NBSIR 84-2853

Measuring the Corrosion Rate of Reinforcing Steel in Concrete

U.S. DEPARTMENT OF COMMERCE
National Bureau of Standards
National Measurement Laboratory
Center for Materials Science
Metallurgy Division
Washington, DC 20234

April 1984

Sponsored by:
Federal Highway Administration
Washington, DC 20590

NBSIR 84-2853

MEASURING THE CORROSION RATE OF REINFORCING STEEL IN CONCRETE

E. Escalante, M. Cohen, A. H. Kahn

U.S. DEPARTMENT OF COMMERCE
National Bureau of Standards
National Measurement Laboratory
Center for Materials Science
Metallurgy Division
Washington, DC 20234

April 1984

Sponsored by:
Federal Highway Administration
Washington, DC 20590

U.S. DEPARTMENT OF COMMERCE, Malcolm Baldrige, *Secretary*
NATIONAL BUREAU OF STANDARDS, Ernest Ambler, *Director*

MEASURING THE CORROSION RATE OF REINFORCING STEEL IN CONCRETE

The deterioration of concrete bridge decks as a result of corrosion action on the reinforcing steel is a serious and costly problem to the nation (1). It is recognized that this corrosion is caused by the diffusion of chloride ions into the concrete resulting in the breakdown of the passive iron oxide film that normally protects the steel. The corrosion product that forms generates sufficient pressure to crack the concrete allowing moisture, oxygen, and more chloride to enter, further accelerating the deterioration process (2).

Methods have been developed that: a) block the chloride from coming in contact with the steel, (b) stop the chloride from entering the concrete, (c) remove the chloride from the vicinity and (e) others, such as cathodic protection, which modify the reactions at the surface of the steel (3-7). All of these approaches require some method to evaluate their effectiveness, and the most direct method is to measure the corrosion rate of the steel before and after a modification to the concrete.

Procedure

The plan for this study is divided into the following five phases which are: A literature review, preliminary studies, measurements in concrete, field measurements, and the development of a microprocessor system. More than half of these phases have been accomplished and reported as will be described.

Phase I. Literature Review - At the outset, a literature search was undertaken to identify the information available on the corrosion of steel in concrete and methods for measurement. A total of 394 papers, reports, and talks from 1964 through 1978 were identified on the subject, and are indexed under subject and author (8). The Subject Index was further subdivided into six subgroups such as general survey, factors affecting corrosion, measurement techniques, protection techniques, concrete design, and related fields. In the Author Index, the references are listed alphabetically by author.

A study of this literature revealed several interesting facts. For example, it was found that over thirty references discussed the effect of chloride on corrosion, but only one reference discussed the effect of oxygen content. Furthermore, the effect of moisture was considered in just four of these references. Not one study on the corrosion of steel in an alkaline environment was found. Clearly, most of the attention had been directed towards the effects of chloride.

Phase II. Preliminary Studies - From the information gathered through the literature search, it was evident that more information was needed on the effects of oxygen, pH, and moisture, and therefore an experimental program was designed to evaluate these factors. Three environments were used on this phase of the study, and these were alkaline solutions of $Ca(OH)_2$ with controlled oxygen, Cl^-, and pH; wet sand/salt mixtures; and mortar (9).

Flat steel specimens 100 x 20 x 1 mm were immersed in controlled solutions and the initiation time for visible corrosion was noted. At a pH of 10.5 or less, corrosion started immediately on immersion. However, as the pH was raised to 12, initiation times increased to 200h, and at higher pH corrosion did not occur during the period of the experiment. The experiment was repeated in the presence of chloride, and it was found that chloride did reduce the initiation time to corrosion but otherwise had little effect. Once initiated, the corrosion rates were constant in all cases. Not surprisingly, the oxygen content of the solution had a large effect on the corrosion rate of the steel with lower concentrations leading to lower corrosion rates.

Further experiments were carried out as described which indicated that the interaction of oxygen, pH, and chloride is more complex and in combination can develop conditions that lead to corrosion even at high values of pH. Figure 1 is a summary of these data showing these three parameters. Illustrated is a three dimensional figure with boundaries within which corrosion does not occur. However, outside of this boundary corrosion of steel is observed. For example, where it was shown earlier that chloride or oxygen acting alone could not cause corrosion of steel at a pH of 12.5, this figure reveals that at an oxygen concentration as low as 2 ppm and at a Cl^- concentration of 6 mol/l, corrosion does occur, and can also develop at lower concentrations of Cl^- at higher oxygen contents.

To observe the effect of moisture on the corrosion of steel, specimens were imbedded in a mixture of sand, $Ca(OH)_2$, and NaCl. The mixture was held in small perforated crucibles in an atmosphere of flowing air with controlled moisture content. This configuration allowed the specimens to be removed for weight loss determination during the course of the experiment. Specimens in a moisture saturated sand mixture did not corrode, but during the same period, specimens in a dry mixture began to corrode almost immediately as moisture was introduced. However, the corrosion rate of these specimens decreased steadily as moisture content of the sand mixture increased; and at saturation, the corrosion ceased. Clearly, the degree of moisture, which affects oxygen concentration, plays a very important role as expected, but the cessation of corrosion at saturation was not anticipated.

After the study of steel in solutions and in sand mixtures was completed, the work was extended to include steel in mortar. Specimens as described above were cast with a 5 mm cover of chloride-containing mortar. After 24h, half of the specimens were immersed in a $Ca(OH)_2$/NaCl solution and half were allowed to dry for one week before immersion. The corrosion current of each specimen was determined by polarizing the specimen from -10 to +10 mV with a potentiostat and plotting the overvoltage as a function of the polarizing current. As in the case of the sand mixture, the corrosion of the steel in mortar that was kept continuously wet was near zero, but the specimens that were allowed to dry for one week, and then immersed, initially developed a high corrosion rate that decreased with time to a very low value.

In summary, these studies have shown that at low concentrations of O_2 and Cl^-, pH controls the initiation of corrosion. But as O_2 and Cl^- increase, corrosion can initiate even at a pH of 12.5. It is hypothesized that in the moisture saturated concrete, the concentrations of Cl^- and oxygen are sufficiently low for conditions to exist within the "no corrosion" boundary of Figure 1. However, as the concrete dries, the Cl^- concentration in the pore water eventually increases to saturation. This drying process also allows oxygen to diffuse more rapidly through the concrete setting up conditions outside of the "no corrosion" boundary where corrosion of steel does take place. Once corrosion is initiated, the pH is reduced at the anodic areas by the corrosion reaction, making it easier to reinitiate corrosion in future times when moisture and oxygen are available.

Phase III. Measurements in Concrete - The purpose of this portion of the work was to compare several polarization techniques for measuring the rate of corrosion of steel in concrete(9). All polarization techniques are essentially the same in that the measurement is based on the observation that the polarizing current necessary to change the potential of a metal in an electrolyte, in this case concrete, is directly related to the rate of corrosion of the metal. The techniques differ only by how this change in potential is achieved. Initially, four techniques were used in this study and are described as follows: Stern and Geary consider the derivative of the potential of the specimen versus the applied current for deviations from the corrosion potential not greater than 10 mV (10). This slope is related to the corrosion current, I_{corr}, by the relationship:

$$\lim_{dE \to 0} \frac{dE}{dI} = \frac{b_a \cdot b_c}{2.3(b_a + b_c)I_{corr}}$$

where b_a and b_c are the anodic and cathodic Tafel Slopes respectively. For calculation of the corrosion current, I_{corr}, values for the Tafel slopes must be obtained from other sources, or by trial and error for the best fit to the data. In this case ΔE was 6 mV and the assumption made that $b_a = b_c = 100$ mV. Thus:

$$I_{corr} = \frac{21.7\Delta I}{\Delta E}$$

The second technique used is an empirical approach described by Schwedtfeger in which I_{corr} is determined from the change in slope of the polarization curve (11). The cathodic corrosion current, I_p, and the anodic corrosion current, I_q, are found at this "break" in the curve, and the corrosion current is then calculated from:

$$I_{corr} = \frac{I_p\ I_q}{I_p + I_q}$$

The technique requires that polarization, ΔE, extend over a range of ±100 mV so that the "breaks" in the polarization curve may be observed.

The three point method of Barnartt is the third technique that has been used (12). Three voltage/current data points are measured along the polarization curve, and from this b_a, b_c, and I_{corr} are calculated as follows. If the first potential point is at ΔE, then the second and third are at $2\Delta E$ and $-2\Delta E$ respectively. The resulting currents at each potential are measured and substituted into the ratios

$$r_1 = I_{2\Delta E}/I_{-2\Delta E}$$

and $$r_2 = I_{2\Delta E}/I_{\Delta E}$$

which in turn are related to each other through the quadratic equation

$$U^2 - r_2 U + r_1 = 0$$

The roots of this equation are

$$\exp \frac{2.3\ \Delta E}{b_a} \text{ and } \exp \frac{2.3\ \Delta E}{b_c}$$

from which b_a and b_c are calculated. Using the relationship of I/I_{corr} versus ΔE, the corrosion current for the process is found.

The fourth technique is a computer analysis of polarization data developed by Mansfield (13). The computer makes a best fit analysis to theoretical curves generated from the Stern-Geary equation for different values of Tafel slopes and in this way b_a, b_c, and I_{corr} are found.

Each of the polarization techniques described employs a three electrode system having the steel specimen as one electrode, a voltage reference as a second electrode, and a counter electrode as a third electrode from which polarizing current is applied to the specimen. The circuit used for these measurements is Holler's (14) which incorporates a Wheatstone bridge for IR compensation as illustrated in Figure 2. The IR compensating design takes advantage of the fact that when a polarizing current is applied to the specimen, its rate of change in potential due to polarization is very small compared to the rate of change in potential due to IR. By repeated applications of current of short duration (e.g. 0.5s), and by adjusting the Wheatstone balancing resistance, the IR component is compensated. Once this balance is achieved, then the polarizing potential, free of IR, can be measured for all values of applied current.

Small concrete slabs 55 x 30 x 5 cm were cast each with three imbedded 1.27 cm diameter steel rods. Two small stainless steel rods were also encased in the concrete and were used to determine the resistivity of the slab. The concrete was treated in two ways. Four of the slabs were cast with 0.06 parts NaCl to one part cement (38.4 lbs NaCl/cu.yd), and the remaining eight slabs were cast without Cl^-. After curing for approximately thirty days, two of the Cl^- free slabs were immersed in a saturated solution of NaCl for six days, withdrawn, and left in a dry laboratory atmosphere. Two other slabs were immersed in a 3.5% NaCl solution for 24h, withdrawn, and also left in a laboratory atmosphere. Corrosion measurements showed that steel specimens in Cl^- free slabs developed very low corrosion rates (<0.007 mg/dm^2/day). However, those slabs with Cl^- in the concrete immediately developed a corrosion rate of 4.4 mg/dm^2/day or more than two orders of magnitude greater than those in Cl^- free concrete. Figure 3 illustrates the decreasing rate of corrosion as the concrete dried. In the case where the chloride free concrete slabs were immersed for six days in a sodium chloride saturated solution, the corrosion rate increased by over an order of magnitude within a few hours as shown in Figure 4. The second group of chloride free concrete slabs that were immersed for only 24h in a solution of 3.5% NaCl behaved similarly to those immersed longer in a more concentrated solution. The average weight loss of steel on these specimens, based on electrochemical measurements, was 0.33 mg/dm^2/day. The resistivity of the concrete was also measured with time, and the data reveal that the resistivity decreases as moisture increases. However, the effect of immersion on the corrosion rate is much greater than its effect on the resistivity of the concrete. As the concrete dries and the resistivity increases above 7,000 ohm-cm, the resistance appears to have a direct influence on the degree of corrosion observed.

Weight losses were calculated from the electrochemicaly determined corrosion currents measured as a function of time, and were compared to gravimetric weight loss measurements on the same cleaned steel rods. A comparison of the data from the four electrochemical techniques described is shown in Table 1 and indicates that all methods overestimated the weight loss of steel. The Stern-Geary method overestimated weight loss by 17% while Schwerdtfeger's, Mansfeld's, and Barnartt's techniques overestimated weight loss by 40%, 56%, and 83% respectively. Since the differences in the corrosion rates between a corroding and a non-corroding system are over an order of magnitude, then even the data with an overestimation of 83% will reveal the existence of high corrosion.

In summary, the data indicate that the corrosion of steel in chloride free concrete is negligible and less than 0.007 mg/dm^2/day while the corrosion rate in chloride contaminated concrete is more than an order of magnitude greater with an average of 0.33 mg/dm^2/day, but can be as high as 4.4 mg/dm^2/day. Comparison of the gravimetrically determined weight loss to the weight losses calculate from the polarization data indicates that the electrochemical measurements overestimate weight loss by 17% to 83%. However, even with an overestimation of 83%, the polarization techniques will differentiate between a corroding and a noncorroding condition of steel in concrete.

An important part of this study is the characterization of the current distribution during polarization measurements. This distribution must be known in order to determine the area of the steel being polarized which, in turn, makes it possible to calculate a corrosion current density or weight loss of steel per unit length of steel bar.

Under controlled laboratory conditions, determining this area is not a serious problem since the specimens are small and the geometry of the electrodes can be arranged so that the entire specimen electrode is polarized with a uniform current. But in extending these polarization measurements to a bridge deck we are faced with limitations brought on by the relatively large size of the structure. The major constraint being that, unlike our small slabs in the laboratory, we cannot polarize the entire bridge at once, but can only polarize a small area at a time, and it is this area that must be characterized.

The experimental procedure used is simple in principle. In an attempt to simulate the large scale geometry in a small slab, it was decided to use a long line specimen in concrete and polarize it from a point or small line current source. The potential of the steel in the concrete would be monitored by reference electrodes positioned along its length. Three variables were considered as having an effect on the distribution of the current, and these are the distance between the counter electrode and the specimen, the amount of current applied, and the resistivity of the concrete.

Using small concrete slabs similar to those described earlier, a new geometric arrangement of electrodes was designed as shown in Figure 5. As before, the steel rod is imbedded horizontally along the length of the concrete slab. The counter electrode is a small diameter (3 mm) rod positioned vertically adjacent to the steel specimen. Three counter electrodes at different distances (1,2,4cm) from the specimen are illustrated. Ten reference electrodes, imbedded in the concrete, are located 8.5 cm apart along the length of the steel rod and 2 cm from its surface. This arrangement of electrodes allowed a polarizing current to be applied to the steel specimen from a small vertical line source. Any changes along the length of the specimen are detected by the reference electrodes. Measurements were made using one of the three counter electrodes. The resistivity of the concrete was controlled by controlling its moisture content. During an experimental run, the potential along the length of the specimen was measured before any current was applied. Subsequent potential measurements were made with reference to this initial base potential. The polarizing current was increased in increments and the potential measurements repeated until a maximum of approximately 10 mV was achieved. Initial measurements were made manually using the Holler bridge circuit for compensation of IR. More recent measurements were made using a computer controlled system which applies a cycling direct current of 5 seconds on and 0.25 seconds off. The potential of the specimen was then read during the current off cycle so that the IR component was zero. Figure 6 illustrates the distribution of current as indicated by the change in potential along the length of a steel rod in concrete. This curve is a measure of the extent of current distribution along the length of a horizontal steel rod when the current source is a small vertical rod in the concrete. The shape of the curve is an indication of how uniformly the current is distributed within the polarized area. This type of data have been obtained as a function of counter electrode distance, applied current, and resistivity of concrete. Preliminary analysis of the data indicates that, as expected, the extent of the current distribution decreases as the resistivity of the concrete increases, and the general shape of the current distribution curve does not appear to be affected. In addition, as the electrode distance is decreased, the area of steel rod polarized also decreases. The largest effect observed is that of the applied current. As the current is increased, the area of distribution increases. But perhaps the most important observation is that current distribution is very limited and 90% of the current encompasses a distance of less than 10x the distance between the specimen and the current source electrode.

A mathematical model has been developed that takes these variables into consideration and generates a current distribution curve along a cylinder from a point source in free space as follows. The electric current distribution in a conductive medium containing electrodes held at arbitrarily fixed potentials may be obtained by a standard application of potential theory. For the case of an infinitely long cylindrical conductor embedded in an electrolyte with a point source of current at a fixed distance from the cylinder, an extension of the formulas given by Jackson gives a complete solution for the current distribution in the electrolyte (15).

The following geometric arrangement is treated as illustrated in Figure 7. Consider an infinitely long metallic cylinder of radius a, held at ground potential, and at a distance, d, from the axis of the cylinder, a point electrode emits a total current I_0. By application of the methods stated above, the current density at any point, P, on the surface of the cylinder is described.

$$J(Z,\theta) = \frac{I_0}{\pi^2 a} \sum_{m=0}^{\infty} \frac{\cos m\theta}{\varepsilon_m} \int_0^{\infty} dk \cos kz \frac{K_m(kd)}{K_m(ka)}$$

where

$J(Z,\theta)$	=	Current density at point P on the cylinder
I_0	=	Total current emitted by source at d
m	=	Integer index for summation
k	=	Variable of integration
ε_m	=	2 for m = 0; 1 for m > 0
$k_m(x)$	=	A modified Bessel function of the second kind, order m, argument x, see ref. (16)

Although this formula is a closed mathematical expression with a precisely defined meaning, its evaluation to a specified accuracy requires numerical integration of a function which is rapidly oscillating and the summation of a convergent series. Use of digital computing methods to produce graphical or tabular representation of the results is essential. A preliminary evaluation of the function, illustrated in Figure 8, demonstrates that it does follow the general shape of the measured data. Further analysis of this function and its correlation to the data is required for a more accurate assessment of their relationship.

Phase IV. Field Measurements - After gaining some experience in performing the corrosion measurements on concrete slabs in the laboratory, the measurements were extended to the field on an operating bridge deck. With the cooperation of the Virginia Highway Department arrangements were made to perform corrosion rate measurements on a bridge deck (Number 2063) on Interstate 66 near the Washington, DC Beltway. The primary objective of this portion of the study was to evaluate the background electrical noise and determine whether the measurements could be made.

Measurements were made on two separate decks that were a few hundred meters apart. Two sets of measurements were made on a seven year old deck (area A) repaired and widened (area B) eighteen months earlier and now in use, and one set was made on a deck (area C) over a year old but not yet opened to traffic. The major difference between the two decks is that the older deck with its new widened section had been exposed to de-icing salts while the new deck, barricaded to traffic, was not salt contaminated.

During the measurements, electrical contact was made to the steel rebar with modified locking pliers. The counter electrode was a 500 cm^2 rectangle of aluminum foil wrapped in a towel wetted with a saturated solution of

$Ca(OH)_2$. A 6 cm diameter hole was cut out of the center of the counter electrode to accommodate the $Cu/CuSO_4$ reference electrode positioned at this point on a small wetted sponge. Using the Stern-Geary technique and the Holler bridge circuit, measurements were made at areas A, B, and C, and a portable Tetronix Model 214 oscilloscope was employed to monitor the background electrical noise.

The background electrical noise was found to be a very high frequency signal beyond the kilohertz range of our instrument, but more importantly, it was of very low amplitude and less than 1 mV. Apparently the steel structure of the deck acts as a very good antenna for radio frequency radiation. The corrosion measurements revealed that area A on the 7 year structure developed a corrosion current of 9.8 μA, while area B on the recently widened section of the same structure had a corrosion current of 6.9 μA. The new deck not open to traffic displayed the lowest corrosion current of 1.2 μA. Thus, though the data could not be verified by examination of the steel, it did seem reasonable. That is the oldest contaminated structure did display the highest corrosion and the new uncontaminated deck the lowest corrosion. Furthermore, it was found that the measurements could be made and that background interference was negligible.

Phase V. Microprocessor System - It is recognized that the corrosion measurements are tedious, time consuming, and require some knowledge of the concepts involved, and because these measurements lend themselves to automation, the ultimate goal of this project is to develop an instrument that will make these measurements automatically. Field measurements would then be performed with a microprocessor controlled device which would relieve the operator of most of the detailed adjustment, data acquisition, and calculations required. This device is to be as simple allowing minimally trained personnel to perform the measurements. A description of the design of such a device follows.

The instrument is designed around a set of commercially available microcomputer printed circuit cards. These are based on a MC 6809 microprocessor circuit giving the system a very capable and powerful controller. The modules used in the system are:

1.) The processor card

2.) Battery backed-up memory

3.) Serial, parallel, and timing card

4.) Analog to digital converter

5.) Digital to analog converter (modified)

6.) Vacuum fluorescent display

In addition to these commercial cards, several other cards have been constructed at NBS, and serve two functions. First, to provide an interface between the user and the system are the following:

1.) Keyboard and display panel

2.) Test and keyboard control

Second, to interface the instrument to the experimental cell, these cards were constructed:

3.) Voltage to current converter

4.) Input amplifier and distribution card

Schematics of these cards and the modification to the digital to analog converter are illustrated in Figures 9 through 13.

The instrument operates in the following manner. At turn on, if the test switch is on, the unit is in a display test mode. The system will not function except for the display which will demonstrate the full character set on a character by character basis. When the test switch is deactivated, followed by a reset, the system goes into operational mode and instruction messages appear on the display.

In the operational mode, the processor requests information from the user, and when all necessary parameters are stored, sets up the system for zero levels and enters a wait state until the operator lets it know that final hook up to the cell is complete. This information is transmitted to the instrument by pressing the "RUN" key thereby activating the microprocessor to begin data acquisition. Output current is generated by a precision voltage to current converter. Voltage driving this converter is derived from a 10 bit digital to analog converter under control of the processor. The full scale output current range is also selectable by the processor software. Input potentials are measured by a 12 bit analog to digital converter. Three channels are used in this measurement. One channel measures the current applied, another the counter electrode to working electrode potential, and the third the reference electrode to working electrode potential where the working electrode is steel in the concrete. The data are then saved in a battery backed up memory remaining there until deliberately erased. When data acquisition is complete, calculations are performed and the results displayed. A permanent record can then be printed out and the data erased, or another experiment can be performed while the data is saved for later printing. The software has been developed as a system of modular routines, the majority of which are single function stand alone subroutines (Appendix A). Links between modules have been made absolute, and therefore the overall program is not relocatable. However, relocation can be put into the system with little difficulty, if it should prove desirable. The software design is such that changes to the measurement technique can be made by changing 3 or 4 of the modules. Such a change is necessary to the present modules since interfacing problems have resulted in an error in the measurements. These changes are in the process of being implemented.

Summary

This National Bureau of Standards study has revealed the following information.

1.) A total of 394 papers, reports, and talks from 1964 to 1978 were found pertinent to the subject of the corrosion of steel in concrete. A study of this literature indicated that information was lacking on the effect of oxygen, pH, and moisture.

2.) Oxygen concentration, chloride concentration, and pH have been found to control the initiation of the corrosion of steel in concrete. Moisture content of the concrete also plays an important role through its effect on the concentration of oxygen and chloride.

3.) The comparison of four polarization techniques for measuring the corrosion of steel in concrete reveals that all the techniques overestimated the weight loss from 17 to 83%. Current distribution measurements indicate that the extent of current distribution along a steel bar in concrete is very limited and of the order of 10 times the distance between the specimen and the current source. A mathematical model describing this current distribution is being developed.

4.) Corrosion rate measurements on operating bridge decks verify that polarization techniques can be used in this situation, and furthermore, background electrical noise is negligible.

5.) A microprocessor controlled instrument for making corrosion rate measurements has been designed and constructed. However, complications in interfacing the cards have developed requiring a change in its design. These improvements are in the process of implementation.

REFERENCES

1. An Economic Analysis of the Environmental Impact of Highway Deicing, U. S. Environmental Protection Agency, EPA-600/276105, May 1976.

2. Cornet, I., Bresler, B., Galvanized Steel in Concrete: Literature Review and Assessment of Performance, Galvanized Reinforcement for Concrete-II, International Lead Zinc Research Organization, Inc., Vol. 2, May 1981.

3. Clifton, J. R., Protection of Reinforcing Bars with Organic Coatings, Materials Performance, 15, No. 5, 14-17, May 1976.

4. Internally Sealed Concrete Guide to Construction and Heat Treatment, Federal Highway Administration, Implementation Package 77-9, April 1977.

5. Slater, J. E., Lankard, D. R., Moreland, P. J., Electrochemical Removal of Chlorides from Concrete Bridge Decks, Materials Performance, 15, 11, 21-26, 1976.

6. Lundquist, J. T., Rosenberg, A. M., Gaidis, J. M., Calcium Nitrite as an Inhibitor of Rebar Corrosion in Chloride Containing Concrete Materials Performance, 18, 36-40, March 1979.

7. Fromm, H. J., Wilson, G. P., Cathodic Protection of Bridge Decks: A Study of Three Ontario Bridges, Transportation Research Board, Transportation Research Record No. 604, 38-47, 1976.

8. Escalante, E., ITO, S., A Bibliography on the Corrosion and Protection of Steel in Concrete, National Bureau of Standards Special Publication 550, August 1979.

9. Escalante, E., Ito, S., Cohen, M., Measuring the Rate of Corrosion of Reinforcing Steel in Concrete, National Bureau of Standards NBSIR 80-2012, March 1980.

10. Stern, M., Geary, A. L., A Theoretical Analysis of the Shape of Polarization Curves, J. Electrochemical Society, 104, 56, January 1957.

11. Schwerdtfeger, W. J., McDorman, O. N., Measurement of the Corrosion Rate of a Metal From Its Polarization Characteristics, J. Electrochemical Society, 99, 407, October 1952.

12. Barnartt, S., Two-Point and Three-Point Methods for the Investigation of Electrode Reaction Mechanisms, Electrochimica ACTA, 15, 1313, 1970.

13. Mansfeld, F., Tafel Slopes and Corrosion Rates from Polarization Resistance Measurements, Corrosion, 29, No. 10, October 1973.

14. Holler, H. D., Studies of Galvanic Couples, J. Electrochemical Society, 97, 271, September 1950.

15. Jackson, J.D., Classical Electrodynamics, John Wilesy and Sons, N.Y., 84, 1962.

16. Handbook of Mathematical Functions with Formulas, Graphs, and Mathematical Tables, National Bureau of Standards, Applied Mathematics Series No. 55, 1964.

TABLE 1

Comparison of Weight Loss Determined Gravimetrically

to

Weight Loss Calculated From Polarization Data

Technique	Weight Loss, mg	Overestimate %
Gravimetric	147	--
Stern-Geary	172	17
Schwerdtfeger	206	40
Mansfeld	230	57
Barnartt	269	83

FIGURES

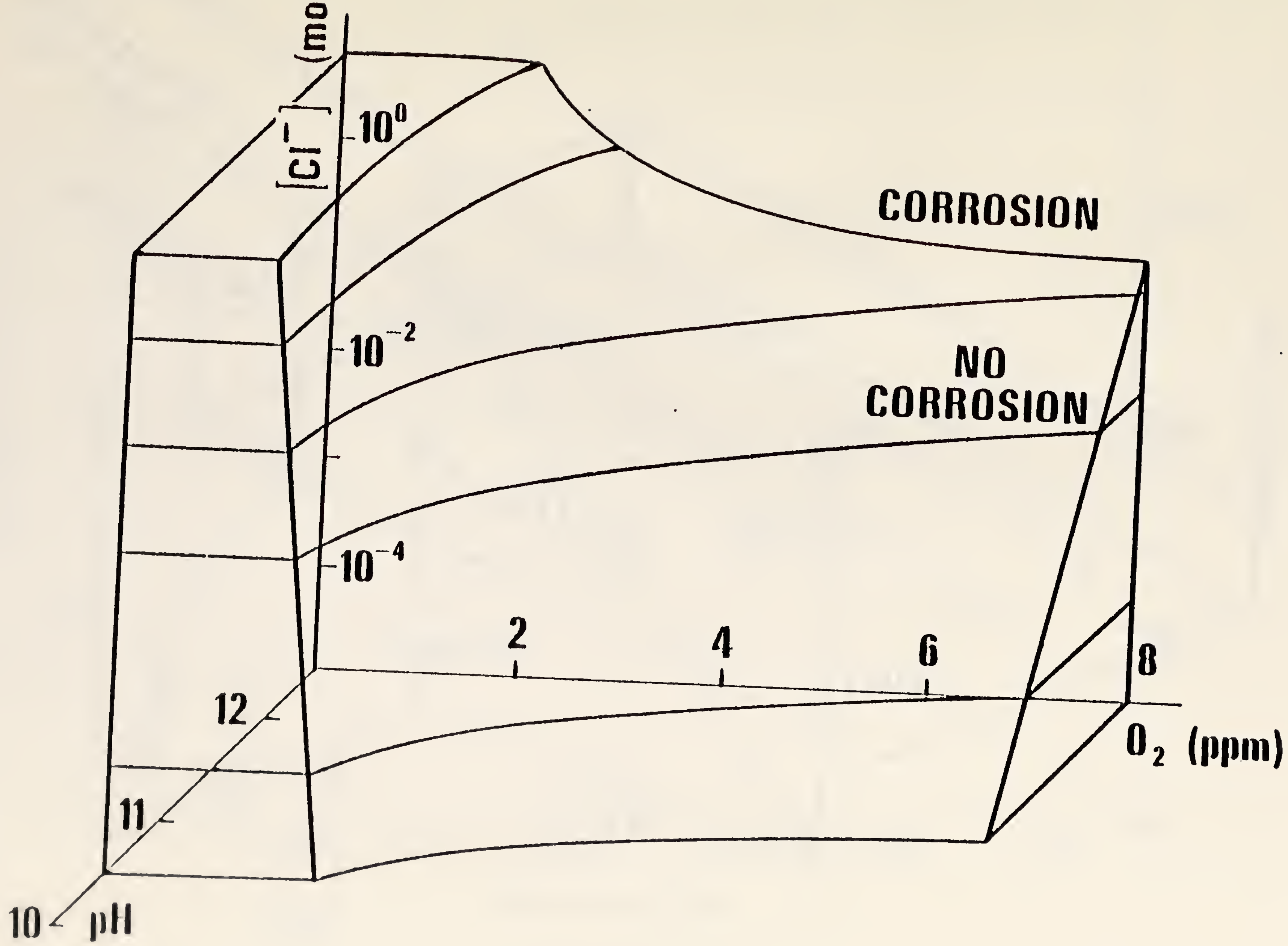

[Cl⁻] (mol/1)
10⁰
10⁻²
10⁻⁴
CORROSION
NO
CORROSION
2
4
6
8
O₂ (ppm)
10
11
12
pH

Figure 1

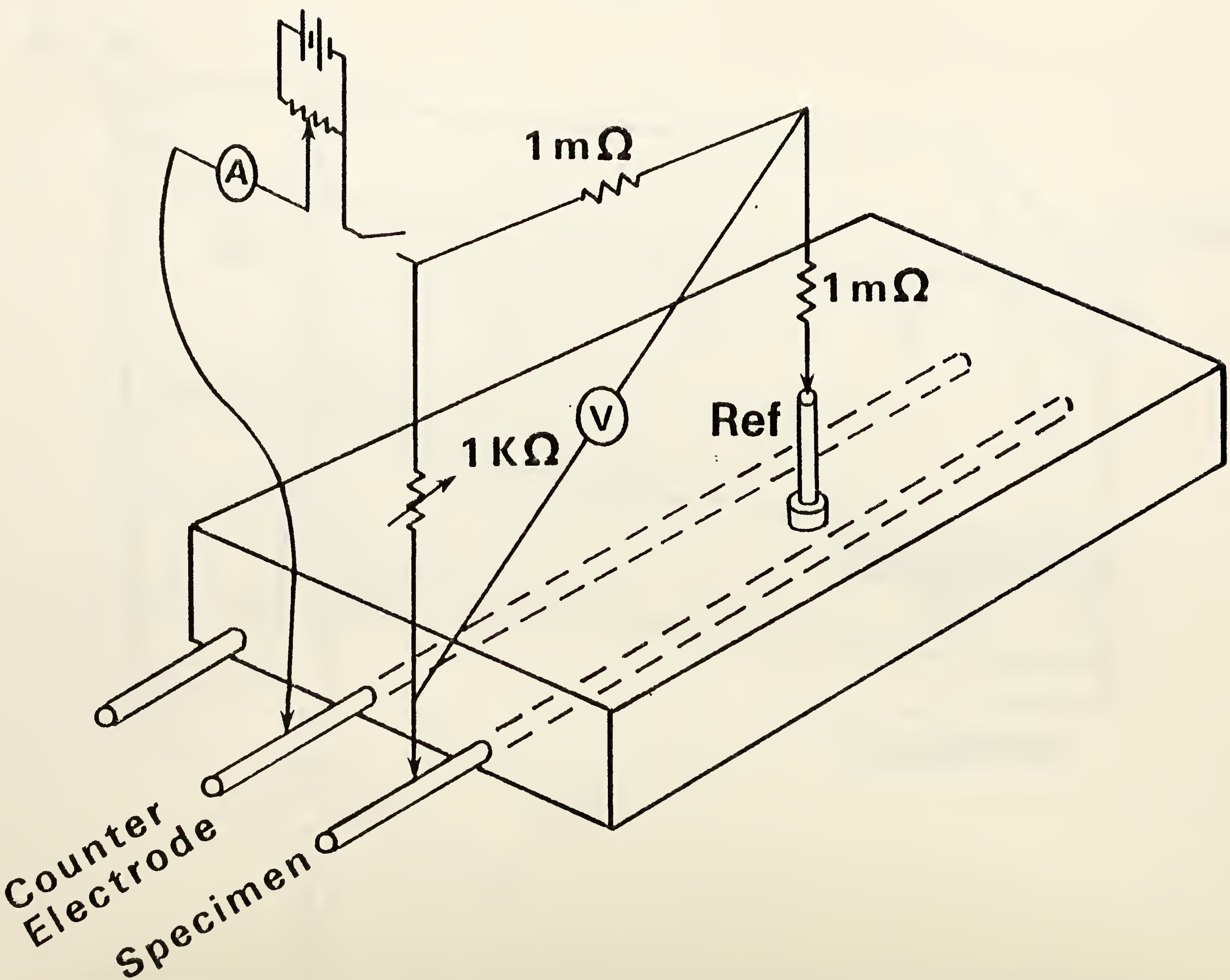


Figure 2

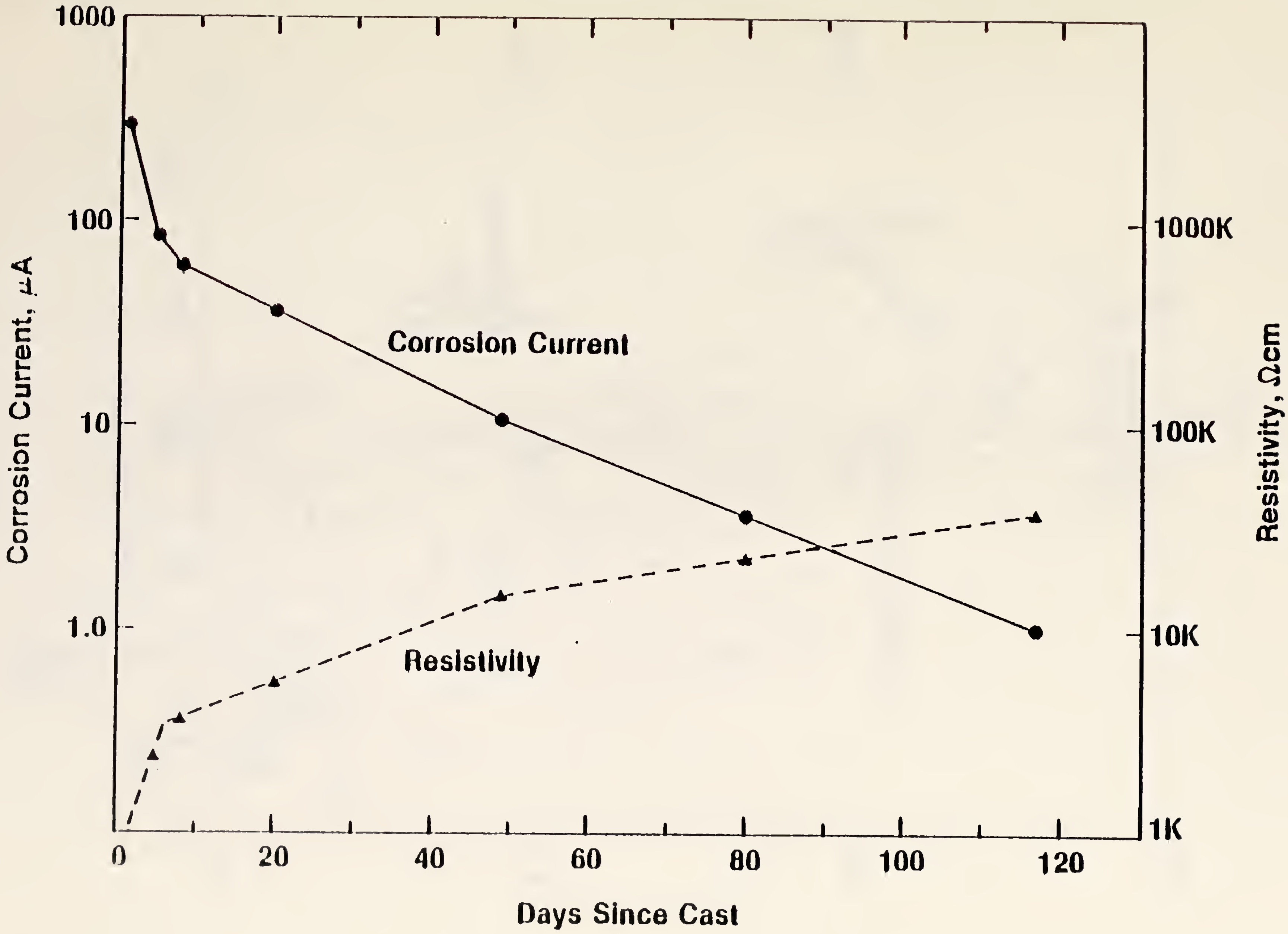
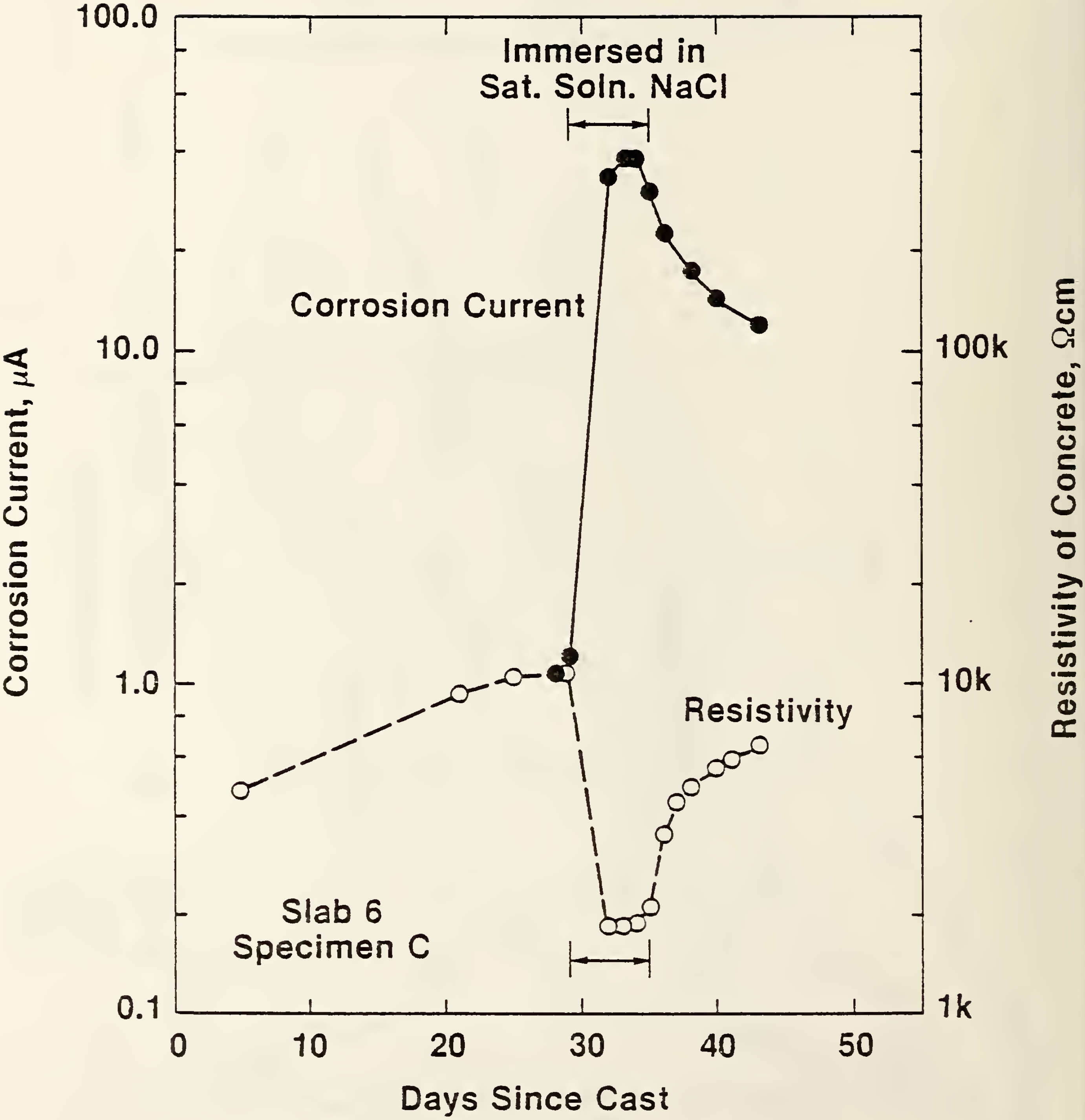

1000
100
10
1.0
Corrosion Current, μA
1000K
100K
10K
1K
Resistivity, Ωcm
Corrosion Current
Resistivity
0
20
40
60
80
100
120
Days Since Cast

Figure 3



100.0
10.0
1.0
0.1
100k
10k
1k
0
10
20
30
40
50
Immersed in
Sat. Soln. NaCl
Corrosion Current
Resistivity
Slab 6
Specimen C
Corrosion Current, μA
Resistivity of Concrete, Ωcm
Days Since Cast

Figure 4

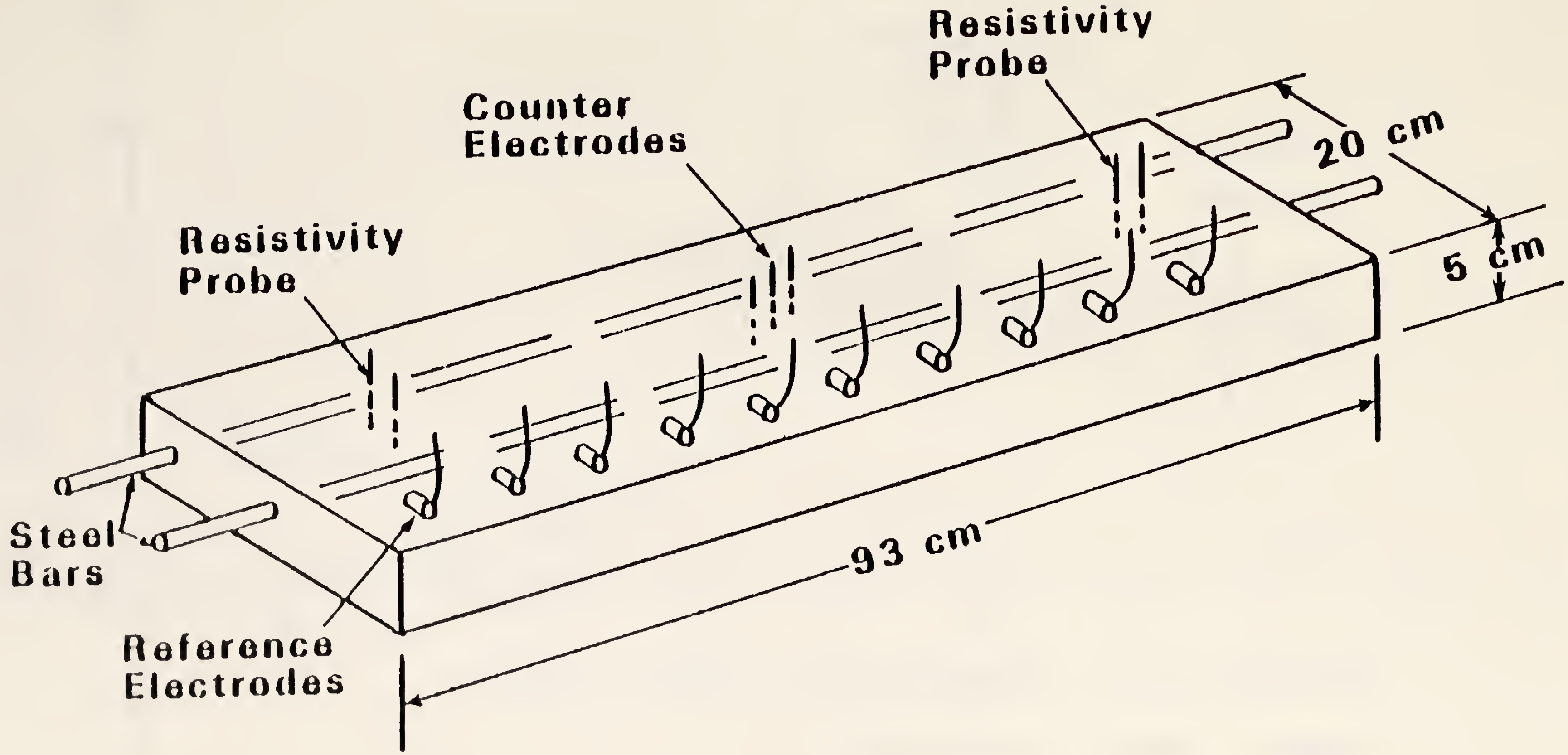

Resistivity Probe
Counter Electrodes
Resistivity Probe
20 cm
5 cm
Steel Bars
93 cm
Reference Electrodes

Figure 5

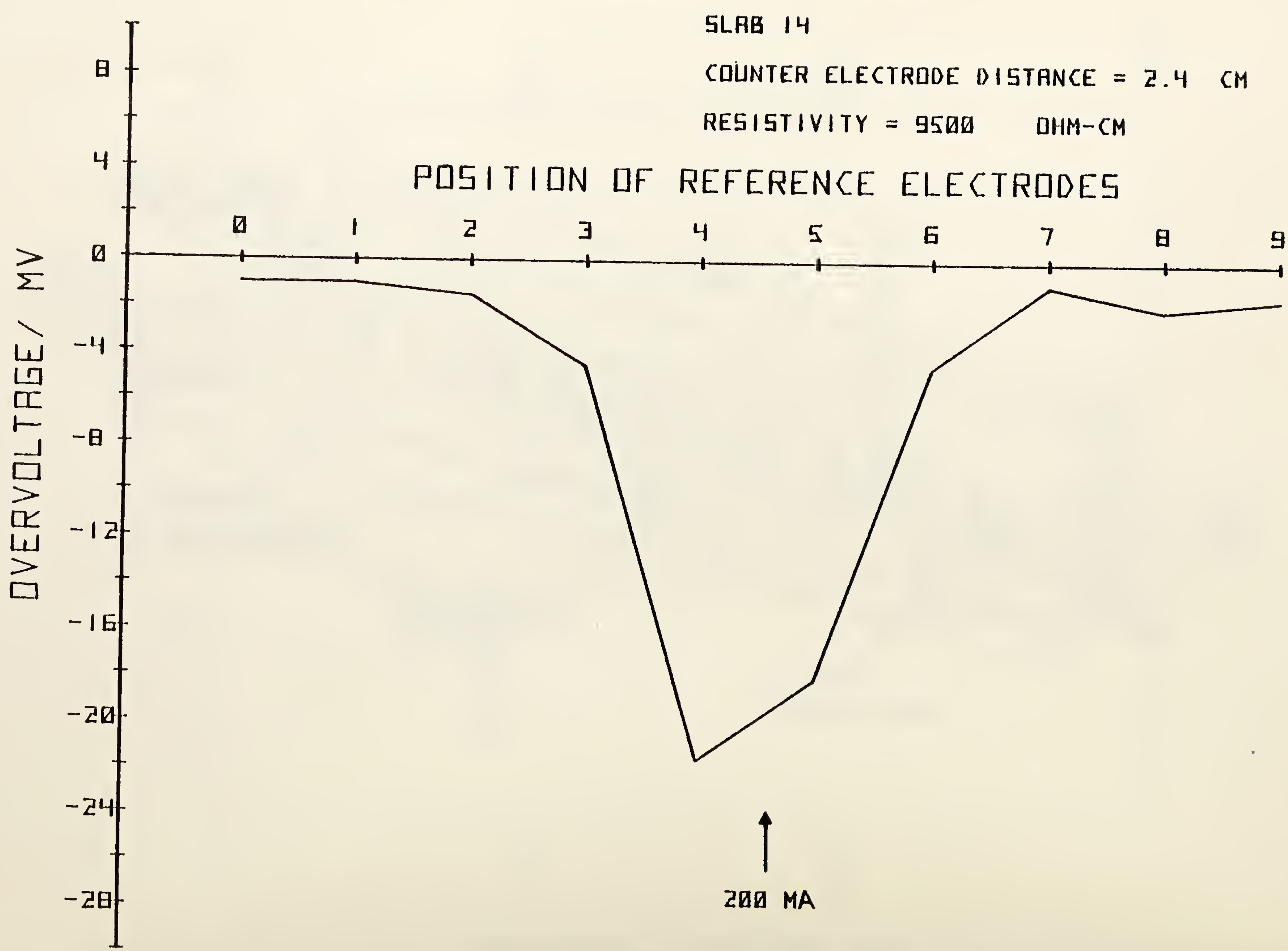

SLAB 14
COUNTER ELECTRODE DISTANCE = 2.4 CM
RESISTIVITY = 9500 OHM-CM
POSITION OF REFERENCE ELECTRODES
0
1
2
3
4
5
6
7
8
9
OVERVOLTAGE/ MV
8
4
0
-4
-8
-12
-16
-20
-24
-28
200 MA

Figure 6

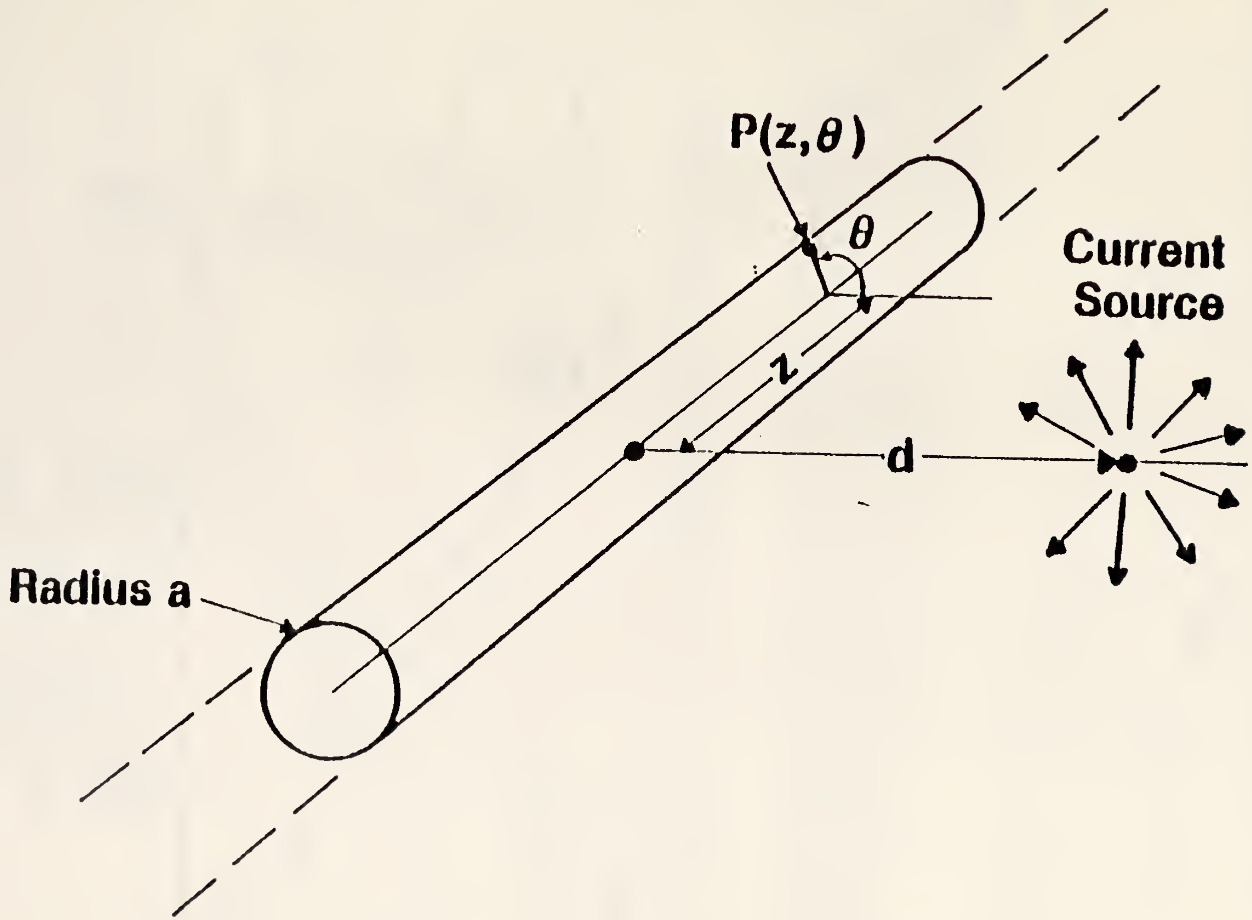
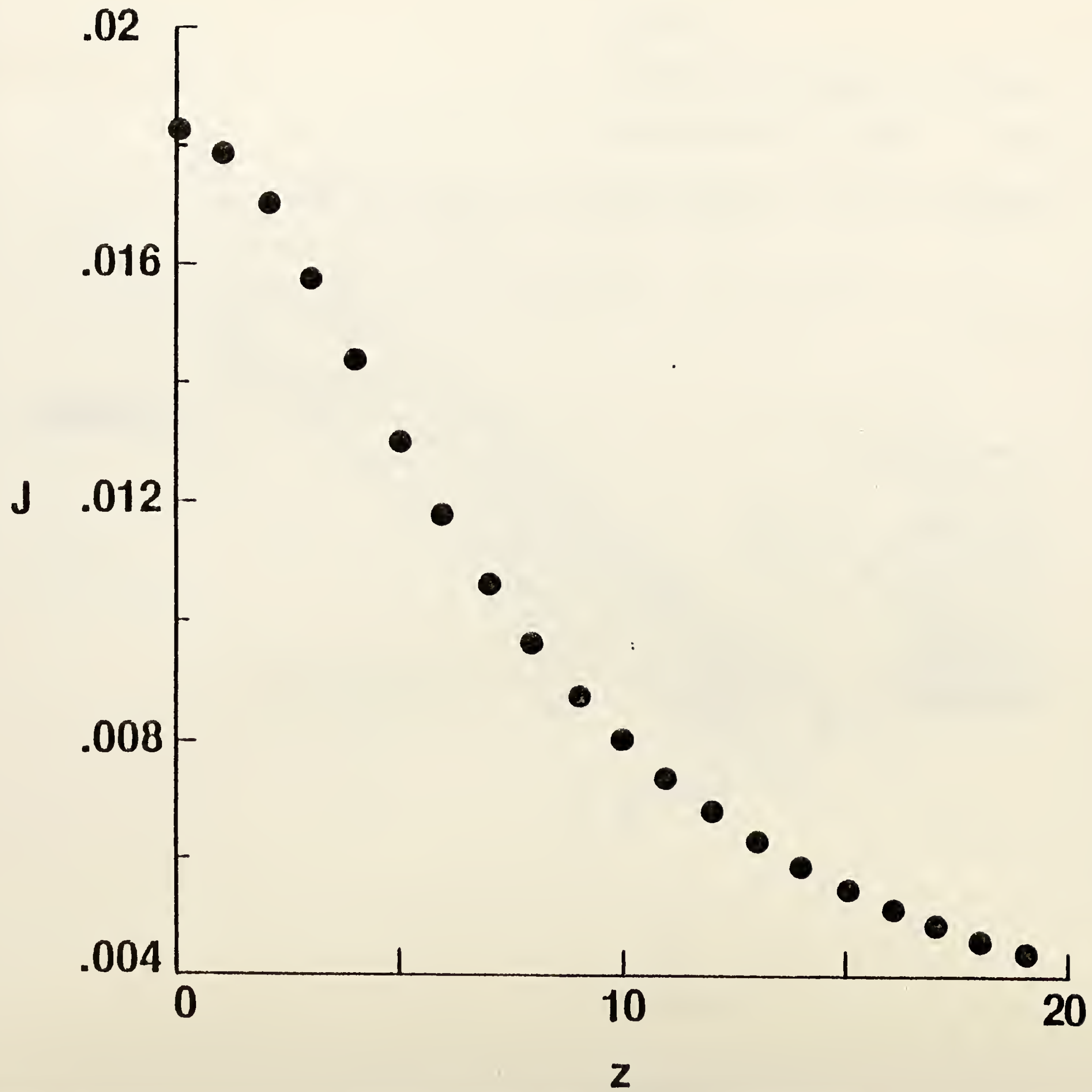


Figure 7



.02
.016
J
.012
.008
.004
0
10
20
z

PANEL WIRING

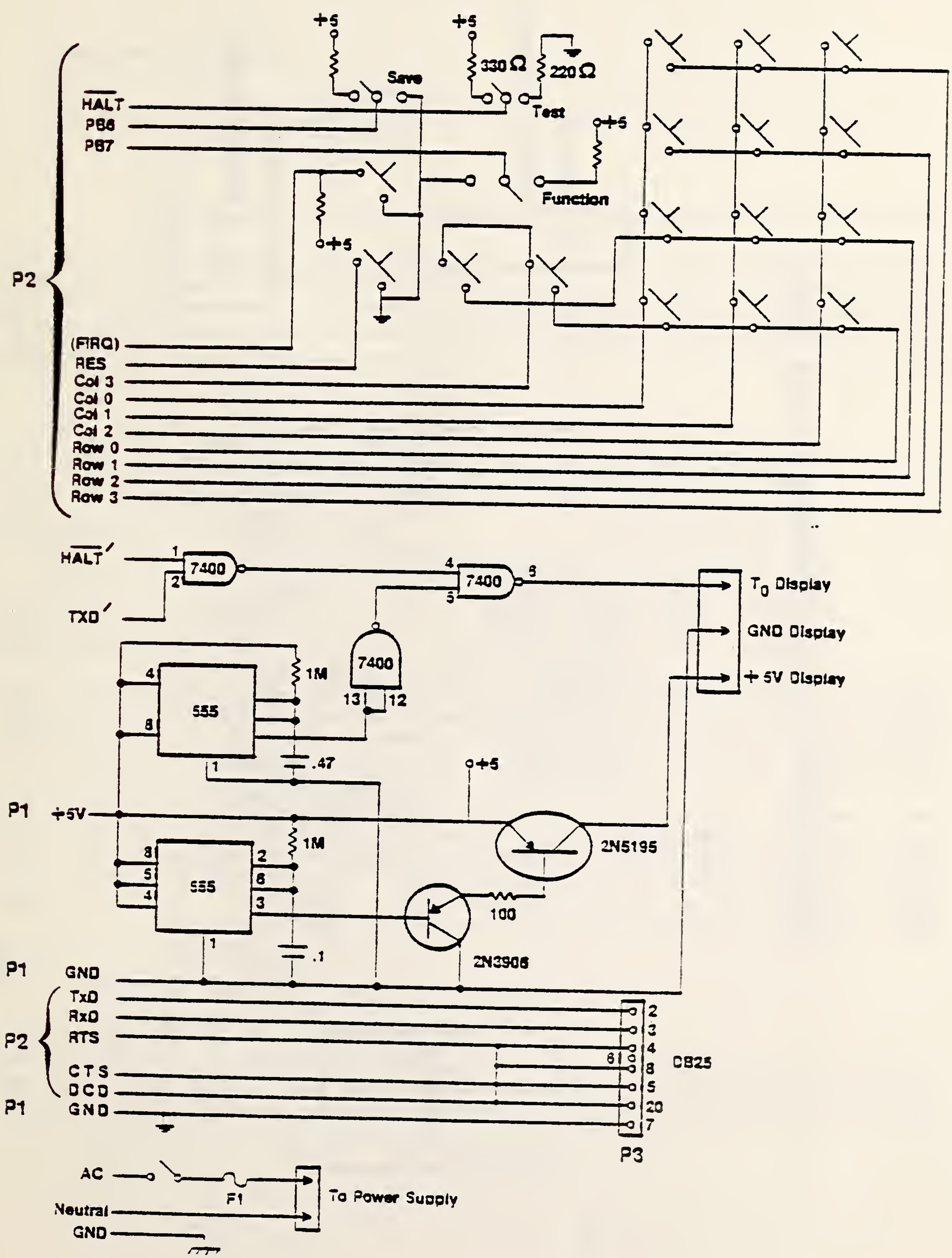


Figure 9

Modification to 1614 Module (10 Bit DAC)

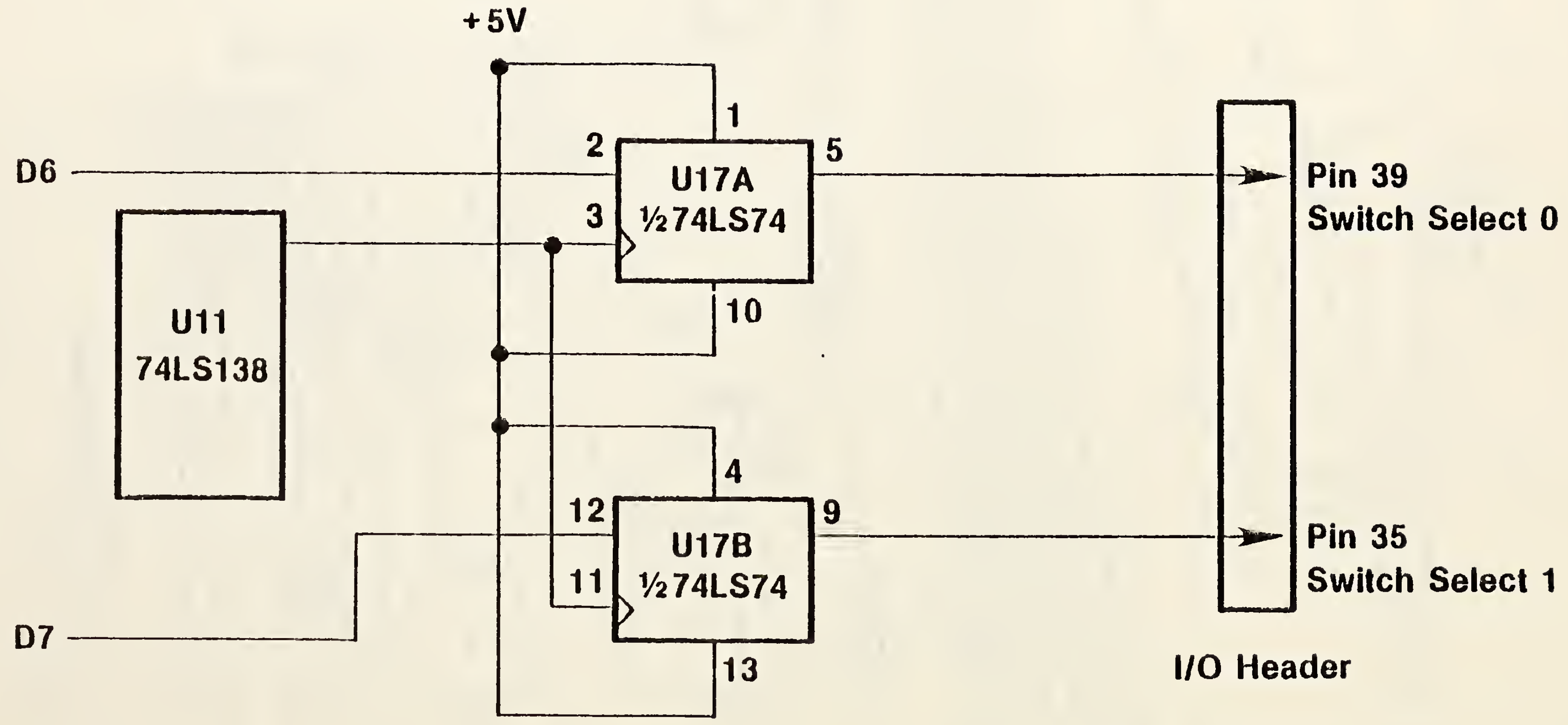


Figure 10

Keyboard Scanner Card

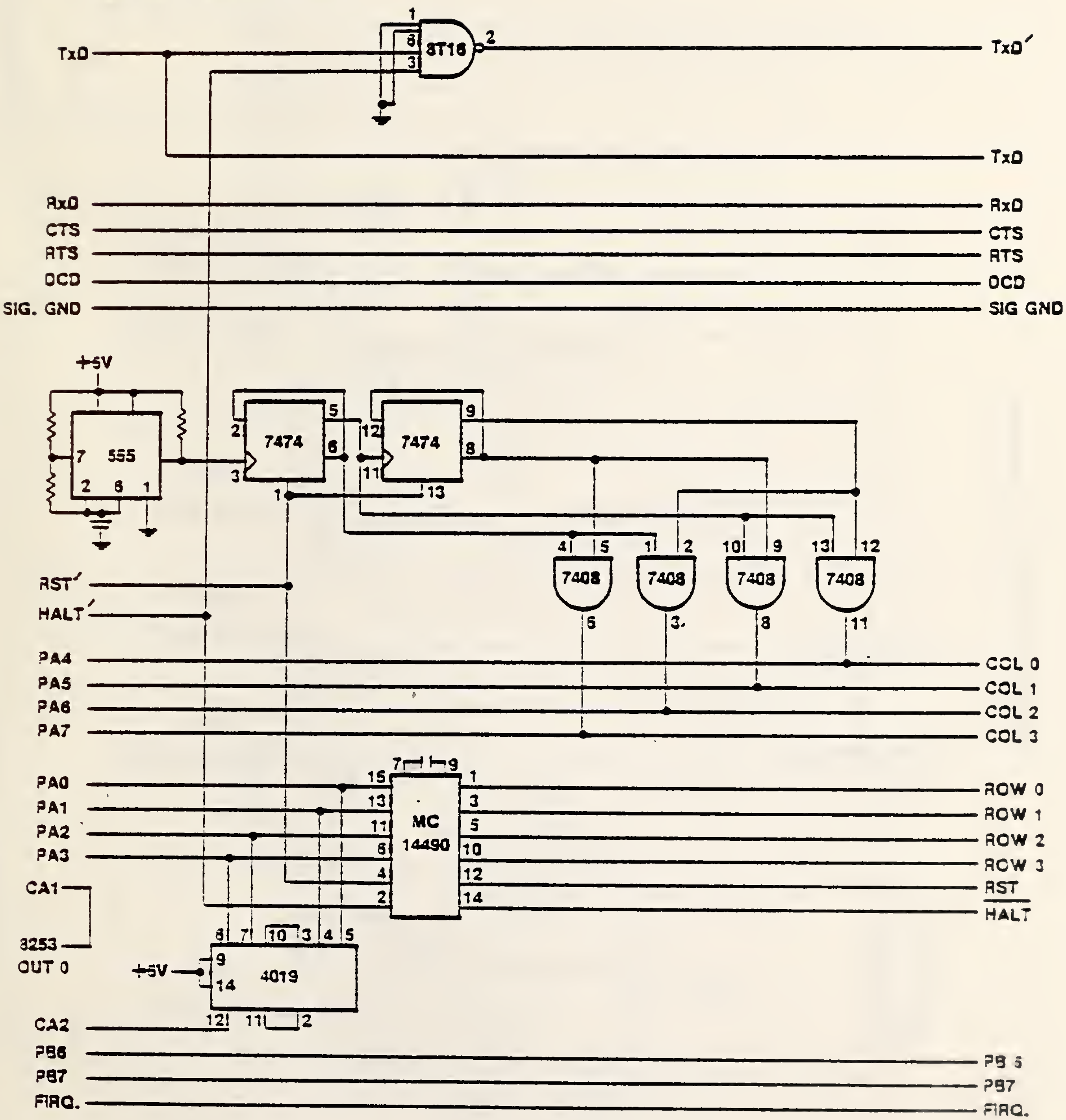
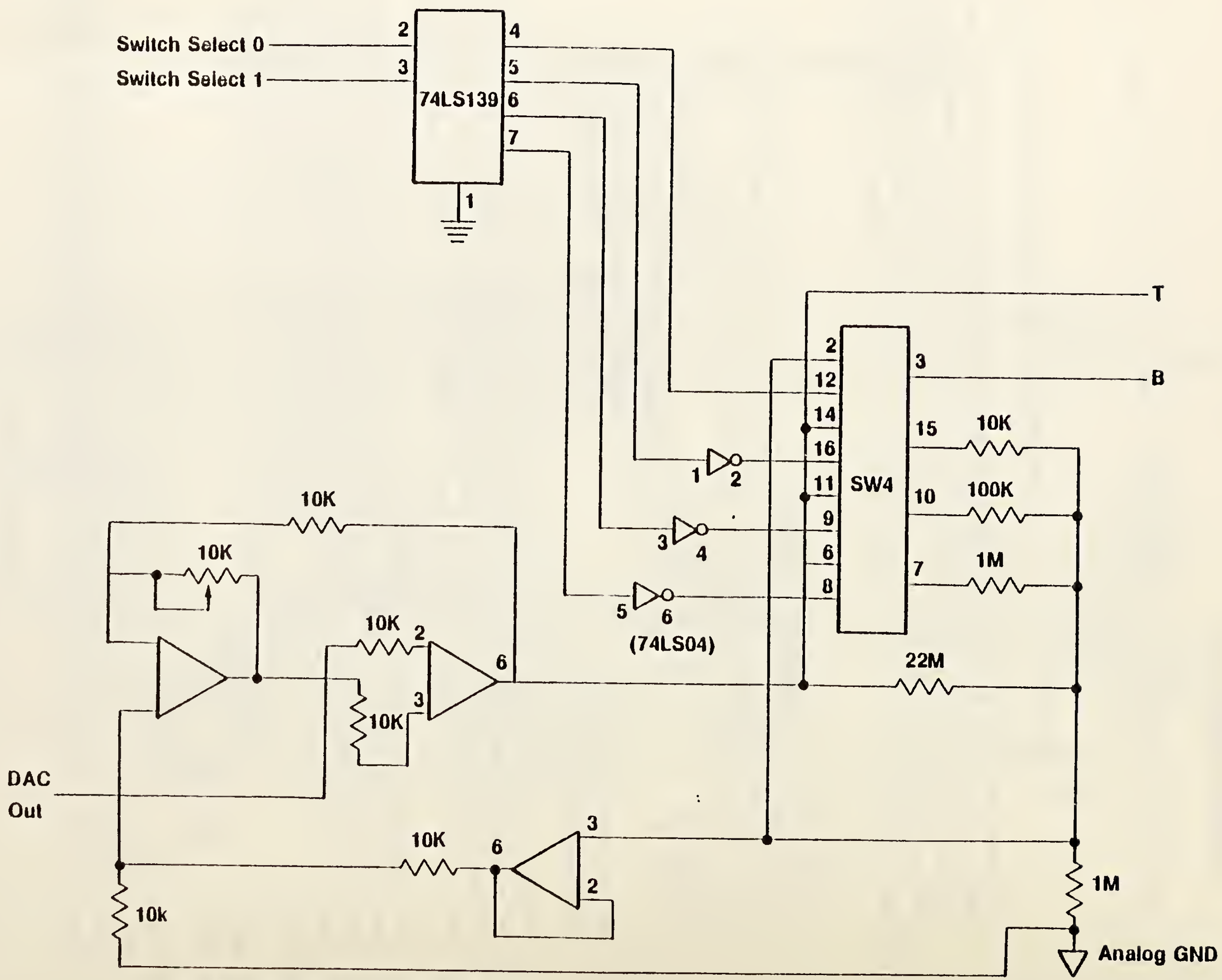


Figure 11

Voltage to Current Converter



Offset Trim and Bipolar Power Not Shown

Figure 12

Input Card

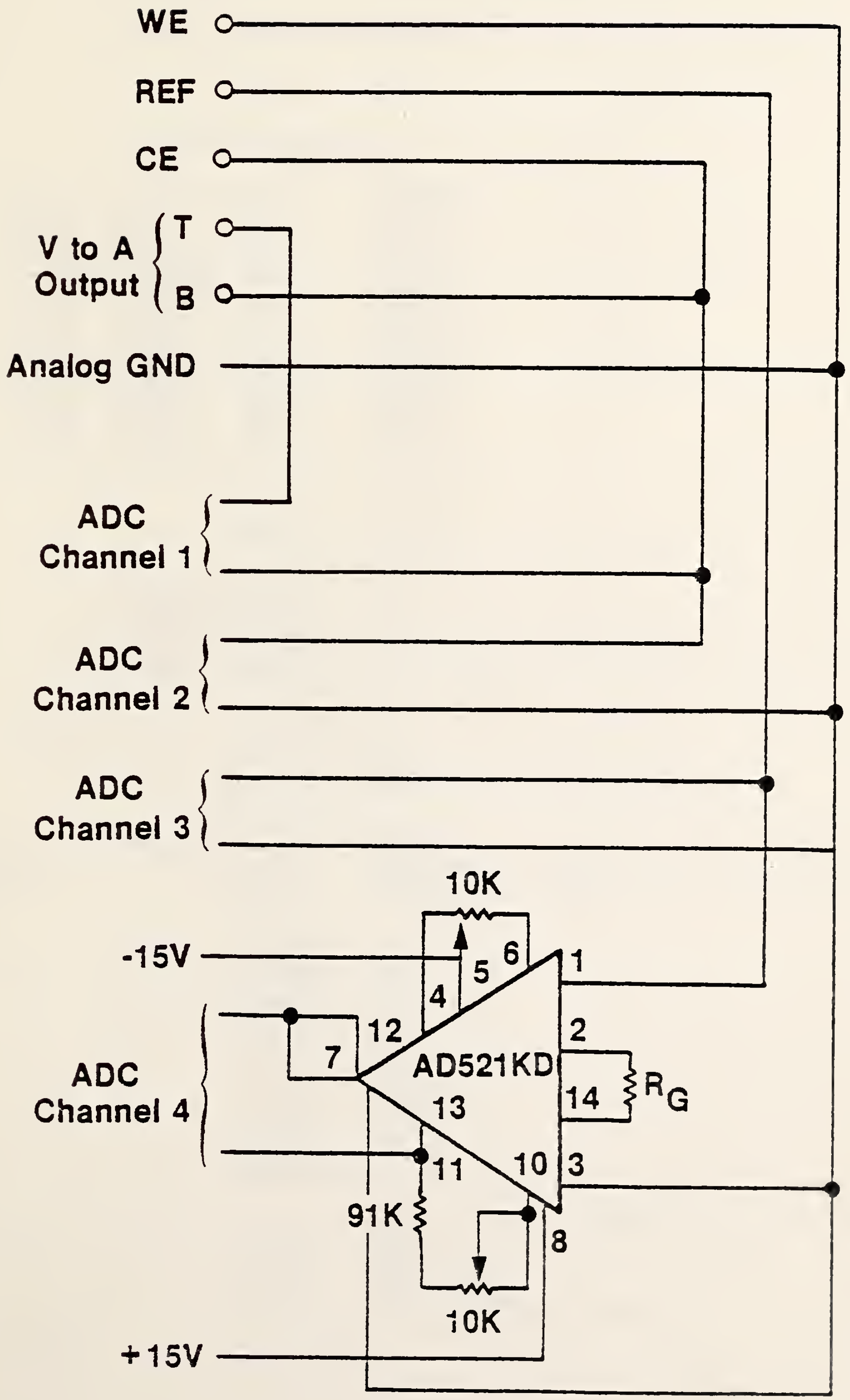


Figure 13

```
                     ************************************************

                     *                    BEGIN

                     * THIS ROUTINE ISSUES PROMPTS AND STORES CONSTANT
                     * DATA FOR THE PORTABLE INSTRUMENT

                     ************************************************

                             OPT     EXP

                     * SYSTEM EQUATES

               0800  BUFFER  EQU     $0800
               FECA  WFS     EQU     $FECA
               F9A3  GETDATA EQU     $F9A3
               0743  STARTD  EQU     $0743
               070B  DATBUF  EQU     $070B
               FB9C  PACKS   EQU     $FB9C
               FA83  CVDTB   EQU     $FA83
               0731  RUNFLG  EQU     $0731
               FB6A  GETR    EQU     $FB6A
               F194  MEASUR  EQU     $F194


0745                         ORG     $0745

0745                 RUNSET  RMB     4
0749                 TIMSET  RMB     4
074D                 AREAST  RMB     4
0751                 WORK    RMB     5

               0007          SETDP   $07

F000                         ORG     $F000

F000 0F   31         BEGIN   CLR     RUNFLG
F002 BE   0800               LDX     #BUFFER    GET START OF BUFFER
F005 9F   0B                 STX     DATBUF     SAVE IT

                     MSG     MACRO
                             LBSR    WFS
                             FCB     $0D,$0A
                             FCC     '&1'
                             FCB     $0
                             LDB     #$&2
                             LBSR    GETDATA
                             BEQ     &0
                             ENDM

F007                 ID      MSG     "ID        (15 CHAR.-MAX) = ","10"
F007 17   0EC0               LBSR    WFS
F00A 0D 0A                   FCB     $0D,$0A
F00C 49 44 20 28             FCC     'ID (15 CHAR.-MAX) = '
F010 31 35 20 43
F014 48 41 52 2E
F018 2D 4D 41 58
F01C 29 20 3D 20
```

```
F020 00                          FCB     $0
F021 C6    10                    LDB     #$10
F023 17    097D                  LBSR    GETDATA
F026 27    DF                    BEQ     ID
                                 ENDM
F028                    DATE     MSG     "DATE      (DDMMYR) = ",7
F028 17    0E9F                  LBSR    WFS
F02B 0D 0A                       FCB     $0D,$0A
F02D 44 41 54 45                 FCC     'DATE (DDMMYR) = '
F031 20 28 44 44
F035 4D 4D 59 52
F039 29 20 3D 20
F03D 00                          FCB     $0
F03E C6    07                    LDB     #$7
F040 17    0960                  LBSR    GETDATA
F043 27    E3                    BEQ     DATE
                                 ENDM
F045                    TIME     MSG     "TIME      TO NEXT MEASUREMENT (MMSS)= ",5
F045 17    0EB2                  LBSR    WFS
F048 0D 0A                       FCB     $0D,$0A
F04A 54 49 4D 45                 FCC     'TIME TO NEXT MEASUREMENT (MMSS)= '
F04E 20 54 4F 20
F052 4E 45 58 54
F056 20 4D 45 41
F05A 53 55 52 45
F05E 4D 45 4E 54
F062 20 28 4D 4D
F066 53 53 29 3D
F06A 20
F06B 00                          FCB     $0
F06C C6    05                    LDB     #$5
F06E 17    0932                  LBSR    GETDATA
F071 27    D2                    BEQ     TIME
                                 ENDM

F073 9E    0B                    LDX     DATBUF     POINT TO DATA
F075 34    20                    PSHS    Y          SAVE Y
F077 30    1B                    LEAX    -5,X       POINT TO MINUTES DATA
F079 17    00C0                  LBSR    CVTIM      GO CONVERT TO BINARY
F07C 86    3C                    LDA     #60        LOAD SEC/MIN
F07E 3D                          MUL                MULTIPLY
F07F DD    49                    STD     TIMSET
F081 9E    0B                    LDX     DATBUF     POINT TO DATA, AGAIN
F083 30    1D                    LEAX    -3,X       POINT TO SECONDS
F085 17    00B4                  LBSR    CVTIM      CONVERT TO BINARY
F088 D3    49                    ADDD    TIMSET
F08A DD    49                    STD     TIMSET
F08C 35    20                    PULS    Y

F08E                    AREA     MSG     "AREA      OF SAMPLE (0-65535 CM↑2)= ",6
F08E 17    0E39                  LBSR    WFS
F091 0D 0A                       FCB     $0D,$0A
F093 41 52 45 41                 FCC     'AREA OF SAMPLE (0-65535 CM↑2)= '
F097 20 4F 46 20
F09B 53 41 4D 50
F09F 4C 45 20 28
F0A3 30 2D 36 35
F0A7 35 33 35 20
F0AB 43 4D 5E 32
```

```
F0AF 29 3D 20
F0B2 00                          FCB    $0
F0B3 C6    06                    LDB    #$6
F0B5 17    0BEB                  LBSR   GETDATA
F0B8 27    D4                    BEQ    AREA
                                 ENDM
F0BA 9E    0B                    LDX    DATBUF     POINT TO INPUT DATA
F0BC 34    20                    PSHS   Y          SAVE Y
F0BE 8D    76                    BSR    INSAV
F0C0 DD    4D                    STD    AREAST     SAVE BINARY AREA
F0C2 35    20                    PULS   Y

F0C4                     RUNS    MSG    "RUNS      COUNT (100 OR LESS)= ",6
F0C4 17    0E03                  LBSR   WFS
F0C7 0D 0A                       FCB    $0D,$0A
F0C9 52 55 4E 53                 FCC    'RUNS COUNT (100 OR LESS)= '
F0CD 20 43 4F 55
F0D1 4E 54 20 28
F0D5 31 30 30 20
F0D9 4F 52 20 4C
F0DD 45 53 53 29
F0E1 3D 20
F0E3 00                          FCB    $0
F0E4 C6    06                    LDB    #$6
F0E6 17    0BBA                  LBSR   GETDATA
F0E9 27    D9                    BEQ    RUNS
                                 ENDM
F0EB 9E    0B                    LDX    DATBUF     POINT TO DATA
F0ED 34    20                    PSHS   Y          SAVE Y
F0EF 8D    45                    BSR    INSAV
F0F1 DD    45                    STD    RUNSET     SAVE BINARY NUMBER OF RUNS
F0F3 35    20                    PULS   Y

F0F5 8E    0B00          CRT000  LDX    #BUFFER    POINT TO START OF INPUT DATA
F0F8 A6    80            CHARN   LDA    X+         GET CHAR
F0FA 81    0D                    CMPA   #$0D       IS IT A CR?
F0FC 27    5C                    BEQ    CG         IF SO, CHANGE IT
F0FE 9C    0B            CNTCM   CPX    DATBUF     HAVE WE REACHED END?
F100 26    F6                    BNE    CHARN      IF NOT, GET ANOTHER CHAR.
F102 8D    5B                    BSR    DATSAV
F104 CC    F194                  LDD    #MEASUR    POINT TO MEASUREMENT ROUTINE
F107 DD    31                    STD    RUNFLG
F109 17    0DBE                  LBSR   WFS
F10C 0D 0A                       FCB    $0D,$0A
F10E 50 52 45 53                 FCC    'PRESS RUNWHEN READY FOR MEASUREMENT'
F112 53 20 52 55
F116 4E 20 57 48
F11A 45 4E 20 52
F11E 45 41 44 59
F122 20 46 4F 52
F126 20 4D 45 41
F12A 53 55 52 45
F12E 4D 45 4E 54
F132 00                          FCB    00
F133 16    0A34                  LBRA   GETR

F136 30    1A            INSAV   LEAX   -6,X
F138 17    0948                  LBSR   CVDTB      CONVERT TO BINARY
F13B 39                          RTS
```

```
F13C 108E 0751      CVTIM   LDY     #WORK
F140 C6   03                LDB     #$3
F142 86   30                LDA     #$30
F144 A7   A0        THIRTY  STA     Y+
F146 5A                     DECB
F147 26   FB                BNE     THIRTY
F149 C6   02                LDB     #$2
F14B A6   80        SIXTY   LDA     X+          GET DATA
F14D A7   A0                STA     Y+          SAVE IT
F14F 5A                     DECB
F150 26   F9                BNE     SIXTY
F152 1F   21                TFR     Y,X
F154 30   1B                LEAX    -5,X
F156 17   092A              LBSR    CVDTB
F159 39                     RTS

F15A 4F             CG      CLRA
F15B A7   1F                STA     -1,X        REPLACE CR BY ZERO
F15D 20   9F                BRA     CNTCM


F15F 34   20        DATSAV  PSHS    Y           SAVE Y
F161 109E 43                LDY     STARTD      GET CMOS MEMEORY START
F164 8E   0800              LDX     #BUFFER     POINT TO DATA
F167 DC   0B                LDD     DATBUF
F169 83   0800              SUBD    #BUFFER
F16C 47                     ASRA                DIVIDE BY 4
F16D 56                     RORB
F16E 47                     ASRA
F16F 56                     RORB
F170 17   0A29              LBSR    PACKS
F173 109F 43                STY     STARTD      SAVE NEXT CMOS DATA ADDRESS
F176 9E   0B                LDX     DATBUF      POINT TO DATA INPUT AREA
F178 35   A0                PULS    Y,PC        RESTORE Y,RETURN

                            END

0 ERROR(S) DETECTED

SYMBOL TABLE:

AREA    F08E   AREAST 074D   BEGIN  F000   BUFFER 0800   CG      F15A
CHARN   F0FB   CNTCM  F0FE   CRT000 F0F5   CVDTB  FA83   CVTIM   F13C
DATBUF  070B   DATE   F028   DATSAV F15F   GETDAT F9A3   GETR    FB6A
ID      F007   INSAV  F136   MEASUR F194   PACKS  FB9C   RUNFLG  0731
RUNS    F0C4   RUNSET 0745   SIXTY  F14B   STARTD 0743   THIRTY  F144
TIME    F045   TIMSET 0749   WFS    FECA   WORK   0751
```

```
                 ******************************************************

                 *                       MEASUR

                 * THIS ROUTINE HANDLES THE MEASUREMENT SEQUENCE IN
                 * THE PORTABLE INSTRUMENT.  THE MEASURED DATA IS
                 * STORED IN THE CMOS MEMORY AND THE ADDRESS UPDATED
                 * FOR LATER RETRIEVAL IF DESIRED.

                 ******************************************************

                 * SYSTEM EQUATES

           0700  INDEX    EQU    $0700
           0702  ADCVAL   EQU    $0702
           0704  CHNUM    EQU    $0704
           0705  SWNUM    EQU    $0705
           0706  DACVAL   EQU    $0706
           070B  DATBUF   EQU    $070B
           0733  RUNCNT   EQU    $0733
           0743  STARTD   EQU    $0743
           0745  RUNSET   EQU    $0745
           0749  TIMSET   EQU    $0749
           B004  SAVST1   EQU    $B004
           DE50  DDAC     EQU    $DE50

           0007           SETDP  $07

           FB2F  CGTIME   EQU    $FB2F
           FB39  TIMCHK   EQU    $FB39
           FB52  CHSELT   EQU    $FB52
           FB67  SMSELT   EQU    $FB67
           FB79  CONVRT   EQU    $FB79
           FB9D  INITDA   EQU    $FB9D
           FBB0  SWNSET   EQU    $FBB0
           FBBF  SWNZER   EQU    $FBBF
           FBCB  LDDAC    EQU    $FBCB
           FECA  WFS      EQU    $FECA

0751                      ORG    $0751

0751             WORK     RMB    5
0756             REVFLG   RMB    1
0757             DACINI   RMB    2
0759             OPNCRK   RMB    2
075B             DACADJ   RMB    2

F194                      ORG    $F194

F194 17    0D33  MEASUR   LBSR   WFS        OUTPUT MESSAGE
F197 0D 0A                FCB    $0D,$0A
F199 57 4F 52 4B          FCC    'WORKING'
F19D 49 4E 47
F1A0 00                   FCB    $00
F1A1 0F    33             CLR    RUNCNT
F1A3 CC    01FF           LDD    #$01FF     CORRESPONDS TO ZERO OUT
F1A6 DD    57             STD    DACINI     SAVE INITIAL VALUE
F1A8 CC    0033           LDD    #$0033     CORRESPONDS TO 1/2 MICROAMP
```

```
F1AB DD    5B                  STD    DACADJ    SAVE ADJUSTMENT VALUE
F1AD 9E    43                  LDX    STARTD    GET START OF CMOS MEMORY AREA
F1AF BF    B004                STX    SAVST1    SAVE START OF MEMORY AREA

F1B2 CC    000A       SELCH    LDD    #$000A
F1B5 ED    84                  STD    X
F1B7 17    0675                LBSR   CGTIME    SET TIMER FOR 1 MSEC TIMEOUTS
F1BA 86    01                  LDA    #$01      SET A TO D FOR CE TO WE, I-0FF
F1BC 97    04                  STA    CHNUM
F1BE 17    0691                LBSR   CHSELT    SELECT CHANNEL
F1C1 17    06A3                LBSR   SMSELT    START SAMPLING MODE
F1C4 17    0672                LBSR   TIMCHK    WAIT FOR TIMEOUT
F1C7 17    06AF                LBSR   CONVRT    MEASURE
F1CA ED    81                  STD    X++       SAVE VOLTAGE, I-OFF ZERO
F1CC 0C    04                  INC    CHNUM
F1CE 17    0681                LBSR   CHSELT
F1D1 17    0693                LBSR   SMSELT
F1D4 17    06A2                LBSR   CONVRT    GET OPEN CIRCUIT POTENTIAL

F1D7 86    00                  LDA    #$00
F1D9 97    04                  STA    CHNUM
F1DB 17    0674                LBSR   CHSELT    SET A TO D TO CURRENT CHANNEL
F1DE 17    0686                LBSR   SMSELT    SET TO SAMPLING MODE
F1E1 17    041C                LBSR   SETIO     GET ZERO CURRENT VALUES

F1E4 DC    5B         CURSET   LDD    DACADJ    GET ADJUSTMENT VALUE
F1E6 44                        LSRA             HALVE IT
F1E7 56                        RORB
F1E8 D3    57                  ADDD   DACINI    ADD INITIAL VALUE
                      *        ****************************
                      *        ALTERNATE IS SUBTRACTION FOR
                      *           OTHER POLARITY
                      *        ****************************
F1EA DD    06                  STD    DACVAL

F1EC 0C    33         NEXT     INC    RUNCNT
F1EE 9F    51                  STX    WORK      SAVE INDEX
F1F0 17    0646                LBSR   TIMCHK    WAIT FOR TIMOUT
F1F3 17    06D5                LBSR   LDDAC TURNON SET CURRENT
F1F6 17    06B7                LBSR   SWNSET    SET SWITCH
F1F9 17    0630       NEXT T-  LBSR   TIMCHK    WAIT FOR ANOTHER TIMEOUT
F1FC 17    067A                LBSR   CONVRT    GET POTENTIAL DROP OVER RESISTOR
F1FF ED    81                  STD    X++       SAVE R DROP
F201 0C    04                  INC    CHNUM
F203 17    064C                LBSR   CHSELT
F206 17    065E                LBSR   SMSELT    SET TO READ CE TO WE POTENTIAL
F209 17    066D                LBSR   CONVRT    GET IT
F20C ED    81                  STD    X++       SAVE CE TO WE, I-0N
F20E 0C    04                  INC    CHNUM
F210 17    063F                LBSR   CHSELT
F213 17    0651                LBSR   SMSELT
F216 17    0620                LBSR   TIMCHK
F219 17    0447                LBSR   SWZERO    TURN OFF CURRENT
F21C 17    065A                LBSR   CONVRT    GET WE-REF, I-0FF
F21F ED    81                  STD    X++       SAVE IT

F221 0A    04                  DEC    CHNUM     POINT TO CE TO WE
F223 17    062C                LBSR   CHSELT
F226 17    0629                LBSR   CHSELT
```

```
F229 17     064D                LBSR    CONVRT     GET WE TO CE CURRENT OFF
F22C ED     81                  STD     X++        SAVE IT
F22E BF     8004                STX     SAVST1     SAVE NEXT DATA BLOCK ADDRESS

F231 0F     04                  CLR     CHNUM
F233 17     061C                LBSR    CHSELT
F236 17     062E                LBSR    SMSELT     SELECT CURRENT READ,  AGAIN
F239 0D     56                  TST     REVFLG     HAVE WE FOUND RANGE?
F23B 2B     06                  BMI     RDONE      YES, THEN CALCULATE PARAMETERS
F23D 8D     3B                  BSR     CALCI      DO CHECK FOR APPROPRIATE RANGES
F23F 9E     51                  LDX     WORK       RESET INDEX
F241 20     A9                  BRA     NEXT
                                [illegible] TURN ON
F243 DC     45          RDONE   LDD     RUNSET     GET # OF RUNS TO BE PERFORMED
F245 1027   008D                LBEQ    CALCII     IF ZERO GO CALCULATE
F249 D0     33                  SUBB    RUNCNT
F24B 102F   0087                LBLE    CALCII     IF DONE, DO FINAL CALCULATIONS
F24F 0F     4B                  CLR     TIMSET+2
F251 0F     4C                  CLR     TIMSET+3
F253 CC     2580                LDD     #$2580
F256 ED     84                  STD     X          SET TIMEOUT FOR ONE SEC
F258 17     05D4                LBSR    CGTIME
F25B 17     05DB        TIMEOT  LBSR    TIMCHK
F25E DC     4B                  LDD     TIMSET+2
F260 C3     0001                ADDD    #$0001
F263 DD     4B                  STD     TIMSET+2
F265 1093   49                  CMPD    TIMSET     ARE WE DONE?
F268 27     02                  BEQ     TIMUP
F26A 20     EF                  BRA     TIMEOT
F26C CC     000A        TIMUP   LDD     #$000A     SET FOR 1 MILLISEC TIMEOUT
F26F ED     84                  STD     X
F271 17     05BB                LBSR    CGTIME

F274 BE     8004        MORE    LDX     SAVST1
F277 16     FF72                LBRA    NEXT

F27A                            ORG     $F27A

F27A                    CALCI   RMB     23         RESERVE AREA FOR NEXT PROGRAM
F291                    TWOS    RMB     69
F2D6                    CALCII  RMB     2          ADDRESS OF CALCULATION PROGRAM

F600                            ORG     $F600

F600 86     03          SETIO   LDA     #$03
F602 97     05                  STA     SWNUM      POINT TO 5 MICROAMPERE FULL SCALE
F604 DC     57                  LDD     DACINI     GET ZERO CURRENT EQUIVALENT
F606 DD     06                  STD     DACVAL     SETUP TO OUTPUT IT

F608 DC     02          OPNSAV  LDD     ADCVAL     GET OPEN CIRCUIT VOLTAGE
F60A ED     81                  STD     X++        SAVE IT IN CMOS MEMORY
F60C 17     FC82                LBSR    TWOS
F60F DD     59                  STD     OPNCRK
F611 17     0225                LBSR    TIMCHK
F614 17     02B4                LBSR    LDDAC      SET FOR ZERO CURRENT
F617 17     0296                LBSR    SWNSET     TURN ON ZERO CURRENT
F61A 17     021C                LBSR    TIMCHK
F61D 0C     04                  INC     CHNUM      POINT TO CE TO WE VOLTAGE
F61F 17     0230                LBSR    CHSELT
```

```
F622 17   0242     -6 ... LBSR   SMSELT
F625 17   0251            LBSR   CONVRT    GET APPLIED VOLTAGE-I ON
F628 DC   02              LDD    ADCVAL    RETRIEVE CE TO WE POTENTIAL
F62A ED   81              STD    X++       SAVE IT
F62C 86   00              LDA    #$00
F62E 97   04              STA    CHNUM     POINT TO CURRENT CHANNEL AGAIN
F630 17   021F            LBSR   CHSELT
F633 17   0231            LBSR   SMSELT
F636 17   0200            LBSR   TIMCHK
F639 17   023D            LBSR   CONVRT    GET DROP ACROSS R
F63C 17   0228            LBSR   SMSELT
F63F 17   027D            LBSR   SWNZER    TURN OFF ZERO CURRENT
F642 DC   02              LDD    ADCVAL    RETRIEVE DROP ACROSS R
F644 ED   81              STD    X++       SAVE IT
F646 39                   RTS

F647 DD   06     SETIO1   STD    DACVAL    SAVE ZERO CURRENT VALUE
F649 17   027F            LBSR   LDDAC     SET CURRENT TO NEW ZERO
F64C 17   0261            LBSR   SWNSET    TURN ON CURRENT
F64F 17   01E7            LBSR   TIMCHK
F652 17   0224            LBSR   CONVRT    GET NEW CURRENT ZERO
F655 17   0267            LBSR   SWNZER    TURN OFF SWITCH
F658 DC   02              LDD    ADCVAL    RETRIEVE VALUE
F65A ED   14              STD    -12,X     SAVE IT IN ZERO LOCATION
F65C 17   020B            LBSR   SMSELT
F65F 39                   RTS

F660 17   023A   SWBEGN   LBSR   INITDA    INITIALIZE DAC
F663 9F   00     SWZERO   STX    INDEX     SAVE INDEX
F665 8E   DE50            LDX    #DDAC     SET OUTPUT TO ZERO ROUTINE
F668 CC   8000            LDD    #$8000
F66B ED   84              STD    X
F66D E7   05              STB    5,X       TURN OFF SWITCH
F66F 9E   00              LDX    INDEX     RESTORE INDEX
F671 39                   RTS

F672             END

0 ERROR(S) DETECTED

SYMBOL TABLE:

ADCVAL 0702   CALCI  F27A   CALCII F2D6   CGTIME F82F   CHNUM  0704
CHSELT F852   CONVRT F879   CURSET F1E4   DACADJ 075B   DACINI 0757
DACVAL 0706   DATBUF 0708   DDAC   DE50   END    F672   INDEX  0700
INITDA F89D   LDDAC  F8C8   MEASUR F194   MORE   F274   NEXT   F1EC
OPNCRK 0759   OPNSAV F608   RDONE  F243   REVFLG 0756   RUNCNT 0733
RUNSET 0745   SAVST1 8004   SELCH  F1B2   SETIO  F600   SETIO1 F647
SMSELT F867   STARTD 0743   SWBEGN F660   SWNSET F8B0   SWNUM  0705
SWNZER F8BF   SWZERO F663   TIMCHK F839   TIMEOT F25B   TIMSET 0749
TIMUP  F26C   TWOS   F291   WFS    FECA   WORK   0751
```

```
                    ***************************************************

                    *                  CALCI

                    * THIS ROUTINE DOES CALCULATIONS FOR THE ADJUST-
                    * MENT OF THE CURRENT TO RESULT IN A FIXED REF TO
                    * WORKING ELECTRODE POTENTIAL


                    ****************************************************

                    * SYSTEM EQUATES

             0759   OPNCRK  EQU    $0759
             0733   RUNCNT  EQU    $0733
             075B   DACADJ  EQU    $075B
             0706   DACVAL  EQU    $0706
             0705   SWNUM   EQU    $0705
             0756   REVFLG  EQU    $0756
             0757   DACINI  EQU    $0757
             FECA   WFS     EQU    $FECA
             0731   RUNFLG  EQU    $0731
             FB68   GETR    EQU    $FB68
             FC00   STARTCL EQU    $FC00     COLD START ON ERROR
             F647   SETIO1  EQU    $F647

F27A                        ORG    $F27A

             0007           SETDP  $07

F27A 0A   33        CALCI   DEC    RUNCNT    REDUCE RUN COUNT TO REFLECT
                    *                        DATA NOT YET FOUND
F27C EC   1E                LDD    -2,X      GET REF TO WE POTENTIAL
F27E A3   12                SUBD   -14,X     GET DIFFERENCE FROM OPEN CIRCUIT
F280 2B   19                BMI    POLAR
F282 84   0F        TEST1   ANDA   #$0F      CLEAR UPPER 4 BITS
F284 1083 000A              CMPD   #$000A    IS THE DIFFERENCE 10 MV?
F288 2D   1B                BLT    ADJUST
F28A 96   56                LDA    REVFLG
F28C 8A   F0                ORA    #$F0      SET FLAG TO INDICATE CURRENT FOUND

F28E 97   56                STA    REVFLG
F290 39             RETURN  RTS

F291 85   08        TWOS    BITA   #$08
F293 27   03                BEQ    PLUS
F295 8A   F8                ORA    #$F8      BIT 3--SET,SIGN EXTEND,ALL ONES
F297 39                     RTS
F298 84   07        PLUS    ANDA   #$07      IF BIT 3 IS ZERO
F29A 39                     RTS              SIGN EXTEND ALL ZEROS, AND RETURN

F29B 8D   02        POLAR   BSR    NEGD      GET TWO'S COMPLEMENT OF D
F29D 20   E3                BRA    TEST1

F29F 43             NEGD    COMA
F2A0 53                     COMB
F2A1 C3   0001              ADDD   #$01
F2A4 39                     RTS
```

```
F2A5 DC   06        ADJUST  LDD    DACVAL      GET OLD CURRENT SET
F2A7 D3   5B                ADDD   DACADJ
F2A9 1083 03FF              CMPD   #$3FF       HAVE WE REACHED FULL SCALE?
F2AD 2C   03                BGE    RANGE
F2AF DD   06        CHANGE  STD    DACVAL
F2B1 39                     RTS

F2B2 DC   57        RANGE   LDD    DACINI      GET INITIAL VALUE
F2B4 0A   05                DEC    SWNUM
F2B6 27   03                BEQ    ERROR3
F2B8 16   03BC              LBRA   SETIO1

F2BB 17   0C0C      ERROR3  LBSR   WFS
F2BE 0D 0A                  FCB    $0D,$0A
F2C0 4F 55 54 20            FCC    'OUT OF RANGE'
F2C4 4F 46 20 52
F2C8 41 4E 47 45
F2CC 00                     FCB    $00
F2CD CC   FC00              LDD    #STARTCL
F2D0 DD   31                STD    RUNFLG
F2D2 17   0893              LBSR   GETR

F2D6                        ORG    $F2D6

F2D6                CALCII  RMB    2           ADDRESS OF NEXT ROUTINE

                            END

0 ERROR(S) DETECTED

SYMBOL TABLE:

ADJUST F2A5   CALCI  F27A   CALCII F2D6   CHANGE F2AF   DACADJ 075B
DACINI 0757   DACVAL 0706   ERROR3 F2BB   GETR   FB6B   NEGD   F29F
OPNCRK 0759   PLUS   F29B   POLAR  F29B   RANGE  F2B2   RETURN F290
REVFLG 0756   RUNCNT 0733   RUNFLG 0731   SETIO1 F647   STARTC FC00
SWNUM  0705   TEST1  F282   TWOS   F291   WFS    FECA
```

```
*****************************************************

*                    CALCII

* THIS ROUTINE DOES FINAL CALCULATIONS PRIOR TO THE
* OUTPUT OF DATA. RESULTS ARE CALCULATED AND FORMAT
* DETERMINED.
* ALGORITHMS ARE:  R'=(CE TO WE)/V(ACROSS RESISTOR)
*                          TIMES R

* OHMS X CM↑2 = KV(REF TO WE)-V(OPEN CIRCUIT)/I XAREA

*                 K IS CONVERSION FACTOR TO VOLTS/AMPS


***********************************************************


* SYSTEM EQUATES

        0700   INDEX    EQU    $0700
        0705   SWNUM    EQU    $0705
        070B   DATBUF   EQU    $070B
        074D   AREAST   EQU    $074D
        0751   WORK     EQU    $0751
        0759   OPNCRK   EQU    $0759
        0743   STARTD   EQU    $0743
        B002   SAVST0   EQU    $B002
        B004   SAVST1   EQU    $B004

        F29F   NEGD     EQU    $F29F
        F7C0   MUL16    EQU    $F7C0
        FAB1   CVBTD    EQU    $FAB1
        F77E   DIV16    EQU    $F77E
        FB68   GETR     EQU    $FB68
        FECA   WFS      EQU    $FECA

075D                    ORG    $075D

075D           APVIOF   RMB    2
075F           POTENT   RMB    2
0761           APVION   RMB    2
0763           IRSW     RMB    2
0765           RONEA    RMB    2
0767           RONEC    RMB    1
0768           EXPON    RMB    1
0769           KONEA    RMB    2
076B           KTHREE   RMB    1
076C           RDAT0    RMB    1
076D           RDAT1    RMB    9
0776           KDAT0    RMB    1
0777           KDAT1    RMB    8
077F           KDAT6    RMB    2
0781           TEN      RMB    2

0790                    ORG    $0790

0790           PRESTR   RMB    6
```

```
0796                    WSTORE  RMB     5
079B                    CNT     RMB     1

              0007              SETDP   $07

F2D6                            ORG     $F2D6

F2D6 CC   000A          CALCII  LDD     #$0A
F2D9 DD   B1                    STD     TEN         SET SHIFT MULTIPLIER
F2DB 34   20                    PSHS    Y           SAVE Y
F2DD 109E 43                    LDY     STARTD      POINT TO ZERO DATA
F2E0 EC   83                    LDD     --X         GET MEASURED VALUE I-OFF
F2E2 A3   A1                    SUBD    Y++         SUBTRACT ZERO VALUE
F2E4 2A   02                    BPL     STORE1
F2E6 8D   B7                    BSR     NEGD        IF NEGATIVE, COMPLEMENT
F2E8 DD   5D            STORE1  STD     APVIOF
F2EA EC   83                    LDD     --X         GET OPEN CIRCUIT VALUE
F2EC A3   A1                    SUBD    Y++         SUBTRACT MEASURED VALUE
F2EE 2A   02                    BPL     STORE2
F2F0 8D   AD                    BSR     NEGD
F2F2 DD   5F            STORE2  STD     POTENT      SAVE IT
F2F4 27   7D                    BEQ     ERROR5
F2F6 EC   83                    LDD     --X         GET GET APPLIED VOLTAGE,I-ON
F2F8 A3   A1                    SUBD    Y++         SUBTRACT CE TO WE, I-OFF
F2FA 2A   02                    BPL     STORE3
F2FC 8D   A1                    BSR     NEGD
F2FE DD   61            STORE3  STD     APVION
F300 27   71                    BEQ     ERROR5
F302 EC   83                    LDD     --X         GET DROP ACROSS RESISTOR
F304 A3   A1                    SUBD    Y++         SUBTRACT ZERO CURRENT VALUE
F306 2A   02                    BPL     STORE4
F308 8D   95                    BSR     NEGD
F30A DD   63            STORE4  STD     IRSW
F30C 27   65                    BEQ     ERROR5
F30E 35   20                    PULS    Y

F310 9F   0B            RAPPAR  STX     DATBUF      SAVE INDEX
F312 0F   9B                    CLR     CNT
F314 0F   68                    CLR     EXPON
F316 34   20                    PSHS    Y           SAVE Y
F318 8E   0763                  LDX     #IRSW
F31B 108E 0761                  LDY     #APVION     POINT TO VOLTAGE(CE TO WE)
F31F 17   025E                  LBSR    SHIFT
F322 DD   65                    STD     RONEA       SAVE BINARY RESULT
F324 8E   076D                  LDX     #RDAT1      POINT TO ASCII STORAGE AREA
F327 17   07B7                  LBSR    CVBTD       MAKE ASCII DATA AND STORE
F32A 35   20                    PULS    Y

F32C 8E   F3AC          CVR     LDX     #RTBL       POINT TO TABLE OF RESISTANCES
F32F D6   05                    LDB     SWNUM       GET WORKING SWITCH NUMBER
F331 3A                         ABX                 POINT TO VALUE
F332 A6   84                    LDA     X
F334 27   56                    BEQ     ERROR6
F336 97   67                    STA     RONEC       SAVE RESISTANCE
F338 8E   076D                  LDX     #RDAT1
F33B 17   00F6                  LBSR    DECIMP      GO SET UP ASCII OUTPUT DATA
>F33E 16  006F                  LBRA    CORRK

F341 35   20            ERROR4  PULS    Y
```

```
F343 17   0B84                LBSR   WFS
F346 0D 0A                    FCB    $0D,$0A
F348 44 41 54 41              FCC    'DATA EXCEED ALLOWABLE POTENTIAL LIMITS!'
F34C 20 45 58 43
F350 45 45 44 20
F354 41 4C 4C 4F
F358 57 41 42 4C
F35C 45 20 50 4F
F360 54 45 4E 54
F364 49 41 4C 20
F368 4C 49 4D 49
F36C 54 53 21
F36F 00                       FCB    $00
F370 16   07F5                LBRA   GETR

F373 17   0B54       ERROR5   LBSR   WFS
F376 0D 0A                    FCB    $0D,$0A
F378 44 41 54 41              FCC    'DATA READS ZERO!'
F37C 20 52 45 41
F380 44 53 20 5A
F384 45 52 4F 21
F388 00                       FCB    $00
F389 16   07DC                LBRA   GETR

F38C 17   0B3B       ERROR6   LBSR   WFS
F38F 0D 0A                    FCB    $0D,$0A
F391 53 57 49 54              FCC    'SWITCH WAS SET TO ZERO!'
F395 43 48 20 57
F399 41 53 20 53
F39D 45 54 20 54
F3A1 4F 20 5A 45
F3A5 52 4F 21
F3A8 00                       FCB    $00
F3A9 16   07BC                LBRA   GETR

F3AC 00              RTBL     FCB    $00        VALUES CORRESPONDING TO SWITCH
F3AD 04 05 06                 FCB    $04,$05,$06 THESE ARE EXPONENTS

                     ***********************************************
                     *     FOR EPRI INSTRUMENT USE $02,$03,$04
                     ***********************************************

F3B0 86   00         CORRK    LDA    #$00       SET CONSTANT FOR CONVERSION
F3B2 97   68                  STA    EXPON
F3B4 97   9B                  STA    CNT
F3B6 34   20                  PSHS   Y
F3B8 108E 075F                LDY    #POTENT
F3BC DC   5F                  LDD    POTENT     GET VALUE OF REF TO WE
F3BE 1083 0041                CMPD   #$0041
F3C2 102E FF7B                LBGT   ERROR4     GREATER THAN 65 MILLIVOLTS, ERROR
F3C6 8E   0763                LDX    #IRSW      POINT TO POTENTIAL ACROSS RESISTOR

F3C9 0C   9B                  INC    CNT        SET TO MULTIPLY ANSWER BY TWO
F3CB 17   01B2                LBSR   SHIFT
F3CE 0A   68                  DEC    EXPON      DIVIDE ANSWER BY TEN
F3D0 DD   69                  STD    KONEA
F3D2 35   20                  PULS   Y
F3D4 34   70                  PSHS   X,Y,U
F3D6 8D   10                  BSR    MLAREA     TIMES AREA - STORE ASCII DATA
```

```
F3D8 35   70               PULS   X,Y,U     RESTORE REGISTERS
F3DA BE   8002             LDX    SAVST0
F3DD 9F   00               STX    INDEX
F3DF BE   8004             LDX    SAVST1    GET FINAL ADDRESS OF DATA
F3E2 BF   8002             STX    SAVST0    START FOR SAVE CMOS MEMORY
F3E5 16   0082             LBRA   PUTDAT

F3E8 BE   0769    MLAREA   LDX    #KONEA    POINT TO V/A DATA
F3EB 108E 074D             LDY    #AREAST   POINT TO AREA DATA
F3EF EC   A4               LDD    Y         GET AREA
F3F1 26   05               BNE    MLD       IF NON-ZERO GO
F3F3 CC   0001             LDD    #$0001    IF AREA ZERO DEFAULT
F3F6 ED   A4               STD    Y         TO 1
F3F8 CE   0751    MLD      LDU    #WORK     POINT TO RESULT STORAGE LOCATION
F3FB 17   03C2             LBSR   MUL16

                  ****************************************

                  * THE NEXT ROUTINE ROUNDS OFF DATA TO 16 BIT

                  ****************************************

F3FE EC   C4      MLCHAR   LDD    U         FETCH STORED RESULT
F400 27   09               BEQ    CNTMC     IF NON-ZERO - ROUNDOFF NEEDED
F402 8D   16               BSR    ROUNDF
F404 EC   42               LDD    2,U       GET DATA
F406 17   01AF             LBSR   SHIFTB    FINISH ROUNDOFF PROCEDURE
F409 ED   42               STD    2,U       SAVE RESULT
F40B EC   42      CNTMC    LDD    2,U       GET ROUNDED DATA
F40D DD   69               STD    KONEA
F40F 8E   0777             LDX    #KDAT1    POINT TO ASCII STORAGE AREA
F412 17   069C             LBSR   CVBTD     MAKE ASCII AND STORE
F415 8E   0777             LDX    #KDAT1
F418 20   1A               BRA    DECIMP

F41A 84   FF      ROUNDF   ANDA   #$FF      BYTE ZERO?
F41C 27   09               BEQ    BYTE2
F41E 44                    LSRA             SHIFT D RIGHT
F41F 56                    RORB
F420 17   01A0             LBSR   UPDATE    ELSE ADJUST LOWER HALF
F423 0C   9B               INC    CNT
F425 20   F3      LOOPR    BRA    ROUNDF    DO NEXT BIT

F427 C4   FF      BYTE2    ANDB   #$FF      IS BYTE ZERO?
F429 27   08               BEQ    ENDRND    GO, DONE
F42B 54                    LSRB             ELSE SHIFT NEXT BIT
F42C 17   0194             LBSR   UPDATE
F42F 0C   9B               INC    CNT
F431 20   F4      LOOPB    BRA    BYTE2
F433 39           ENDRND   RTS

F434 96   67      DECIMP   LDA    RONEC     GET RESISTANCE EXPONENT
F436 9B   68               ADDA   EXPON     ADJUST EXPONENT
F438 8B   04               ADDA   #$04      POSITION ADJUST FOR DECIMAL POINT
F43A 97   68               STA    EXPON     SAVE IT
F43C 81   0A               CMPA   #$0A      IS EXPONENT 10
F43E 27   0F               BEQ    TENEXP
F440 80   0A               SUBA   #$0A      IS EXPONENT >10
F442 2B   06               BMI    SINGEX
```

```
F444 1F   89              TFR   A,B        SAVE ONES DIGIT
F446 86   01              LDA   #$01       SET TENS
F448 20   08              BRA   DECDAT
F44A 4F          SINGEX   CLRA             SET TENS TO ZERO
F44B D6   68              LDB   EXPON      GET ONES DIGIT
F44D 20   03              BRA   DECDAT
F44F 5F          TENEXP   CLRB             SET ONES DIGIT TO ZERO
F450 86   01              LDA   #$01       SET TENS TO ONE

F452 8A   30     DECDAT   ORA   #$30
F454 CA   30              ORB   #$30       MAKE EXPONENT ASCII
F456 E7   07              STB   7,X
F458 A7   06              STA   6,X
F45A A6   84              LDA   X
F45C A7   1F              STA   -1,X
F45E 86   2E              LDA   #'.        PUT DECIMAL POINT IN DATA
F460 A7   84              STA   X
F462 86   45              LDA   #'E        PUT EXPONENTIAL SIGN IN DATA
F464 A7   05              STA   5,X
F466 39                   RTS

F46A                      ORG   $F46A


F46A             PUTDAT   RMB   2          STARTING ADDRESS OUTPUT DATA


F580                      ORG   $F580

F580 EC   A4     SHIFT    LDD   0,Y        GET MULTIPLICAND
F582 DD   90              STD   PRESTR
F584 34   20              PSHS  Y          SAVE Y POINTER
F586 108E 0790            LDY   #PRESTR
F58A 9F   00              STX   INDEX      SAVE POINTER
F58C 8E   0781   SHIFT1   LDX   #TEN       POINT TO MULTIPLIER
F58F CE   0751            LDU   #WORK      POINT TO STORAGE
F592 17   022B            LBSR  MUL16      GO MULTIPLY
F595 0A   68              DEC   EXPON      ADJUST EXPONENT
F597 DC   51              LDD   WORK       GET UPPER 16 BITS
F599 26   06              BNE   USERND     IF NOT ZERO, ROUND OFF
F59B DC   53              LDD   WORK+2     GET DATA
F59D ED   A4              STD   0,Y
F59F 20   EB              BRA   SHIFT1     IF ZERO, DO AGAIN
F5A1 17   FE76   USERND   LBSR  ROUNDF     ROUND OFF
F5A4 35   20              PULS  Y
F5A6 DC   53     USEABL   LDD   WORK+2     GET USABLE PART OF DATA
F5A8 27   22              BEQ   ERROR7
F5AA ED   A4              STD   0,Y        SAVE VALUE
F5AC 9E   00              LDX   INDEX      RESTORE POINTER

F5AE 34   40     PROCES   PSHS  U          SAVE U REGISTER
F5B0 CE   0796            LDU   #WSTORE    POINT TO SCRATCH AREA
F5B3 17   01C8            LBSR  DIV16
F5B6 35   40              PULS  U
F5B8 0D   9B     SHIFTB   TST   CNT        TEST ADJUSTMENT COUNT
F5BA 27   06              BEQ   ENDPRO     IF ZERO NONE NECESSARY
F5BC 58                   ASLB             ELSE MULTIPLY ANSWER BY 2
F5BD 49                   ROLA
F5BE 0A   9B              DEC   CNT
```

```
F5C0 20    F6              BRA    SHIFTB
F5C2 39           ENDPRO   RTS                 RETURN

F5C3 06    53     UPDATE   ROR    WORK+2       SHIFT LOWER WORD
F5C5 06    54              ROR    WORK+3
F5C7 24    02              BCC    BACK
F5C9 0C    54              INC    WORK+3       IF BIT WAS SET ADJUST LAST BIT
F5CB 39           BACK     RTS

F5CC 35    20     ERROR7   PULS   Y
F5CE 16    FDA2            LBRA   ERROR5

                           END

0 ERROR(S) DETECTED

SYMBOL TABLE:

APVIOF 075D  APVION 0761  AREAST 074D  BACK   F5CB  BYTE2  F427
CALCII F2D6  CNT    079B  CNTMC  F40B  CORRK  F3B0  CVBTD  FAB1
CVR    F32C  DATBUF 070B  DECDAT F452  DECIMP F434  DIV16  F77E
ENDPRO F5C2  ENDRND F433  ERROR4 F341  ERROR5 F373  ERROR6 F38C
ERROR7 F5CC  EXPON  0768  GETR   FB6B  INDEX  0700  IRSW   0763
KDAT0  0776  KDAT1  0777  KDAT6  077F  KONEA  0769  KTHREE 076B
LOOPS  F431  LOOPR  F425  MLAREA F3EB  MLCHAR F3FE  MLD    F3FB
MUL16  F7C0  NEGD   F29F  OPNCRK 0759  POTENT 075F  PRESTR 0790
PROCES F5AE  PUTDAT F46A  RAPPAR F310  RDAT0  076C  RDAT1  076D
RONEA  0765  RONEC  0767  ROUNDF F41A  RTBL   F3AC  SAVST0 8002
SAVST1 8004  SHIFT  F580  SHIFT1 F58C  SHIFTB F5B8  SINGEX F44A
STARTD 0743  STORE1 F2EB  STORE2 F2F2  STORE3 F2FE  STORE4 F30A
SWNUM  0705  TEN    0781  TENEXP F44F  UPDATE F5C3  USEABL F5A6
USERND F5A1  WFS    FECA  WORK   0751  WSTORE 0796
```

```
                *********************************************

                *                 PUTDATA

                * THIS ROUTINE OUTPUTS THE FINAL RESULTS TO THE
                * DISPLAY.  IT THEN PROMPTS FOR CMOS MEMORY SAVE,
                * AND THEN FOR CONTINUATION.  IF NOTHING FURTHER
                * IS DESIRED. IT GOES INTO AN ENDLESS WAIT LOOP
                * UNTIL POWER DOWN.


                *********************************************

                * SYSTEM EQUATES

          0700  INDEX   EQU    $0700
          0800  BUFFER  EQU    $0800
          076C  RDAT0   EQU    $076C
          0776  KDAT0   EQU    $0776
          DE20  ACIA    EQU    $DE20
          0756  REVFLG  EQU    $0756
          070B  DATBUF  EQU    $070B
          0743  STARTD  EQU    $0743

          DE28  KEYPAD  EQU    $DE28
          F949  KBINCH  EQU    $F949
          FC00  STARTCL EQU    $FC00
          FB7D  PGMSAV  EQU    $FB7D
          FBCC  UNPAKS  EQU    $FBCC
          FECA  WFS     EQU    $FECA
          FF08  WRC     EQU    $FF08
          F2D6  CALCII  EQU    $F2D6

          0007          SETDP  $07

F46A                    ORG    $F46A

F46A 0D   56    PUTDAT  TST    REVFLG    IS THIS FIRST OUTPUT POINT?
F46C 26   02            BNE    HEADER    IF SET, OUTPUT HEADER
F46E 20   22            BRA    ROUT      ELSE OUTPUT NEXT DATA POINT

F470 0F   56    HEADER  CLR    REVFLG
F472 17   00C5          LBSR   CRLF
F475 8E   0800          LDX    #BUFFER   POINT TO HEADER DATA

F478 C6   0F            LDB    #15       OUTPUT ID
F47A 108E DE20          LDY    #ACIA
F47E 17   00C4          LBSR   OUTDAT
F481 A6   80            LDA    X+        INCREMENT X, CHECK FOR LAST CHAR.
F483 1026 00D8          LBNE   ERROR7

F487 17   00C4  OUTSS   LBSR   OUT4S
F48A C6   06            LDB    #6
F48C 17   00B6          LBSR   OUTDAT    OUTPUT DATE
F48F 17   009C          LBSR   CHARIN    WAIT FOR CHARACTER
```

```
F492 17    00A5        ROUT     LBSR    CRLF
F495 17    0A32                 LBSR    WFS
F498 52 27 20 3D                FCC     "R' = ",$00
F49C 20 00
F49E 8E    076C                 LDX     #RDAT0      POINT TO R'DATA
F4A1 C6    09                   LDB     #9
F4A3 17    009F                 LBSR    OUTDAT      OUTPUT R'
F4A6 17    0A21                 LBSR    WFS
F4A9 20 4F 48 4D                FCC     ' OHMS',$00
F4AD 53 00
F4AF 8D    7D                   BSR     CHARIN      WAIT FOR CHARACTER

F4B1 17    0086        KOUT     LBSR    CRLF
F4B4 17    0A13                 LBSR    WFS
F4B7 4B 20 3D 20                FCC     'K = ',$00
F4BB 00
F4BC 8E    0776                 LDX     #KDAT0
F4BF C6    09                   LDB     #9
F4C1 17    0081                 LBSR    OUTDAT      OUTPUT K
F4C4 17    0A03                 LBSR    WFS
F4C7 20 4F 48 4D                FCC     ' OHM X CM↑2  ',$00
F4CB 20 58 20 43
F4CF 4D 5E 32 20
F4D3 20 00
F4D5 8D    57                   BSR     CHARIN      WAIT FOR CHARACTER
F4D7 9E    43                   LDX     STARTD      GET BEGINNING OF FILE
F4D9 30    08                   LEAX    8,X         OFFSET BY ZERO LOCATIONS
F4DB 9C    0B                   CPX     DATBUF      HAVE WE REACHED END OF DATA
F4DD 27    05                   BEQ     EXIT
F4DF 9E    0B                   LDX     DATBUF      ELSE, LOAD POINTER TO NEXT DATA
F4E1 16    FDF2                 LBRA    CALCII      GO DO IT

F4E4 8D    54          EXIT     BSR     CRLF
F4E6 17    09E1                 LBSR    WFS
F4E9 53 41 56 45                FCC     'SAVE DATA?   YES = 1',$00
F4ED 20 44 41 54
F4F1 41 3F 20 20
F4F5 20 59 45 53
F4F9 20 3D 20 31
F4FD 00
F4FE 8D    2E                   BSR     CHARIN
F500 C1    31                   CMPB    #$31        IS IT A ONE?
F502 27    31                   BEQ     DSAVE
F504 8D    34          ANOTHR   BSR     CRLF
F506 17    09C1                 LBSR    WFS
F509 57 41 4E 54                FCC     'WANT ANOTHER RUN?  YES = 1',$00
F50D 20 41 4E 4F
F511 54 48 45 52
F515 20 52 55 4E
F519 3F 20 20 59
F51D 45 53 20 3D
F521 20 31 00

F524 8D    0B                   BSR     CHARIN
F526 C1    31                   CMPB    #$31
F528 1027 06D4                  LBEQ    STARTCL
F52C 20    FE                   BRA     *           THIS IS AN ENDLESS LOOP!
```

```
F52E B6   DE2B      CHARIN  LDA    KEYPAD     CLEAR KEYPAD
F531 17   0415              LBSR   KBINCH
F534 39                     RTS

F535 17   0645      DSAVE   LBSR   PGMSAV     DISABLE CMOS MEMORY
F538 20   CA                BRA    ANOTHR

F53A 108E DE20      CRLF    LDY    #ACIA
F53E 17   0989              LBSR   WFS
F541 0D 0A 00               FCB    $0D,$0A,$00
F544 39                     RTS

F545 A6   80        OUTDAT  LDA    X+
F547 17   09BE              LBSR   WRC        OUTPUT CHAR.
F54A 5A                     DECB
F54B 26   F8                BNE    OUTDAT
F54D 39                     RTS

F54E 9F   00        OUT4S   STX    INDEX
F550 108E DE20              LDY    #ACIA
F554 17   0973              LBSR   WFS
F557 20 20 20 20            FCB    $20,$20,$20,$20,$00
F55B 00
F55C 9E   00                LDX    INDEX
F55E 39                     RTS

F55F 86   3F        ERROR7  LDA    #'?
F561 17   09A4              LBSR   WRC
F564 16   FF20              LBRA   OUTSS

F567                END

0 ERROR(S) DETECTED

SYMBOL TABLE:

ACIA   DE20   ANOTHR F504   BUFFER 0800   CALCII F2D6   CHARIN F52E
CRLF   F53A   DATBUF 070B   DSAVE  F535   END    F567   ERROR7 F55F
EXIT   F4E4   HEADER F470   INDEX  0700   KBINCH F949   KDAT0  0776
KEYPAD DE2B   KOUT   F4B1   OUT4S  F54E   OUTDAT F545   OUTSS  F487
PGMSAV FB7D   PUTDAT F46A   RDAT0  076C   REVFLG 0756   ROUT   F492
STARTC FC00   STARTD 0743   UNPAKS FBCC   WFS    FECA   WRC    FF08
```

DIV151 F78E DIV153 F799 DIV16 F77E DIV163 F7A1 DIV165 F7AB
DIV167 F7AD

NOT FOUND

```
                                    ************************************************

                                    *                          MUL16

                                    * THIS ROUTINE IS A 16 BIT BY 16 BIT MULTIPLY!
                                    * ONE NUMBER IS POINTED TO BY X, THE OTHER BY Y.
                                    * U POINTS TO STORAGE FOR THE RESULT.  THIS
                                    * ROUTINE GENERATES A 32 BIT UNSIGNED RESULT.

                                    *  (A:B) X (C:D) =         BDH:BDL
                                    *                    +     BCH:BCL
                                    *                    +     ADH:ADL
                                    *                    + ACH:ACL
                                    *                     ----------------

                                    ************************************************

  F7C0                                    ORG     $F7C0

  F7C0 6F    C4             MUL16         CLR     0,U
  F7C2 6F    41                           CLR     1,U
  F7C4 A6    01                           LDA     1,X           LEAST SIGN. BYTE NUMBER A
  F7C6 E6    21                           LDB     1,Y           LEAST SIGN. BYTE NUMBER B
  F7C8 3D                                 MUL
  F7C9 ED    42                           STD     2,U           SAVE RESULT
  F7CB A6    84                           LDA     0,X           MOST SIGN. BYTE NUMBER A
  F7CD E6    21                           LDB     1,Y           LEAST SIGN. BYTE NUMBER B
  F7CF 3D                                 MUL
  F7D0 E3    41                           ADDD    1,U
  F7D2 ED    41                           STD     1,U
  F7D4 24    02                           BCC     AB1
  F7D6 6C    C4                           INC     0,U

  F7D8 A6    01             AB1           LDA     1,X           LEAST SIGN. BYTE NUMBER A
  F7DA E6    A4                           LDB     0,Y           MOST SIGN. BYTE NUMBER B
  F7DC 3D                                 MUL
  F7DD E3    41                           ADDD    1,U
  F7DF ED    41                           STD     1,U
  F7E1 24    02                           BCC     AB2
  F7E3 6C    C4                           INC     0,U

  F7E5 A6    84             AB2           LDA     0,X           MOST SIGN. BYTE NUMBER A
  F7E7 E6    A4                           LDB     0,Y           MOST SIGN. BYTE NUMBER B
  F7E9 3D                                 MUL
  F7EA E3    C4                           ADDD    0,U
  F7EC ED    C4                           STD     0,U

  F7EE 39                                 RTS

  F7EF                      END

0 ERROR(S) DETECTED

SYMBOL TABLE:

AB1     F7D8    AB2     F7E5    END     F7EF    MUL16   F7C0
```

```
            *********************************************

            *                 ASTIMER

            * THIS ROUTINE INITIALIZES THE TIMER IN THE
            * MODULAS ONE SYSTEM, AND ALSO SETS TIMER
            * PERIOD AND CHECKS FOR TIMEOUT OF TIMER 0.
            * HANDLING OF TIMER ONE AND TWO CAN BE DONE
            * IF DESIRED. SEE DATA SHEETS FOR TIMER AND
            * MODULE 1140 MANUAL

            *********************************************


            * SYSTEM EQUATES

       0034 CTRLT0 EQU   $34      INITIAL CONTROL STATE TIMER 0
       0060 CTRLT1 EQU   $60      INITIAL CONTROL STATE TIMER 1
       0099 CTRLT2 EQU   $99      INITIAL CONTROL STATE TIMER 2
       0025 MSBT0  EQU   $25      MOST SIGNIFICANT BYTE COUNTER 0
       00B0 LSBT0  EQU   $80      LEAST SIGN. BYTE COUNTER 0
       00FF LSBT1  EQU   $FF      LEAST SIGN. BYTE COUNTER 1
       0010 MSBT2  EQU   $10      MOST SIGN. BYTE COUNTER 2

            * ADDRESSES

       DE2C CT0    EQU   $DE2C    COUNTER 0 ADDRESS
       DE2D CT1    EQU   $DE2D    COUNTER 1
       DE2E CT2    EQU   $DE2E    COUNTER 2
       DE2F TIMER  EQU   $DE2F    CONTROL WORD FOR PROGRAMMING
       DE29 CRA    EQU   $DE29    PIA CONTROL REGISTER A
       DE2B KPD    EQU   $DE28    DATA REGISTER A--KEYPAD

            *  INITIALIZATION

F800               ORG   $FB00


F800 86    34   INTIME LDA   #CTRLT0  RESET TIMERS
F802 B7    DE2F        STA   TIMER
F805 86    60          LDA   #CTRLT1
F807 B7    DE2F        STA   TIMER
F80A 86    99          LDA   #CTRLT2
F80C B7    DE2F        STA   TIMER
F80F 86    80          LDA   #LSBT0   SET INITIAL VALUES
F811 B7    DE2C        STA   CT0
F814 86    25          LDA   #MSBT0
F816 B7    DE2C        STA   CT0
F819 86    FF          LDA   #LSBT1
F81B B7    DE2D        STA   CT1
F81E 86    10          LDA   #MSBT2
F820 B7    DE2E        STA   CT2
F823 39                RTS


            *********************************************
```

```
                      * WHEN THE TIMER IS READ, IT INDICATES THE
                      * NUMBER OF CLOCK PERIODS ELAPSED SINCE THE
                      * LAST INTERRUPT (TIMEOUT) WAS GENERATED.
                      * HERE, THE TIMER HAS BEEN INITIALIZED TO
                      * GENERATE A TIMEOUT PULSE EVERY SECOND.
                      * THE INPUT CLOCK IS A 9600 HERTZ PULSE INPUT
                      * FROM THE BAUD RATE GENERATOR, AND THE TIMEOUT
                      * APPEARS AT CA1 OF THE PIA.

                      * TO CHANGE THE TIMER PERIOD, THE COUNTER IS
                      * LOADED WITH A NEW VALUE. THIS NEW COUNT
                      * IS EFFECTIVE AFTER THE NEXT TIMEOUT PERIOD.

                      ***********************************************

                      * READ TIMER--A AND B REGISTERS DESTROYED

 FB24 4F              RDTIME  CLRA
 FB25 B7    DE2F              STA     TIMER      LATCH COUNTER 0
 FB28 F6    DE2C              LDB     CT0        GET LSB OF COUNT
 FB2B B6    DE2C              LDA     CT0        GET MSB OF COUNT
 FB2E 39                      RTS

                      * CHANGE TIMER PERIOD--A  DESTROYED
                      *                        X CONTAINS LOCATION
                      *                          OF NEW VALUE

 FB2F A6    01        CGTIME  LDA     1,X        GET LSB OF NEW VALUE
 FB31 B7    DE2C              STA     CT0
 FB34 A6    84                LDA     X          GET MSB OF NEW VALUE
 FB36 B7    DE2C              STA     CT0        COUNTER 0 NOW CONTAINS NEW PERIOD

 FB39 B6    DE29      TIMCHK  LDA     CRA        CHECK FOR TIME OUT
 FB3C 48                      ASLA
 FB3D 24    FA                BCC     TIMCHK
 FB3F B6    DE2B              LDA     KPD        CLEAR IRQ
 FB42 39                      RTS                TIMER NOW OPERATING AT NEW PERIOD

                      * TIMCHK IS ALSO USED AS SUBROUTINE CALL
                      *         FOR TIMING NEEDS.

                              END

0 ERROR(S) DETECTED

SYMBOL TABLE:

CGTIME FB2F   CRA     DE29   CT0     DE2C   CT1     DE2D   CT2     DE2E
CTRLT0 0034   CTRLT1  0060   CTRLT2  0099   INTIME  FB00   KPD     DE2B
LSBT0  0080   LSBT1   00FF   MSBT0   0025   MSBT2   0010   RDTIME  FB24
TIMCHK FB39   TIMER   DE2F
```

NO SUCH FILE

```
                    ****************************************************

                    *                ASATOD

                    * THIS ROUTINE PROVIDES A UTILITY PROGRAM TO
                    * OPERATE THE ADAPTIVE SYSTEMS MODULAS ONE
                    * A TO D MODULE #1642.

                    ****************************************************


                    * SYSTEM EQUATES

              DE10  DADC      EQU     $DE10      BASE ADDRESS OF 1642 MODULE = PIA

0700                          ORG     $0700

0700                INDEX     RMB     2          RESERVE TWO LOCATIONS FOR X TEMP
0702                ADCVAL    RMB     2          LOCATION FOR LATEST ADC VALUE
0704                CHNUM     RMB     1          STORAGE FOR CURRENT CHANNEL

F843                          ORG     $F843

              0007            SETDP   $07

                    * INITIALIZATION---ASSUMES RESET HAS OCCURRED

F843 8E    DE10     INITAD    LDX     #DADC      GET ADDRESS OF PIA
F846 CC    34F0               LDD     #$34F0
F849 A7    01                 STA     1,X        SET A SIDE--ALL INPUT,CA2 OUTPUT
F84B E7    02                 STB     2,X        SET B SIDE--PB4-7 OUT,PB0-3 IN
F84D 86    3C                 LDA     #$3C
F84F A7    03                 STA     3,X        SET CB2--OUTPUT, IRQ'S MASKED
F851 39                       RTS

                    * CHANNEL SELECTION

F852 9F    00       CHSELT    STX     INDEX      SAVE INDEX
F854 8E    DE10               LDX     #DADC      GET BASE ADDRESS
F857 96    04                 LDA     CHNUM
F859 48                       ASLA
F85A 48                       ASLA
F85B 48                       ASLA
F85C 48                       ASLA
F85D A7    02                 STA     2,X        PUT IN CHANNEL SELECT
F85F C6    08                 LDB     #8         DO APPROX. 50 MICROSEC DELAY
F861 5A             ADC1      DECB               FOR OP AMP TO STABILIZE
F862 26    FD                 BNE     ADC1
F864 9E    00                 LDX     INDEX      RESTORE INDEX
F866 39                       RTS

                    * SELECT SAMPLING MODE

F867 9F    00       SMSELT    STX     INDEX      SAVE INDEX
F869 8E    DE10               LDX     #DADC      GET BASE ADDRESS
F86C CC    343C               LDD     #$343C
F86F A7    01                 STA     1,X        SET CA2-LO FOR A/D SAMPLING
```

```
  F871 12                NOP
  F872 12                NOP
  F873 12                NOP
  F874 E7    03          STB     3,X       RESET ADC
  F876 9E    00          LDX     INDEX     RESTORE INDEX
  F878 39                RTS

                  * A/D CONVERSION

  F879 9F    00   CONVRT STX     INDEX     SAVE INDEX
  F87B 8E    DE10        LDX     #DADC
  F87E CC    3C34        LDD     #$3C34
  F881 A7    01          STA     1,X       SET CA2-HI FOR HOLD MODE
  F883 86    04          LDA     #$4
  F885 E7    03          STB     3,X       START ADC CONVERSION (CB2-LO)
  F887 E6    03   ADC3   LDB     3,X       CECK FOR EOC
  F889 2B    03          BMI     ADC4      IF RECEIVED--READ VALUE
  F88B 4A                DECA              IF NOT FOUND, LOOP
  F88C 26    F9          BNE     ADC3      UNTIL LOOP COUNTER = 0

  F88E A6    02   ADC4   LDA     2,X       READ UPPER 4 BITS
  F890 8A    F0          ORA     #$F0      MASK OUTPUT BITS
  F892 E6    84          LDB     0,X       GET LOWER 8 BITS
  F894 40                NEGA              DATA IS COMPLEMENTARY OFFSET BINAR
Y
  F895 50                NEGB              MUST BE NEGATED
  F896 82    00          SBCA    #0        ADJUST OVERFLOW
  F898 DD    02          STD     ADCVAL    SAVE RESULT
  F89A 9E    00          LDX     INDEX     RESTORE INDEX
  F89C 39                RTS


                         END

0 ERROR(S) DETECTED

SYMBOL TABLE:

ADC1   F861   ADC3   F887   ADC4   F88E   ADCVAL 0702   CHNUM  0704
CHSELT F852   CONVRT F879   DADC   DE10   INDEX  0700   INITAD F843
SMSELT F867
```

```
                    ******************************************
                    *               ASDTOA

                    * THIS ROUTINE PROVIDES A UTILITY PROGRAM TO
                    * OPERATE THE ADAPTIVE SYSTEMS MODULAS ONE
                    * D TO A MODULE #1614

                    ******************************************


                                 OPT    PAG

                    * SYSTEM EQUATES

              DE50  DDAC        EQU     $DE50      DAC BASE ADDRESS
              DE55  DSWITC      EQU     $DE55      ANALOG SWITCH SELECT ADDRESS
              0700  INDEX       EQU     $0700      TEMP STORAGE FOR INDEX

0705                            ORG     $0705
0705                SWNUM       RMB     1          STORAGE FOR LATEST SWITCH VALUE
0706                DACVAL      RMB     2          STORAGE FOR LATEST INPUT VALUE
0708                DACOLD      RMB     2          STORAGE FOR PREVIOUS INPUT VALUE

              0007              SETDP   $07

F89D                            ORG     $F89D

                    * INITIALIZATION

F89D 8E    DE50     INITDA      LDX     #DDAC      GET BASE ADDRESS
F8A0 4F                         CLRA
F8A1 C6    06                   LDB     #6         ZERO DAC'S AND TURN SWITCH OFF
F8A3 8D    05                   BSR     DACINZ
F8A5 C6    0A                   LDB     #10
F8A7 8E    0700                 LDX     #INDEX

F8AA A7    80       DACINZ      STA     X+
F8AC 5A                         DECB
F8AD 26    FB                   BNE     DACINZ
F8AF 39                         RTS

                    * TURN SWITCH ON

F8B0 9F    00       SWNSET      STX     INDEX      SAVE INDEX
F8B2 8E    DE55                 LDX     #DSWITC    POINT TO SWITCH SELECT
F8B5 96    05                   LDA     SWNUM      GET SWITCH NUMBER
F8B7 46                         RORA               MOVE NUMBER TO BITS 6 AND 7
F8B8 46                         RORA
F8B9 46                         RORA
F8BA A7    84                   STA     X          SEND TO SWITCH SELECT
F8BC 9E    00                   LDX     INDEX      RESTORE INDEX
F8BE 39                         RTS

                    * TURN SWITCH OFF--ALTERNATE TO SWNSET WITH 00
                    *           DOES NOT CHANGE CONTENTS OF SWNUM
```

```
FBBF 9F    00      SWNZER   STX    INDEX      SAVE INDEX
FBC1 8E    DE55             LDX    #DSWITC
FBC4 86    00               LDA    #$00
FBC6 A7    84               STA    0,X
FBC8 9E    00               LDX    INDEX      RESTORE INDEX
FBCA 39                     RTS

                   * LOAD DAC VALUE

FBCB 9F    00      LDDAC    STX    INDEX      SAVE INDEX
FBCD 8E    DE50             LDX    #DDAC      POINT TO DAC 1
FBD0 D6    06               LDB    DACVAL     GET DAC INPUT
FBD2 D7    08               STB    DACOLD
FBD4 96    07               LDA    DACVAL+1
FBD6 97    09               STA    DACOLD+1
FBD8 54                     LSRB              SHIFT SO THAT 2 LO  BITS ARE IN
FBD9 46                     RORA              BITS 6 AND 7 OF B AND 8 HI BITS
FBDA 56                     RORB              ARE IN A
FBDB 46                     RORA
FBDC 56                     RORB
FBDD ED    84               STD    X          OUTPUT VALUE TO DAC
FBDF 9E    00               LDX    INDEX      RESTORE INDEX
FBE1 39                     RTS

                            END

0 ERROR(S) DETECTED
```

SYMBOL TABLE:

```
DACINZ FBAA   DACOLD 0708   DACVAL 0706   DDAC   DE50   DSWITC DE55
INDEX  0700   INITDA FB9D   LDDAC  F8CB   SWNSET FBB0   SWNUM  0705
SWNZER FBBF
```

```
                ******************************************
                *                SAVECK

                * THIS ROUTINE CHECKS FOR DISABLING OF THE
                * CMOS MEMORY ON STARTUP. THE CONTENTS OF
                * THIS CMOS MEMORY ARE SAVED BY BISABLING
                * THE INPUT TO THE MEMORY. THIS IS DONE BY
                * CLOSURE OF THE STORE KEY (ALTERNATE ACTION)
                * PRIOR TO POWER-OFF OR WHENEVER DESIRED.
                * ON RESET OR POWER-UP, THE UNIT WILL CHECK
                * AND REQUEST RELEASE IF NEEDED. THAT IS THE
                * FUNCTION OF THIS ROUTINE.

                ******************************************


                * SYSTEM EQUATES

           DE2A ORB     EQU    $DE2A     ADDRESS OF DATA REG OF PIA
           DE20 ACIA    EQU    $DE20
           FECA WFS     EQU    $FECA     ROUTINE TO WRITE STRING
           070A SAVFLG  EQU    $070A

           0007         SETDP  $07

FBF0                    ORG    $FBF0

                * CHECK FOR SAVE AND REQUEST RELEASE
                *      ASSUMES INITIALIZATION COMPLETE!

FBF0 108E DE20  SAVECK  LDY    #ACIA     POINT TO COMMUNICATIONS REGISTER
FBF4 F6   DE2A          LDB    ORB       GET PIA B DATA
FBF7 C4   40            ANDB   #$40      IS BIT 6 HI?
FBF9 27   04            BEQ    CONT1     IF NOT HIGH, MEMORY HAS BEEN SAVED

FBFB 0F   0A            CLR    SAVFLG
FBFD 20   31            BRA    CONT      MEMORY AVAILABLE-CONTINUE
FBFF 86   FF    CONT1   LDA    #$FF      SET FLAG .
F901 97   0A            STA    SAVFLG
F903 17   05C4          LBSR   WFS       REQUEST CLEARANCE OF DISABLE
F906 4D 45 4D 4F        FCC    'MEMORY NOT AVAILABLE--UNSAVE!'
F90A 52 59 20 4E
F90E 4F 54 20 41
F912 56 41 49 4C
F916 41 42 4C 45
F91A 2D 2D 55 4E
F91E 53 41 56 45
F922 21
F923 00                 FCB    0
F924 86   04    CHECK   LDA    #$4
F926 C6   40    CHECK1  LDB    #$40
F928 F5   DE2A          BITB   ORB
F92B 27   F7            BEQ    CHECK
F92D 4A                 DECA
F92E 26   F6            BNE    CHECK1    CHECK FOUR TIMES FOR COMPLETION
F930 17   0597  CONT    LBSR   WFS
F933 0D 0A              FCB    $0D,$0A
```

```
F935 52 45 41 44          FCC     'READY'
F939 59
F93A 00                   FCB     0
F93B 39                   RTS

                          END

0 ERROR(S) DETECTED

SYMBOL TABLE:

ACIA   DE20  CHECK  F924  CHECK1 F926  CONT  F930  CONT1  F8FF
ORB    DE2A  SAVECK FBF0  SAVFLG 070A  WFS   FECA
```

```
                        ************************************************

                        *                KPDCTL

                        * THIS ROUTINE CONTROLS THE OPERATION OF THE
                        * KEYPAD ON THE PORTABLE INSTRUMENT. THIS
                        * UTILITY IS CONCERNED ONLY WITH THE IMMEDIATE
                        * OPERATION OF THE KEYPAD AND THE ENTRY OF A
                        * SINGLE CHARACTER, AND ITS TRANSLATION INTO
                        * ASCII. OTHER ASPECTS OF KEYPAD USAGE ARE TO
                        * BE CONTROLLED BY THE CALLING ROUTINES.

                        ************************************************


                        * SYSTEM EQUATES

                 DE2B   KEYPAD  EQU     $DE2B       DRA OF PIA ON 1140 MODULE
                 0700   INDEX   EQU     $0700

                 0007           SETDP   $07

F93C                            ORG     $F93C

                        * INITIALIZATION--ASSUMES RESET

F93C 8E    DE28         INITKP  LDX     #KEYPAD     GET PIA BASE ADDRESS
F93F 86    14                   LDA     #$14        A SIDE-INPUT,IRQ'S MASKED
F941 A7    01                   STA     1,X         CA1 GOES LO,CA2 GOES HI
F943 CC    3F04                 LDD     #$3F04      B SIDE-2 IN, 6 OUT MASK IRQ'S
F946 ED    02                   STD     2,X         CB1 AND CB2 GO L0
F948 39                         RTS

                        * READ KEYPAD--NO MATCH SHOWN BY ZERO BIT SET

F949 9F    00           KBINCH  STX     INDEX       SAVE INDEX
F94B 34    40                   PSHS    U           SAVE U
F94D CE    DE28                 LDU     #KEYPAD
F950 86    40                   LDA     #$40        SET MASK
F952 A5    41           CHARIN  BITA    1,U         DO WE HAVE A CHARACTER(CA2-HI)
F954 27    FC                   BEQ     CHARIN      LOOP, IF NOT
F956 A6    C4                   LDA     U           ELSE, GET CHARACTER
F958 8E    F96D                 LDX     #KEYTBL

F95B A1    81           KBIN1   CMPA    X++
F95D 27    0A                   BEQ     KBIN2
F95F 8C    F989                 CPX     KEYTBL+28 HAVE WE REACHED END OF TABLE?
F962 26    F7                   BNE     KBIN1
F964 5F                         CLRB                SHOW NO MATCH FOUND!
F965 9E    00           KBIN3   LDX     INDEX       RESTORE INDEX
F967 35    C0                   PULS    U,PC        RESTORE U REGISTER, RETURN

F969 E6    1F           KBIN2   LDB     -1,X        SHOW MATCH
F96B 20    FB                   BRA     KBIN3

                        * TABLE TO CONVERT KEYPAD VALUE TO ASCII

F96D 11 0D              KEYTBL  FCB     $11,$0D
```

```
F96F 12 33                  FCB    $12,$33
F971 14 36                  FCB    $14,$36
F973 18 39                  FCB    $18,$39
F975 21 30                  FCB    $21,$30
F977 22 32                  FCB    $22,$32
F979 24 35                  FCB    $24,$35
F97B 28 38                  FCB    $28,38
F97D 41 7F                  FCB    $41,$7F
F97F 42 31                  FCB    $42,$31
F981 44 34                  FCB    $44,$34
F983 48 37                  FCB    $48,$37
F985 81 4A                  FCB    $81,$4A
F987 82 48                  FCB    $82,$48

                            END

0 ERROR(S) DETECTED

SYMBOL TABLE:

CHARIN F952   INDEX  0700   INITKP F93C   KBIN1  F95B   KBIN2  F969
KBIN3  F965   KBINCH F949   KEYPAD DE28   KEYTBL F96D
```

```
                  ******************************************************

                  *                    GETDATA

                  * THIS ROUTINE GETS A DATA STRING FROM THE
                  * KEYPAD OF THE PORTABLE INSTRUMENT. ENTRY
                  * REQUIRES THAT B CONTAIN THE NUMBER OF CHAR.
                  * REQUESTED AND X CONTAINS THE STARTING ADDRESS
                  * FOR STORAGE. THE ROUTINE FORMATS THE STRING
                  * SO THAT LEADING ZEROS ARE INSERTED TO FILL
                  * THE REQUESTED DATA BLOCK. STORAGE IS IN THE
                  * FORM OF ASCII CHARACTERS.

                  * INCLUDED IN THIS ROUTINE IS A SUBROUTINE TO
                  * ALLOW FOR APPROXIMATELY 100 MILLISEC PER
                  * CHARACTER

                  ******************************************************


                  * SYSTEM EQUATES

             F949 KBINCH  EQU     $F949      INPUT CHARACTER FROM KEYPAD
             FECA WFS     EQU     $FECA      STRING OUTPUT ROUTINE
             07B0 INDEX1  EQU     $07B0      TEMPORARY STORAGE FOR INDEX
             FA0C CMDADR  EQU     ERROR+$A
             DE28 KEYPAD  EQU     $DE28
             FF08 WRC     EQU     $FF08

 070B                     ORG     $070B

 070B             DATBUF  RMB     2          STORAGE FOR LATEST BUFFER ADDRESS
 070D             TCOUNT  RMB     1          STORAGE FOR WORKING COUNT
 070E             TCHAR   RMB     1          STORAGE FOR WORKING CHARACTER
 070F             TCHCNT  RMB     1          COUNTER FOR ACESSES TO CHAR.
 0710             TBUFFR  RMB     33         BUFFER AREA FOR STRING STORAGE
 0731             RUNFLG  RMB     2          FLAG TO INDICATE RUN STATUS

             0007         SETDP   $07

 F9A3                     ORG     $F9A3

 F9A3 9E   0B     GETDAT  LDX     DATBUF     GET STORAGE POINTER
 F9A5 D7   0D             STB     TCOUNT
 F9A7 9F   B0             STX     INDEX1     SAVE INDEX
 F9A9 8E   0710           LDX     #TBUFFR    POINT TO TEMPORARY BUFFER
 F9AC 0F   0E             CLR     TCHAR
 F9AE B6   DE28           LDA     KEYPAD     CLEAR ANY RESIDUAL CHARACTER IRQ
 F9B1 8D   96     DATA1   BSR     KBINCH     GET A CHARACTER
 F9B3 27   4D             BEQ     ERROR      ZERO BIT IS ERROR FLAG FOR KBINCH
>F9B5 17   007F           LBSR    CHARCK     CHECK FOR TIMEOUT
 F9B8 0A   0D             DEC     TCOUNT     UPDATE CHAR. COUNT
 F9BA 26   2E             BNE     NXTCHR
 F9BC C1   0D             CMPB    #$0D
 F9BE 26   5A             BNE     ERROR1     LAST CHAR. MUST BE CR
 F9C0 E7   80     RET     STB     X+
 F9C2 34   40             PSHS    U          SAVE U
 F9C4 1F   13             TFR     X,U
```

```
F9C6 9E    80                 LDX    INDEX1     POINT TO STORAGE AREA
F9C8 DF    80                 STU    INDEX1
F9CA D6    0D                 LDB    TCOUNT     FETCH REMAINING NUMBER OF CHAR
F9CC 27    07                 BEQ    CHARS1
F9CE 86    30                 LDA    #$30       LOAD A WITH ASCII ZERO
F9D0 A7    80        ZEROS    STA    X+
F9D2 5A                       DECB
F9D3 26    FB                 BNE    ZEROS
F9D5 34    20        CHARS1   PSHS   Y          SAVE Y
F9D7 108E 0710                LDY    #TBUFFR
F9DB A6    A0        CHARS    LDA    Y+
F9DD A7    80                 STA    X+
F9DF 109C 80                  CMPY   INDEX1
F9E2 27    02                 BEQ    RET1
F9E4 20    F5                 BRA    CHARS
F9E6 9F    0B        RET1     STX    DATBUF
F9E8 35    E0                 PULS   Y,U,PC     CLEAN STACK, RETURN

F9EA C1    0D        NXTCHR   CMPB   #$0D       IS CHAR. A CR
F9EC 26    02                 BNE    NXTONE
F9EE 20    D0                 BRA    RET

F9F0 C1    7F        NXTONE   CMPB   #$7F       IS IT A DELETE
F9F2 26    02                 BNE    CHAR
F9F4 5F                       CLRB              SHOW CANCELLATION OF ENTRY
F9F5 39                       RTS

F9F6 E7    80        CHAR     STB    X+         SAVE CHARACTER
F9F8 1F    98                 TFR    B,A
F9FA 17    050B               LBSR   WRC        ECHO CHARACTER
F9FD 58                       ASLB
F9FE 2B    4D                 BMI    CMD        IF CMD, PROCESS
FA00 20    AF                 BRA    DATA1

FA02 17    04C5      ERROR    LBSR   WFS        OUTPUT STRING
FA05 49 4C 4C 45              FCC    'ILLEGAL CHARACTER!'
FA09 47 41 4C 20
FA0D 43 48 41 52
FA11 41 43 54 45
FA15 52 21
FA17 00                       FCB    0
FA18 20    18                 BRA    WAIT

FA1A 17    04AD      ERROR1   LBSR   WFS
FA1D 54 30 30 20              FCC    'T00 MANY CHARACTERS!'
FA21 4D 41 4E 59
FA25 20 43 48 41
FA29 52 41 43 54
FA2D 45 52 53 21
FA31 00                       FCB    0

FA32 17    FF14      WAIT     LBSR   KBINCH     WAIT FOR ANY CHAR. INPUT
FA35 5F                       CLRB              SHOW ERROR
FA36 39                       RTS

FA37 D1    0E        CHARCK   CMPB   TCHAR      COMPARE TO PREVIOUS CHAR.
FA39 27    07                 BEQ    CKMATC
FA3B D7    0E        NEW      STB    TCHAR      SAVE CHAR. IF NO MATCH
FA3D 86    03                 LDA    #3
```

```
FA3F 97   0F                 STA    TCHCNT    SET FOR THREE INPUTS OF CHAR.
FA41 39                      RTS

FA42 0A   0F        CKMATC   DEC    TCHCNT
FA44 27   F5                 BEQ    NEW
FA46 35   02                 PULS   A
FA48 35   02                 PULS   A
FA4A 16   FF64               LBRA   DATA1

FA4D 8E   FA0C      CMD      LDX    #CMDADR   GET CMD BASE ADDRESS
FA50 57                      ASRB             RESTORE CHAR
FA51 3A                      ABX
FA52 6E   84                 JMP    X


FA54 20   07        CMDH     BRA    CMDH1     HALT COMMAND ROUTINE

                    ******************************************

                    * LOCATION RUNFLG MUST BE LOADED WITH THE
                    * ADDRESS OF THE ROUTINE TO BE STARTED ON RUN
                    * COMMAND. ELSE, AN ERROR WILL BE RETURNED.

                    ******************************************

FA56 DC   31        CMDR     LDD    RUNFLG    RUN COMMAND ROUTINE
FA58 27   0C                 BEQ    ERROR2
FA5A 34   06                 PSHS   D         PUT D INTO SUBROUTINE RETURN CALL
FA5C 39                      RTS

                    *******************************************

                    * THE HALT COMMAND ENTERS A LOOP AND STAYS THERE
                    * UNTIL A RUN COMMAND IS RECEIVED.
                    * UPON RECEIPT OF THE R CMD. THE PROCESSOR WILL
                    * RESUME AT THE ROUTINE THAT WAS IN OPERATION
                    * WHEN HALT WAS CALLED.  THIS IS DONE BY
                    * REPORTING AN ERROR AND LETTING THE ROUTINE
                    * USE ITS EXCEPTION CALL.

                    *******************************************

FA5D 17   FEE9      CMDH1    LBSR   KBINCH    WAIT FOR A CHARACTER
FA60 C1   4A                 CMPB   #$4A      IS IT AN R?
FA62 26   F9                 BNE    CMDH1     IF NOT LOOP AGAIN
FA64 5F                      CLRB             ELSE, SHOW ERROR
FA65 39                      RTS              AND RETURN

FA66 20   9A        ERROR2   BRA    ERROR

                             END

0 ERROR(S) DETECTED

SYMBOL TABLE:

CHAR   F9F6   CHARCK FA37   CHARS  F9DB   CHARS1 F9D5   CKMATC FA42
CMD    FA4D   CMDADR FA0C   CMDH   FA54   CMDH1  FA5D   CMDR   FA56
DATA1  F9B1   DATBUF 070B   ERROR  FA02   ERROR1 FA1A   ERROR2 FA66
```

```
GETDAT F9A3    INDEX1 0780    KBINCH F949    KEYPAD DE28    NEW    FA3B
NXTCHR F9EA    NXTONE F9F0    RET    F9C0    RET1   F9E6    RUNFLG 0731
TBUFFR 0710    TCHAR  070E    TCHCNT 070F    TCOUNT 070D    WAIT   FA32
WF[illegible]    FECA    WRC    FF0B    ZEROS  F9D0
```

NO SUCH FILE

```
                        ***************************************************

                        *                CLRBUF

                        * THIS ROUTINE CLEARS AREAS OF MEMORY IN THE AS
                        * SYSTEM. THE STARTING AND ENDING ADDRESSES ARE TO
                        * BE PLACED IN THE LOCATIONS INDICATED, BY THE
                        * CALLING PROGRAM. A AND X ARE DESTROYED.

                        ***************************************************

 07D8                           ORG     $07D8

              0007              SETDP   $07

 07D8                   TBUFST  RMB     2         START ADDRESS TO BE HERE
 07DA                   TBUFND  RMB     2         END ADDRESS TO BE HERE

 FA69                           ORG     $FA69

 FA69 34   40                   PSHS    U         SAVE U REGISTER
 FA6B DE   DA                   LDU     TBUFND    LOAD U WITH END ADDRESS
 FA6D 4F                        CLRA
 FA6E 36   02           NEXT    PSHU    A         STORE A IN LAST LOCATION
 FA70 1193 DB                   CMPU    TBUFST    HAVE WE DONE ALL
 FA73 26   F9                   BNE     NEXT      LOOP IF NOT DONE
 FA75 36   02                   PSHU    A
 FA77 35   C0                   PULS    U,PC      RESTORE U, RETURN

                                END

0 ERROR(S) DETECTED

SYMBOL TABLE:

NEXT    FA6E   TBUFND 07DA   TBUFST 07D8
```

```
                    ******************************************

                    *           BIN2BCD

                    * THIS ROUTINE CONVERTS A 16 BIT BINARY
                    * NUMBER TO BCD AND STORES THE RESULTING
                    * 5 CHARACTERS.
                    * ON START, D CONTAINS THE BINARY NUMBER,
                    * AND X CONTAINS THE POINTER TO DECIMAL
                    * STORAGE

                    ******************************************

                    * SYSTEM EQUATES

               FA79 K10K    EQU    $FA79

                    * TEMPORARY STORAGE

073E                        ORG    $073E

073E                SAVEA   RMB    1          ACCUMULATOR A
073F                SAVEX   RMB    2          STORE DATA POINTER
0741                SAVEX1  RMB    2          POINTER TO CONSTANTS

               0007         SETDP  $07

FAB1                        ORG    $FAB1

FAB1 9F   3F        CVBTD   STX    SAVEX      SAVE DATA POINTER
FAB3 8E   FA79              LDX    #K10K      POINT TO CONSTANTS
FAB6 0F   3E        CVDEC1  CLR    SAVEA      INITIALIZE DEC CHAR
FAB8 A3   84        CVDEC2  SUBD   X
FABA 25   04                BCS    CVDEC5     OVERFLOW
FABC 0C   3E                INC    SAVEA      INC CHAR BEING BUILT
FABE 20   F8                BRA    CVDEC2

FAC0 E3   84        CVDEC5  ADDD   X          RESTORE PARTIAL RESULT
FAC2 36   02                PSHU   A          SAVE A REGISTER
FAC4 9F   41                STX    SAVEX1
FAC6 9E   3F                LDX    SAVEX      POINT TO STORAGE FOR RESULT
FAC8 96   3E                LDA    SAVEA
FACA 8B   30                ADDA   #$30       MAKE ASCII CHARACTER
FACC A7   84                STA    X
FACE 37   02                PULU   A          RESTORE A REGISTER
FAD0 30   01                LEAX   1,X        INCREMENT X
FAD2 9F   3F                STX    SAVEX
FAD4 9E   41                LDX    SAVEX1     POINT TO CONSTANTS
FAD6 30   02                LEAX   2,X
FAD8 8C   FA83              CPX    #K10K+10   ARE WE DONE
FADB 26   D9                BNE    CVDEC1
FADD 39                     RTS

                            END

0 ERROR(S) DETECTED
```

SYMBOL TABLE:

CVBTD	FAB1	CVDEC1	FAB6	CVDEC2	FAB8	CVDEC5	FAC0	K10K	FA79
SAVEA	073E	SAVEX	073F	SAVEX1	0741				

```
                   ****************************************
                   *               BCD2BIN
                   * THIS ROUTINE CONVERTS A STRING OF 5
                   * ASCII NUMBERS TO A 16 BIT BINARY VALUE.
                   * THE INITIAL STRING SHOULD NOT EXCEED
                   * 65,535.
                   * X SHOULD POINT TO THE START OF THE
                   * ASCII STRING.
                   * THE RESULT WILL BE IN D.

                   ****************************************

                   * TEMPORARY STORAGE

073A                       ORG     $073A

073A               TEMP    RMB     1          BCD STORAGE
073B               TEMP1   RMB     2          BINARY STORAGE
073D               COUNT   RMB     1          STORAGE FOR CHARACTER COUNT

             0007          SETDP   $07

                   * CONSTANTS

FA79                       ORG     $FA79

FA79 2710          K10K    FDB     10000
FA7B 03E8                  FDB     1000
FA7D 0064                  FDB     100
FA7F 000A                  FDB     10
FA81 0001                  FDB     1

FA83 0F   3A       CVDTB   CLR     TEMP
FA85 0F   3B               CLR     TEMP1
FA87 0F   3C               CLR     TEMP1+1    INITIALIZE STORAGE
FA89 108E FA79             LDY     #K10K
FA8D 86   05               LDA     #$05
FA8F 97   3D               STA     COUNT

FA91 5F            NEXT    CLRB
FA92 A6   80               LDA     X+         GET ASCII CHARACTER
FA94 80   30               SUBA    #$30       MAKE BCD
FA96 27   0E               BEQ     SUM
FA98 97   3A               STA     TEMP       SAVE IT
FA9A 4F                    CLRA
FA9B 5F                    CLRB
FA9C E3   A4       CV      ADDD    Y          GET CONSTANT VALUE
FA9E 0A   3A               DEC     TEMP
FAA0 26   FA               BNE     CV         ADD UNTIL DONE
FAA2 D3   3B               ADDD    TEMP1
FAA4 DD   3B               STD     TEMP1
FAA6 0A   3D       SUM     DEC     COUNT
FAA8 27   04               BEQ     EXIT
FAAA 31   22               LEAY    2,Y
FAAC 20   E3               BRA     NEXT
```

```
 FAAE DC   3B        EXIT    LDD     TEMP1
 FAB0 39                     RTS

                             END

0 ERROR(S) DETECTED

SYMBOL TABLE:

COUNT  073D   CV     FA9C   CVDTB  FA63   EXIT   FAAE   K10K   FA79
NEXT   FA91   SUM    FAA6   TEMP   073A   TEMP1  073B
```

```
                    ******************************************************

                    *                    FIRQHDL

                    * THIS ROUTINE IS AN INTERRUPT HANDLER FOR FIRQ
                    * WHICH CHOOSES BETWEEN A JUMP TO MONITOR AND
                    * A REPEAT ROUTINE. SELECTION IS MADE BY
                    * MEANS OF THE STATUS OF THE FUNCT SWITCH ON THE
                    * PANEL OF THE PORTABLE INSTRUMENT. THE FIRQ IS
                    * ISSUED BY PRESSING THE KEY MARKED RPT(MONITOR).

                    ******************************************************


                    * SYSTEM EQUATES
              07F6  EFIN      EQU     $07F6      STORAGE LOCATION FOR FIRQ VECTOR
              DE2A  FRQFLG    EQU     $DE2A      PIA B SIDE DATA REGISTER
              FC3C  NMI       EQU     $FC3C      ENTRY ROUTINE FOR AS01
              0700  INDEX     EQU     $0700

0733                          ORG     $0733

              0007            SETDP   $07

0733                RUNCNT    RMB     1          NUMBER OF REPEATS PERFORMED
0734                RPTADR    RMB     2          ADDRESS TO WHICH REPEAT GOES

FADE                          ORG     $FADE

                    * INITIALIZATION

FADE 9F    00       FRQINZ    STX     INDEX      SAVE X
FAE0 8E    FAF1               LDX     #FRQVEC    GET ADDRESS FOR FIRQ ROUTINE
FAE3 36    10                 PSHU    X          SAVE IN STACK
FAE5 8E    FC3C               LDX     #NMI       POINT TO MONITOR
FAE8 9F    34                 STX     RPTADR     SAVE AS DEFAULT FIRQ
FAEA 9E    00                 LDX     INDEX
FAEC 36    32                 PSHU    Y,X,A
FAEE 1C    0F                 ANDCC   #$0F       ENABLE FIRQ
FAF0 39                       RTS

                    * INTERRUPT HANDLER

FAF1 F6    DE2A     FRQVEC    LDB     FRQFLG     READ PIA FOR FUNCTION FLAG
FAF4 2A    06                 BPL     REPEAT     IF BIT7 = 0 REPEAT ROUTINE
FAF6 8E    FC3C     DEBUG     LDX     #NMI       DO MONITOR STARTUP
FAF9 AF    61                 STX     1,S        SET LOCATION TO JUMP TO
FAFB 3B                       RTI                JUMP TO AS01

FAFC 0C    33       REPEAT    INC     RUNCNT
FAFE 6E    9F 0734            JMP     [RPTADR]


                              END

0 ERROR(S) DETECTED
```

SYMBOL TABLE:

```
DEBUG  FAF6   EFIN   07F6   FRQFLG DE2A   FRQINZ FADE   FRQVEC FAF1
INDEX  0700   NMI    FC3C   REPEAT FAFC   RPTADR 0734   RUNCNT 0733
```

```
           **************************************************

           *                    STARTUP

           * THIS ROUTINE IS ENTERED WHEN RESET IS RECEIVED BY
           * THE ADAPTIVE SCIENCE SYSTEM. IT PERFORMS ALL
           * REQUIRED INITIALIZATIONS, AND THEN POINTS TO THE
           * START OF THE ROUTINE, AND WAITS FOR A RUN COMMAND

           **************************************************

           * SYSTEM EQUATES

      FB00 INTIME  EQU    $FB00     INITIALIZE TIMER
      FB43 INITAD  EQU    $FB43     INITIALIZE A TO D
      FB9D INITDA  EQU    $FB9D     INITIALIZE D TO A
      F93C INITKP  EQU    $F93C     INITIALIZE KEYPAD
      07D8 TBUFST  EQU    $07D8
      07DA TBUFND  EQU    $07DA
      FC00 RES     EQU    $FC00     INITIALIZE MONITOR AND ACIA
      FADE FRQINZ  EQU    $FADE     SET UP FIRQ ROUTINE
      FBF0 SAVECK  EQU    $FBF0     CMOS MEMORY SAVE CHECK
      070A SAVFLG  EQU    $070A
      FA69 CLRBUF  EQU    $FA69
      8010 BEGDAT  EQU    $8010     START OF CMOS DATA STORAGE
      0743 STARTD  EQU    $0743
      0731 RUNFLG  EQU    $0731
      F949 KBINCH  EQU    $F949
      FA56 CMDR    EQU    $FA56
      8002 SAVSTO  EQU    $8002     POINTER TO CMOS DATA STORAGE
      F000 BEGIN   EQU    $F000     LOCATION FOR START OF
           * [illegible] EQU $F664        SEQUENCE

      0007         SETDP  $07

FB1F               ORG    $FB1F

FB1F 36   20       STARTUP PSHU   Y
FB21 8D   BB               BSR    FRQINZ
FB23 86   07               LDA    #$07
FB25 1E   BB               EXG    A,DP
FB27 17   FCD6             LBSR   INTIME
FB2A 17   ~~FB70~~ FB33        LBSR   INITDA [illegible]
FB2D 17   FD13             LBSR   INITAD
FB30 17   FE09             LBSR   INITKP
FB33 8E   0000             LDX    #$0000
FB36 9F   D8               STX    TBUFST    POINT TO START OF SCRATCHPAD
FB38 8E   07D7             LDX    #$07D7
FB3B 9F   DA               STX    TBUFND    POINT TO END OF SCRATCHPAD
FB3D 17   FF29             LBSR   CLRBUF
FB40 8E   0800             LDX    #$0800
FB43 9F   D8               STX    TBUFST    POINT TO DATA STORAGE
FB45 8E   0FFF             LDX    #$0FFF
FB48 9F   DA               STX    TBUFND    POINT TO END OF DATA STORAGE
FB4A 17   FF1C             LBSR   CLRBUF
FB4D 17   FDA0             LBSR   SAVECK
FB50 0D   0A               TST    SAVFLG    HAS CMOS MEMORY BEEN PROTECTED?
FB52 26   20               BNE    SAVDAT    YES, GET START OF UNPROTECTED AREA
```

```
FB54 8E    8010              LDX    #BEGDAT
FB57 9F    43                STX    STARTD    SAVE POINTER TO DATA AREA
FB59 9F    DB                STX    TBUFST    CLEAR CMOS MEM.
FB5B 8E    87FF              LDX    #BEGDAT+$7EF
FB5E 9F    DA                STX    TBUFND
FB60 17    FF06              LBSR   CLRBUF
FB63 CC    F000      GO      LDD    #BEGIN
FB66 DD    31                STD    RUNFLG    POINT TO PROGRAM ENTRY
FB68 17    FDDE      GETR    LBSR   KBINCH
FB6B C1    4A                CMPB   #$4A
FB6D 26    F9                BNE    GETR      IF CHARACTER NOT R LOOP
FB6F 17    FEE4              LBSR   CMDR
FB72 27    AB                BEQ    STARTUP   TRY AGAIN, IF ERROR

FB74 BE    8002      SAVDAT  LDX    SAVSTO    GET NEXT DATA LOCATION
FB77 9F    43                STX    STARTD    STORE IN POINTER
FB79 20    E8                BRA    GO


                             END

0 ERROR(S) DETECTED

SYMBOL TABLE:

BEGDAT 8010   BEGIN  F000   CLRBUF FA69   CMDR   FA56   FRQINZ FADE
GETR   FB68   GO     FB63   INITAD FB43   INITDA FB9D   INITKP F93C
INTIME FB00   KBINCH F949   RES    FC00   RUNFLG 0731   SAVDAT FB74
SAVECK FBF0   SAVFLG 070A   SAVSTO 8002   STARTD 0743   STARTU FB1F
TBUFND 07DA   TBUFST 07DB
```

```
                 ****************************************************

                 *                    PGMSAVE

                 * THIS ROUTINE WILL DISABLE AND ENABLE ACCESS
                 * TO THE CMOS MEMORY WHILE THE PORTABLE UNIT
                 * IS POWERED UP.  TO RETAIN MEMORY DURING
                 * POWER OFF, THE SAVE BUTTON MUST BE DEPRESSED

                 ****************************************************

                 * SYSTEM EQUATES

            DE2A ORB      EQU    $DE2A      DATA REGISTER PIA B SIDE
FB7D                      ORG    $FB7D

FB7D CC   7F04   PGMSAV   LDD    #$7F04     DISABLE CMOS MEMORY
FB80 7F   DE2B            CLR    ORB+1      CLEAR CONTROL REGISTER B
FB83 FD   DE2A            STD    ORB        SET B SIDE BIT 7-IN, REST-OUT
FB86 86   3F              LDA    #$3F
FB88 B7   DE2A            STA    ORB        SET BIT 6 LO
FB8B 39                   RTS

FB8C 86   7F     UNSAVE   LDA    #$7F
FB8E B7   DE2A            STA    ORB        SET BIT 6 HI
FB91 39                   RTS

FB92 CC   3F04   EXT      LDD    #$3F04     RETURN TO EXTERNAL CONTROL
FB95 7F   DE2B            CLR    ORB+1      CLEAR CONTROL REGISTER B
FB98 FD   DE2A            STD    ORB        SET B SIDE 2-IN 6-OUT
FB9B 39                   RTS

                          END

0 ERROR(S) DETECTED

SYMBOL TABLE:

EXT    FB92   ORB    DE2A   PGMSAV FB7D   UNSAVE FB8C
```

```
                        *************************************************
                        *                UN/PACK
                        * THIS ROUTINE PACKS FOUR 6 BIT CHARACTERS(ASCII)
                        * INTO THREE 8 BIT BYTES, OR UNPACKS THREE EIGHT
                        * BIT BYTES INTO FOUR BYTES.  RECONVERSION TO ASCII
                        * IS ALSO DONE. ON ENTRY*
                        * FOR PACKING, X POINTS TO THE START OF DATA
                        *              Y POINTS TO THE START OF STORAGE
                        *              D CONTAINS THE NUMBER OF 4 CHAR BLOCKS
                        * FOR UNPACK,  X CONTAINS THE START OF STORAGE
                        *              Y CONTAINS THE START OF CHAR. STORAGE
                        *              D CONTAINS THE NUMBER OF 4 CHAR BLOCKS
                        *****************************************************
FB9C                            ORG     $FB9C
FB9C 4C                 PACKS   INCA            ADJUST COUNT OF MS BYTE
FB9D 34   06                    PSHS    D       PUT COUNT ON THE STACK
FB9F 8D   0A            PAC1    BSR     PACK    PACK 4 INTO 3
FBA1 6A   61                    DEC     1,S     LS COUNT
FBA3 26   FA                    BNE     PAC1
FBA5 6A   E4                    DEC     0,S     MS BYTE
FBA7 26   F6                    BNE     PAC1
FBA9 35   86                    PULS    D,PC    CLEAN UP STACK,RETURN
FBAB EC   81            PACK    LDD     X++     GET FIRST TWO CHARACTERS
FBAD 58                         ASLB
FBAE 58                         ASLB
FBAF 58                         ASLB
FBB0 49                         ROLA
FBB1 58                         ASLB
FBB2 49                         ROLA
                        * ACCA IS NOW PACKED AND ACCB CONTAINS 4 BITS
                        *         LEFT JUSTIFIED
FBB3 A7   A0                    STA     Y+      STORE FIRST PACKED BYTE
FBB5 A6   80                    LDA     X+      GET THIRD CHAR.
FBB7 44                         LSRA
FBB8 44                         LSRA
FBB9 84   0F                    ANDA    #$0F    CLEAN UP MS NYBBLE
                        * ACCA HAS MS NYBBLE AND ACCB HAS LS NYBBLE
FBBB 34   04                    PSHS    B       PUT B IN TEMP STORAGE
FBBD AA   E0                    ORA     S+      PUT CHAR TOGETHER, CLEAN STACK
FBBF A7   A0                    STA     Y+      STORE SECOND PACKED BYTE
FBC1 EC   81                    LDD     X++     GET LAST TWO CHAR.
                        * PICK UP 2 LSB FROM A AS 2 MSB IN B
FBC3 58                         ASLB
FBC4 58                         ASLB
```

```
FBC5 44                  LSRA
FBC6 56                  RORB
FBC7 44                  LSRA
FBC8 56                  RORB
FBC9 E7    A0            STB     Y+         STORE THIRD PACKED BYTE
FBCB 39                  RTS

           ***************************************************

FBCC 34    20    UNPAKS  PSHS    Y
FBCE 4C                  INCA               ADJUST COUNTER MS BYTE
FBCF 34    06            PSHS    D          PUT COUNT ON THE STACK
FBD1 8D    0A    UNP1    BSR     UNPACK     UNPACK 3 INTO 4
FBD3 6A    61            DEC     1,S        LS COUNT
FBD5 26    FA            BNE     UNP1
FBD7 6A    E4            DEC     S          MS COUNT
FBD9 26    F6            BNE     UNP1
FBDB 35    A6            PULS    D,Y,PC     CLEANSTACK,RETURN

           * UNPACK RETURNS FOUR RIGHT-JUSTIFIED 6-BIT
           * CHARACTERS FROM THREE PACKED 8-BIT BYTES

FBDD 34    06    UNPACK  PSHS    D          SAVE D REGISTER
FBDF EC    80            LDD     X+
FBE1 44                  LSRA               :
FBE2 56                  RORB               : 16-BIT SHIFT, TWO PLACES
FBE3 44                  LSRA               :
FBE4 56                  RORB               :

           * ACCA IS NOW UNPACKED

FBE5 54                  LSRB
FBE6 54                  LSRB

           * ACCB IS NOW UNPACKED

FBE7 ED    A1            STD     Y++        STORE 1ST AND 2ND CHARS
FBE9 EC    80            LDD     X+         GET 2ND AND THIRD BYTES
FBEB 58                  ASLB               :
FBEC 49                  ROLA               : ANOTHER SHIFT, TWO PLACES
FBED 58                  ASLB               :
FBEE 49                  ROLA               :
FBEF 84    3F            ANDA    #$3F       CLEAR TOP 2 BITS

           * ACCA IS NOW UNPACKED

FBF1 E6    80            LDB     X+         GET 3RD BYTE AGAIN
FBF3 C4    3F            ANDB    #$3F

           * BOTH ARE NOW UNPACKED

FBF5 ED    A1            STD     Y++        STORE THIRD AND FOURTH CHARS
FBF7 35    86            PULS    D,PC       RECOVER D, RETURN

                         END
```

```
0 ERROR(S) DETECTED

SYMBOL TABLE:

PAC1   FB9F   PACK   FBAB   PACKS  FB9C   UNP1   FBD1   UNPACK FBDD
UNPAKS FBCC
```

NBS-114A (REV. 2-80)

U.S. DEPT. OF COMM. BIBLIOGRAPHIC DATA SHEET (See instructions)	1. PUBLICATION OR REPORT NO. NBSIR 84-2853	2. Performing Organ. Report No.	3. Publication Date April 1984

4. TITLE AND SUBTITLE

Measuring the Rate of Corrosion of Reinforcing Steel in Concrete

5. AUTHOR(S)

E. Escalante, M. Cohen, and A. H. Kahn

6. PERFORMING ORGANIZATION (If joint or other than NBS, see instructions)

NATIONAL BUREAU OF STANDARDS
DEPARTMENT OF COMMERCE
WASHINGTON, D.C. 20234

7. Contract/Grant No.
PO-8-3-0118

8. Type of Report & Period Covered
Interim 1980 - 1983

9. SPONSORING ORGANIZATION NAME AND COMPLETE ADDRESS (Street, City, State, ZIP)

Federal Highway Administration
Paving and Structural Materials Group, HRS-22
Washington, DC 20590

10. SUPPLEMENTARY NOTES

FHWA Contract Manager: Y. P. Virmani

☐ Document describes a computer program; SF-185, FIPS Software Summary, is attached.

11. ABSTRACT (A 200-word or less factual summary of most significant information. If document includes a significant bibliography or literature survey, mention it here)

The progress on a research program directed at developing a nondestructive method for measuring the corrosion of steel in concrete as related to bridge deck deterioration is reported. This report summarizes the past work and describes the new developments on this project. The five phases described are: 1) a literature review, 2) preliminary studies, 3) measurements in concrete, 4) field measurements, and 5) development of a microprocessor system.

12. KEY WORDS (Six to twelve entries; alphabetical order; capitalize only proper names; and separate key words by semicolons)

Bridge deck corrosion; corrosion in concrete; corrosion of steel, polarization technique; rebar corrosion.

13. AVAILABILITY

[X] Unlimited
[ ] For Official Distribution. Do Not Release to NTIS
[ ] Order From Superintendent of Documents, U.S. Government Printing Office, Washington, D.C. 20402.
[X] Order From National Technical Information Service (NTIS), Springfield, VA. 22161

14. NO. OF PRINTED PAGES
86

15. Price
$11.50