
14 11402 010401         MOV     R4,R1
15 11404 006301         ASL     R1
16 11406 006301         ASL     R1
17 11410 060401         ADD     R4,R1
18 11412 006301         ASL     R1
19 11414 060501         ADD     R5,R1
20 11416 005104         COM     R4
21 11420 005105         COM     R5
22 11422 000207         RTS     PC
23
24 11424 116700 CHANNL: MOVB    PROGNO,R0
25 11430 005300         DEC     R0
26
27 11432 105711         TSTB    (R1)
28 11434 100002         BPL     1$
29 11436 110311         MOVB    R3,(R1)
30 11440 000207         RTS     PC
31
32 11442 120311 1$:     CMPB    R3,(R1)
33 11444 001410         BEQ     2$
34 11446 132741         BITB    #20,-(R1)
35 11452 001405         BEQ     2$
36 11454 142711         BICB    #20,(R1)
37 11460 062700         ADD     #4,R0
38 11464 000207         RTS     PC
39
40 11466 005026 2$:     CLR     (SP)+
41 11470 000167         JMP     EXITH1
```

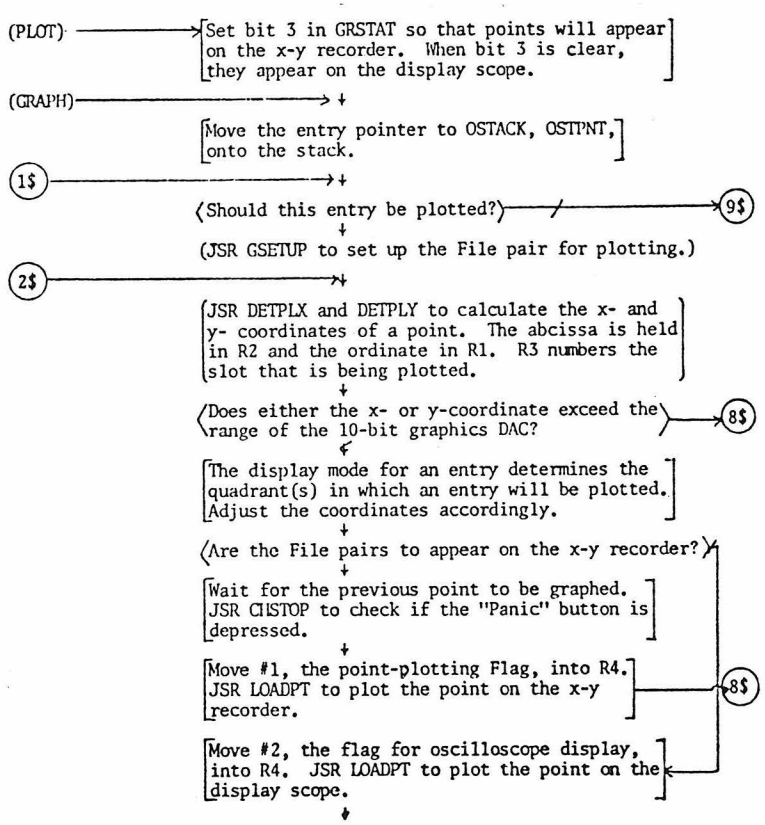


```
2                       .SBTTL  GRAPHICS CONTROL
3
4
5 011474 016702 PUSH:   MOV     OSTPNT,R2
6 011500 020227         CMP     R2,#84.+OSTACK
7 011504 103401         BLO     1$
8 011506 104410         TRAP    10
9
10 11510 105712 1$:     TSTB    (R2)
11 11512 001404         BEQ     2$
12 11514 062702         ADD     #12.,R2
13 11520 010267         MOV     R2,OSTPNT
14
15 11524 116722 2$:     MOVB    PRFILE,(R2)+
16 11530 110422         MOVB    R4,(R2)+
17 11532 012722         MOV     #1,(R2)+
18 11536 005022         CLR     (R2)+
19 11540 012722         MOV     #1,(R2)+
20 11544 005022         CLR     (R2)+
21 11546 010312         MOV     R3,(R2)
22
23 11550 012737 RFRESH: MOV     #100,@#165236
24 11556 005037         CLR     @#165236
25 11562 032737 3$:     BIT     #4,@#165230
26 11570 001774         BEQ     3$
27 11572 004767         JSR     PC,GRAPH
28 11576 000167         JMP     NEWCOM
29
30 11602 016702 POP:    MOV     OSTPNT,R2
31 11606 005012         CLR     (R2)
32 11610 162702         SUB     #12.,R2
33
34 11614 020227         CMP     R2,#OSTACK
35 11620 103753         BLO     RFRESH
36
37 11622 010267         MOV     R2,OSTPNT
38 11626 005304         DEC     R4
39 11630 003364         BGT     POP
40 11632 000746         BR      RFRESH
41
42
43 11634 005723 XMOVE:  TST     (R3)+
44 11636 005723 XEXP:   TST     (R3)+
45 11640 005723 YMOVE:  TST     (R3)+
46 11642 005723 YEXP:   TST     (R3)+
47
48 11644 066703         ADD     OSTPNT,R3
49 11650 010413         MOV     R4,(R3)
50 11652 000736         BR      RFRESH
51
52 11654 010467 MOVCRS: MOV     R4,CURSOR
53 11660 000733         BR      RFRESH
```

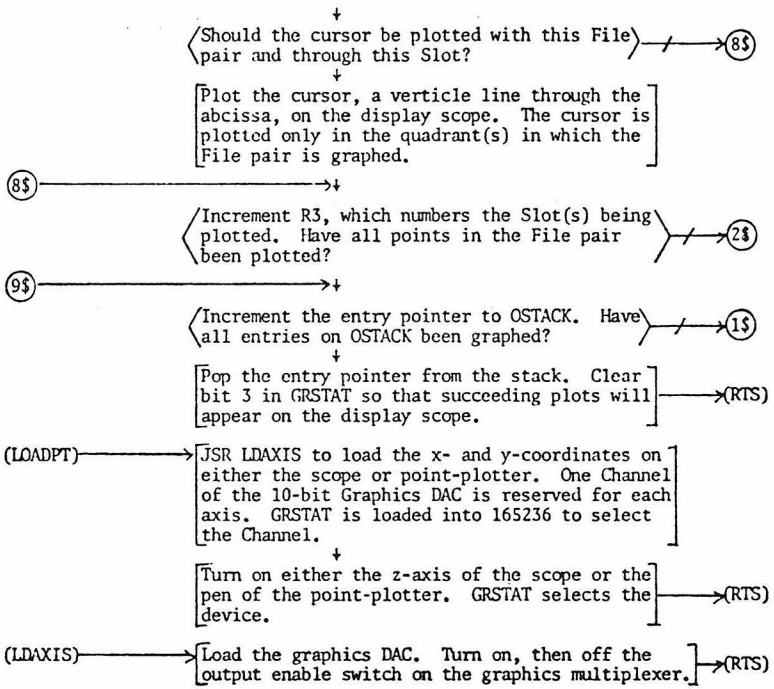


GRAPHICS CONTROL

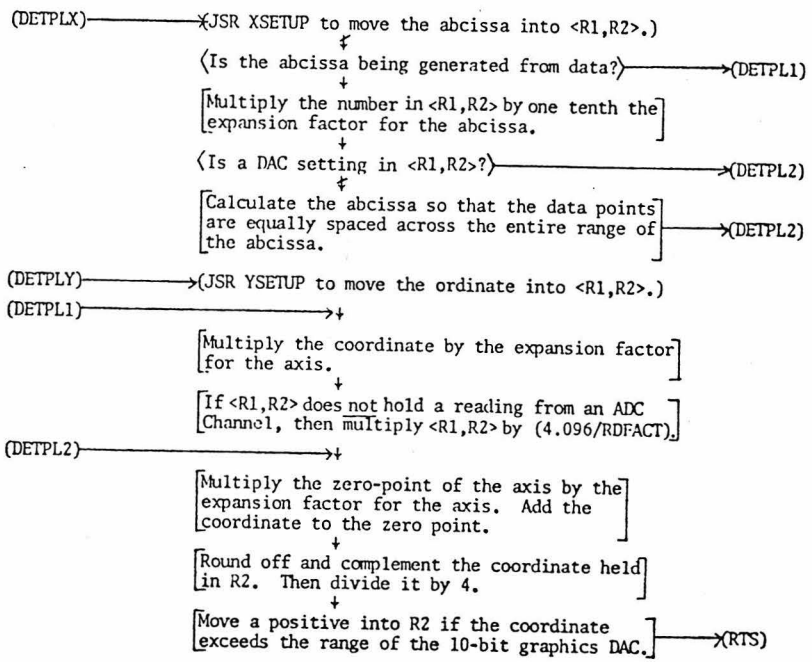
```
1 011662 052767 PLOT:   BIS     #10,GRSTAT
2
3 011670 016746 GRAPH:  MOV     OSTPNT,-(SP)
4
5 011674 011605 1$:     MOV     (SP),R5
6 011676 005765         TST     10.(R5)
7 011702 001516         BEQ     9$
8 011704 004767         JSR     PC,GSETUP
9
10 11710 011605 2$:     MOV     (SP),R5
11 11712 004767         JSR     PC,DETPLY
12 11716 003104         BGT     8$
13 11720 010246         MOV     R2,-(SP)
14 11722 004767         JSR     PC,DETPLX
15 11726 012601         MOV     (SP)+,R1
16 11730 005702         TST     R2
17 11732 003075         BGT     8$
18
19 11734 022715         CMP     #5,(R5)
20 11740 001414         BEQ     4$
21 11742 006201         ASR     R1
22 11744 006202         ASR     R2
23 11746 022715         CMP     #3,(R5)
24 11752 003402         BLE     3$
25 11754 062701         ADD     #-1000,R1
26 11760 032715 3$:     BIT     #2,(R5)
27 11764 001002         BNE     4$
28 11766 062702         ADD     #-1000,R2
29
30 11772 005767 4$:     TST     GRSTAT
31 11776 001417         BEQ     6$
32
33 12000 004767 5$:     JSR     PC,PANIC
34 12004 032737         BIT     #2,@#165230
35 12012 001372         BNE     5$
36 12014 032737         BIT     #1,@#165230
37 12022 001766         BEQ     5$
38
39 12024 012704         MOV     #1,R4
40 12030 004767         JSR     PC,LOADPT
41 12034 000435         BR      8$
42
43 12036 012704 6$:     MOV     #2,R4
44 12042 004767         JSR     PC,LOADPT
45
46 12046 021667         CMP     (SP),OSTPNT
47 12052 001026         BNE     8$
48 12054 020367         CMP     R3,CURSOR
49 12060 001023         BNE     8$
```



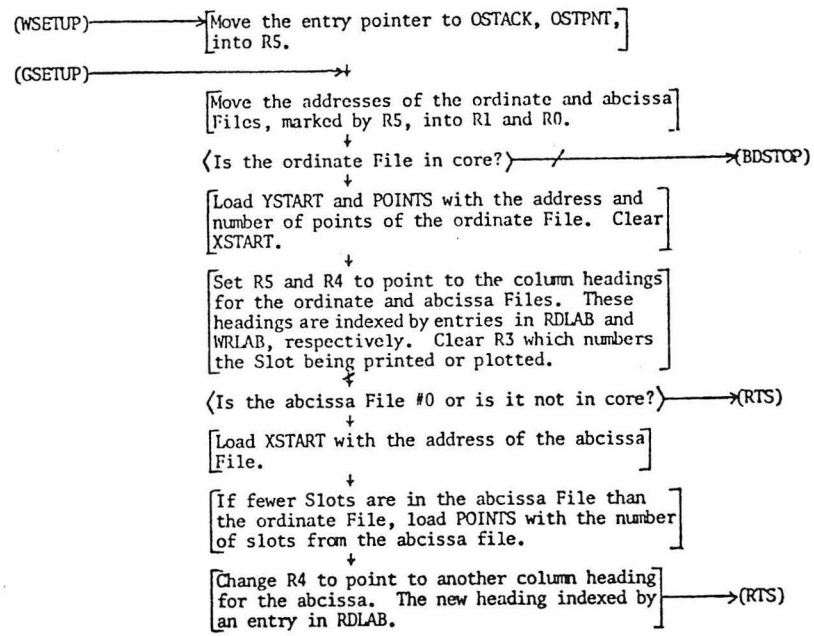
```
1 012062 005001          CLR     R1
2 012064 012746          MOV     #200,-(SP)
3 012070 022715          CMP     #5,(R5)
4 012074 001406          BEQ     7$
5 012076 006216          ASR     (SP)
6 012100 022715          CMP     #3,(R5)
7 012104 003402          BLE     7$
8 012106 062701          ADD     #-1000,R1
9 012112 004767 7$:      JSR     PC,LOADPT
10 12116 062701          ADD     #-10,R1
11 12122 005316          DEC     (SP)
12 12124 003372          BGT     7$
13 12126 005026          CLR     (SP)+
14
15 12130 005203 8$:      INC     R3
16 12132 026703          CMP     POINTS,R3
17 12136 003264          BGT     2$
18
19 12140 162716 9$:      SUB     #12.,(SP)
20 12144 021627          CMP     (SP),#OSTACK
21 12150 103251          BHIS    1$
22
23 12152 005026          CLR     (SP)+
24 12154 005067          CLR     GRSTAT
25 12160 000207          RTS     PC
26
27
28 12162 012700 LOADPT:  MOV     #GRSTAT,R0
29 12166 010237          MOV     R2,@#165234
30 12172 004767          JSR     PC,LDAXIS
31 12176 052710          BIS     #4,(R0)
32 12202 010137          MOV     R1,@#165234
33 12206 004767          JSR     PC,LDAXIS
34
35 12212 050410          BIS     R4,(R0)
36 12214 011037          MOV     (R0),@#165236
37 12220 040410          BIC     R4,(R0)
38 12222 011037 EXITG1:  MOV     (R0),@#165236
39 12226 000207          RTS     PC
40
41 12230 052710 LDAXIS:  BIS     #40,(R0)
42 12234 011037          MOV     (R0),@#165236
43 12240 042710          BIC     #44,(R0)
44 12244 000766          BR      EXITG1
```



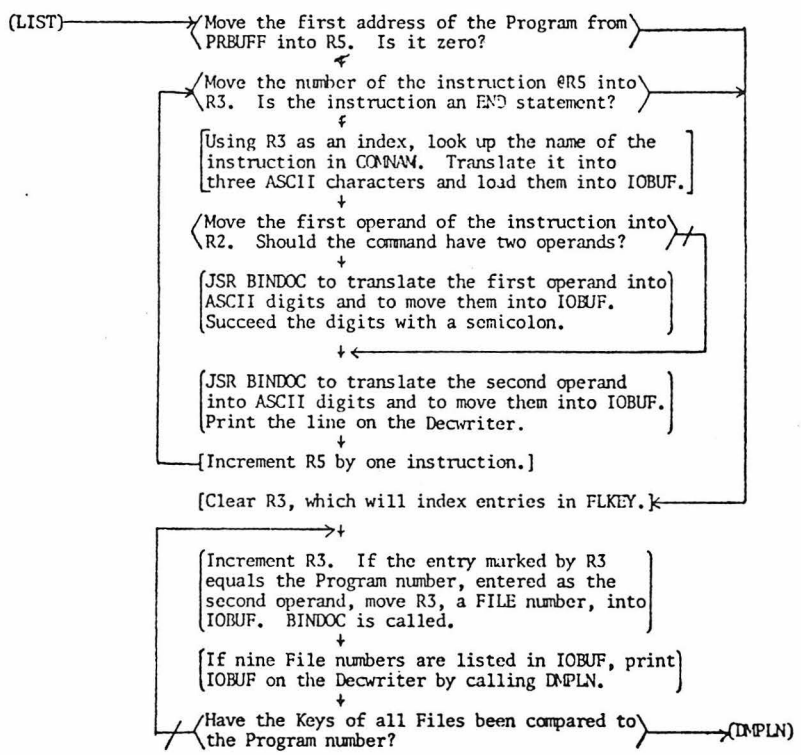
```
1 012246 004767 DETPLX: JSR     PC,XSETUP
2 012252 005700         TST     R0
3 012254 001022         BNE     DETPL1
4
5 012256 011500         MOV     (R5),R0
6 012260 004767         JSR     PC,MULT
7 012264 012700         MOV     #10.,R0
8
9 012270 005704         TST     R4
10 12272 100033         BPL     DETPL2
11 12274 010302         MOV     R3,R2
12 12276 005001         CLR     R1
13 12300 016700         MOV     POINTS,R0
14 12304 112767         MOVB    #12.,SHIFT
15 12312 000423         BR      DETPL2
16
17 12314 005725 DETPLY: TST     (R5)+
18 12316 004767         JSR     PC,YSETUP
19
20 12322 011500 DETPL1: MOV     (R5),R0
21 12324 004767         JSR     PC,MULT
22
23 12330 022704         CMP     #8.,R4
24 12334 003014         BGT     DETHOP
25 12336 112767         MOVB    #12.,SHIFT
26 12344 012700         MOV     #1000.,R0
27 12350 004767         JSR     PC,DIVID
28 12354 006304         ASL     R4
29 12356 016400         MOV     RDFACT(R4),R0
30 12362 004767 DETPL2: JSR     PC,DIVID
31
32 12366 010146 DETHOP: MOV     R1,-(SP)
33 12370 010246         MOV     R2,-(SP)
34 12372 112767         MOVB    #-3.,SHIFT
35 12400 012500         MOV     (R5)+,R0
36 12402 005001         CLR     R1
37 12404 012502         MOV     (R5)+,R2
38 12406 100001         BPL     3$
39 12410 005301         DEC     R1
40 12412 004767 3$:     JSR     PC,MULT
41
42 12416 062602         ADD     (SP)+,R2
43 12420 005501         ADC     R1
44 12422 062601         ADD     (SP)+,R1
45 12424 001007         BNE     4$
46
47 12426 006202         ASR     R2
48 12430 005202         INC     R2
49 12432 006202         ASR     R2
50 12434 005102         COM     R2
51 12436 022702         CMP     #-2000,R2
52 12442 103402         BLO     5$
53 12444 012702 4$:     MOV     #1,R2
54 12450 000207 5$:     RTS     PC
```



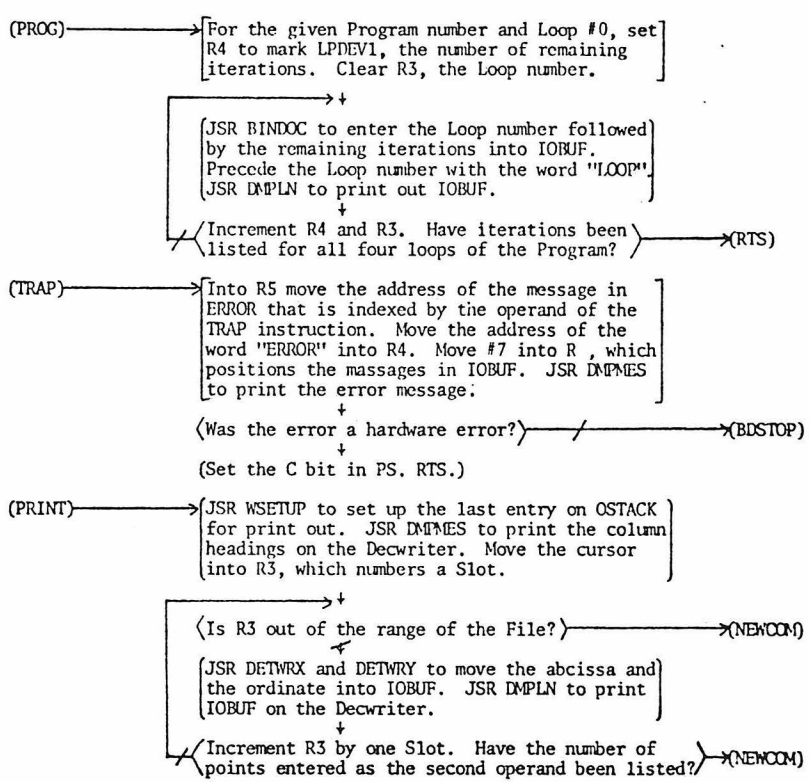
```
1 012452 016705 WSETUP: MOV     OSTPNT,R5
2
3 012456 012702 GSETUP: MOV     #YSTART,R2
4 012462 112501         MOVB    (R5)+,R1
5 012464 006301         ASL     R1
6 012466 016101         MOV     FLBUFF(R1),R1
7 012472 010122         MOV     R1,(R2)+
8 012474 001002         BNE     1$
9 012476 000167         JMP     BDSTOP
10
11 12502 112500 1$:     MOVB    (R5)+,R0
12 12504 012122         MOV     (R1)+,(R2)+
13
14 12506 005721         TST     (R1)+
15 12510 112103         MOVB    (R1)+,R3
16 12512 116304         MOVB    RDLAB(R3),R4
17 12516 112103         MOVB    (R1)+,R3
18 12520 116305         MOVB    WRLAB(R3),R5
19 12524 005003         CLR     R3
20
21 12526 006300         ASL     R0
22 12530 016001         MOV     FLBUFF(R0),R1
23 12534 010112         MOV     R1,(R2)
24 12536 001407         BEQ     3$
25
26 12540 021142         CMP     (R1),-(R2)
27 12542 002001         BGE     2$
28 12544 011112         MOV     (R1),(R2)
29
30 12546 116105 2$:     MOVB    4(R1),R5
31 12552 116505         MOVB    RDLAB(R5),R5
32
33 12556 062704 3$:     ADD     #LABTAB,R4
34 12562 062705         ADD     #LABTAB,R5
35 12566 012700         MOV     #13.,R0
36 12572 000207         RTS     PC
```



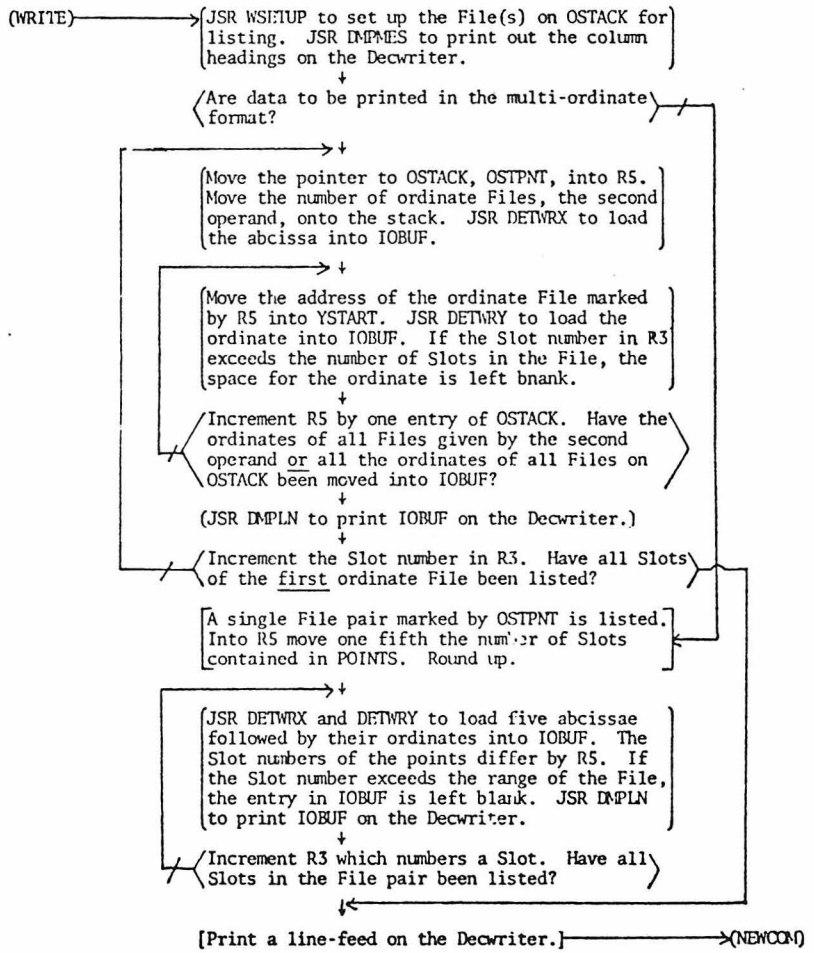
```
1                       .SBTTL  LISTING CONTROL
2
3
4 012574 006304 LIST:   ASL     R4
5 012576 016405         MOV     PRBUFF(R4),R5
6 012602 001447         BEQ     6$
7
8 012604 111503 1$:     MOVB    (R5),R3
9 012606 100415         BMI     6$
10
11 12610 006303         ASL     R3
12 12612 016302         MOV     COMNAM(R3),R2
13 12616 006302         ASL     R2
14 12620 012703         MOV     #IOBUF,R3
15 12624 005001 2$:     CLR     R1
16 12626 012700         MOV     #32.,R0
17 12632 004767         JSR     PC,MULT
18 12636 062701         ADD     #72,R1
19 12642 110123         MOVB    R1,(R3)+
20 12644 020327         CMP     R3,#3+IOBUF
21 12650 103765         BLO     2$
22
23 12652 112503         MOVB    (R5)+,R3
24 12654 112502         MOVB    (R5)+,R2
25 12656 105763         TSTB    COMFL1(R3)
26 12662 100006         BPL     3$
27
28 12664 004767         JSR     PC,BINDOC
29 12670 112737         MOVB    #';,@#13.+IOBUF
30 12676 000403         BR      4$
31 12700 062767 3$:     ADD     #13.,OUTPNT
32
33 12706 012502 4$:     MOV     (R5)+,R2
34 12710 004767         JSR     PC,BINDOC
35 12714 004767         JSR     PC,DMPLN
36 12720 000731         BR      1$
37
38 12722 006204 6$:     ASR     R4
39 12724 005003         CLR     R3
40
41 12726 005203 7$:     INC     R3
42 12730 120463         CMPB    R4,FLKEY(R3)
43 12734 001012         BNE     8$
44 12736 010302         MOV     R3,R2
45 12740 004767         JSR     PC,BINDOC
46
47 12744 022767         CMP     #117.,OUTPNT
48 12752 003003         BGT     8$
49 12754 004767         JSR     PC,DMPLN
50 12760 000762         BR      7$
51
52 12762 022703 8$:     CMP     #111.,R3
53 12766 003357         BGT     7$
54 12770 000167         JMP     DMPLN
```



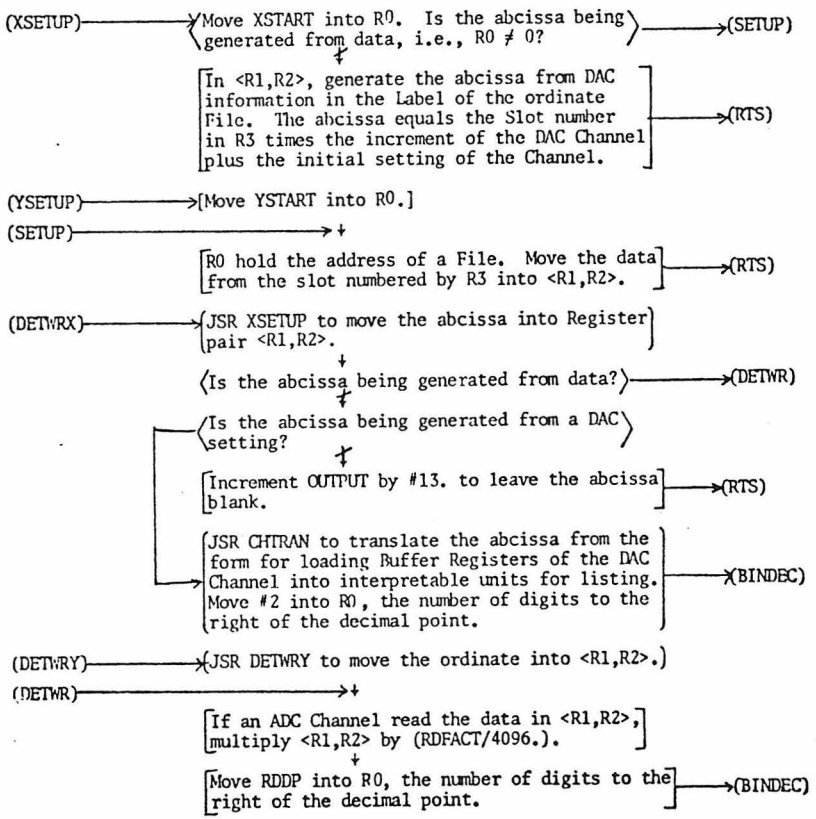
```
1 012774 006304 PROG:   ASL     R4
2 012776 006304         ASL     R4
3 013000 006304         ASL     R4
4 013002 062704         ADD     #LPITER,R4
5 013006 005003         CLR     R3
6
7 013010 010302 2$:     MOV     R3,R2
8 013012 004767         JSR     PC,BINDOC
9 013016 012402         MOV     (R4)+,R2
10 13020 004767         JSR     PC,BINDOC
11 13024 012737         MOV     #'LO,@#IOBUF
12 13032 012737         MOV     #'OP,@#2+IOBUF
13 13040 004767         JSR     PC,DMPLN
14
15 13044 005203         INC     R3
16 13046 022703         CMP     #4,R3
17 13052 003356         BGT     2$
18 13054 000207         RTS     PC
19
20
21 13056 013700 TRAP:   MOV     @#54,R0
22 13062 005060         CLR     256(R0)
23
24 13066 162716         SUB     #2,(SP)
25 13072 113605         MOVB    @(SP)+,R5
26 13074 010516         MOV     R5,(SP)
27 13076 006305         ASL     R5
28 13100 006305         ASL     R5
29 13102 062605         ADD     (SP)+,R5
30 13104 012704         MOV     #ERROR,R4
31 13110 060405         ADD     R4,R5
32 13112 012700         MOV     #7,R0
33 13116 004767         JSR     PC,DMPMES
34
35 13122 022705         CMP     #40.+ERROR,R5
36 13126 001402         BEQ     1$
37 13130 000167         JMP     BDSTOP
38 13134 000261 1$:     SEC
39 13136 000207         RTS     PC
40
41
42 13140 010446 PRINT:  MOV     R4,-(SP)
43 13142 004767         JSR     PC,WSETUP
44 13146 004767         JSR     PC,DMPMES
45 13152 016703         MOV     CURSOR,R3
46
47 13156 020367 3$:     CMP     R3,POINTS
48 13162 002120         BGE     EXITW2
49
50 13164 004767         JSR     PC,DETWRX
51 13170 004767         JSR     PC,DETWRY
52 13174 004767         JSR     PC,DMPLN
53
54 13200 005203         INC     R3
55 13202 105316         DECB    (SP)
56 13204 003364         BGT     3$
57 13206 000506         BR      EXITW2
```



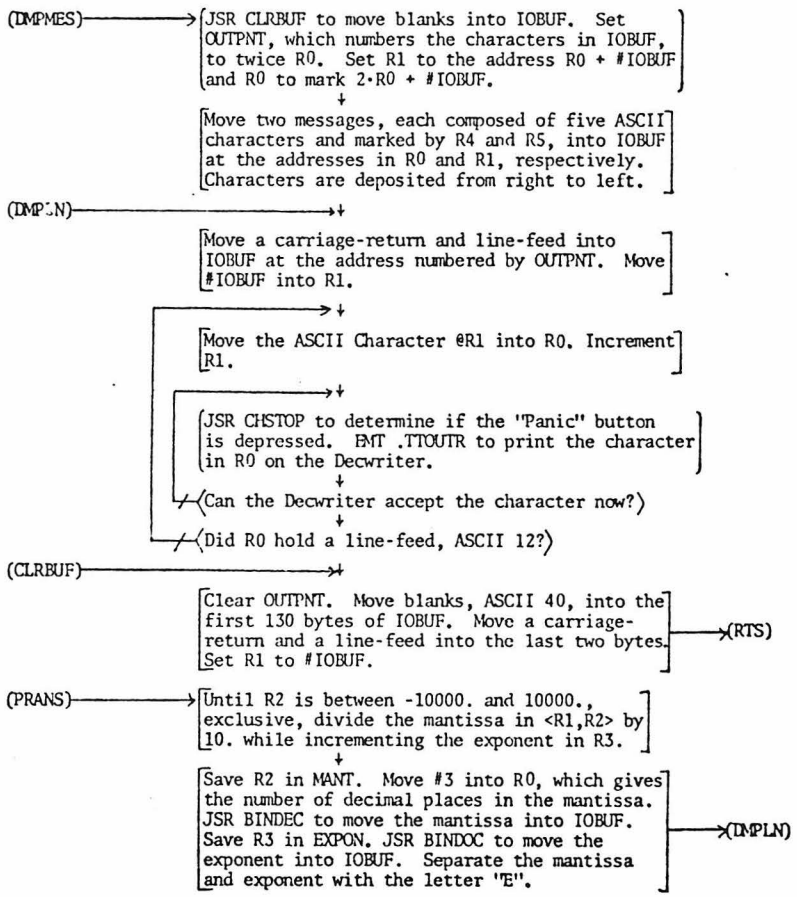
```
1 013210 010446 WRITE:  MOV     R4,-(SP)
2 013212 004767         JSR     PC,WSETUP
3 013216 004767         JSR     PC,DMPMES
4
5 013222 011600 1$:     MOV     (SP),R0
6 013224 001447         BEQ     5$
7 013226 016705         MOV     OSTPNT,R5
8 013232 111502         MOVB    (R5),R2
9 013234 006302         ASL     R2
10 13236 016267         MOV     FLBUFF(R2),YSTART
11 13244 010046         MOV     R0,-(SP)
12 13246 004767         JSR     PC,DETWRX
13
14 13252 111502 2$:     MOVB    (R5),R2
15 13254 006302         ASL     R2
16 13256 016267         MOV     FLBUFF(R2),YSTART
17 13264 027703         CMP     @YSTART,R3
18 13270 003403         BLE     3$
19 13272 004767         JSR     PC,DETWRY
20 13276 000403         BR      8$
21 13300 062767 3$:     ADD     #13.,OUTPNT
22
23 13306 162705 8$:     SUB     #12.,R5
24 13312 020527         CMP     R5,#OSTACK
25 13316 103402         BLO     4$
26 13320 005316         DEC     (SP)
27 13322 003353         BGT     2$
28
29 13324 005026 4$:     CLR     (SP)+
30 13326 004767         JSR     PC,DMPLN
31 13332 005203         INC     R3
32 13334 026703         CMP     POINTS,R3
33 13340 003330         BGT     1$
34 13342 000430         BR      EXITW2
35
36 13344 016702 5$:     MOV     POINTS,R2
37 13350 005001         CLR     R1
38 13352 062702         ADD     #2,R2
39 13356 012700         MOV     #5,R0
40 13362 004767         JSR     PC,DIVID
41 13366 010205         MOV     R2,R5
42
43 13370 011603 6$:     MOV     (SP),R3
44 13372 004767 7$:     JSR     PC,DETWRX
45 13376 004767         JSR     PC,DETWRY
46 13402 060503         ADD     R5,R3
47 13404 026703         CMP     POINTS,R3
48 13410 003370         BGT     7$
49
50 13412 004767         JSR     PC,DMPLN
51 13416 005216         INC     (SP)
52 13420 020516         CMP     R5,(SP)
53 13422 003362         BGT     6$
54
55 13424 004767 EXITW2: JSR     PC,DMPLN
56 13430 005026         CLR     (SP)+
57 13432 000167         JMP     NEWCOM
```



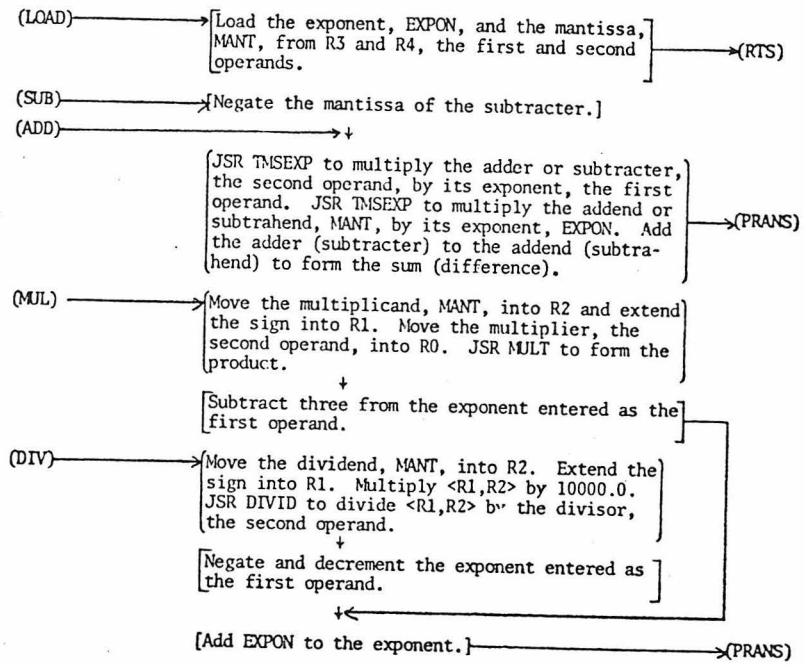
```
1 013436 016700 XSETUP: MOV     XSTART,R0
2 013442 001022         BNE     SETUP
3
4 013444 016704         MOV     YSTART,R4
5 013450 005001         CLR     R1
6 013452 016402         MOV     8.(R4),R2
7 013456 100001         BPL     1$
8 013460 005301         DEC     R1
9 013462 010300 1$:     MOV     R3,R0
10 13464 004767         JSR     PC,MULT
11 13470 066402         ADD     6(R4),R2
12 13474 005501         ADC     R1
13 13476 116404         MOVB    5(R4),R4
14 13502 000207         RTS     PC
15
16 13504 016700 YSETUP: MOV     YSTART,R0
17
18 13510 116004 SETUP:  MOVB    4(R0),R4
19 13514 062700         ADD     #10.,R0
20 13520 010301         MOV     R3,R1
21 13522 006301         ASL     R1
22 13524 006301         ASL     R1
23 13526 060001         ADD     R0,R1
24 13530 012102         MOV     (R1)+,R2
25 13532 011101         MOV     (R1),R1
26 13534 000207         RTS     PC
27
28
29 13536 004767 DETWRX: JSR     PC,XSETUP
30
31 13542 005700         TST     R0
32 13544 001021         BNE     DETWR
33 13546 005704         TST     R4
34 13550 100004         BPL     2$
35 13552 062767         ADD     #13.,OUTPNT
36 13560 000207         RTS     PC
37
38 13562 004767 2$:     JSR     PC,CHTRAN
39 13566 012700         MOV     #100.,R0
40 13572 004767         JSR     PC,DIVID
41 13576 012700         MOV     #2,R0
42 13602 000420         BR      EXITW1
43
44 13604 004767 DETWRY: JSR     PC,YSETUP
45
46 13610 022704 DETWR:  CMP     #8.,R4
47 13614 003411         BLE     3$
48 13616 006304         ASL     R4
49 13620 016400         MOV     RDFACT(R4),R0
50 13624 112767         MOVB    #-12.,SHIFT
51 13632 004767         JSR     PC,MULT
52 13636 006204         ASR     R4
53
54 13640 116400 3$:     MOVB    RDDP(R4),R0
55 13644 000167 EXITW1: JMP     BINDEC
```



```
1 013650 004767 DMPMES: JSR     PC,CLRBUF
2 013654 010067         MOV     R0,OUTPNT
3 013660 006367         ASL     OUTPNT
4 013664 060001         ADD     R0,R1
5 013666 060100         ADD     R1,R0
6 013670 012702         MOV     #5,R2
7
8 013674 114440 1$:     MOVB    -(R4),-(R0)
9 013676 114541         MOVB    -(R5),-(R1)
10 13700 005302         DEC     R2
11 13702 003374         BGT     1$
12
13 13704 016700 DMPLN:  MOV     OUTPNT,R0
14 13710 112760         MOVB    #15,IOBUF(R0)
15 13716 112760         MOVB    #12,IOBUF+1(R0)
16 13724 012701         MOV     #IOBUF,R1
17
18 13730 112100 2$:     MOVB    (R1)+,R0
19 13732 004767 3$:     JSR     PC,PANIC
20 13736                .TTOUTR
21 13740 103774         BCS     3$
22 13742 022700         CMP     #12,R0
23 13746 001370         BNE     2$
24
25 13750 005067 CLRBUF: CLR     OUTPNT
26 13754 012701         MOV     #132.+IOBUF,R1
27 13760 012741         MOV     #5015,-(R1)
28 13764 012741 4$:     MOV     #20040,-(R1)
29 13770 022701         CMP     #IOBUF,R1
30 13774 103773         BLO     4$
31 13776 000207         RTS     PC
32
33
34 14000 005701 PRANS:  TST     R1
35 14002 100404         BMI     2$
36 14004 001011         BNE     3$
37 14006 022702         CMP     #10000.,R2
38 14012 101014         BHI     4$
39 14014 022701 2$:     CMP     #-1,R1
40 14020 001003         BNE     3$
41 14022 022702         CMP     #-10000.,R2
42 14026 103406         BLO     4$
43
44 14030 005203 3$:     INC     R3
45 14032 012700         MOV     #10.,R0
46 14036 004767         JSR     PC,DIVID
47 14042 000756         BR      PRANS
48
49 14044 010267 4$:     MOV     R2,MANT
50 14050 012700         MOV     #3,R0
51 14054 004767         JSR     PC,BINDEC
52 14060 110367         MOVB    R3,EXPON
53 14064 010302         MOV     R3,R2
54 14066 004767         JSR     PC,BINDOC
55 14072 112737         MOVB    #'E,@#20.+IOBUF
56 14100 000701         BR      DMPLN
```

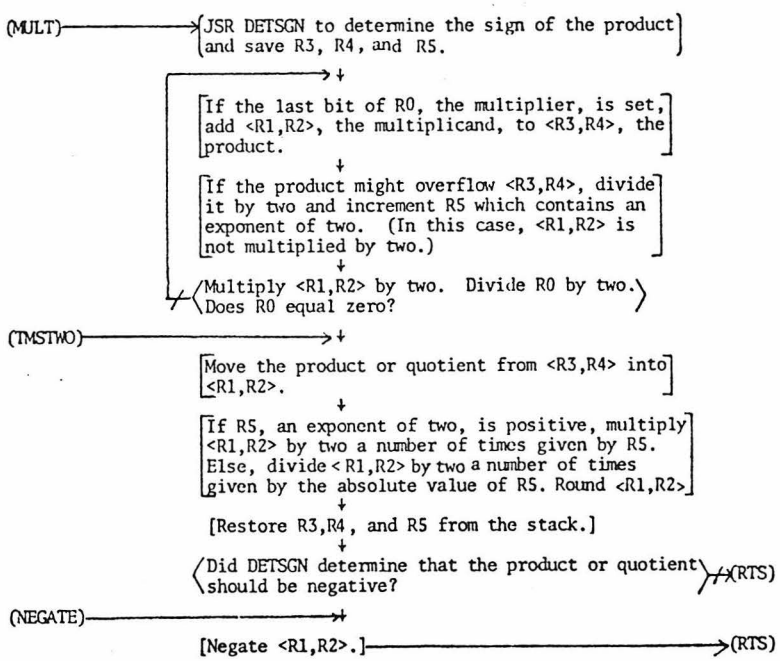


```
2                       .SBTTL  NUMERICAL ROUTINES
3
4
5 014102 110367 LOAD:   MOVB    R3,EXPON
6 014106 010467         MOV     R4,MANT
7 014112 000207         RTS     PC
8
9
10 14114 005404 SUB:    NEG     R4
11
12 14116 010402 ADD:    MOV     R4,R2
13 14120 004767         JSR     PC,TMSEXP
14 14124 010205         MOV     R2,R5
15 14126 010104         MOV     R1,R4
16
17 14130 016702         MOV     MANT,R2
18 14134 116703         MOVB    EXPON,R3
19 14140 004767         JSR     PC,TMSEXP
20
21 14144 060502         ADD     R5,R2
22 14146 005501         ADC     R1
23 14150 060401         ADD     R4,R1
24 14152 000712         BR      PRANS
25
26
27 14154 005001 MUL:    CLR     R1
28 14156 016702         MOV     MANT,R2
29 14162 100001         BPL     1$
30 14164 005301         DEC     R1
31
32 14166 010400 1$:     MOV     R4,R0
33 14170 004767         JSR     PC,MULT
34 14174 162703         SUB     #3,R3
35 14200 000416         BR      MULEP
36
37 14202 005001 DIV:    CLR     R1
38 14204 016702         MOV     MANT,R2
39 14210 100001         BPL     2$
40 14212 005301         DEC     R1
41
42 14214 012700 2$:     MOV     #10000.,R0
43 14220 004767         JSR     PC,MULT
44 14224 010400         MOV     R4,R0
45 14226 004767         JSR     PC,DIVID
46
47 14232 005403         NEG     R3
48 14234 005303         DEC     R3
49 14236 066703 MULEP:  ADD     EXPON,R3
50 14242 000656         BR      PRANS
```

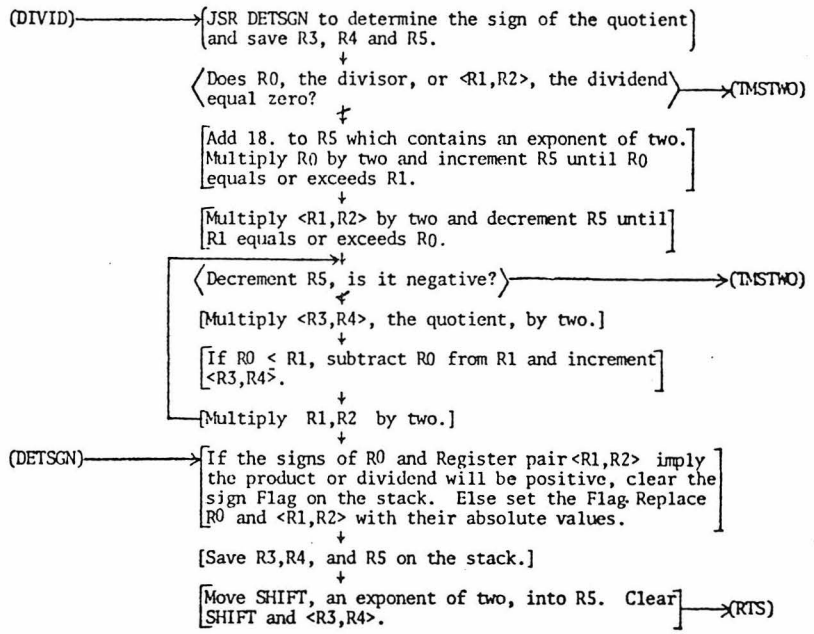


NUMERICAL ROUTINES

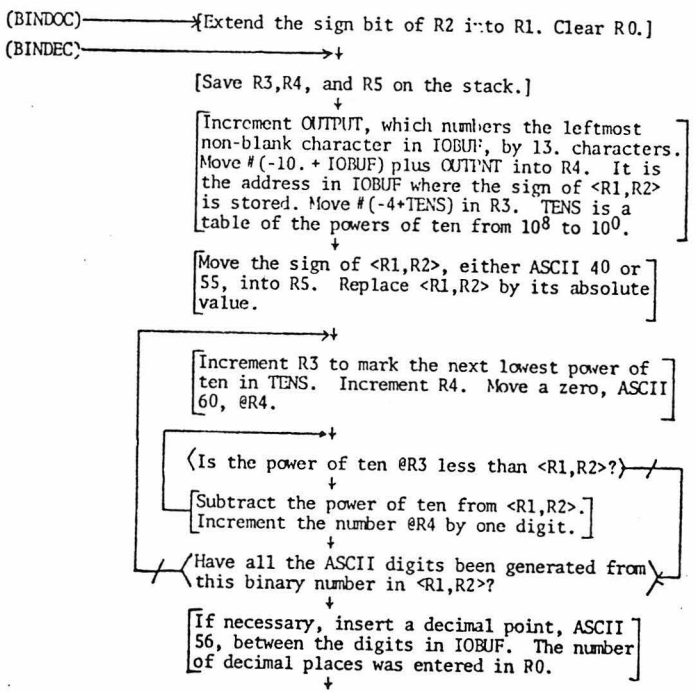
```
1 014244 005046 MULT:   CLR     -(SP)
2 014246 004367         JSR     R3,DETSGN
3
4 014252 032700 1$:     BIT     #1,R0
5 014256 001403         BEQ     2$
6 014260 060204         ADD     R2,R4
7 014262 005503         ADC     R3
8 014264 060103         ADD     R1,R3
9
10 14266 022701 2$:     CMP     #20000,R1
11 14272 101004         BHI     3$
12 14274 005205         INC     R5
13 14276 006203         ASR     R3
14 14300 006004         ROR     R4
15 14302 000402         BR      4$
16
17 14304 006302 3$:     ASL     R2
18 14306 006101         ROL     R1
19 14310 000241 4$:     CLC
20 14312 006000         ROR     R0
21 14314 001356         BNE     1$
22
23 14316 010402 TMSTWO: MOV     R4,R2
24 14320 010301         MOV     R3,R1
25
26 14322 005705         TST     R5
27 14324 001421         BEQ     4$
28 14326 100410         BMI     3$
29 14330 006302 1$:     ASL     R2
30 14332 006101         ROL     R1
31 14334 005305         DEC     R5
32 14336 003374         BGT     1$
33 14340 000413         BR      4$
34
35 14342 006201 2$:     ASR     R1
36 14344 006002         ROR     R2
37 14346 005205         INC     R5
38 14350 022705 3$:     CMP     #-1,R5
39 14354 003372         BGT     2$
40 14356 062702         ADD     #1,R2
41 14362 005501         ADC     R1
42 14364 006201         ASR     R1
43 14366 006002         ROR     R2
44
45 14370 012605 4$:     MOV     (SP)+,R5
46 14372 012604         MOV     (SP)+,R4
47 14374 012603         MOV     (SP)+,R3
48
49 14376 005726         TST     (SP)+
50 14400 001403         BEQ     EXITN1
51
52 14402 005401 NEGATE: NEG     R1
53 14404 005402         NEG     R2
54 14406 005601         SBC     R1
55 14410 000207 EXITN1: RTS     PC
```



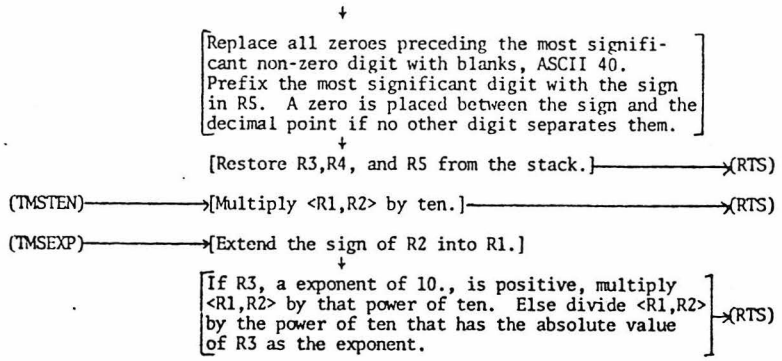
```
1 014412 005046 DIVID:  CLR     -(SP)
2 014414 004367         JSR     R3,DETSGN
3
4 014420 005702         TST     R2
5 014422 001002         BNE     1$
6 014424 005701         TST     R1
7 014426 001733         BEQ     TMSTWO
8 014430 005700 1$:     TST     R0
9 014432 001731         BEQ     TMSTWO
10
11 14434 062705         ADD     #18.,R5
12 14440 020001 2$:     CMP     R0,R1
13 14442 103003         BHIS    3$
14 14444 006300         ASL     R0
15 14446 005205         INC     R5
16 14450 000773         BR      2$
17
18 14452 020001 3$:     CMP     R0,R1
19 14454 101404         BLOS    4$
20 14456 006302         ASL     R2
21 14460 006101         ROL     R1
22 14462 005305         DEC     R5
23 14464 000772         BR      3$
24
25 14466 005305 4$:     DEC     R5
26 14470 100712         BMI     TMSTWO
27
28 14472 006304         ASL     R4
29 14474 006103         ROL     R3
30
31 14476 020100         CMP     R1,R0
32 14500 103402         BLO     5$
33 14502 005204         INC     R4
34 14504 160001         SUB     R0,R1
35
36 14506 006302 5$:     ASL     R2
37 14510 006101         ROL     R1
38 14512 000765         BR      4$
39
40 14514 005700 DETSGN: TST     R0
41 14516 100003         BPL     1$
42 14520 005400         NEG     R0
43 14522 005266         INC     2(SP)
44 14526 005701 1$:     TST     R1
45 14530 100001         BPL     2$
46 14532 004767         JSR     PC,NEGATE
47 14536 005366         DEC     2(SP)
48
49 14542 010446 2$:     MOV     R4,-(SP)
50 14544 010546         MOV     R5,-(SP)
51 14546 010346         MOV     R3,-(SP)
52
53 14550 116705         MOVB    SHIFT,R5
54 14554 105067         CLRB    SHIFT
55 14560 005003         CLR     R3
56 14562 005004         CLR     R4
57 14564 000207         RTS     PC
```



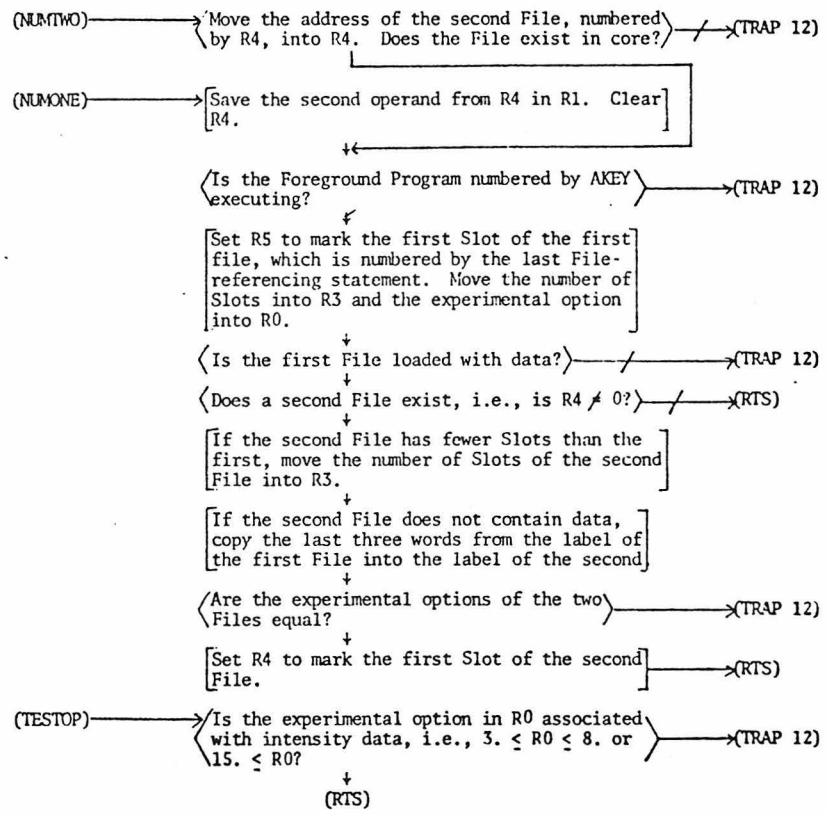
```
1 014566 005000 BINDOC: CLR     R0
2 014570 005001         CLR     R1
3 014572 005702         TST     R2
4 014574 100001         BPL     BINDEC
5 014576 005301         DEC     R1
6
7 014600 010346 BINDEC: MOV     R3,-(SP)
8 014602 010446         MOV     R4,-(SP)
9 014604 010546         MOV     R5,-(SP)
10
11 14606 062767         ADD     #13.,OUTPNT
12 14614 016704         MOV     OUTPNT,R4
13 14620 062704         ADD     #-10.+IOBUF,R4
14 14624 010446         MOV     R4,-(SP)
15 14626 012703         MOV     #-4+TENS,R3
16
17 14632 012705         MOV     #40,R5
18 14636 005701         TST     R1
19 14640 100004         BPL     1$
20 14642 004767         JSR     PC,NEGATE
21 14646 012705         MOV     #55,R5
22
23 14652 062703 1$:     ADD     #4,R3
24 14656 005204         INC     R4
25 14660 112714         MOVB    #60,(R4)
26
27 14664 026301 2$:     CMP     2(R3),R1
28 14670 003370         BGT     1$
29 14672 002402         BLT     3$
30 14674 021302         CMP     (R3),R2
31 14676 101006         BHI     4$
32
33 14700 105214 3$:     INCB    (R4)
34 14702 166301         SUB     2(R3),R1
35 14706 161302         SUB     (R3),R2
36 14710 005601         SBC     R1
37 14712 000764         BR      2$
38
39 14714 020327 4$:     CMP     R3,#32.+TENS
40 14720 103754         BLO     1$
41
42 14722 012604         MOV     (SP)+,R4
43 14724 010402         MOV     R4,R2
44 14726 010201         MOV     R2,R1
45 14730 005302         DEC     R2
46 14732 005400         NEG     R0
47 14734 001002         BNE     8$
48 14736 005201         INC     R1
49 14740 000407         BR      6$
50
51 14742 062700 8$:     ADD     #9.,R0
52 14746 112422 5$:     MOVB    (R4)+,(R2)+
53 14750 005300         DEC     R0
54 14752 002375         BGE     5$
55 14754 112712         MOVB    #56,(R2)
```



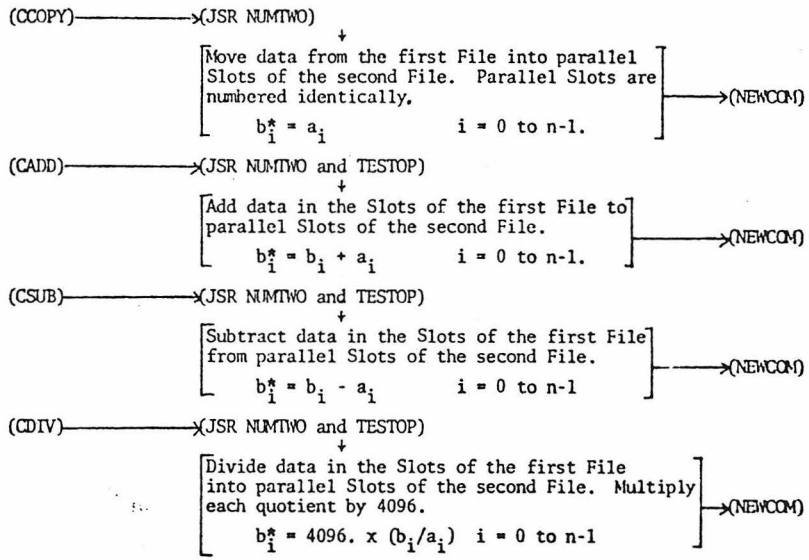
```
1 014760 112721 6$:     MOVB    #40,(R1)+
2 014764 122711         CMPB    #60,(R1)
3 014770 001010         BNE     7$
4 014772 122761         CMPB    #40,1(R1)
5 015000 001404         BEQ     7$
6 015002 122761         CMPB    #56,1(R1)
7 015010 001363         BNE     6$
8 015012 110541 7$:     MOVB    R5,-(R1)
9
10 15014 012605         MOV     (SP)+,R5
11 15016 012604         MOV     (SP)+,R4
12 15020 012603         MOV     (SP)+,R3
13 15022 000207         RTS     PC
14
15
16 15024 010246 TMSTEN: MOV     R2,-(SP)
17 15026 010146         MOV     R1,-(SP)
18 15030 006302         ASL     R2
19 15032 006101         ROL     R1
20 15034 006302         ASL     R2
21 15036 006101         ROL     R1
22 15040 062601         ADD     (SP)+,R1
23 15042 062602         ADD     (SP)+,R2
24 15044 005501         ADC     R1
25 15046 006302         ASL     R2
26 15050 006101         ROL     R1
27 15052 000207         RTS     PC
28
29 15054 005001 TMSEXP: CLR     R1
30 15056 005702         TST     R2
31 15060 100001         BPL     5$
32 15062 005301         DEC     R1
33
34 15064 005703 5$:     TST     R3
35 15066 001413         BEQ     7$
36 15070 002404         BLT     6$
37 15072 004767         JSR     PC,TMSTEN
38 15076 005303         DEC     R3
39 15100 000771         BR      5$
40
41 15102 012700 6$:     MOV     #10.,R0
42 15106 004767         JSR     PC,DIVID
43 15112 005203         INC     R3
44 15114 002772         BLT     6$
45 15116 000207 7$:     RTS     PC
```



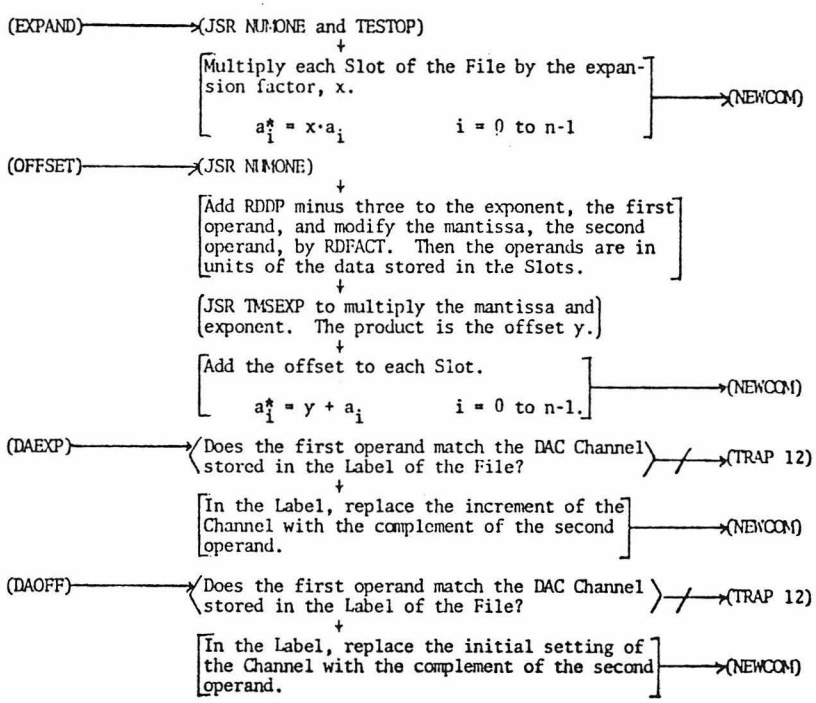
```
1                       .SBTTL  DATA ANALYSIS ROUTINES
2
3
4 015120 006304 NUMTWO: ASL     R4
5 015122 016404         MOV     FLBUFF(R4),R4
6 015126 001454         BEQ     E10A
7 015130 001002         BNE     EPHOP
8
9 015132 010401 NUMONE: MOV     R4,R1
10 15134 005004         CLR     R4
11
12 15136 116700 EPHOP:  MOVB    AKEY,R0
13 15142 006300         ASL     R0
14 15144 005760         TST     PRCNT(R0)
15 15150 001043         BNE     E10A
16
17 15152 011203         MOV     (R2),R3
18 15154 062702         ADD     #4,R2
19 15160 010205         MOV     R2,R5
20 15162 062705         ADD     #6,R5
21 15166 111200         MOVB    (R2),R0
22 15170 100433         BMI     E10A
23
24 15172 005704         TST     R4
25 15174 001417         BEQ     EXITA1
26
27 15176 021403         CMP     (R4),R3
28 15200 002001         BGE     1$
29 15202 011403         MOV     (R4),R3
30
31 15204 062704 1$:     ADD     #4,R4
32 15210 105714         TSTB    (R4)
33 15212 100004         BPL     2$
34 15214 012224         MOV     (R2)+,(R4)+
35 15216 012224         MOV     (R2)+,(R4)+
36 15220 012224         MOV     (R2)+,(R4)+
37 15222 000404         BR      EXITA1
38
39 15224 121400 2$:     CMPB    (R4),R0
40 15226 001014         BNE     E10A
41 15230 062704         ADD     #6,R4
42 15234 000207 EXITA1: RTS     PC
43
44
45 15236 022700 TESTOP: CMP     #2,R0
46 15242 002006         BGE     E10A
47 15244 022700         CMP     #8.,R0
48 15250 002371         BGE     EXITA1
49 15252 022700         CMP     #14.,R0
50 15256 002766         BLT     EXITA1
51
52 15260 104412 E10A:   TRAP    12
```



```
1 015262 004767 CCOPY:  JSR     PC,NUMTWO
2
3 015266 012524 1$:     MOV     (R5)+,(R4)+
4 015270 012524         MOV     (R5)+,(R4)+
5 015272 005303         DEC     R3
6 015274 003374         BGT     1$
7 015276 000446         BR      EXITA2
8
9 015300 004767 CADD:   JSR     PC,NUMTWO
10 15304 004767         JSR     PC,TESTOP
11
12 15310 062524 2$:     ADD     (R5)+,(R4)+
13 15312 005514         ADC     (R4)
14 15314 062524         ADD     (R5)+,(R4)+
15 15316 005303         DEC     R3
16 15320 003373         BGT     2$
17 15322 000434         BR      EXITA2
18
19 15324 004767 CSUB:   JSR     PC,NUMTWO
20 15330 004767         JSR     PC,TESTOP
21
22 15334 162524 3$:     SUB     (R5)+,(R4)+
23 15336 005614         SBC     (R4)
24 15340 162524         SUB     (R5)+,(R4)+
25 15342 005303         DEC     R3
26 15344 003373         BGT     3$
27 15346 000422         BR      EXITA2
28
29 15350 004767 CDIV:   JSR     PC,NUMTWO
30 15354 004767         JSR     PC,TESTOP
31
32 15360 012402 4$:     MOV     (R4)+,R2
33 15362 011401         MOV     (R4),R1
34 15364 012500         MOV     (R5)+,R0
35 15366 005725         TST     (R5)+
36 15370 112767         MOVB    #12.,SHIFT
37 15376 004767         JSR     PC,DIVID
38 15402 010124         MOV     R1,(R4)+
39 15404 010264         MOV     R2,-4(R4)
40 15410 005303         DEC     R3
41 15412 003362         BGT     4$
42 15414 000167 EXITA2: JMP     NEWCOM
```

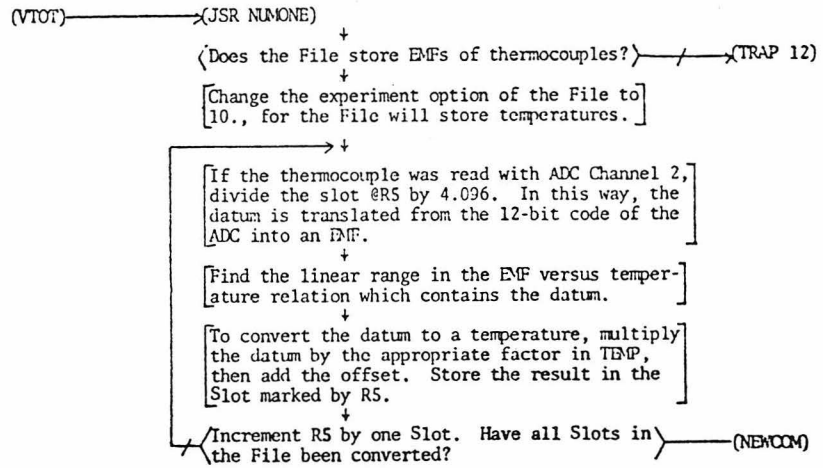


```
1 015420 004767 EXPAND: JSR     PC,NUMONE
2 015424 010104         MOV     R1,R4
3
4 015426 012502 1$:     MOV     (R5)+,R2
5 015430 011501         MOV     (R5),R1
6 015432 012700         MOV     #1000.,R0
7 015436 112767         MOVB    #10.,SHIFT
8 015444 004767         JSR     PC,DIVID
9 015450 112767         MOVB    #-10.,SHIFT
10 15456 010400         MOV     R4,R0
11 15460 004767         JSR     PC,MULT
12 15464 010125         MOV     R1,(R5)+
13 15466 010265         MOV     R2,-4(R5)
14
15 15472 005303         DEC     R3
16 15474 003354         BGT     1$
17 15476 000746         BR      EXITA2
18
19 15500 010346 OFFSET: MOV     R3,-(SP)
20 15502 004767         JSR     PC,NUMONE
21
22 15506 010304         MOV     R3,R4
23 15510 010102         MOV     R1,R2
24 15512 116003         MOVB    RDDP(R0),R3
25 15516 162703         SUB     #3,R3
26 15522 062603         ADD     (SP)+,R3
27 15524 010046         MOV     R0,-(SP)
28 15526 004767         JSR     PC,TMSEXP
29 15532 012600         MOV     (SP)+,R0
30
31 15534 022700         CMP     #8.,R0
32 15540 003410         BLE     2$
33 15542 112767         MOVB    #12.,SHIFT
34 15550 006300         ASL     R0
35 15552 016000         MOV     RDFACT(R0),R0
36 15556 004767         JSR     PC,DIVID
37
38 15562 060225 2$:     ADD     R2,(R5)+
39 15564 005515         ADC     (R5)
40 15566 060125         ADD     R1,(R5)+
41 15570 005304         DEC     R4
42 15572 003373         BGT     2$
43 15574 000707         BR      EXITA2
44
45
46 15576 120362 DAEXP:  CMPB    R3,5(R2)
47 15602 001226         BNE     E10A
48
49 15604 005722         TST     (R2)+
50 15606 000403         BR      DAEP
51
52 15610 120362 DAOFF:  CMPB    R3,5(R2)
53 15614 001221         BNE     E10A
54
55 15616 005104 DAEP:   COM     R4
56 15620 010462         MOV     R4,6(R2)
57 15624 000673         BR      EXITA2
```

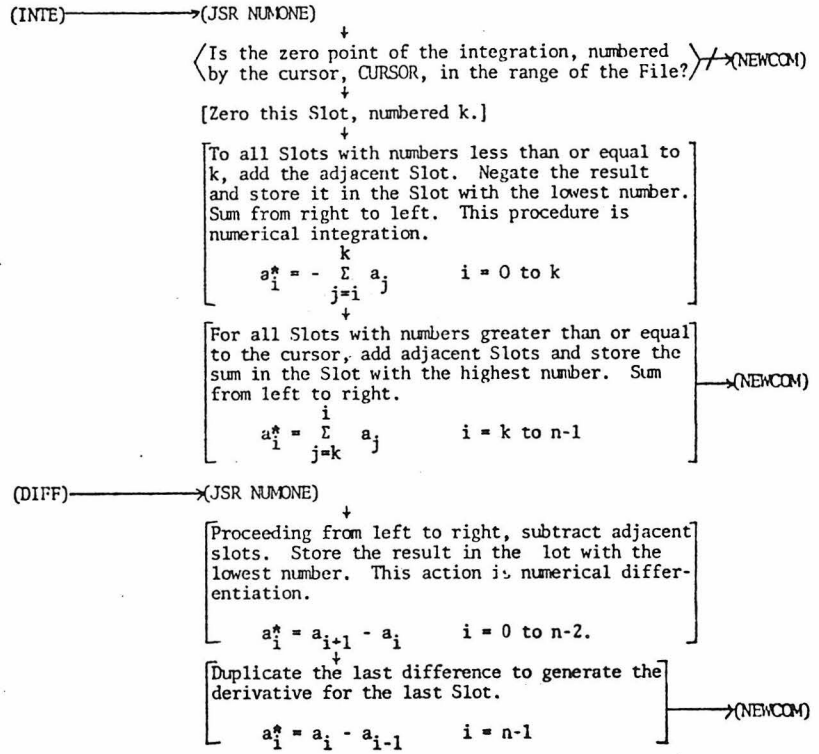


DATA ANALYSIS ROUTINES

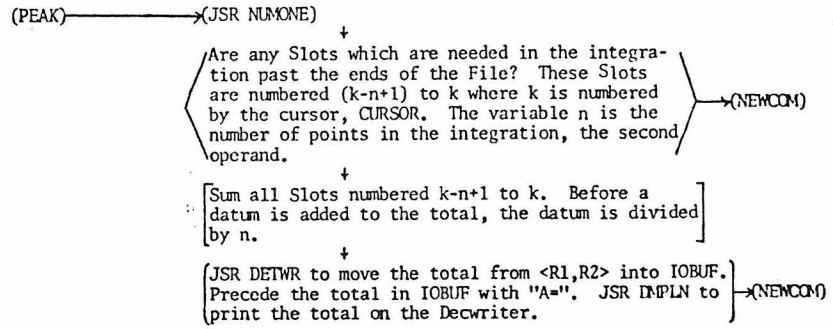
```
1 015626 004767 VTOT:   JSR     PC,NUMONE
2
3 015632 022700         CMP     #2.,R0
4 015636 001403         BEQ     1$
5 015640 022700         CMP     #14.,R0
6 015644 001205         BNE     E10A
7
8 015646 112765 1$:     MOVB    #10.,-6(R5)
9 015654 010046         MOV     R0,-(SP)
10
11 15656 012502 2$:     MOV     (R5)+,R2
12 15660 011501         MOV     (R5),R1
13 15662 022716         CMP     #2.,(SP)
14 15666 001007         BNE     3$
15 15670 012700         MOV     #1000.,R0
16 15674 112767         MOVB    #-12.,SHIFT
17 15702 004767         JSR     PC,MULT
18
19 15706 012704 3$:     MOV     #-4+TEMP,R4
20 15712 062704 4$:     ADD     #4,R4
21 15716 022402         CMP     (R4)+,R2
22 15720 002774         BLT     4$
23
24 15722 012400         MOV     (R4)+,R0
25 15724 112767         MOVB    #-10.,SHIFT
26 15732 004767         JSR     PC,MULT
27 15736 061402         ADD     (R4),R2
28 15740 005501         ADC     R1
29 15742 010125         MOV     R1,(R5)+
30 15744 010265         MOV     R2,-4(R5)
31
32 15750 005303         DEC     R3
33 15752 003341         BGT     2$
34 15754 005026         CLR     (SP)+
35 15756 000616         BR      EXITA2
```



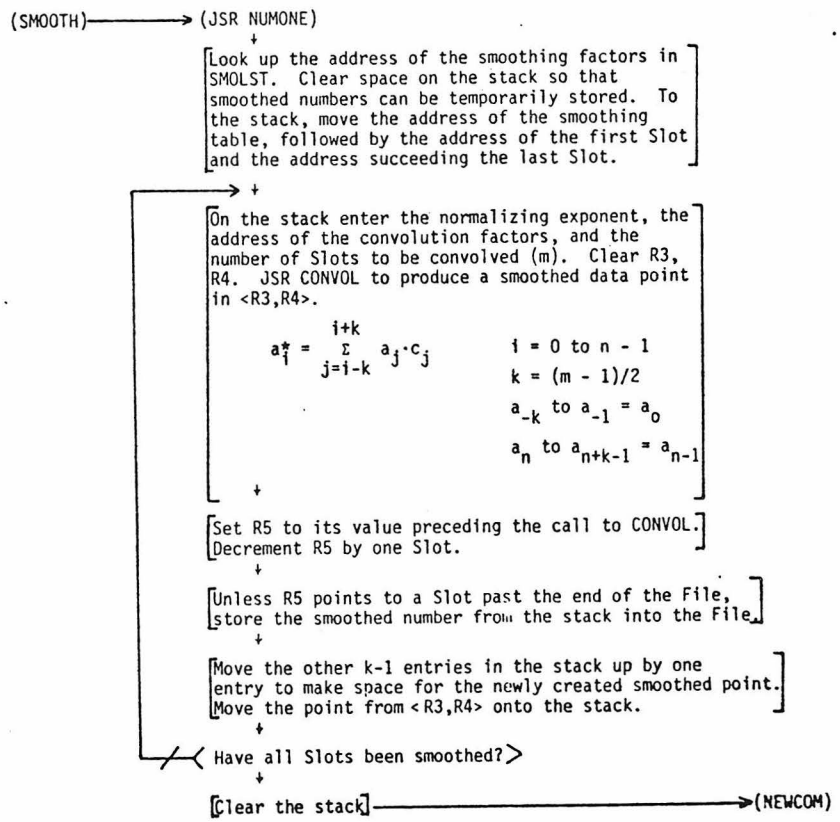
```
1 015760 004767 INTE:   JSR     PC,NUMONE
2
3 015764 016704         MOV     CURSOR,R4
4 015770 020403         CMP     R4,R3
5 015772 002122         BGE     EXITA3
6
7 015774 006304         ASL     R4
8 015776 006304         ASL     R4
9 016000 060504         ADD     R5,R4
10 16002 010401         MOV     R4,R1
11 16004 010400         MOV     R4,R0
12 16006 005024         CLR     (R4)+
13 16010 005024         CLR     (R4)+
14 16012 010402         MOV     R4,R2
15
16 16014 005303 1$:     DEC     R3
17 16016 020005         CMP     R0,R5
18 16020 101411         BLOS    2$
19 16022 005440         NEG     -(R0)
20 16024 064410         ADD     -(R4),(R0)
21 16026 005440         NEG     -(R0)
22 16030 005660         SBC     2(R0)
23 16034 064410         ADD     -(R4),(R0)
24 16036 005560         ADC     2(R0)
25 16042 000764         BR      1$
26
27 16044 005303 2$:     DEC     R3
28 16046 002474         BLT     EXITA3
29 16050 062122         ADD     (R1)+,(R2)+
30 16052 005512         ADC     (R2)
31 16054 062122         ADD     (R1)+,(R2)+
32 16056 000772         BR      2$
33
34
35 16060 004767 DIFF:   JSR     PC,NUMONE
36
37 16064 010504         MOV     R5,R4
38 16066 062704         ADD     #4,R4
39 16072 005303         DEC     R3
40
41 16074 162425 3$:     SUB     (R4)+,(R5)+
42 16076 005615         SBC     (R5)
43 16100 162415         SUB     (R4)+,(R5)
44 16102 005415         NEG     (R5)
45 16104 005465         NEG     -2(R5)
46 16110 005625         SBC     (R5)+
47 16112 005303         DEC     R3
48 16114 003367         BGT     3$
49
50 16116 014544         MOV     -(R5),-(R4)
51 16120 014544         MOV     -(R5),-(R4)
52 16122 000446         BR      EXITA3
```



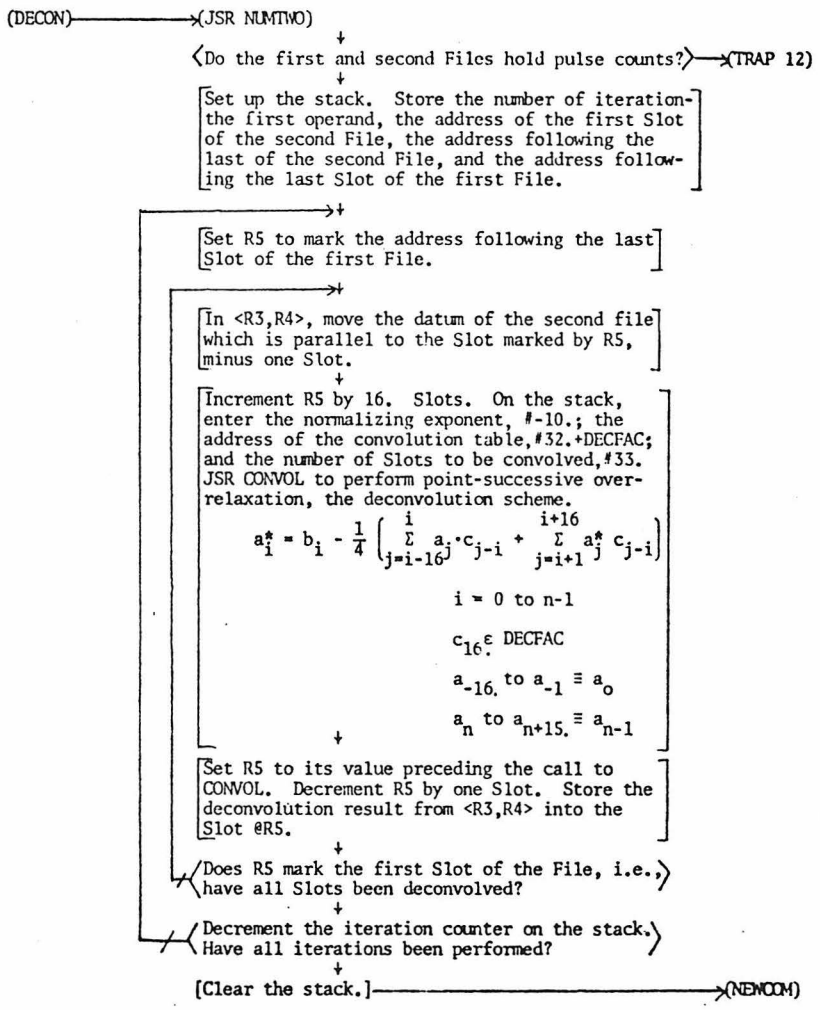
```
1 016124 004767 PEAK:   JSR     PC,NUMONE
2
3 016130 016702         MOV     CURSOR,R2
4 016134 020203         CMP     R2,R3
5 016136 002040         BGE     EXITA3
6 016140 160102         SUB     R1,R2
7 016142 005202         INC     R2
8 016144 100435         BMI     EXITA3
9
10 16146 006302         ASL     R2
11 16150 006302         ASL     R2
12 16152 060205         ADD     R2,R5
13 16154 010103         MOV     R1,R3
14 16156 010104         MOV     R1,R4
15 16160 010046         MOV     R0,-(SP)
16 16162 005046         CLR     -(SP)
17 16164 005046         CLR     -(SP)
18
19 16166 012502 1$:     MOV     (R5)+,R2
20 16170 012501         MOV     (R5)+,R1
21 16172 010400         MOV     R4,R0
22 16174 004767         JSR     PC,DIVID
23 16200 060266         ADD     R2,2(SP)
24 16204 005501         ADC     R1
25 16206 060116         ADD     R1,(SP)
26 16210 005303         DEC     R3
27 16212 003365         BGT     1$
28
29 16214 012601         MOV     (SP)+,R1
30 16216 012602         MOV     (SP)+,R2
31 16220 012604         MOV     (SP)+,R4
32 16222 004767         JSR     PC,DETWR
33 16226 012767         MOV     #'A=,IOBUF
34 16234 004767         JSR     PC,DMPLN
35 16240 000167 EXITA3: JMP     NEWCOM
```



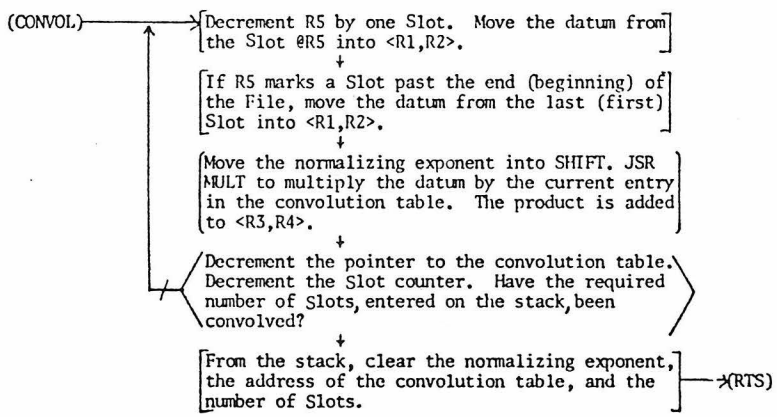
```
1 016244 004767 SMOOTH: JSR     PC,NUMONE
2
3 016250 006301         ASL     R1
4 016252 016101         MOV     SMOLST(R1),R1
5 016256 161106         SUB     (R1),SP
6 016260 010146         MOV     R1,-(SP)
7
8 016262 010546         MOV     R5,-(SP)
9 016264 006303         ASL     R3
10 16266 006303         ASL     R3
11 16270 060305         ADD     R3,R5
12 16272 010546         MOV     R5,-(SP)
13 16274 061105         ADD     (R1),R5
14
15 16276 005003 1$:     CLR     R3
16 16300 005004         CLR     R4
17
18 16302 016600         MOV     4(SP),R0
19 16306 114046         MOVB    -(R0),-(SP)
20 16310 010046         MOV     R0,-(SP)
21 16312 005200         INC     R0
22 16314 011046         MOV     (R0),-(SP)
23 16316 006216         ASR     (SP)
24 16320 005216         INC     (SP)
25 16322 004767         JSR     PC,CONVOL
26
27 16326 010601         MOV     SP,R1
28 16330 062701         ADD     #4,R1
29 16334 010102         MOV     R1,R2
30 16336 017200         MOV     @(R2),R0
31 16342 060001         ADD     R0,R1
32 16344 006300         ASL     R0
33 16346 060005         ADD     R0,R5
34
35 16350 020516         CMP     R5,(SP)
36 16352 103012         BHIS    3$
37 16354 016115         MOV     -2(R1),(R5)
38 16360 011165         MOV     (R1),2(R5)
39 16364 000405         BR      3$
40
41 16366 011161 2$:     MOV     (R1),4(R1)
42 16372 016161         MOV     -2(R1),2(R1)
43 16400 162701 3$:     SUB     #4,R1
44 16404 020102         CMP     R1,R2
45 16406 101367         BHI     2$
46
47 16410 005721         TST     (R1)+
48 16412 010421         MOV     R4,(R1)+
49 16414 010311         MOV     R3,(R1)
50
51 16416 020566         CMP     R5,2(SP)
52 16422 101325         BHI     1$
53
54 16424 067606         ADD     @4(SP),SP
55 16430 062706         ADD     #6,SP
56 16434 000701         BR      EXITA3
```



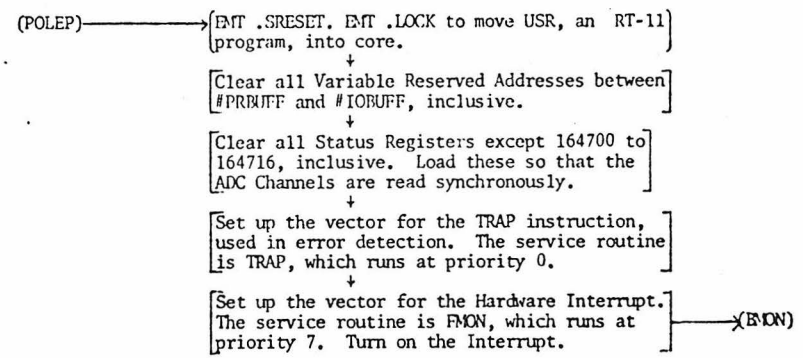
```
1 016436 010346 DECON:  MOV     R3,-(SP)
2 016440 004767         JSR     PC,NUMTWO
3
4 016444 022700         CMP     #8.,R0
5 016450 001401         BEQ     3$
6 016452 104412         TRAP    12
7
8 016454 006303 3$:     ASL     R3
9 016456 006303         ASL     R3
10 16460 060304         ADD     R3,R4
11 16462 010446         MOV     R4,-(SP)
12 16464 010446         MOV     R4,-(SP)
13 16466 010546         MOV     R5,-(SP)
14 16470 060305         ADD     R3,R5
15 16472 010546         MOV     R5,-(SP)
16
17 16474 011605 1$:     MOV     (SP),R5
18
19 16476 016600 2$:     MOV     4(SP),R0
20 16502 014003         MOV     -(R0),R3
21 16504 014004         MOV     -(R0),R4
22 16506 010066         MOV     R0,4(SP)
23
24 16512 062705         ADD     #64.,R5
25 16516 012746         MOV     #-10.,-(SP)
26 16522 012746         MOV     #32.+DECFAC,-(SP)
27 16526 012746         MOV     #33.,-(SP)
28 16532 004767         JSR     PC,CONVOL
29
30 16536 062705         ADD     #68.,R5
31 16542 060345         ADD     R3,-(R5)
32 16544 060445         ADD     R4,-(R5)
33 16546 005565         ADC     2(R5)
34
35 16552 020566         CMP     R5,2(SP)
36 16556 101347         BHI     2$
37 16560 016666         MOV     6(SP),4(SP)
38
39 16566 005366         DEC     8.(SP)
40 16572 003340         BGT     1$
41
42 16574 062706         ADD     #10.,SP
43 16600 000617         BR      EXITA3
```



```
1 016602 014501 CONVOL: MOV     -(R5),R1
2 016604 014502         MOV     -(R5),R2
3
4 016606 016600         MOV     10.(SP),R0
5 016612 020500         CMP     R5,R0
6 016614 103002         BHIS    1$
7 016616 012002         MOV     (R0)+,R2
8 016620 012001         MOV     (R0)+,R1
9
10 16622 016600 1$:     MOV     8.(SP),R0
11 16626 020500         CMP     R5,R0
12 16630 103402         BLO     2$
13 16632 014001         MOV     -(R0),R1
14 16634 014002         MOV     -(R0),R2
15
16 16636 116667 2$:     MOVB    6(SP),SHIFT
17 16644 117600         MOVB    @4(SP),R0
18 16650 004767         JSR     PC,MULT
19 16654 060204         ADD     R2,R4
20 16656 005503         ADC     R3
21 16660 060103         ADD     R1,R3
22
23 16662 005366         DEC     4(SP)
24 16666 005366         DEC     2(SP)
25 16672 003343         BGT     CONVOL
26
27 16674 011666         MOV     (SP),6(SP)
28 16700 062706         ADD     #6,SP
29 16704 000207         RTS     PC
```



```
1                       .SBTTL  INITIALIZATION ROUTINE
2
3
4 016706 000000         .WORD   0
5        000000'        .CSECT  POL
6
7 000000        POLEP:  .SRESET
8 000002                .LOCK
9
10 00004 012701         MOV     #PRBUFF,R1
11 00010 005021 1$:     CLR     (R1)+
12 00012 022701         CMP     #OSTPNT,R1
13 00016 101374         BHI     1$
14
15 00020 012701         MOV     #164000,R1
16 00024 005021 2$:     CLR     (R1)+
17 00026 022701         CMP     #165000,R1
18 00032 101374         BHI     2$
19
20 00034 012701         MOV     #164700,R1
21 00040 012700         MOV     #10,R0
22 00044 010021 3$:     MOV     R0,(R1)+
23 00046 005200         INC     R0
24 00050 022701         CMP     #164720,R1
25 00054 101373         BHI     3$
26
27 00056 012737         MOV     #TRAP,@#34
28 00064 005037         CLR     @#36
29
30 00070 012737         MOV     #FMON,@#270
31 00076 012737         MOV     #340,@#272
32 00104 012737         MOV     #1,@#164776
33
34 00112 000167         JMP     BMON
35       000000'        .END    POLEP
```



Appendix A: Starting Up the POL Operating System

If another programmer is using the POL operating system, one need not start up POL. He can enter commands on the Decwriter in a normal manner and may ignore the rest of this appendix. The sequence for starting up POL is as follows:

(1) The power for the computer, disc drives, Decwriter, peripheral board, and all peripheral devices to be used must be on.

(2) Bootstrap the computer by keying in 167700 on the toggles of the computer console. Lower, then raise, the "Halt" toggle. Next depress the "Load Address" toggle and then the "Start" toggle.

(3) The RT-11 operating system responds by printing

```
RT-11SJ     VØ2B-05
.
```

(4) The user starts up the POL operating system by typing

```
.RUN POL
```

(5) POL commands may now be entered and executed.

To exit POL, type C while depressing the control key. The consent of all programmers is essential before this action is taken! After the "Halt" toggle is lowered then raised, and the "Continue" toggle is depressed, the RT-11 operating system can be used. If the copy of the POL operating system in core is not written over by RT-11 programs, a programmer re-enters POL by typing:

```
.START     4366
```

Otherwise, go back to step 2.

Appendix B: Glossary

ADC: an analog-to-digital converter.

address: a number that references a word of memory.

algorithm: a sequence of arithmetic operations.

allocation error: This error is flagged when a POL Program illegally references a protected resource, i.e., a File or a Peripheral. The error appears on the Decwriter as "ERROR ALLOC".

assembly: the process of translating statements entered on the Decwriter into a form that is understood by the computer.

ASCII: a method for coding typewriter characters into numbers.

Assembly Subroutine: an assembly language subroutine that is associated with one or a group of POL commands, and runs when the command assembles. These subroutines modify the second operand of an instruction and/or detect errors that are unique to an instruction.

asynchronous: If a peripheral is controlled asynchronously, information is read from, or loaded into the peripheral immediately.

Background mode: While POL runs, the computer is in one of two states, the Foreground or Background mode. In the Foreground mode, the computer executes POL Programs that acquire data from experiments. All other tasks are accomplished in the Background mode. Three Background processes (the assembly of POL Programs, the execution of Background Programs, and the assembly and execution of Immediate Instructions) are serviced in the Background mode. No two of these processes are accomplished simultaneously.

Background processes: See Background mode.

Background program: a POL Program that analyzes or displays data.

binary: base two.

bit: Each word in memory consists of sixteen consecutive bits. Bits are either set (have value one) or are clear (have value zero).

branch: the process by which the computer skips over a section of a program to execute other statements in the program.

Buffer Register: a location in memory that holds information that is passed between a peripheral and the computer.

byte: a group of eight consecutive bits.

call: the process through which a routine enters a subroutine.

channel: a Slot.

Channel: one line that runs into or out of a multiplexer. All Channels associated with an ADC or a DAC are numbered, usually zero through seven.

clear: A bit is clear if it has value zero.

code: a group of assembly language instructions.

command: Several POL or assembly language commands form a program. Each command consists of a mnemonic, or name, and one or two operands.

command number: The sixty-three POL commands are numbered zero through sixty-two. Table 11.1 shows what number is associated with each command.

Command Subroutine: an assembly language subroutine that executes all POL commands associated with a single mnemonic.

complement: a binary operation that sets all bits in a number that were clear and clears all bits that were set.

core: the part of the computer that stores numbers and assembly language instructions. All entries in core are binary numbers.

CPU: the central processing unit; the part of the computer that executes assembly language instructions.

DAC: a digital-to-analog converter.

decimal: base ten.

directive: a command.

disc: a mass storage disc; the RX01 floppy disc unit.

display scope: the storage oscilloscope, a peripheral, on which experimental data are graphed.

DVM: a digital voltmeter.

8-Channel ADC: a multiplexed ADC, a peripheral, which has eight Channels to read data from experiments.

8-Channel DAC: a multiplexed DAC, a peripheral, which has eight Channels to control experimental parameters.

EMT: an assembly language instruction that initiates the execution of Program Requests.

EOF error: an end-of-file error. This error is flagged when an attempt is made to access information from, or load information into a location that is past the beginning or end of a record in core, e.g., a Program. A EOF error appears on the Decwriter as "ERROR EOF".

error message: The POL operating system prints out ten different error messages to inform a user that he has made an error in programming a POL instruction. Errors are flagged during both assembly and execution of an instruction.

execution: running or enacting an assembled instruction or program.

experimental option: a byte in the Label of a File that indicates the type of data stored in the File and which Peripheral read the data.

file: a list of related information, a record.

File: a POL record that stores data from experiments in core. A File consists of two parts – a Label which describes the data in the File and a string of Slots. Each Slot holds a data point.

File number: Each of the 110. Files in POL are numbered 1 through 110.

File-referencing statement: the S:FILE or I:FILE statement in POL. These statements number the File on which a POL Program operates.

Fixed Reserved Addresses: locations in core that are accessed by the POL operating system. These locations are never modified.

flag: to detect (an error).

Foreground mode: See Background mode.

Foreground Program: a POL Program whose principal function is to read data from experiments.

Function Subroutines: a POL subroutine that performs tasks required in the execution of several Command Subroutines. Function Subroutines abbreviate the length of the POL operating system.

graphics DAC: a multiplexed DAC, a peripheral, that drives the x- and y-axes of the display scope and point-plotter.

hardware error: This error is flagged when the computer makes a mistake in transferring information between core and disc. A hardware error appears on the Decwriter as "ERROR HRDWR".

Hardware Interrupt: This interrupt forces the CPU to suspend temporarily executing of Background Processes and to enter the Foreground mode. In this mode, Foreground Programs run.

Immediate Instructions: a POL instruction that executes by itself rather than in a Program. After they are entered on the Decwriter, Immediate Instructions are processed instantly. Note that some instructions can be executed either as Immediate Instructions or in a Background Program.

index: a number which marks one piece of information from a table of information in core. The index is added to the first address, or label, of the table to give the address of the piece of information.

instruction: a command.

interrupt: a signal that temporarily causes the CPU to suspend the execution of one program so that another may be run. The PDP-11/10 computer uses a priority system so that more important programs are processed first.

Interrupt: a Hardware Interrupt.

Interrupt Period: a time interval of sixteen milliseconds after which another Hardware Interrupt is generated.

interrupt vector: two words in core which provide information so that an interrupt may be serviced (acknowledged). The first word holds the address of the service routine, the program that runs after the interrupt is generated. The second word contains a new processor status word which establishes the priority, or importance, of running the service routine.

I/O operation: an input/output operation between a peripheral and the computer.

JSR: an assembly language instruction that initiates the execution of a subroutine.

jump: a branch.

Key: the number of a Foreground Program. In POL Keys are associated with Files and Peripherals for protection.

label: a name associated with an address in an assembly language program.

Label: a string of five consecutive words that prefix the Slots in a File and describe the data contained in the File.

listing: the printout of an assembled program.

location: an address in core.

loop: a method which allows a section of a program to be executed iteratively.

Loop: a loop in a POL Program.

Management Routines: assembly language routines in the POL operating system that control the assembly and execution of POL commands.

manipulation error: This error is flagged when a data analysis routine might operate on data which are inappropriate to the routine. The error appears on the Decwriter as "ERROR MANIP".

mark: to number an address in memory.

memory: core.

mnemonic: the name associated with an instruction.

multiplexed: A peripheral is multiplexed so that a single element in the peripheral (e.g., an ADC) can accept signals from or send signals to many devices. In this way, an expensive element need not be duplicated.

narrow converter: The output of any Channel of the DAC unit is the sum of the outputs from two 12-bit D/A converters, the wide and narrow converters.

octal: base eight.

operand: a number or information associated with an instruction. POL instructions may have none, one, or two operands. Two operands are separated by a semicolon. The second operand is the single operand associated with a POL instruction, or the operand that follows the semicolon. The first operand prefixes the semicolon.

order error: This error is flagged when POL instructions are not programmed in the proper sequence. The error appears on the Decwriter as "ERROR ORDER".

output stack: an area in core set aside by the POL system. The output stack is used in displaying data.

"Panic" button: a switch that is depressed by the user to suspend the printing on the Decwriter or graphing on the point-plotter.

parallel entries: Two entries in different records in core are said to be parallel if the entries are indexed by the same numbers.

Period: an Interrupt Period.

peripheral: a device which sends information to, or accepts information from the computer.

Peripheral: a peripheral which controls an experimental parameter or reads data from an experiment.

Peripheral Address: a location in core that passes information between the computer and a peripheral, or monitors this passing of information. These locations are called Buffer Registers and Status Registers, respectively.

pointer: a location in core or a Register that holds the address of a block of information in core.

point-plotter: a peripheral that graphs experimental data point-by-point on a piece of paper.

POL: Peripheral Oriented Language, the computer language which is the subject of this manual.

priority: See interrupt.

process: to execute.

processor status word: a word in core that describes the state of the CPU. This word delineates the priority of the program that is currently executing.

program: a collection of commands that are processed in succession by the computer.

Program: a program consisting of POL commands.

Program number: POL Programs are numbered zero through four. The Background Program is assigned number zero while the four Foreground Programs are allotted numbers one through four.

Programming environment: POL instructions can be executed in three Programming environments – as Immediate Instructions, in Background Programs, and in Foreground Programs. Many instructions may be entered legally in certain environments only.

Program Request: a function performed by the RT-11 operating system when an EMT trap is generated. The most useful of the functions are those which involve peripherals. (See the RT-11 Manual for more information.)

pulse counter: a peripheral that sums pulses from an electron multiplier.

queue: to put in a line or sequence.

RADIX 50: a method of coding certain typewriter characters into numbers.

register: a device that stores information.

Register: a General Programming Register, R0 through R5, of the PDP-11/10 computer. When information is held in two consecutive Registers, these are referred to as a Register pair and appear in angle brackets, e.g., ⟨R1,R2⟩.

Reserved Address: a location in core that holds information for the POL operating system.

RTS: an assembly language command that ends the execution of a subroutine and returns control of the CPU to the calling routine.

service routine: an assembly language routine that is entered when an interrupt is generated.

set: A bit is set if it has value one.

Slot: a location in a File that is two words in length and stores a single data point. The n Slots of a File are numbered zero through (n-1).

software: the assembly language programs of a computer.

space: an area in core memory.

stack: an area in core that is set aside by the RT-11 operating system to store information temporarily. The stack pointer, SP or R6, marks the word in the stack that is currently being accessed or modified.

statement: a command.

Status Register: a location in core that controls the passing of data between the computer and a peripheral.

STOP message: a message printed on the Decwriter to signal that a POL program has ceased executing. This message usually denotes that execution has ended normally (without error).

storage pointer: a location in the Label of a File. This location holds the address of the Slot in which the next piece of data read by a Foreground Program will be stored.

Subinterval: Each Interrupt Period is divided into eight Subintervals that are two milliseconds in length. These Subintervals are numbered zero through seven. A single Channel of the 8-Channel ADC and 8-Channel DAC may be read or set in each Subinterval.

synchronously: If a Peripheral is operated synchronously, it is loaded or accessed in a Subinterval of an Interrupt Period.

system clock: a location in core that notes the length of time, in milliseconds, the POL operating system has been running. This location is zeroed when POL starts executing and is incremented by sixteen milliseconds after each Hardware Interrupt is generated.

table: a sequence of addresses in core that store related information. A label is assigned to the first address of a table.

transfer error: This error is flagged when the name of a disc record is referenced incorrectly. The error appears on the Decwriter as "ERROR TRANS."

trap: an interrupt generated by an assembly language instruction.

twos complement: the standard method for negating binary numbers. The twos complement of a number is obtained by complementing a number, then incrementing it by one.

usage error: This error is flagged when an instruction is entered in a Programming environment that is illegal for the instruction. The error appears on the Decwriter as "ERROR USAGE".

Variable Reserved Address: a location in core that stores information for the POL operating system. Unlike Fixed Reserved Addresses, Variable Reserved Addresses can be accessed and modified.

vector: an interrupt vector.

voltmeter: a digital voltmeter.

wide converter: See narrow converter.

word: a group of sixteen consecutive bits.

@: This symbol means "at the address".

#: When this symbol prefixes a label, the address of the label is referenced. If this symbol prefixes a number, the value of the number, not an address in core, is referenced.

Appendix C: The Construction of Data Files

Because the precocious programmer may be confined by the dearth of data reduction routines in POL, he can manipulate Files stored on disc with RT-11 programs. If the results of a manipulation are formatted in the POL Filing scheme and stored on disc, these results can be displayed with the POL operating system. This appendix describes the format of Files and the coding of their data.

All Files consists of two parts: a Label followed by Slots. The latter are consecutively numbered from 0 to n-1 where n is the number of Slots in the file. Each Slot consists of two words (32 bits). The Label, which is five words long, describes the data in the File. In the first word, the number of Slots (n) is noted. The next word contains the storage pointer. It is manipulated in Foreground Programs and has no significance in data analysis. In the lower order byte of the third word resides the experimental option; the higher order byte stores the number of the DAC Channel which was monitored while obtaining the data. If either of these bytes is not loaded, negative one abides in the appropriate location. See Table 7.4 for option and Channel assignments. If a DAC Channel was monitored, the fourth and fifth words hold the initial setting and increment of the Channel. Suppose the wide converter can set the experimental parameter associated with the Channel to a maximum value of W, and the fourth word of the label stores S. Then the initial setting of the parameter is W·S/40960. The units of the parameter and W are those given in Table 7.4. If channel 0 were used and S were 20480., the initial D/A setting

corresponds to 1000.0 eV (since W is 2000.0 eV.). The same algorithm deciphers increments. The abscissa of any data slot m is calculated from the relation:

$$\text{abscissa} = \text{initial value} + m \cdot \text{increment}$$

All information in the label and all data points are coded in binary. Negative numbers are in two's complement representation. Although data are coded in binary, some data need further translation before they are accessible to the user.

Data obtained from A/D Channels are in a 12-bit code. A Slot contains the coded value C, and R is the maximum value of an experimental variable particular to the A/D Channel. (See Table 7.4 for Channel assignments of the A/D converter.) Each reading of that channel, T, is deciphered by the algorithm:

$$T = R \cdot C/4096$$

When A/D Channel 0 produces a reading of 2048., the voltage of the LEED gun is 750.0 eV (since R = 1500.0 eV). If the R:AD statement accumulates rather than averages data (see § 4.2), then the data must be normalized prior to conversion. That is, the data must be divided by the number of samples accumulated.

Pulse counts, times, and temperatures are not modified when these are stored in Files. Binary data that correspond to these parameters translate directly into counts, milliseconds, or degrees Kelvin.

DVM readings are stored in the binary code of the decimal display.

As always, negative numbers are in two's complement form. However, the position of the decimal point is not recorded. When a particular parameter is read, the range setting, and hence the decimal point, is fixed by Table 7.4. To convert DVM data to the units in Table 7.4, readings are multiplied by the conversion factors in Table C, that notes how the factor varies with the application mode.

Table C: Correlation between Application Modes of DVMs and the Correction Factor for Translating Binary Readings

Mode	Correction Factors
0	$10.^{-2}$
1	$10.^{-2}$
2	$10.^{-3}$
3	$10.^{-3}$
4	$10.^{-2}$
5	$10.^{-3}$
6	$10.^{-3}$

Appendix D: How the Second Operands of Some Commands are Modified in Assembly

The LIST command prints out an assembled version of the Program numbered by the single operand. Only the first three characters of each mnemonic are displayed. In the format for entering two operands, these are separated by a semicolon in the listing. Most operands are printed in the same decimal (base ten) code in which they were entered. Occasionally, the second operand is modified in assembly. Table D catalogs which operands are modified and how these operands are assembled. Except for settings of the DAC Channels, the operands are assembled through a simple arithmetic algorithm. By employing the algorithms given in this appendix and the output from the LIST command, a user can back-calculate the second operands that were entered on the Decwriter.

DAC settings are assembled as follows. Let y be the second operand of the S:DA or I:DA command that is entered on the Decwriter and w be the maximum value of the experimental parameter monitored by the DAC Channel. Values of w for each Channel are listed in Table 7.4. The settings of the wide and narrow converters for a Channel, s and i, are given by the algorithms

$$s = \sim[4096. \cdot y/w] + 1$$

$$i = \sim\left\lfloor \frac{40960.}{w} \cdot \left(\frac{(s+1)\cdot w}{4096.} + y \right) \right\rfloor$$

where ~ denotes the complement operation. If y is associated with

the S:DA command and s is zero, one is subtracted from s before i is determined. Note that i always assumes values between -1 and -15., inclusive. For the S:DA command, s is between -1. and -4096., inclusive, but may assume values from -4096. to 4096., inclusive, for the I:DA command. The second operand of the S:DA or I:DA commands, which is one word in length, is assembled by placing s into the 12. most significant bits and i into the 4 least significant bits of the word. For the DAOFF or DAEXP commands, the second operand is assembled from the algorithm

$$\sim[40960. \cdot y/w]$$

where y is the second operand that is entered on the Decwriter. The definition of w is unchanged.

Table D: How the Second Operands of Some Commands are Assembled

The second operand is:	The operand is associated with the instructions:	The algorithm for assembly is:[1]
an alphabetic name	F:GET,F:PUT,F:RID, P:GET,P:PUT,P:RID	The first three ASCII characters ($y_\alpha, y_\beta, y_\gamma$) are assembled into a RADIX 50 word. $1600.\cdot(y-64.)+40.\cdot(y-64.)+(y-64.)$
time	W:CLOCK,R:AD,R:PC	$y\cdot(1000./16.)$
a D/A setting	S:DA,I:DA	see text
a modification to a D/A setting	DAOFF,DAEXP	see text
a number of Slots	S:STORE	$4\cdot y$
a number of Slots	I:STORE	$4\cdot(y-1)$
a zero-point for graphing	X:MOVE,Y:MOVE	$327.68\cdot y$
a mantissa	ADD,SUB,MUL,DIV LOAD,OFFSET	$1000.\cdot y$
an expansion factor	EXPAND	$1000.\cdot y$

1. The variable y is the second operand that is entered on the Decwriter.

APPENDIX C

THE DESIGN OF A HIGH-CURRENT POWER SUPPLY WITH A SERVOMECHANISM FOR CONTROLLING THE TEMPERATURE OF A SUBSTRATE

This appendix details the design of a high current power supply that heated the substrate in this study and also describes the operation of a servomechanism that controlled the temperature of the substrate. The circuitry for both devices is shown in Fig. 1. The operation of the power supply and the servomechanism will be elucidated through a description of the four components - the EMF amplifier, the comparator, the manual control and the current source in the circuitry.

The EMF amplifier is a chopper stabilized amplifier (AD260K), which is isolated from ground as are all components in Fig. 1. To select the gain of the amplifier, switch S_3 is set to one of the five ranges shown in Fig. 1. When the EMF of a Pt-Pt10%Rh thermocouple on the substrate is being amplified to provide an input signal to the servomechanism, the gain is fixed at x100. The design of the EMF amplifier is described in the product information for the AD260K (Analog Devices, Inc. Norwood, Mass.).

The comparator divides a 0-10 V signal from the computer by ten, subtracts this signal from the output of the EMF amplifier, and then passes the difference through an integrator with a time constant of 1 sec. The signals from the EMF amplifier and the computer enter the comparator through points R and C, respectively. If the dc gain of the servomechanism were infinite, a 10V signal from the computer would cause the thermocouple EMF to be 10 mV (~900 K). (Ideally, the dc gain is infinite because the integrator has infinite gain at low frequencies.) This servomechanism does not program the temperature of the sample linearly, but programs the EMF of the thermocouple linearly. However,

the temperature is nearly proportional to the EMF over limited temperature ranges. (See Appendix B). Diodes D_1 and D_2 limit the output of the comparator to 1.2 V.

Either the output of the comparator or the manual control provides a reference signal (V_s) to fix the output current of the current source (I). The source of the control signal is selected by switch S_1. When S_1 is in position P, the compartor provides the reference signal; when in position M, the manual control provides the signal. In the latter case, the potentiometer R_{10} is used to vary the reference signal continuously. The maximum current that may be generated from the current source is selected with switch S_2 and may be set at 10, 12 or 15 A.

The gain for programming the current source (I/V_s) is $R_6^{-1}=10\ \Omega^{-1}$. That is, when V_s is 1 V, the current I is 10 V. As is shown below, the response time of the current source is fast compared to that of the servomechanism. The current source heats the sample conductively by passing a current through two 10 mil Ta wires that support the substrate. The measured resistance of these wires (R_c) is about 0.1 Ω at room temperature. The EMF of the thermocouple on the sample changes by 0.1 to 1 mV per ampere of current that is passed through the Ta wires. The measured response time for this change is 1 to 10 sec. Hence, the EMF at equilibrium and response time are non-linear functions of the applied current.

Using conventional techniques to analyze the frequency response of the servomechanism, its control scheme can be shown to be stable. As

determined from the information given above, the transfer function for programming the EMF of the thermocouple with the voltage from the computer is 1 mV per 1V. Moreover, the transfer function has a pair of poles that are nearly critically damped. That is, the thermal time constant for heating the substrate nearly equals the time constant of the servomechanism. If the time constant of the servomechanism were larger, the equilibrium value of the thermocouple EMF would be obtained more slowly. If the time constant of the servomechanism were smaller, power surges from the current source, which occur when the response of the servomechanism is under damped, melt the Ta wires supporting the sample. (This point has been proven experimentally.) If the servomechanism is implemented with the scheme shown in Fig. 1, the thermocouple EMF at equilibrium is stable to 1 μV (~0.1 K) whereas the measured response time is about 5 sec.

The operation of the current source is described as follows. A constant voltage source (V_i) capable of supplying 18 A at 10 V furnishes the unregulated power to the current source. The current that heats the crystal (I) is developed across the resistor R_6 and the Ta wires with resistance R_c. The amplifier A_1 generates the difference of the product IR_6 and the programming voltage V_s to regulate the transistor Q_3. If the current I is larger than V_s/R_6, Q_3 will divert current from the base of the current amplifier formed by the Darlington pair of Q_1 and Q_2. Hence, the current I will be decreased. If the current I is smaller than V_s/R_6, Q_3 will pass less current so that more current is amplified by the Darlington pair to increase I.

More rigorously, the dependence of I upon V_s and V_i is simply

$$I = [G_\infty V_A + G_0 V_i/T][1/T + 1]^{-1}$$

where

$$G_\infty = R_6^{-1}$$

$$G_0 = \left\{R_1 + \beta^2 \left[(R_c + R_6) \| \frac{1}{C_3 s}\right]\right\}^{-1} \approx R_1^{-1}$$

$$T = \frac{\beta R_6}{R_6 + R_c} \frac{R_3}{R_4} \frac{\left[R_1 \| \beta^2 (R_c + R_6)\right]}{R_2} \times$$

$$\left\{1 + \left[(R_6 + R_c) \| \frac{R_1}{\beta^2}\right] \cdot C_3 s\right\}^{-1} \left\{1 + R_3 C_2 s\right\}^{-1} \left\{1 + R_p C_1 s\right\}^{-1}$$

Also, β is defined as the dc short-circuit current gain in the common-emitter configuration, which is approximately 20 for Q_1, Q_2 and Q_3; R_p is the parasitic resistance between the collector and base of Q_3 (~100 KΩ); s is complex frequency; and $\|$ means "in parallel with". From the analysis, one can conclude the following. The current I is proportional to V_s/R_6 within 0.5%. The dc offset current through the substrate is $G_0 V_i/T$=1 mA. The close-loop gain (T) is 200 at low frequencies and has three poles near 100 H_z. These poles roll off the gain at high frequencies to effect stability and yet maintain a large gain to filter the 60 cycle noise from V_i. The ac noise component of the current I was measured as 5 mA peak-to-peak. Moreover, the close-loop gain of the current source was determined to be stable using conventional frequency response techniques. The thermal stability of β for each transistor is not important as long as Q_1, Q_2 and Q_3 are mounted on heat sinks to dissipate the large amount of power used to regulate the current I.

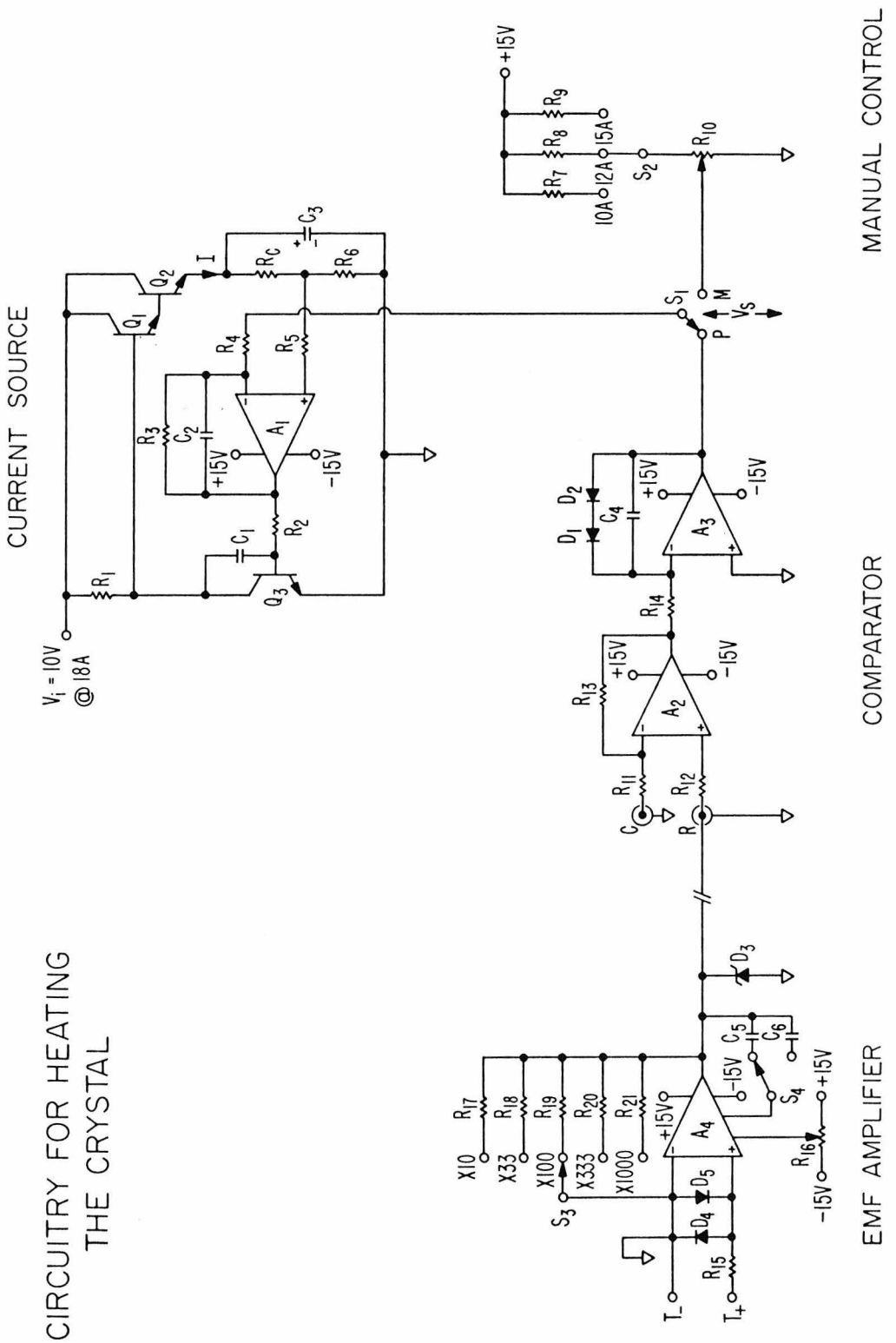

CIRCUITRY FOR HEATING
THE CRYSTAL
CURRENT SOURCE
COMPARATOR
MANUAL CONTROL
EMF AMPLIFIER
V_i = 10V
@ 18A
R_1
Q_1
Q_2
Q_3
I
R_C
C_3
R_6
R_4
R_5
R_3
C_2
A_1
+15V
-15V
C_1
R_2
S_1
P
M
V_S
+15V
R_9
R_8
15A
R_7
12A
10A
S_2
R_{10}
D_1
D_2
C_4
A_3
R_{14}
R_{13}
A_2
R_{11}
R_{12}
C
R
D_3
C_5
C_6
S_4
R_{17}
R_{18}
R_{19}
R_{20}
R_{21}
X10
X33
X100
X333
X1000
S_3
A_4
D_4
D_5
R_{15}
R_{16}
I_
I+

Fig.1

R_1	40Ω/5W/5%
R_2	100Ω/1W/5%
R_3	20KΩ/½W
R_4	200Ω/½W
R_5	200Ω/½W
R_6	0.1Ω/40W/5%
R_7	700Ω/1W
R_8	560Ω/1W
R_9	430Ω/1W
R_{10}	50Ω/2W
R_{11}	100KΩ
R_{12}	9KΩ
R_{13}	10KΩ
R_{14}	1MΩ
R_{15}	30Ω/½W
R_{16}	50KΩ
R_{17}	3KΩ
R_{18}	10KΩ
R_{19}	30KΩ
R_{20}	100KΩ
R_{21}	300KΩ

C_1	0.02μF
C_2	0.05μF
C_3	0.018F/15V
C_4	1μF/Mylar
C_5	0.1μF/Mylar
C_6	1μF/Mylar
D_1	1N4005
D_2	1N4005
D_3	1N4740
D_4	1N914B
D_5	1N914B
Q_1	MJ802
Q_2	MJ802
Q_3	2N3055
A_1	741
A_2	741
A_3	AD0042C
A_4	AD260K

All resistors are ¼W/1% unless noted otherwise.