252
4-11-66

MASTER

RELEASED FOR ANNOUNCEMENT
IN NUCLEAR SCIENCE ABSTRACTS

IOWA STATE UNIVERSITY

AMES LABORATORY

RESEARCH AND
DEVELOPMENT
REPORT

U.S.A.E.C.

DISCLAIMER

This report was prepared as an account of work sponsored by an agency of the United States Government. Neither the United States Government nor any agency Thereof, nor any of their employees, makes any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government or any agency thereof. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States Government or any agency thereof.

DISCLAIMER

Portions of this document may be illegible in electronic image products. Images are produced from the best available original document.

CFSTI PRICES

H.C. \$3.00 ; MN .75

IS-1225

Mathematics and Computers (UC-32)
TID-4500 (47th Ed.), November 1, 1965

UNITED STATES ATOMIC ENERGY COMMISSION

Research and Development Report

DIAN - A SYSTEM FOR PROGRAMMING
A DIGITAL COMPUTER IN TERMS
OF ANALOG COMPONENTS

By

G. J. Farris and L. E. Burkhart

RELEASED FOR ANNOUNCEMENT

IN NUCLEAR SCIENCE ABSTRACTS

January, 1966

Ames Laboratory
at
Iowa State University of Science and Technology
F. H. Spedding, Director
Contract W-7405 eng-82

IS-1225

This report is distributed according to the category Mathematics and Computers (UC-32) as listed in TID-4500 (47th Ed.), November 1, 1965.

LEGAL NOTICE

This report was prepared as an account of Government sponsored work. Neither the United States, nor the Commission, nor any person acting on behalf of the Commission:

- A. Makes any warranty or representation, expressed or implied, with respect to the accuracy, completeness, or usefulness of the information contained in this report, or that the use of any information, apparatus, method, or process disclosed in this report may not infringe privately owned rights; or
- B. Assumes any liabilities with respect to the use of, or for damages resulting from the use of any information, apparatus, method, or process disclosed in this report.

As used in the above, "person acting on behalf of the Commission" includes any employee or contractor of the Commission, or employee of such contractor, to the extent that such employee or contractor of the Commission, or employee of such contractor prepares, disseminates, or provides access to, any information pursuant to his employment or contract with the Commission, or his employment with such contractor.

Printed in USA. Price \$ 3.00 . Available from the Clearinghouse for
Federal Scientific and Technical Information, National
Bureau of Standards, U. S. Department of
Commerce, Springfield,
Virginia

Table of Contents

Part I PROGRAMMING GUIDE

Introduction	1
Basic Components of the System	2
Preparation of a DIAN Flowsheet.	5
Coding the Input Cards	10
Specification of Output.	16
The Complete Input Deck.	17
Application of the DIAN System to Boundary Value Problems.	18
Error Messages	19

Part II SYSTEM DESCRIPTION

The Source Program	21
Flow Charts.	23
Flow Chart A, Fortran Main Program	33
Flow Chart BA, DDA Subprogram.	34
Flow Chart BB, DDA Subprogram.	36
Flow Chart BC, DDA Subprogram.	37
Flow Chart BD, DDA Subprogram.	38
Flow Chart C, INTERP Subprogram.	40
Flow Chart D, Subroutine for Integrators	41
Flow Chart E, Subroutine for Transport Delay Generators.	42
Flow Chart F, Subroutine for Empirical Function Generators	43

Part III EXAMPLE PROBLEMS

Purpose.	45
Example 1. Initial Value Problem.	45
Example 2. Boundary Value Problem	49
Example 3. Simulation of a Mixer-Settler Extractor.	56

APPENDIX

IBM 7074 Listing	64
Modifications for 20K System	71
IBM 7090 Listing	73

IS-1225

DIAN - A SYSTEM FOR PROGRAMMING A DIGITAL
COMPUTER IN TERMS OF ANALOG COMPONENTS

G. J. Farris* and L. E. Burkhart

ABSTRACT

DIAN (DIgital-ANalog) is a programming system by which a digital computer may be programmed in a manner very similar to the technique of programming an analog computer.

In using DIAN, the programmer need be concerned only with the relationships between inputs and outputs of basic computational components which perform specific mathematical operations.

The system is compatible with the Fortran symbolic language and may be used as part of a larger digital computer program. It may be applied to the more general types of boundary-value problems arising in process simulation, to the initial value problems ordinarily solved on analog computers, to non-linear problems, and to control problems involving large transport delays.

The system may be used to simulate a very large analog computer or it may be considered to be a separate technique of problem solution with its own unique features.

*Present Address: Union Carbide Corporation, Nuclear Division, Oak Ridge,
Tennessee

PART I

PROGRAMMING GUIDE

Introduction

DIAN (DIgital-ANalog) is a programming system by means of which a digital computer may be programmed in a manner very similar to the technique of programming an analog computer.

An electronic analog computer consists of a collection of computational building blocks, or modules, each of which performs a specific mathematical operation. A problem is prepared for solution on an analog computer by drawing a diagram indicating the components required and their interconnections for performing a specific set of mathematical operations.

Similarly, in using DIAN, the programmer need be concerned only with the relationships between inputs and outputs of basic computational components which perform specific mathematical operations. He need not be concerned with the digital computer programming required to accomplish these mathematical operations or with the mechanics of interconnecting the components.

The system may be used to simulate a very large analog computer or it may be considered to be a separate technique of problem solution with its own unique features.

Three versions of the system are described in this manual, one for an IBM 7070 or 7074 computer with 10,000 words of core storage (10K system), one for use on the same type of machine with a 20,000 word core (20 K system), and one for an IBM 7090. All three require two work tapes in addition to the input and output tapes.

Basic Components of the System

The eight basic components of the system are given in Table 1.

The integrators used in DIAN are somewhat different from analog computer integrators in that they use two types of inputs, primary and secondary. If these integrators were completely analogous to analog computer integrators, the primary input would always be the differential of the independent variable. However, the primary input to an integrator in the DIAN system may be the differential of any function of the independent or dependent variables. This arrangement provides one major advantage over analog computers - it makes possible the generation of many nonlinear functions by using combinations of the basic components.

Function multipliers are used to generate the differential of the product of two functions. As with the integrators, one of the functions may consist of the sum of several other functions.

Constant multipliers are analogous to analog computer potentiometers except that the value of the constant is not restricted to values less than one.

Transport delay generators permit an exact representation of time lag due to material transport time. This feature is useful in control problems and is another advantage which the system has over analog computers since it is difficult to obtain satisfactory transport delay without distortion on an analog computer.

The relay, also given in Table 1, may be used for branching during the problem solution by adding or dropping functions, depending on the value of a control variable.

NAME	SYMBOL	INPUT CARD FORMAT	INPUT
1. Integrator	$Y = \int_{t_0}^t (dy_1 + dy_2 + dy_3) + Y_0$	FORMAT(2I5,E15.7,4I5)	1. Component Number 2. Identification Code (1) 3. Initial Value (Y_0) 4. Primary Input 5. Secondary Inputs
2. Constant Multiplier		FORMAT (2I5, 2E15.7,I5)	1. Component Number 2. Identification Code (2) 3. Constant (a) 4. Primary Input
3. Summer		FORMAT (6I5)	1. Component Number 2. Identification Number (3) 3. Secondary Inputs
4. Relay	$dx (Y_L \leq Y \leq Y_U)$ $0 \text{ (all other } Y)$	FORMAT (2I5, 3E15.7, 2I5)	1. Component Number 2. Identification Code (4) 3. Initial Value (Y_0) 4. Lower Limit (Y_L) 5. Upper Limit (Y_U) 6. Primary Input 7. Secondary Input

Table 1. Basic components of the DIAN system.

NAME	SYMBOL	INPUT CARD FORMAT	INPUT
5. Function Multiplier		FORMAT (2I5,2E15.7,4I5)	<ol style="list-style-type: none"> 1. Component Number 2. Identification Code (5) 3. Initial Value (U_0) 4. Initial Value (V_0) 5. Primary Input 6. Secondary Inputs
6. Absolute Value Generator		FORMAT (3I5)	<ol style="list-style-type: none"> 1. Component Number 2. Identification Code (6) 3. Primary Input
7. Transport Delay Generator		FORMAT (4I5)	<ol style="list-style-type: none"> 1. Component Number 2. Identification Code (7) 3. No. of Independent Variable Increments of Delay 4. Primary Input
8. Empirical Function Generator		FORMAT (2I5,E15.7,I5,A5,I5)	<ol style="list-style-type: none"> 1. Component Number 2. Identification Code (8) 3. Initial Value (Y_0) 4. No. of Data Points 5. Interpolation Method (P or L) 6. Secondary Input
O. Independent Variable Generator		FORMAT (I5, 3E15.7, I5)	<ol style="list-style-type: none"> 1. Identification Code (O) 2. Initial Value 3. Final Value 4. Increment Size 5. Increments Between Output Points

Table 1 (Continued). Basic components of the DIAN system.

Empirical function generators, which are represented schematically as shown in Figure 3, permit functional relationships to be defined empirically in terms of discrete data points.

The normal relationships between inputs and outputs of the components are as shown in these figures. However, the sign of the output of any component may be reversed if desired. This is indicated on the diagram by replacing the plus sign by a minus sign.

In addition to these eight basic types of components, each problem requires one or more independent variable generators which have no input but whose output is the increment of the independent variable.

Preparation of a DIAN Flowsheet

A computer diagram showing the components required and their interconnection is prepared in a manner similar to that used for analog computers.

For example, to solve an n^{th} order ordinary differential equation, first solve for the highest order derivative, then multiply by the differential of the independent variable. That is, an equation in the form

$$F(t, y, \frac{dy}{dt}, \dots, \frac{d^n y}{dt^n}) = 0$$

is rearranged to the form

$$d(\frac{d^{n-1}y}{dt^{n-1}}) = G(t, y, \frac{dy}{dt}, \dots, \frac{d^{n-1}y}{dt^{n-1}}) dt$$

Next, assume the differential on the left is available as a secondary input to an integrator which uses dt as the primary input. The output of this integrator, $d(\frac{d^{n-2}y}{dt^{n-2}})$, may be used as a secondary input to another integrator with a primary input of dt to produce $d(\frac{d^{n-3}y}{dt^{n-3}})$.

Continuing in sequence all the derivatives will be generated. The output of the last integrator will be ydt. The term tdt is generated by using dt as both the primary and the secondary input to an integrator. When all terms of the function Gdt have been generated by properly combining the various derivatives, they are combined to form $d(\frac{dn-1y}{dt^{n-1}})$ which is the input for the first integrator.

If special functions are required, they may be generated from combinations of basic components. Examples of the method are shown in Figures 1 and 2. Many different nonlinear functions of the independent and dependent variables may be generated in this manner.

In preparing the flow diagram, the integrator inputs need not be scaled since the digital computer uses floating point arithmetic.

During preparation of the diagram each component is assigned a serial number from 1 to 1400 (500 with the 10K system). Integrators must have serial numbers between 1 and 1000 (350 with the 10K system), and function multipliers and empirical function generators must have serial numbers between 1001 and 1400 (351 and 500 with the 10K system). Independent variable generators must have a serial number of 0, regardless of how many of them are used for the problem. The total number of components possible is 1400 for the 7090 and 20K/7074 versions. For the 10K/7074 version this number is 500. The number of empirical function generators plus transport delay generators is limited to 15 (10 with the 10K system).

A typical DIAN flowsheet showing the component interconnections is given in Figure 3. This particular flowsheet will be developed in greater detail in a later section of the report.

EXPONENTIAL FUNCTION GENERATOR

LOGARITHMIC AND RECIPROCAL FUNCTIONS

VARIABLE TO A POWER n

Figure 1. Examples of non-linear function generation.

SINE - COSINE GENERATOR

TANGENT GENERATOR

Figure 2. Examples of non-linear function generation.

Figure 3. Illustration of a typical DIAN flowsheet.

Coding the Input Cards

For input to the digital computer each of the components is represented by one data card containing (1) the component serial number, (2) a code number identifying the type of component, (3) primary and secondary input specifications consisting of the serial numbers of other components whose output form the input to the component, plus (4) additional information such as initial values for integrators, constants for constant multipliers, and delay times for transport delay generators. Each card is read under a specific FORMAT statement. The FORMAT statement and required coding information for each type of component are shown in Table 1. Typical input cards for each type of component in the DIAN system are shown in Figures 4, 5, 6, and 7.

Integrator cards contain the component number, identification code, initial value, one primary input, and three secondary inputs and are read by the digital computer according to the Fortran statement - FORMAT (2I5, E15.7, 4I5).

Function multipliers use a similar format except that they require two initial values for U_0 and V_0 in that order, where U is the primary input variable and V is the sum of the secondary input variables.

Input cards for constant multipliers contain the component number, identification code, the constant, and one primary input.

Input cards for summers contain component and identification numbers plus four secondary inputs.

Cards for absolute value generators contain only the component and identification numbers plus one primary input.

Transport delay generator cards require component and identification numbers, the number of independent variable increments in the delay time,

COMPONENT NUMBER	IDENTIFICATION (INTEGRATOR)	INITIAL VALUE	PRIMARY INPUT	SECONDARY INPUT	SECONDARY INPUT	SECONDARY INPUT
17	1	5.0000000E-03	0	-16	+0	+0

INPUT CARD FOR
AN INTEGRATOR

Figure 4a. Input card for an integrator.

[illegible]

Figure 4b. Input card for a multiplier.

[illegible]

Figure 6a. Input card for a summer.

[illegible]

Figure 6b. Input card for a relay.

[illegible]

Figure 7a. Input card for a transport delay generator.

[illegible]

Figure 7b. Input card for an empirical function generator.

and one primary input. Up to fifteen transport delay generators may be used for a single problem and the length of the time delay may range from 2 to 100 increments of the independent variable. The increment size may not be changed during the solution of problems involving transport delay generators.

Input cards for relays contain component and identification numbers, the initial value of the control variable, lower and upper limits on the range of the control variable, plus one primary and one secondary input specification.

The secondary input variable is the control variable for the relay (for example, dy in Table 1).

Cards for empirical function generators contain the component number and identification code, the initial value of the input variable, the number of data points to be used with the component, the letters P or L to specify linear or parabolic interpolation between points, and one secondary input. Each empirical function generator card is followed by cards containing the data points in order of increasing value of the input variable, each data point consisting of two numbers, a value of the input variable and a corresponding value of the output variable, according to the specification - FORMAT (4E15.7). An example is shown in Figure 26.

In specifying inputs for components, primary inputs are always positive and secondary inputs are negative except in the following case. Components which may have more than one secondary input require dummy secondary inputs if less than the maximum number are used. Positive zeros may be used for this purpose and they will be ignored by the DIAN system. For example, in Figure 4a the integrator which this card represents required only one secondary input and, consequently, the input card must contain

two dummy inputs. A minus sign before the identification code of any component reverses the sign of its output.

The independent variable generator is a special type of component which has no input and whose output is the differential of the independent variable. Coding cards for these components contain a zero in column five as an identification code and component number. Initial and final values of the independent variable (the integration limits) and the independent variable increment are specified by the next three numbers. The last number specifies the frequency with which output is produced by the program by specifying the number of independent variable increments in each output interval. Up to thirty of these components may be used with a single problem in order to change increment size or output frequency during the computation. This corresponds to breaking up an integral into several parts.

Specification of Output

Output for a problem is specified by listing the serial numbers of the integrators which hold the integrated values of the desired output variables and by supplying a format for printing the output line. At any time during the computation, a particular integrator holds the current value of its secondary input variable. The independent variable is always an output variable and need not be specified. A maximum of forty-seven other output variables may be specified on one to eight input cards, each of which contains six integrator numbers and which are read according to a FORMAT (6I5).

These output variables are printed in the order in which they are listed on the cards (except that the independent variable is always the

first variable printed) according to a Fortran "FORMAT" statement supplied as input with the problem. The output format card contains a seventy character Fortran "FORMAT" statement excluding the word "FORMAT". That is, a left parenthesis is placed in column one and a right parenthesis in column seventy.

The Complete Input Deck

Since several problems may be run sequentially in a multifile run, the first input card is a control card containing the number of problems in the run in columns one through five. This number is right - justified in the number field. This is followed by the individual coding decks for each problem.

The first card of a problem input deck is a title card containing a problem title of up to seventy characters which is reproduced on the first page of output. The letter B as the first character of this title designates a boundary value problem (see the following section). Any other character may be used in position one of this card for initial value problems. The second card is a control card which supplies information needed to read the following cards properly. It contains the total numbers of the various types of components, the number of independent variable generators, and the number of output variables (excluding the independent variable), in the order: integrators, constant multipliers, summers, relays, function multipliers, absolute value generators, transport delay generators, empirical function generators, independent variable generators, and output variables. This card is read according to a FORMAT (10I5) statement.

Following these two cards are the input cards for all of the components. They are grouped according to component type. That is, all of the integrators are grouped together, then all of the constant multipliers, etc., in the order of the identification codes as listed in Table 1. Data cards for empirical function generators are placed immediately following the card for the component with which they are used.

Cards for independent variable generators are included next, followed by the output variable specification cards and the output format card.

If more than one problem is included in the run, the next card would be the title card for the next problem.

This entire input deck is used as input data for the object programs given in the Appendix of this manual.

Application of the DIAN System to Boundary Value Problems

DIAN contains provisions for a subroutine which can be used to extend its applicability to more general types of boundary value problems requiring an interpolation procedure and iterative solution. For such problems a final rather than initial value of one of the variables is known. An additional input card is required to provide data necessary to carry out the iterative solution.

Since the particular components whose initial values are changed after each iteration depend on the nature of the problem, it is not possible to write a single subroutine which would be applicable to all boundary value problems. Therefore, programming of boundary value problems requires rewriting and recompilation of the Fortran subroutine. A sample subroutine for a particular boundary value problem is included as a part of the library system.

In order to avoid recompilation of the main program for each boundary value problem, the main program uses a subroutine called "INTERP" for every problem. When the system is being used for initial value problems, a dummy subroutine is used which, if written in Fortran, would consist of the three statements:

```
SUBROUTINE INTERP
RETURN
END
```

The object deck for this dummy subroutine consists of a title card, one condensed card, and one execute card which can be prepared from examples in a Fortran reference manual without actually compiling the three statements.

For a boundary value problem this dummy subroutine should be replaced by the actual interpolation subroutine for that problem. The total core storage area which may be used for this subroutine is limited to 1100 words for the two large systems (600 with the 10K/7074 system).

Error Messages

Certain error conditions will cause the problem to be terminated automatically and a message typed on the console log sheet.

After each floating point arithmetic operation, a check is made to determine whether the operation has resulted in an attempt to generate a number larger than 10^{51} . If so, an error condition results and the message "Floating Overflow" is typed. This may result from an error in the input coding or from instability of the integration procedure caused by the use of too large an increment size.

When empirical function generators are used, an error condition results if the value of the input variable is outside the range of the

tabulated data. The message "Variable Exceeds Range of Table" will be typed. This may also result from integration instability.

If a read or write operation on one of the two work tapes results in ten consecutive error conditions, the message "Ten Tape Errors" will be typed.

PART II
SYSTEM DESCRIPTION

The Source Program

There are three parts of the source program for the DIAN system. Input and output is handled by a program written in Fortran which serves as the control program. It contains two subprograms, one written in Fortran which is used in conjunction with boundary value problems and one written in Autocoder or MAP which actually performs the computations and comprises the main part of the system. These programs need be compiled only once to produce an object deck in machine language which uses the input decks described previously.

When the input coding cards for the individual components are read by the input portion of the program the numbers are stored sequentially in COMMON storage area. An initialization procedure in the DDA subprogram converts this block of storage into a sequence of machine language instructions which, in effect, establishes the component interconnections and provides linkages with closed subroutines which perform the mathematical operations.

Five blocks of working storage, each containing one storage word for each component, are used in carrying out the computations. The five storage words associated with each component contain the necessary data to describe the status of the component at any time during the computation and to carry out the mathematical operation assigned to the component. These five registers are a Y register which contains the current integrated value of the input variable, an R register used in double precision accumulation of the inputs to a component, a DZ register which contains the

output of the component, and YNM1 and YNM2 registers which hold the previous two values of Y for use with the numerical integration formulas.

A major difference between analog computers and conventional digital differential analyzers or DDA simulators such as DIAN is that all mathematical operations are performed simultaneously on an analog computer (parallel operation) whereas the digital method, using present digital computers, requires that the components be serviced serially.

In performing the computations using DIAN the digital computer works sequentially through the coding block assembled from the input data. For each component in the sequence the first instruction causes a working index to be loaded with the component serial number. The second instruction causes a branch to the appropriate closed subroutine. The remaining storage words for that component are values of indices to be used by the subroutine in determining the locations in the working storage blocks from which to obtain the necessary data for carrying out the mathematical operation. After the computation required to update the component has been completed, the results are stored in the appropriate registers and control is transferred to the instructions for the next component in the sequence.

After all of the components have been updated a check is made to determine whether the integration limit or an output point has been reached. If not, the independent variable is incremented and the servicing procedure repeated.

The numerical integration procedure used by DIAN consists of the following forward integration formulas:

$$y_1 = y_0 + (\Delta x)(y'_0)$$

$$y_2 = y_1 + (\Delta x) \left(\frac{3y'_1}{2} - \frac{y'_0}{2} \right)$$

$$y_{n+1} = y_n + (\Delta x) \left(\frac{23y'_n}{12} - \frac{4}{3}y'_{n-1} + \frac{5}{12}y'_{n-2} \right)$$

The first two formulas are used only for the first two increments in starting the integration procedure. To compensate for the resulting reduced accuracy, the integration increment size is automatically reduced temporarily at the beginning of the problem or after a change has been made in the increment size during computation.

Flow Charts

The operation of the three main sections of DIAN are represented in the following flow charts. Figure 8 represents the Fortran program which processes input and output for the system. The computation subprogram (DDA) is represented in four parts in Figures 9-12. Figure 13 is a flow chart for the Fortran subprogram INTERP which is used with boundary value problems. The last three flow charts, Figures 14-16 represent three of the closed subroutines which are included within the DDA subprogram for carrying out the mathematical operations. These flow charts are for the subroutines used by integrators, transport delay generators, and empirical function generators. Since the subroutines for the other components are relatively simple, flow charts are not given for them.

Each of the flow charts is described in detail in the following pages.

Figure 8. Flow chart for Fortran main program.

Figure 9. Flow chart for first portion of DDA subprogram.

Figure 10. Flow chart for BRANCH portion of DDA subprogram.

Figure 11. Flow chart for integration step control portion of DDA subprogram.

BD

Figure 12. Flow chart for end portion of DDA subprogram.

Figure 13. Flow chart for INTERP subprogram.

D

Figure 14. Flow chart for integrators.

E

Figure 15. Flow chart for transport delay generators.

F

Figure 16. Flow chart for empirical function generators.

Flow Chart A, Figure 8; FORTRAN Main Program

- Block A1: Problem is initialized, control variables read in, and problem title transferred to output.
- Block A2: All component cards are read in and stored in COMMON storage area designated "CODE". Data for empirical function generators are read into proper storage area. All independent variable generator data are read in and stored.
- Block A3: Locations of desired output variables, and output format are read in.
- Block A4: The first independent variable generator is moved into the working area.
- Block A5: Control is transferred to the DDA subprogram, which carries out the computations.
- Block A6: Control may be returned to the main program for several reasons shown in subsequent blocks. In block A6 a check is made to see if an output point has been reached.
- Block A7: Output variables are written out and control transferred back to the DDA subprogram.
- Block A8: A check is made to see whether control was returned to the main program because of an error condition.
- Block A9: If a boundary value problem is being solved, control is transferred to the INTERP subprogram; otherwise this block is skipped.

- Block A10: A check is made to see if the integration limits are to be extended.
- Block A11: A new independent variable generator is moved into the working area.
- Block A12: If another iteration is required for a boundary value problem, the problem is re-started with the first independent variable generator.
- Block A13: A check is made to see if the boundary value problem work tape must be processed.
- Block A14: During each iteration the output for a boundary value problem is stored on a work tape. If a new iteration is required, the tape will be re-wound by the INTERP subprogram and the incorrect data discarded. After the boundary condition has been matched or when an error condition occurs this work tape is processed by the main program to produce the desired output.
- Block A15: If this is a multi-file run then the next problem is started; otherwise the computations are finished.

Flow chart BA, Figure 9 ; DDA Subprogram

- Block BA1: When control is transferred to this subprogram it must determine the current status of the problem solution. The first check is to determine whether this is merely a return after writing an output point.

- Block BA2: After an output data point has been written, control is returned to the block which originally transferred control to the output routine. (BC3 or BD3)
- Block BA3: A check is made to determine if this is the first independent variable generator for the problem.
- Block BA4: If this is not the beginning of the problem, then a check is made to see if the integration increment has been changed.
- Block BA5: If the first independent variable generator is being used, then a check is made for a new problem or a new iteration for a boundary value problem.
- Block BA6: A new problem is initialized.
- Block BA7-: For a new boundary value problem, parts of the subprogram
BA8 involving magnetic tape procedures must be initialized.
- Block BA9: "SETCOMP" subroutine uses the information contained on the component input cards to compile the block of instructions comprising the subroutine "CODE" which establishes linkages with the computational subroutines and, in effect, establishes component interconnections.
- Block BA10-: For the first problem of a run the core storage areas will
BA11 already be zeroed but for subsequent problems in a multi-file run some areas must be reset to zero between problems.
- Block BA12: Output points are determined by the step size and the number of steps in each output interval. This information is obtained from the current independent variable

generator and is reset before computations are started under control of subroutine "CODE" in block BB1.

Flow Chart BB, Figure 10; DDA Subprogram

- Block BB1: "CODE" is a subroutine compiled by an initialization routine for each problem. It consists of a block of machine-language instructions and data representing each of the components. The first instruction loads an index register with the serial number of the component and the second instruction causes a branch to the appropriate computational subroutine. The index containing the component serial number is used by the subroutine in determining the locations within the working storage areas of the five working registers associated with the component. The remaining words in the coding block contain the information required by the subroutine to carry out the computations. The last instruction of "CODE" causes a return to "BRANCH" after all components have been updated.
- Block BB2: "BRANCH" is a branch point which causes control to be transferred to the appropriate routine, depending upon the status of the computations.
- Block BB3: The first pass through "CODE" initializes all of the components.
- Block BB4: The first starting step in the integration procedure uses a reduced increment size and uses the first starting formula.

- Block BB5: On the second step the second integration formula is used.
- Block BB6: The first two integration steps are completed using the reduced increment size and the full integration formula.
- Block BB7: Subsequent integration steps are made using the full increment size.

Flow Chart BC, Figure 11; DDA Subprogram

- Block BC1: To compensate for the reduced accuracy of the starting integration formulas the increment size is reduced to $1/50$ of the full step size specified by the independent variable generator.
- Block BC2: After one hundred of the reduced steps have been taken, the full integration formula is used with the full step size. The initial values of the integrators must be stored on a work tape to be used later as the points YNM2 for the full size steps.
- Block BC3: The initial values of the output variables are written out.
- Block BC4: For the first reduced size integration step the first starting formula is used.
- Block BC5: The independent variable is incremented before control is returned to "CODE" for the next step.
- Block BC6: For the second starting step in the integration the second starting formula is used.

- Block BC7: After the second starting step the full integration formula using the two past points is used with reduced size increments.
- Block BC8: The program branches, depending upon whether the starting procedure is currently in the first or second fifty steps of the integration with the reduced increments.
- Block BC9-: After fifty of the reduced increments have been taken the
BC10 integrated values held in the Y register of the integrators are stored on a work tape to be used later as the points YNM1 for each integrator with the full step size for the integration increments.
- Block BC11-: After one hundred reduced increments have been completed,
12 the increment size is reset to the full value specified by the independent variable generator.
- Block BC13: The full integration formula requires two previous points, YNM1 and YNM2. These points were stored on a work tape for each integrator as the initial values and as the integrated values of the output variables after fifty reduced increment steps. They are now read from the work tape into the YNM1 and YNM2 registers of the integrators to be used in the integration formulas for the full increment size.

Flow Chart BD, Figure 12; DDA Subprogram

- Block BD1: After each integration step a check is made to determine

whether an output point has been reached.

- Block BD2: If an output point has not been reached, the independent variable generator is incremented and control is returned to "CODE" to continue the computations.
- Block BD3: At an output point control is transferred to the output routine.
- Block BD4: After output, control is returned to this block, which reloads the output departure count.
- Block BD5: After each output a check is made to see if the upper limit of integration specified by the IVG (independent variable generator) has been reached.
- Block BD6: Before control is transferred back to the Fortran program, the current and previous values of the control variable for a boundary value problem are moved into a COMMON storage area for use by the INTERP subprogram.
- Block BD7: The settings of electronic switches and the values of certain index words must be stored for use if control is transferred back to the DDA subprogram by the main Fortran program.
- Block BD8: Control is transferred back to the Fortran program to process an output point, to check for a new IVG, to reset for a new iteration, or to initialize and read input for a new problem.
- Block BD9: When an output point has been reached, this block moves the current values of the output variables to the output storage area.

- Block BD10: If a boundary value problem is not being solved, the output point will be processed by the Fortran program and control returned directly to the output departure point.
- Block BD11: If a boundary value problem is being solved, the output point will be stored on a work tape for processing after the problem has been finished.
- Block BD12: Priority tape operations are used for writing the work tape and computations are continued during this output. At the completion of the tape operation, it is checked for validity after which computations are resumed at the point where they were interrupted.

Flow Chart C, Figure 13; INTERP Subprogram

- Block C1: A check is made for a new problem.
- Block C2: For a new problem, the input card with the control parameters is read in.
- Block C3: A check is made to see if the desired boundary has been reached.
- Block C4: If the boundary has not yet been reached, control is returned to the main program for a new IVG.
- Block C5: If the boundary has been reached the work tape is rewound.
- Block C6: The current value of the control variable is compared with the desired value.

- Block C7: If the problem is finished control is returned to the main program to process the output from the work tape.
- Block C8: If another iteration is required then a check is made to see if the iteration just finished was the first iteration.
- Block C9: For the second iteration a second assumed initial value of the control variable is used.
- Block C10: For subsequent iterations the results of the past two iterations are used to linearly interpolate for a new initial value for the control variable.
- Block C11: Initial values of the components are reset where necessary. The statements required to do this depend upon the problem. This makes it necessary to rewrite and recompile this subrouting for each boundary value problem (see Section III, Example problem 2).
- Block C12: The solution may not converge to the desired boundary value. Therefore, if, after a specified number of iterations, the desired value has not been reached the problem is discontinued and the output from the last iteration is processed.
- Block C13: If another iteration is required, control is transferred back to the Fortran program to restart the problem with the new initial conditions.

Flow Chart D, Figure 14; Subroutine for Integrators

- Block D1: Two passes are made through each of the components in updating

them. A check is made here for the first or second pass.

Block D2: For the first pass, a check is made for the initialization step.

Block D3: In the initialization step the initial value for each integrator, which was specified on its input card, is stored as the current value of the integrated variable.

Block D4: During the first pass in updating an integrator the output of the integrator is computed from the current value of the integrated variable and the two preceeding values.

Block D5: On the second pass the new value of the integrated variable is obtained by summing the inputs for the component and adding this sum to the last integrated value.

Block D6: In updating the components control is switched back and forth between the individual instruction blocks in "CODE" and the computation subroutines. After an integrator has been updated it transfers control to the first instruction of the next component block in "CODE".

Flow Chart E, Figure 15; Subroutine for Transport Delay Generators

Block E1: Each transport delay generator uses a separate block of 100 storage words. The location of the proper storage area for this particular component is determined.

Block E2: A check is made for the first or second pass through the component.

- Block E3-: During the first pass the location of the output value
E4 within the storage block is determined from the delay time specified for the component, and this value is moved to the output register of this delay generator.
- Block E5: On the second pass the input to the delay generator is stored in the location from which the output value was moved during the first pass. This value will become the output after the number of time increments specified for the delay.
- Block E6: Control is returned to the first instruction of the next component in "CODE".

Flow Chart F, Figure 16; Subroutine for Empirical Function Generators

- Block F1: A check is made for the first or second pass through the component.
- Block F2: Each EFG (empirical function generator) uses a specific block of 100 storage words. This routine locates the proper storage area.
- Block F3: Either linear or parabolic interpolation may be used between the data points. This routine obtains the point preceeding the desired value and two points following it from the table.
- Block F4: Linear or parabolic interpolation is used to determine the value of the output variable corresponding to the current value of the input variable.

- Block F5: The previous value of the output variable is subtracted from the current value to obtain the output differential.
- Block F6: On the second pass the inputs to the component are summed and added to the previous value of the input variable to obtain its new value.
- Block F7: Control is returned to the next component instructions in "CODE".

PART III

EXAMPLE PROBLEMS

Purpose

The problems described in this section are quite simple and of little importance in themselves. They are elementary problems designed to illustrate programming techniques of the DIAN system.

Example 1. Initial Value Problem

The two equations given in Figure 17 represent mass and energy balances during the transient operation of a continuous stirred-tank chemical reactor in which a first order exothermic reaction is taking place. The heat of reaction is removed by water circulating through cooling coils in the tank. The variables C and T represent concentration and temperature. The independent variable is time.

An initial value problem based on this system consists of determining the temperature and concentration as a function of time from start-up to steady-state, when the initial temperature and concentration are known.

Generation of the non-linear differential function $Ce^{-E/RT}$ is shown in the top portion of Figure 18. This is then combined with the diagrams for solution of the two simultaneous ordinary differential equations to produce the complete DIAN flowsheet.

Figure 19 is an input listing prepared from the flowsheet in Figure 18. Each line represents a card to be punched to form the complete input deck for solution of the problem. Since several problems may be run sequentially in a multifile run, the first card is an overall control card containing the number of problems in the run.

$$dC = (a_1 - b_1C - K_1C e^{-\frac{E}{RT}}) dt$$

$$dT = (a_2 - b_2T + K_2C e^{-\frac{E}{RT}}) dt$$

Figure 17. Diagram and equations for stirred-tank chemical reactor with first order, exothermic reaction.

Figure 18. DIAN flowsheet for stirred-tank reactor problem.

STATEMENT NUMBER		Cont.	FORTRAN STATEMENT														
1	5	6	7	10	15	20	25	30	35	40	45	50	55	60	65	70	72
1																	
CONCENTRATION AND TEMPERATURE IN A STIRRED TANK REACTOR																	
6			7	2	0	1	0	0	0	1	2						
1			1	3	333333E-03	11	-2	0	0								
2			-1	3	333333E-03	1	-2	0	0								
4			1	4	0785030E-17	3	-4	0	0								
5			1	2	0392515E-19	0	-1001	0	0								
12			1	3	0000000E+02	0	-11	0	0								
17			1	5	0000000E-03	0	-16	0	0								
3			-2	1	1323600E+04	2											
8			2	7	8600000E-13	5											
9			2	1	7373737E+00	0											
10			-2	5	6780000E-03	12											
13			-2	7	8600000E+12	5											
14			2	2	5000000E-05	0											
15			-2	5	0000000E-03	17											
11			3	-8	-9	-10	0										
16			3	-13	-14	-15	0										
10 01			5	4	0785030E-17	5	0000000E-03	4	-16	0	0						
0			0	0000000E+00	3	0000000E+03	1	0000000E+01	30								
17			12	0	0	0	0										
(7HOTIME =E15.7,18X15HCNCONCENTRATION, =E15.7,18X13HTEMPERATURE, =E15.7)																	

Figure 19. Input listing for initial value stirred-tank reactor problem.

The first card in the problem input deck (and the second card in Figure 19) is the title card containing a problem title of up to seventy characters which is reproduced on the first page of output. The letter B as the first character of this title designates a boundary value problem. Any other character in this position would signify an initial value problem.

The next card is a control card which contains the total numbers of the various types of components, in order of increasing identification code numbers, plus the number of independent variable generators and the number of output variables.

Following these two cards are the input cards for all the components, grouped according to component type and in order of increasing identification code numbers.

Cards for independent variable generators come next, followed by the output variable specification cards and the output format card. If more than one problem were included in the run, the next card would be the title card for the next problem.

This entire input package is used as input data for the DIAN object program deck. Output for the problem is shown in Figure 20. The solution required 30 seconds on the IBM 7074.

Example 2. Boundary Value Problem

The stirred-tank reactor problem of Example 1 may be used to illustrate the use of the DIAN system for solving boundary value problems. In this case suppose that one wishes to determine the initial concentration which would produce a given steady-state concentration. Although the differential equations reduce to algebraic equations at steady-state,

they cannot be solved explicitly for temperature and concentration since the equations are transcendental.

The problem can be solved using the DIAN system by solving the differential equations as a boundary value problem. Figure 21 is the listing for the subprogram INTERP which is used in solving this problem. The input listing for the problem is shown in Figure 22. The input deck is very similar to that used for the initial value problem of Example 1 except that several independent variable generators are used since it is necessary to check periodically to see whether steady-state has been reached. This is done by the INTERP subprogram, which is entered only at the end of the integration range of an independent variable generator. Also, one additional input card is required by the subprogram.

For a boundary value problem the control variable is specified by listing it as the first output variable. In this case, concentration is specified as the control variable by listing integrator 17 as the location of the first output variable.

In this example a steady-state concentration of 5.0×10^{-3} is desired. To start the problem, two initial values are assumed for the concentration. One of these (5.0×10^{-3}) is used on the input cards as the initial value for integrator number 17, as the constant for constant multiplier number 15 and as one of the initial values for function multiplier number 1001. It is also used indirectly for the constant in constant multiplier number 14 since the constant a_1 is a function of the initial concentration.

The second assumed initial concentration (5.5×10^{-3}) is specified on the last input card, along with the desired steady-state concentration (5.0×10^{-3}), plus limits for use in determining when steady-state has been reached and when the boundary condition has been met. For measuring

CONCENTRATION AND TEMPERATURE IN A STIRRED TANK REACTOR

TIME = 0.000000E 00	CONCENTRATION = 0.500000E-02	TEMPERATURE = 0.300000E 03
TIME = 0.300000E 03	CONCENTRATION = 0.4635067E-02	TEMPERATURE = 0.3048973E 03
TIME = 0.600000E 03	CONCENTRATION = 0.4473963E-02	TEMPERATURE = 0.3057898E 03
TIME = 0.900000E 03	CONCENTRATION = 0.4425432E-02	TEMPERATURE = 0.3059524E 03
TIME = 0.120000E 04	CONCENTRATION = 0.4413106E-02	TEMPERATURE = 0.3059820E 03
TIME = 0.150000E 04	CONCENTRATION = 0.4410234E-02	TEMPERATURE = 0.3059874E 03
TIME = 0.180000E 04	CONCENTRATION = 0.4409597E-02	TEMPERATURE = 0.3059884E 03
TIME = 0.210000E 04	CONCENTRATION = 0.4409461E-02	TEMPERATURE = 0.3059885E 03
TIME = 0.240000E 04	CONCENTRATION = 0.4409433E-02	TEMPERATURE = 0.3059886E 03
TIME = 0.270000E 04	CONCENTRATION = 0.4409427E-02	TEMPERATURE = 0.3059886E 03
TIME = 0.300000E 04	CONCENTRATION = 0.4409426E-02	TEMPERATURE = 0.3059886E 03

Figure 20: Output for Example Problem 1.

SUBROUTINE INTERP

C

```

    DIMENSION IVG(5), CODE(7500), YOUT(48), OUT(48), FMT(14),
1  TITLE(14), N(150), D(1500), B(7005), MAP(7500)
    COMMON CODE, IVG, NCOMP, NOUT, NEFG, NTDG, NABS, NMUL, NREL,
1  NSUM, NCON, NINT, YOUT, OUT, XNM1, X, M7, M6, I2, I1, D, B,
2  TITLE, M5, N, FMT
    EQUIVALENCE (CODE, MAP)
1  FORMAT (4E15.7, I5)
    IF(I1 - LASTI1) 22, 4, 2
2  M9 = 1
    M4 = 0
    READ INPUT TAPE 5, 1, XI2, XFIN, DX, EPS, MAX
    LASTI1 = I1
4  DELTA = ABSF(X - XNM1)
5  IF(DELTA - EPS) 8, 8, 6
6  M6 = 1
    RETURN
8  REWIND 3
    D = ABSF(X - XFIN)
    IF(D - DX) 20, 20, 10
10 GO TO (14, 12), M9
12 XI2 = XI1 + ((XI1 - XI2) * (XFIN - XF1))/(XF1 - X)
14 M9 = 2
    XF1 = X

```

C

```

    THE FOLLOWING STATEMENTS DEPEND ON THE PROBLEM
    XI1 = CODE(38)
    CODE(38) = XI2
    CODE(69) = XI2
    CODE(86) = XI2
    CODE(65) = XI2/200.0

```

C

```

    M4 = M4 + 1
    IF(MAX - M4) 20, 20, 18
18 M6 = 2
    RETURN
20 M6 = 3
    RETURN
22 STOP
    END

```

Figure 21. Card listing for subprogram INTERP.

C FOR COMMENT																	
STATEMENT NUMBER	5	6	7	10	15	20	25	30	35	40	45	50	55	60	65	70	72
1																	
1																	
	BOUNDARY VALUE VERSION OF THE STIRRED TANK REACTOR PROBLEM																
6		7		2		0		1		0		0		8		2	
1		1		3		3333333E-03		11		-2		0		0			
2		-1		3		3333333E-03		1		-2		0		0			
4		1		4		0785030E-17		3		-4		0		0			
5		1		2		0392515E-19		0		-1001		0		0			
12		1		3		0000000E+02		0		-11		0		0			
17		1		5		0000000E-03		0		-16		0		0			
3		-2		1		1323600E+04		2									
8		2		7		8600000E-13		5									
9		2		1		7373737E+00		0									
10		-2		5		6780000E-03		12									
13		-2		7		8600000E+12		5									
14		2		2		5000000E-05		0									
15		-2		5		0000000E-03		17									
11		3		-8		-9		-10		0							
16		3		-13		-14		-15		0							
1001		5		4		0785030E-17		5		0000000E-03		4		-16		0	
0		0		0		0000000E+00		5		0000000E+02		5		0000000E+00		50	
0		5		0		0000000E+02		1		0000000E+03		1		0000000E+01		25	
0		1		0		0000000E+03		1		5000000E+03		1		0000000E+01		25	
0		1		5		0000000E+03		2		0000000E+03		1		0000000E+01		25	
0		2		0		0000000E+03		2		5000000E+03		1		0000000E+01		25	
0		2		0		0000000E+03		3		0000000E+03		1		0000000E+01		25	
0		3		0		0000000E+03		3		5000000E+03		2		0000000E+01		20	
0		3		5		0000000E+03		4		0000000E+03		2		0000000E+01		20	
17		12		0		0		0		0							
(7H OTIME =E15. 7, 18 x 15HC ONCENTRATION, =E15. 7, 18 x 13HT EMPERATURE, =E15. 7)																	
5		50		0		000E-03		5		0000000E-03		1		0000000E-08		1	
		0		0		0000000E-10										10	

Figure 22. DIAN input listing for boundary value problem involving stirred-tank reactor.

the approach to steady-state, the current value of the control variable and its value at the end of the preceding integration step are automatically supplied to the INTERP subprogram by the DDA subprogram. The input data card for this problem also contains the number 10 which specifies that, if the boundary condition has not been met after 10 iterations, the problem should be terminated and the results of the last iteration printed out.

After two steady-state solutions have been computed using the two assumed initial concentrations, a linear interpolation procedure is used to compute a new trial initial concentration.

Before starting each new iteration, the initial values of the components mentioned above (integrator 17, constant multiplier 15, function multiplier 1001, constant multiplier 14) must be reset with the new initial values. The numbers which must be changed before starting the next iteration are located somewhere in the storage block called CODE. To determine the location of an initial value for a given component, the following formulas may be used:

(1) Initial Values of Integrators

$$I = (7 \times NPI) + 3$$

where: NPI = number of integrators preceding the
desired integrator in the input deck

I = the location within CODE, i.e., CODE (I)
is the symbolic storage location of the
desired initial value.

(2) Constants for Constant Multipliers

$$I = (7 \times NINT) + (4 \times NPC) + 3$$

where: NINT = total number of integrators in the input
deck

NPC = number of constant multipliers preceeding
this component in the input deck

(3) Initial Values for Relay Control Variables

$$I = (7 \times NINT) + (4 \times NCON) + (6 \times NSUM) + (7 \times NPR) + 3$$

where: NCON = total number of constant multipliers in
the input deck

NSUM = total number of summers in the input deck

NPR = number of relays preceeding this component
in the input deck.

(4) Initial Values for Function Multipliers

$$I = (7 \times NINT) + (4 \times NCOM) + (6 \times NSUM) + (7 \times NREL) + \\ (8 \times NPR) + K$$

where: NREL = total number of relays in the input deck

NPM = number of function multipliers preceeding
this component in the input deck

$K = 3$ for U_o

$K = 4$ for V_o

(5) Initial Values for Empirical Function Generators

$$I = (7 \times NINT) + (4 \times NCON) + (6 \times NSUM) + (7 \times NREL) \\ + (8 \times NMUL) + (3 \times NABS) + (4 \times NTDG) + (6 \times NPE)$$

where: NMUL = total number of function multipliers in the
input deck

NABS = total number of absolute value generators in the
input deck

NTDG = total number of transport delay generators in the
input deck

NPE = number of empirical function generators preceeding
this component in the input deck.

This example boundary value problem represents one form in which a boundary condition might occur; that is, as the steady-state value of some variable. Another way in which a boundary condition occurs is as a specified value of a control variable at a given value of the independent variable (e.g. temperature at the wall of a pipe). The Fortran program for the INTERP subprogram given in Figure 21 may be modified easily for such a problem. The input variable EPS would be replaced by TFIN (the final value of the independent variable) and statements 4 and 5 would be replaced by:

```
4      Y = IVG(3)
5      IF (TFIN - Y) 8,8,6
```

The subprogram INTERP as listed in Figure 21 is for use with the 20K system. To use it with the 10K system the DIMENSION statement must be changed to correspond to the DIMENSION statement of the main program of the 10K system.

The output for Example Problem 2 is given in Figure 23.

Example 3. Simulation of a Mixer-Settler Extractor

The DIAN system can be used for a variety of simulation problems. Its use in the simulation of countercurrent extraction equipment illustrates the use of the transport delay generator and the arbitrary function generator.

In the mixer-settler extractor shown in Figure 24 the entering liquid streams are mixed intimately in a mixing chamber and flow co-currently to a settling chamber where the two immiscible phases separate and are drawn

BOUNDARY VALUE VERSION OF THE STIRRED TANK REACTOR PROBLEM

TIME = 0.000000E 00	CONCENTRATION = 0.1528728E-01	TEMPERATURE = 0.3000000E 03
TIME = 0.250000E 03	CONCENTRATION = 0.5247677E-02	TEMPERATURE = 0.3045361E 03
TIME = 0.500000E 03	CONCENTRATION = 0.5017152E-02	TEMPERATURE = 0.3056331E 03
TIME = 0.750000E 03	CONCENTRATION = 0.5003718E-02	TEMPERATURE = 0.3058985E 03
TIME = 0.100000E 04	CONCENTRATION = 0.5000897E-02	TEMPERATURE = 0.3059628E 03
TIME = 0.125000E 04	CONCENTRATION = 0.5000215E-02	TEMPERATURE = 0.3059784E 03
TIME = 0.150000E 04	CONCENTRATION = 0.5000050E-02	TEMPERATURE = 0.3059822E 03
TIME = 0.175000E 04	CONCENTRATION = 0.5000010E-02	TEMPERATURE = 0.3059831E 03
TIME = 0.200000E 04	CONCENTRATION = 0.5000000E-02	TEMPERATURE = 0.3059833E 03
TIME = 0.225000E 04	CONCENTRATION = 0.4999998E-02	TEMPERATURE = 0.3059834E 03
TIME = 0.250000E 04	CONCENTRATION = 0.4999997E-02	TEMPERATURE = 0.3059834E 03

Figure 23. Output for Example Problem 2.

off. In the process a solute material crosses the interface from one phase to the other as, for example, uranium might be preferentially extracted from an impure acid solution of dissolved uranium ore by an immiscible organic solvent.

For one of the simpler models of a mixer-settler, one assumes ideal stage behavior, completely immiscible liquids, constant flow rates, and negligible volume changes. If plug flow is assumed for the light phase as it passes through the settler, then this portion of the unit may be represented by a pure time delay. If perfect mixing is assumed in the mixing section and also for the heavy phase as it passes through the settler, then the equations given in Figure 24 represent a single stage extractor.

The equilibrium relation between X and Y (solute concentrations in the heavy and light phases, respectively) may be represented by experimental data points using an empirical function generator.

The DIAN simulation of a single stage extractor is shown in Figures 25 and 26.

To simulate a multistage extractor, many of these single stage simulation programs are used together to form a model of a countercurrent extractor. If the diagram in Figure 24 were a typical stage, n , of a multistage extractor, the input heavy phase would be labeled X_{i-1} instead of X_F . The input light phase would be Y_{i+1} instead of Y_0 . The inputs to constant multipliers 1 and 2 would be $Y_{i+1}dt$, and the constants would be L/v_H and H/v_H .

The DIAN system is particularly useful for such a problem. The large number of transport delays may be obtained easily. The equilibrium relationship can be expressed in terms of an array of experimental data points. The

Figure 24. Diagram of a single stage mixer-settler extractor.

Figure 25. DIAN simulation of a single stage mixer-settler extractor.

assumption of 100 per cent stage efficiency or of even constant stage efficiency is not necessary if such information is known as a function of time or position in the extractor. Flow rates may also be known functions of time.

If very many time or position varying parameters are used with a large multistage extractor -- say 50 or more stages, the size of the problem becomes overwhelming in terms of analog equipment, especially non-linear equipment, and then the use of DIAN is most convenient.

Also, a large linear model of an extractor or other multistage system programmed on an analog computer becomes significantly larger when non-linear parameters are considered. At this stage the problem can be moved from the analog computer to the digital computer without significant re-programming by use of the DIAN system.

A P P E N D I X

IBM 7074 Listing

FORTTRAN Main Program for 10K System

```

C      DIAN      FORTRAN MAIN PROGRAM - 10K SYSTEM
C
      DIMENSION IVG(5), CODE(3000), YOUT(48), OUT(48), FMT(14),
1  TITLE(14), N(150), D(1000), B(2505), MAP(3000)
      COMMON CODE, IVG, NCOMP, NOUT, NEFG, NTDG, NABS, NMUL, NREL,
1  NSUM, NCON, NINT, YOUT, OUT, XNM1, X, M7, M6, I2, I1, D, R,
2  TITLE, M5, N, FMT
      EQUIVALENCE (CODE, MAP)
1  FORMAT (2I5, E15.7, 4I5)
3  FORMAT (2I5, E15.7, I5)
5  FORMAT (6I5)
7  FORMAT (2I5, 3E15.7, 2I5)
9  FORMAT (I5, 3E15.7, I5)
11 FORMAT (I5)
13 FORMAT (14A5)
15 FORMAT (2I5, E15.7, I5, A5, I5)
17 FORMAT (3I5)
19 FORMAT (4I5)
21 FORMAT (10I5)
23 FORMAT (1H1, 25X14A5)
25 FORMAT (4E15.7)
27 FORMAT (2I5, 2E15.7, 4I5)
      READ INPUT TAPE 5, 11, NPROB
      DO5411=1, NPROB
      REWIND 2
      REWIND 3
      READ INPUT TAPE 5, 13, (TITLE(L1), L1=1, 14)
      WRITE OUTPUT TAPE 10, 23, (TITLE(L1), L1=1, 14)
      READ INPUT TAPE 5, 21, NINT, NCON, NSUM, NREL, NMUL, NABS, NTDG,
1  NCFG, NIVG, NOUT
      NCOMP = NINT + NCON + NSUM + NREL + NMUL + NABS + NTDG + NCFG
      K0 = NINT * 7
      K1 = K0 + 1
      K2 = K0 + (NCON * 4)
      K3 = K2 + 1
      K4 = K2 + (NSUM * 6)
      K5 = K4 + 1
      K6 = K4 + (NREL * 7)
      K7 = NOUT + 1
      K8 = 5 * NIVG
      K9 = K6 + 1
      K10 = K6 + (NMUL * 8)
      K11 = K10 + 1
      K12 = K10 + (NABS * 3)
      K13 = K12 + 1
      K14 = K12 + (NTDG * 4)
      K15 = K14 + 1
      K16 = K14 + 6
      K19 = (100 * NTDG) - 77
      IF(NINT) 56, 4, 2
2  READ INPUT TAPE 5, 1, (CODE(K), K=1, K0)
4  IF(NCON) 56, 8, 6
6  READ INPUT TAPE 5, 3, (CODE(K), K=K1, K2)
8  IF(NSUM) 56, 12, 10
10 READ INPUT TAPE 5, 5, (CODE(K), K=K3, K4)
12 IF(NREL) 56, 16, 14
14 READ INPUT TAPE 5, 7, (CODE(K), K=K5, K6)
16 IF(NMUL) 56, 20, 18
18 READ INPUT TAPE 5, 27, (CODE(K), K = K9, K10)
20 IF(NABS) 56, 24, 22
22 READ INPUT TAPE 5, 17, (CODE(K), K = K11, K12)
24 IF(NTDG) 56, 28, 26
26 READ INPUT TAPE 5, 19, (CODE(K), K = K13, K14)

```

```

28 IF(NEFG) 56, 34, 30
30 DO32I4=1, NEFG
   READ INPUT TAPE 5, 15, (CODE(K), K=K15, K16)
   K19 = K19 + 100
   K20 = K19 - 1 + (2 * MAP(K16 - 2))
   READ INPUT TAPE 5, 25, (D(K), K=K19, K20)
   K15 = K15 + 6
32 K16 = K16 + 6
34 READ INPUT TAPE 5, 9, (N(K), K=1, K8)
   READ INPUT TAPE 5, 5, (YOUT(J1), J1 = 1, NOUT)
   READ INPUT TAPE 5, 13, (FMT(L3), L3=1, 14)
36 DO46I2=1, NIVG
   DO38I3=1, 5
   L2 = (5 * I2) - 5 + 13
38 IVG(I3) = N(L2)
40 CALL DDA
   GO TO (42, 44, 48, 50, 46), M5
42 WRITE OUTPUT TAPE 10, FMT, (OUT(L4), L4 = 1, K7)
   GO TO 40
44 CALL INTERP
   GO TO (46, 36, 50), M6
46 CONTINUE
48 GO TO 54
50 REWIND 3
   DO52I4=1, M7
   READ TAPE 3, (OUT(L4), L4=1, K7)
   WRITE OUTPUT TAPE 10, FMT, (OUT(L4), L4=1, K7)
52 CONTINUE
54 CONTINUE
56 STOP
   END

```

DDA Subprogram for 10K System (written in Autocoder)

AA00 EXECUTE	CNTRL7	DDA SUBPROGRAM	AB26	BE RESET
AA02	DC		AB28	ZST1 TEMP+3
AA04		ADDA @	AB30	ZA1 +PRIORITY
AA06		+0+3+325+0+9100000001	AB32	STD1 BRANCHCOMD(6,9)
AA08 ORIGIN	CNTRL325,COUNTA		AB34	ZA1 BRANCHCOMD
AA10	XZA 98,20		AB36	ZST1 159
AA12	RS 98,1WSTORAGE		AB38	ZA1 +TRPRIORITY
AA14	RG 98,SUBSTORE		AB40	STD1 BRANCHCOMD(6,9)
AA16	ZA1 156		AB42	ZA1 BRANCHCOMD
AA18	ZST1 TEMP		AB44	ZST1 158
AA20	ZA1 TEMP+1		AB46	CD TITLE(0),6
AA22	ZST1 102		AB48	BE **2
AA24	ZA1 TEMP+2		AB50	B **3
AA26	ZST1 101		AB52	CD TITLE(1),2
AA28	BES 12,0+X27		AB54	BE **3
AA30	CD 12(9),1		AB56	ESN 11
AA32	BE DIAN		AB58	B **13
AA34	B EXTENDY		AB60	ESF 11
AA36 OUTPUT	ZA1 Y		AB62	ZA1 +23
AA38	ZST1 OUT		AB64	S1 NOUT
AA40	XL 60,YOUT		AB66	BM1 **3
AA42	XA 60,1		AB68	ESN 6
AA44	XLIN 61,YOUT-1+X60		AB70	B **4
AA46	ZA1 Y+X61		AB72	ESF 6
AA48	ZST1 OUT+X60		AB74	A1 OUTRDW(2,5)
AA50	BIX 60,**-3		AB76	STD1 OUTRDW(6,9)
AA52	ESN 12		AB78	ZA1 OUTAREARDW(2,5)
AA54	ESF 13		AB80	S1 NOUT
AA56	BES 11,HALT2+9		AB82	STD1 OUTAREARDW(6,9)
AA58	ESF 5		AB84	ZA1 +AA
AA60	TSM 12		AB86	STD1 BLXCOMD(6,9)
AA62	PTW 12,OUTAREARDW		AB88	XL 47,+7
AA64	BES 5,*		AB90	XL 53,+0
AA66	BIX 50,0+X27		AB92	ZA1 XZACOMD
AA68	B 0+X27		AB94	ZA2 BLXCOMD
AA70	B EFGI		AB96 SETCOMP	ZA3 47(5)
AA72	B DELAYM		AB98	STD3 *(5)+1
AA74	B ABSM		AC00	ZA3 CONST(0)
AA76	B MULM		AC02	STD3 *(9)+7
AA78	B DSNM		AC04	ZA3 NINT+X47
AA80	B ADDM		AC06	BZ3 **7
AA82	B CONM		AC08	XL 52,9993
AA84	B INTM		AC10	XA 52,1
AA86 AA	B BRANCH-1		AC12	AS1 CODE+X53
AA88	B INTP		AC14	AS2 CUVE+1+X53
AA90	B CONP		AC16	XA 53,7
AA92	B ADDP		AC18	BIX 52,**-3
AA94	B DSNP		AC20	BIX 47,SETCOMP
AA96	B MULP		AC22	ZST1 CODE+X53
AA98	B ABSP		AC24	ZST2 CODE+1+X53
AB00	B DELAYP		AC26	XL 80,+10010
AB02	B EFGP		AC28	ZA1 +340
AB04	BSF 1,**+2		AC30	C1 NCOMP
AB06 BRANCH	B WW		AC32	BL RESET
AB08	XL* 66,+0		AC34	ZA1 YRDW(2,5)
AB10	B CODE		AC36	S1 NCOMP
AB12 DIAN	ESN 3		AC38	STD1 YRDW(6,9)
AB14	ESF 4		AC40	ZA1 YNM1RDW(2,5)
AB16	ESF 8		AC42	S1 NCOMP
AB18	ESF 9		AC44	STD1 YNM1RDW(6,9)
AB20	XL 50,+0		AC46	ZA1 YNM2RDW(2,5)
AB22	ZA1 11		AC48	S1 NCOMP
AB24	C1 TEMP+3		AC50	STD1 YNM2RDW(6,9)

AC52	RESET	ZA1	IVG+3	AD78	XS	70,1	
AC54		ZST1	TEM+3	AD80	PR	0+X70	
AC56		ZA1	BWW	AD82	CORRECT	XL	80,+10010
AC58		ZST1	BRANCH	AD84	PR	++1	
AC60		TRW	11	AD86	BES	6,++3	
AC66		XL	29,VCOMP	AD88	TW	12,OUTROW	
AC68		ZA1	NMUL	AD90	PR	++1	
AC70		A1	NEFG	AD92	TSM	12	
AC72		XL	69,9991	AD94	XL	80,+10010	
AC74		ZA1	NTDG	AD96	BSF	5,0+X87	
AC76		SL1	2	AD98	TRPRIORITY	TRB	11
AC78		XL	68,9991	AE00	BIX	80,++2	
AC80		XA	68,1	AE02	B	RETRY-4	
AC82		ZA1	+0	AE04	XL	71,161	
AC84		ZST1	CB2	AE06	MSP	71	
AC86		ZST1	CB3	AE08	XS	71,1	
AC87		ZST1	R+X29	AE10	PR	0+X71	
AC88		ZST1	DZ+X29	AE12	TW11PRI	TRB	11
AC90		BIX	29,+-2	AE14	TSK	11	
AC91		ZST1	R+350+X69	AE16	B	TRPRIORITY+1	
AC92		ZST1	DZ+350+X69	AE18	FIRST50	ESF	10
AC94		BIX	69,+-2	AE20	TWZ	11,YRDW	
AC96		ZA2	NTDG	AE22	TRW	11	
AC98		BZ2	++3	AE24	ESN	7	
AD00		ZST1	D-1+X68	AE26	XL	23,+48	
AD02		BIX	68,+-1	AE28	TEST	BIX	28,GOON
AD04	BB	ZA1	ACS	AE30	BLX	27,OUTPUT	
AD06		ZST1	CB1	AE32	XL	28,IVG+4	
AD08		ZA1	IVG+1	AE34	XA	28,1	
AD10		ZST1	Y	AE36	ZA1	DZ	
AD12		XL	28,IVG+4	AE38	BM1	ENDRANGE+1	
AD14		XA	28,1	AE40	ZA1	Y	
AD16		ZA1	+9	AE42	FM	ZZ	
AD18		STD1	112(0)	AE44	ZA2	IVG+2	
AD20		PC	PRICONWORD	AE46	C2	9991	
AD22		B	GOON+5	AE48	BH	GOON	
AD24	EXTENDY	ESF	3	AE50	ENDRANGE	B	HALT2
AD26		ESN	4	AE52	ZA1	Y	
AD28		ZA1	IVG+3	AE54	FM	ZZ	
AD30		S1	TEM+3	AE56	C1	IVG+2	
AD32		BZ1	++2	AE58	B	ENDRANGE-1	
AD34		B	RESET	AE60	BIX	23,GOON	
AD36		ZA1	IVG+1	AE62	BSF	8,++5	
AD38		FA	DZ	AE64	XS	23,1	
AD40		ZST1	Y	AE66	ESF	7	
AD42		B	EXTENDY-6	AE68	ESN	8	
AD44	PRIORITY	ZA3	0+X99	AE70	B	++3	
AD46		BSN	5,++2	AE72	BES	10,FIRST50	
AD48		XL	87,97	AE74	B	SECOND50	
AD50		CD	9993(1),2	AE76	GOON	ZA1	DZ
AD52		BE	CORRECT	AE77	FA	R	
AD54		BIX	80,RETRY	AE78	FA	Y	
AD56		PR	++1	AE80	ZST1	Y	
AD58		TYP	TAPE	AE81	ZST2	R	
AD60		NUP		AE82	ESF	2	
AD62		B	HALT1+3	AE84	ESN	1	
AD64	RETRY	TRB	12	AE86	BV2	++1	
AD66		ISK	12	AE88	OV1	BRANCH+1	
AD68		ZA3	162	AE90	B	BRANCH+1	
AD70		CD	9993(0),0	AE92	HW	ESF	3
AD72		BE	CORRECT+4	AE94	ZA1	IVG+3	
AD74		MSP	9993	AE96	ZA2	+0	
AD76		XL	70,9993	AE98	FD	F50	

AF00	ZST1 DZ	AG26	FA TEM
AF02	ZA1 +TW11PRI	AG28	FBV HALT1
AF04	STD1 BRANCHCMD(6,9)	AG30 DD	FM DZ+X22
AF06	ZA1 BRANCHCMD	AG32	FBV HALT1
AF08	ZST1 156	AG34	ZST1 DZ+X21
AF10	ZA1 +6	AG36	ESN 1
AF12	STD1 111(0)	AG38	BSF 2,++2
AF14	TWZ 11,YROW	AG40	B ++3
AF16	BES 4,++2	AG42	ZS1 DZ+X21
AF18	BLX 27,OUTPUT	AG44	ZST1 DZ+X21
AF20	ESN 7	AG46	B CC+2
AF22	ZA1 BFF	AG48 CC	ESN 1
AF24	ZST1 BRANCH	AG50	ESF 2
AF26	B GOON	AG52	ZA1 2+X24
AF28 FF	ZA1 BCS	AG54	BM1 ++3
AF30	ZST1 CB1	AG56	BZ1 ++2
AF32	ZA1 BCS+1	AG58	B 2+X24
AF34	ZST1 CB2	AG60	XSN 24,0
AF36	ZA1 BMH	AG62	BIX 24,CC+2
AF38	B FF-2	AG64	B CC+2
AF40 HH	XL 26,++2	AG66 INLZT	ZA1 0+X24
AF42	ZA1 CCS+X26	AG68	ZST1 Y+X21
AF44	ZST1 CB1+X26	AG70	B CC
AF46	BIX 26,--2	AG72 CONM	ESN 2
AF48	ESN 10	AG74 CONP	BSF 1,++2
AF50	ZA1 BGMB	AG76	B CC+1
AF52	ZST1 BRANCH	AG78	BES 3,INLZT
AF54	XL 23,++46	AG80	XLIN 22,1+X24
AF56	B GOON	AG82	ZA1 Y+X21
AF58 SECOND50	ZA1 BTEST	AG84	B DD
AF60	ZST1 BRANCH	AG86 MULM	ESN 2
AF62	ZA1 IVG+3	AG88 MULD	BES 1,++2
AF64	ZST1 DZ	AG90	B TT
AF66	ESN 9	AG92	BES 3,VV
AF68	ZA1 +8	AG94	XLIN 22,2+X24
AF70	STD1 111(0)	AG96	ZA1 Y+X21
AF72	TR 11,YNM2RDW	AG98	FS YNM2+X21
AF74	TR 11,YNM1RDW	AH00	ZST1 YNM2+X21
AF76 BTEST	B TEST	AH02	FA Y+X21
AF78 BWH	B WH	AH04	FBV HALT1
AF80 BFF	B FF	AH06	FM DZ+X22
AF82 BMH	B HH	AH08	ZST1 DZ+X21
AF84 BGMB	B GOON-8	AH10	ZA1 YNM2+X21
AF86 INTM	ESN 2	AH12	FM YNM1+X21
AF88 INTP	BSF 1,++2	AH14	FBV HALT1
AF90	B INPSM	AH16	FA DZ+X21
AF92	BES 3,INLZT	AH18	BIX 24,DD+1
AF94	XLIN 22,1+X24	AH19	B DD+1
AF96	ZA1 YNM2+X21	AH20 TT	XLIN 22,2+X24
AF98	FM CB3	AH22	ZA1 DZ+X22
AG00	FBV HALT1	AH24	FA YNM1+X21
AG02	ZST1 TEM	AH26	FBV HALT1
AG04	ZA1 YNM1+X21	AH28	ZST1 YNM1+X21
AG06	ZST1 YNM2+X21	AH30	ZA1 Y+X21
AG08	FM CB2	AH32	ZST1 YNM2+X21
AG10	FBV HALT1	AH34	BIX 24,INPSM
AG12	FA TEM	AH36 VV	B INPSM
AG14	FBV HALT1	AH38	ZA1 0+X24
AG16	ZST1 TEM	AH40	ZST1 YNM1+X21
AG18	ZA1 Y+X21	AH42	BIX 24,INLZT
AG20	ZST1 YNM1+X21	AH44	B INLZT
AG22	FM CB1		ESF 2
AG24	FBV HALT1		ZS1 2+X24

AH46	BMI	EE	AI64	FS	3+X98
AH48	XLIN	22,9991	AI66	FD	TEM+4
AH50	ZA1	DZ+X22	AI68	FBV	HALT1
AH52	FA	R+X21	AI70	FS	TEM+5
AH54	FBV	HALT1	AI72	FD	TEM+1
AH56	ZST1	R+X21	AI74	FBV	HALT1
AH58	XSN	24,0	AI76	ZST1	TEM+4
AH60	BIX	24,INPSM	AI78	ZA1	Y+X21
AH62	B	INPSM	AI80	FS	2+X98
AH63	EE	BZ1 *-2	AI82	FM	TEM+4
AH64	ZA1	Y+X21	AI84	FA	TEM+5
AH65	FA	R+X21	AI86	PP	ZST1 TEM+1
AH66	FBV	HALT1	AI88	ZA1	Y+X21
AH67	ZST1	Y+X21	AI90	FS	0+X98
AH68	ZST2	R+X21	AI92	FM	TEM+1
AH69	B	2+X24	AI94	FA	1+X98
AH70	EFGM	ESN 2	AI96	FBV	HALT1
AH72	EFGP	BES 1,*+2	AI98	B	0+X56
AH74	B	EE-3	AJ00	EMPINIT	A1 +D
AH76	ZA1	66(2,5)	AJ02		ZST1 YNM2(1,5)+X21
AH78	SL1	2	AJ04		A1 1+X24
AH80	XLIN	55,9991	AJ06		A1 1+X24
AH82	BES	3,EMPINIT	AJ08		S1 +1
AH84	XSN	98,2	AJ10		STD1 YNM2(6,9)+X21
AH86	ZA3	Y+X21	AJ12		MSM YNM2+X21
AH88	C3	D+X55	AJ14		ZA1 0+X24
AH90	BL	HALT3	AJ16		ZST1 Y+X21
AH92	LEH	YNM2+X21	AJ18		B EFGP+6
AH94	B	HALT3	AJ20	ADDM	ESN 2
AH96	ZA1	1+X98	AJ22	ADDW	BSF 1,*+2
AH98	C3	0+X98	AJ24		B CC+1
AI00	BE	*+2	AJ26		ZA1 +0
AI02	BLX	56,SLOPE	AJ28		ZST1 DZ+X21
AI04	BES	3,*+5	AJ30	GG	ZS1 0+X24
AI06	ZST1	TEM+2	AJ32		BMI LL
AI08	FS	YNM1+X21	AJ34		XLIN 22,9991
AI10	ZST1	DZ+X21	AJ36		ZA1 DZ+X22
AI12	ZA1	TEM+2	AJ38		FA DZ+X21
AI14	ZST1	YNM1+X21	AJ40		FBV HALT1
AI16	XA	66,1	AJ42		ZST1 DZ+X21
AI18	XA	24,1	AJ44		XSN 24,0
AI20	B	CC+1	AJ46		BIX 24,GG
AI22	SLOPE	XS 98,2	AJ48		B GG
AI24	ZS2	98(2,5)	AJ50	LL	BZ1 *-2
AI26	S2	+5	AJ52		XSN 24,0
AI28	C2	YNM2(6,9)+X21	AJ54		XS 24,3
AI30	BL	HALT3	AJ56		BIX 24,DD+3
AI32	ZA1	2+X98	AJ58		B DD+3
AI34	FS	0+X98	AJ60	ABSM	ESN 2
AI36	ZST1	TEM+1	AJ62	ABSP	XS 24,1
AI38	ZA1	3+X98	AJ64		BSF 1,*+2
AI40	FS	1+X98	AJ66		B CC+1
AI42	FD	TEM+1	AJ68		XLIN 22,1+X24
AI44	FBV	HALT1	AJ70		ZA1 DZ+X22
AI46	CD	2(9)+X24,3	AJ72		MSP 9991
AI48	BE	PP	AJ74		B DD+2
AI50	ZST1	TEM+5	AJ76	DSNM	ESN 2
AI52	ZA1	4+X98	AJ78	DSNP	BSF 1,*+3
AI54	FS	2+X98	AJ80		XA 24,2
AI56	ZST1	TEM+4	AJ82		B INPSM
AI58	FA	TEM+1	AJ84		BES 3,MM
AI60	ZST1	TEM+1	AJ86		ZA1 1+X24
AI62	ZA1	5+X98	AJ88		CI Y+X21

AJ90	BH	ZERO	AL20	ZST1	TEMP+1
AJ92	ZA1	2+X24	AL22	ZA1	101
AJ94	C1	Y+X21	AL24	ZST1	TEMP+2
AJ96	BL	ZERO	AL26	XZA	98,20
AJ98	XLIN	22,3+X24	AL28	BCB	1,*
AK00	ZA1	DZ+X22	AL30	ZA1	TEMP
AK02	XA	24,2	AL32	ZST1	156
AK04	B	DD+2	AL34	RS	98,SUBSTORE
AK06 ZERO	ZA1	+0	AL36	RG	98,IWSTORAGE
AK08	B	*-3	AL38	ZA1	+0
AK10 MM	ZA1	0+X24	AL40	STD1	112(0)
AK12	XA	24,2	AL42	ZA1	+3
AK14	B	INLZT+1	AL44	BES	13,*+8
AK18 DELAYM	ESN	2	AL46	ZA1	+1
AK20 DELAYP	BES	3,IVIT	AL48	BES	12,*+4
AK22	XL	65,Y+X21	AL49	ZA1	+5
AK24	XA	66,1	AL50	BES	11,*+2
AK26	BES	1,KK	AL51	ZA1	+2
AK28	XLIN	22,1+X24	AL52	ZST1	M5
AK30	ZA1	DZ+X22	AL54	B	0+X94
AK32	BES	9,*+5	AL56	BES	11,*-2
AK34	FA	D-1+X65	AL58	ZA1	+4
AK36	ZST1	D-1+X65	AL60	B	*-4
AK38	BES	8,*+3	AL62 HALT3	TYP	OUTRANGE
AK39	B	CC+1	AL64	NOP	
AK40	ZST1	D-1+X65	AL66	B	HALT1+2
AK42	BIX	65,*+2	AL68 OUTAREARDW	DRDW	OUT,OUT+23
AK44	XL	65,YNM2+X21	AL70 OUTRDW	DRDW	OUT+24,OUT+47
AK46	XU	65,Y+X21	AL72 YRDW	DRDW	Y+1,Y+350
AK48	B	CC+1	AL74 YNMIRDW	DRDW	YNM1+1,YNM1+350
AK50 KK	BES	7,CC+1	AL76 YNM2RDW	DRDW	YNM2+1,YNM2+350
AK52	BES	8,*+3	AL78 IWSTORAGE	DRDW	-*+2,*+11
AK54	ZA1	D-1+X65	AL80 SUBSTORE	DRDW	-*+11,*+20
AK56	B	DD+2	AL82	DA	20
AK58	BIX	65,*-2	AL84		0,9
AK60	B	CC+1	AL86 OVERFLOW	DC	-RDW
AK66 INIT	BES	1,NN	AL88		@FLOATING OVERFLOW@
AK68	ZA1	66(2,5)	AL90 OUTRANGE	DC	-RDW
AK70	SL1	2	AL92		@VARIABLE EXCEEDS RANGE OF TABLE@
AK72	ZA2	0+X24	AL94 TAPE	DC	-RDW
AK74	A2	9991	AL96		@TEN TAPE ERRORS@
AK76	A1	+1	AL98	DC	
AK78	SL1	4	AM00 CONST		+7467834600
AK80	A1	9992	AM02 BRANCHCMD		+0100000000
AK82	ZST1	Y+X21	AM04 PRICONWORD		+1111110011
AK84	ZST1	YNM2+X21	AM06 XZACOMD		+4600210000
AK86 NN	XA	66,1	AM08 BLXCOMD		+0200240000
AK88	B	CC+1	AM10 ZZ		+5110000001
AK90 HALT1	TYP	OVERFLOW	AM12 F50		+5,0F+1
AK92	NOP		AM14 CB1		+0F
AK94	BLX	27,OUTPUT	AM16 CB2		+0F
AK96	ESN	13	AM18 CB3		+0F
AK98	B	*+2	AM20 ACS		+1F
AL00 HALT2	ESF	13	AM22 BCS		+15F-1,-5F-1
AL02	ESF	12	AM24 CCS		+1.9166667F,-1.3333333F,+.41666667F
AL04	ZA1	50(1,5)	AM26 TEMP		+0F,+0F,+0F,+0F
AL06	ZST1	M7	AM28 TEM		+0F,+0F,+0F,+0F,+0F,+0F
AL08	XLIN	75,YOUT	AM30 M5	EQU	3338
AL10	ZA1	Y+X75	AM32 TITLE	EQU	3339
AL12	ZST1	X	AM33 R	EQU	3353
AL14	ZA1	YNM1+X75	AM34 OZ	EQU	3854
AL16	ZST1	XNM1	AM36 Y	EQU	4355
AL18	ZA1	102	AM38 YNM2	EQU	4856

```

AM40 YNMI      EQU  5357
AM42 D         EQU  5858
AM44 I1        EQU  6858
AM46 I2        EQU  6859
AM48 M6        EQU  6860
AM50 M7        EQU  6861
AM52 X         EQU  6862
AM54 XNMI      EQU  6863
AM56 OUT       EQU  6864
AM58 YOUT      EQU  6912
AM60 NINT      EQU  6960
AM62 NCON      EQU  6961
AM64 NSUM      EQU  6962
AM66 NREL      EQU  6963
AM68 NMUL      EQU  6964
AM70 NABS      EQU  6965
AM72 NTDG      EQU  6966
AM74 NEFG      EQU  6967
AM76 NOUT      EQU  6968
AM78 NCOMP     EQU  6969
AM80 IVG       EQU  6970
AM82 CODE      EQU  6975
AM84 END       CNTRL325

```

Modifications for 20K System

The listings given for the IBM 7074/10K version of DIAN may be modified quite easily to give the 20K version, which has a maximum of 1400 components.

The Fortran part of the system (Figure 33) is modified by making the following changes in the DIMENSION statement: CODE(7500), D(1500), B(7005), MAP(7500).

More extensive changes are required in the DDA subprogram to accomplish the expansion to 1400 components. Some of the changes correspond to the above changes in the dimension and location of the arrays in Common storage in the add-storage mode. One instruction must be added to the program:

```

AH81      XA      55,10000

```

Next, the instructions listed below must be changed as shown:

```

AA42      XA      60,10001
AB88      XL      47,+100000007
AB90      XZA     53,10000
AC28      ZA1     +990

```

AC80		XA	68,10001
AC91		ZST1	R+1000+X69
AC92		ZST1	DZ+1000+X69
AK76		A1	+10001
AL72	YRDW	DRDW	Y+1,Y+1000
AL74	YNM1RDW	DRDW	YNM1+1,YNM1+1000
AL76	YNM2RDW	DRDW	YNM2 + 1, YNM2 + 1000

Finally, to match locations of variables in the two programs, the variables in Common storage must be given new addresses by changing the instructions as shown in Table 2.

Table 2. Variable addresses for 20K system.

Number	Variable	Operation	Address
AM30	M5	EQU	3823
AM32	TITLE	EQU	3824
AM33	R	EQU	3838
AM34	DZ	EQU	5239
AM36	Y	EQU	6640
AM38	YNM2	EQU	8041
AM40	YNM1	EQU	9442
AM42	D	EQU	10843
AM44	I1	EQU	12343
AM46	I2	EQU	12344
AM48	M6	EQU	12345
AM50	M7	EQU	12346
AM52	X	EQU	12347
AM54	XNM1	EQU	12348
AM56	OUT	EQU	12349
AM58	YOUT	EQU	12397
AM60	NINT	EQU	12445
AM62	NGON	EQU	12446
AM64	NSUM	EQU	12447
AM66	NREL	EQU	12448
AM68	NMUL	EQU	12449
AM70	NABS	EQU	12450
AM72	NTDG	EQU	12451
AM74	NEFG	EQU	12452
AM76	NOUT	EQU	12453
AM78	NCOMP	EQU	12454
AM80	IVG	EQU	12455
AM82	CODE	EQU	12460

IBM 7090 Listing

FORTRAN Main Program

```

C  DIAN MAIN PROGRAM FOR INPUT-OUTPUT PROCESSING          7090 FORTRAN IV
  DIMENSION  IVG(5), CODE(7500), MAP(7500), YOUT(48),OUT(48),Y(140),
  1,FMT(13), TITLE(13),N(150),D(1500),R(1401),YNM1(1401),YNM2(1401),
  2 DZ(1401)
  COMMON  M2, FMT, N, M5, TITLE, R, DZ, Y, YNM2, YNM1,
  1 0, I1, I2, M6, M7, X, XNM1, OUT,
  2 YOUT, NINT, NCON, NSUM, NREL, NMUL, NABS,
  3NTDG, NEFG, NOUT, NCOMP, IVG, CODE
  EQUIVALENCE (CODE, MAP)
  1 FORMAT (2I5, E15.7, 4I5)
  3 FORMAT (2I5, E15.7, 15)
  5 FORMAT (6I5)
  7 FORMAT (2I5, 3E15.7, 2I5)
  9 FORMAT (15, 3E15.7, 15)
 11 FORMAT (15)
 13 FORMAT (13A6)
 15 FORMAT (2I5, E15.7, 15, A5, 15)
 17 FORMAT (3I5)
 19 FORMAT (4I5)
 21 FORMAT (10I5)
 23 FORMAT (1H1, 25X13A6)
 25 FORMAT (4E15.7)
 27 FORMAT (2I5, 2E15.7, 4I5)
 29 FORMAT (33H0 VARIABLE EXCEEDS RANGE OF TABLE)
 31 FORMAT (19H0 FLOATING OVERFLOW)
  READ (5, 11) NPROB
  DO68I1=1, NPROB
  REWIND 9
  REWIND 11
  READ (5, 13)(TITLE(I1), I1=1, 13)
  READ (5, 21) NINT, NCON, NSUM, NREL, NMUL, NABS, NTDG, NEFG,
  1 NIVG, NOUT
  NCOMP = NINT + NCON + NSUM + NREL + NMUL + NABS + NTDG + NEFG
  K0 = NINT * 7
  K1 = K0 + 1
  K2 = K0 + (NCON * 4)
  K3 = K2 + 1
  K4 = K2 + (NSUM * 6)
  K5 = K4 + 1
  K6 = K4 + (NREL * 7)
  K7 = NOUT + 1
  K8 = 5 * NIVG
  K9 = K6 + 1
  K10 = K6 + (NMUL * 8)
  K11 = K10 + 1
  K12 = K10 + (NABS * 3)
  K13 = K12 + 1
  K14 = K12 + (NTDG * 4)
  K15 = K14 + 1
  K16 = K14 + 6
  K19 = (100 * NTDG) - 99
  IF(NINT) 70, 4, 2
  2 READ (5, 1)(CODE(K), K=1, K0)
  4 IF(NCON) 70, 8, 6
  6 READ (5, 3)(CODE(K), K=K1, K2)
  8 IF(NSUM) 70,12, 10
 10 READ (5,5)(CODE(K), K=K3, K4)
 12 IF(NREL) 70,16, 14
 14 READ (5, 7)(CODE(K), K=K5, K6)
 16 IF(NMUL) 70,20, 18
 18 READ (5, 27)(CODE(K), K=K9, K10)
 20 IF(NABS) 70, 24, 22

```

```

22 READ (5, 17)(CODE(K), K=K11, K12)
24 IF(NTDG) 70, 28, 26
26 READ (5, 19)(CODE(K), K=K13, K14)
28 IF(NEFG) 70, 34, 30
30 DO32I4=1, NEFG
   READ (5, 15)(CODE(K), K=K15, K16)
   K19 = K19 + 100
   K20 = K19 - 1 + (2 * MAP(K16-2))
   READ (5, 25)(D(K), K=K19, K20)
   K15 = K15 + 6
32 K16 = K16 + 6
34 READ (5, 9)(N(K), K=1, K8)
   READ (5, 5)(YOUT(J1), J1=1, NOUT)
   READ (5, 13)(FMT(L3), L3=1, 13)
   WRITE (6, 23)(TITLE(L1), L1=1, 13)
   IF(NCOMP.GT.990) GO TO 36
   K17 = NCOMP
   GO TO 30
36 K17 = 1001
38 DO60I2=1, NIVG
   DO40I3=1, 5
   L2 = (5 * I2) - 5 + I3
40 IVG(I3) = N(L2)
42 CALL DDA
   M7 = M7
   GO TO (44, 58, 62, 64, 60, 46, 48, 52, 54, 56), M5
44 WRITE (6, FMT)(OUT(L4), L4=1, K7)
   GO TO 42
46 WRITE (11)(OUT(L4), L4=1, K7)
   GO TO 42
48 WRITE (9)(Y(L5), L5=2, K17)
   GO TO (42, 50), M2
50 REWIND 9
   GO TO 42
52 READ (9)(YNM2(L5), L5=2, K17)
   READ (9)(YNM1(I5), L5=2, K17)
   REWIND 9
   GO TO 42
54 WRITE (6, 29)
   GO TO 42
56 WRITE (6, 31)
   GO TO 42
58 CALL INTERP
   GO TO (60, 38, 64), M6
60 CONTINUE
62 GO TO 68
64 REWIND 11
   DO66I4=1, M7
   READ (11)(OUT(L4), L4=1, K7)
   WRITE (6, FMT)(OUT(L4), L4=1, K7)
66 CONTINUE
68 CONTINUE
70 STOP
   END

```

DDA Subprogram (written as MAP)

```

*      DDA SUBPROGRAM FOR DIAN
DDA    ENTRY DDA
      SAVE (1,2,4)1
      LDI TEMP
      CLA 8
      STO TEMP+1
      CLA HALT1
      STO 8
      STZ 0
      OFT S12
      TRA 127
      CLA 12
      SUB =1
      TZE DIAN
      TRA EXTY
OUTPUT CLA Y
      STO OUT
      LXA NOUT,1
      PXA ,1
      PAC ,4
      CLA YOUT-1,4
      PAC ,2
      CLA Y,2
      STO OUT,4
      TIX *-6,1,1
      OSI S12
      RIS S13
      OFT S11
      TRA HALT2+8
      CLA M7
      ADD =1
      STO M7
      CLA =6
RET     STO M5
      STI TEMP
      CLA TEMP+1
      STO 8
      RETURN DDA,0
      TRA EFGM
      TRA DELAYM
      TRA ABSM
      TRA MULM
      TRA DSNM
      TRA ADDM
      TRA CONM
      TRA INTM
AA      TRA BR-1
      TRA INTP
      TRA CONP
      TRA ADDP
      TRA DSNP
      TRA MULP
      TRA ABSP
      TRA DELAYP
      TRA EFGP
      ONT S1
BR      TRA MW
      RIS S1
      STZ 166
      TRA CODE
DIAN    OSI S3
      RIS S4
      RIS S8
      RIS S9
      STZ M7
      CLA 11
      SUB TEMP+3
      TZE RESET
      ADD TEMP+3
      STO TEMP+3

```

```

CLA TITLE
ARS 30
SUB =022
TZE **3
OSI S11
TRA **2
RIS S11
CLA =0
STT 127
SETCMP PSE 97
CLA =0777777777777777
PAX ,2
LXA CONST+4,1
CLA NINT+8,1
PAX ,4
TZE **11
CLA CODE-1,2
ADD AXTCOM
STO CODE-1,2
CLA CODE,2
ADD TSXCOM
STO CODE,2
PAX ,2
SUB CONST+8,1
PAX ,2
TIX *-9,4,1
TIX *-13,1,1
MSE 97
TRA **6
STZ CODE-1,2
STZ CODE,2
TRA *-15
TSXCOM TSX AA,4
AXTCOM AXT 0,1
RESET CLA IVG+3
      STO TEM+3
      CLA BWW
      STO BR
      STZ CB2
      STZ CB3
      LXA NCOMP,1
      PXA ,1
      PAC ,2
      STZ R,2
      STZ DZ,2
      TIX *-4,1,1
      STZ R
      STZ DZ
      CLA NMUL
      ADD NEFG
      PAX ,1
      PXA ,1
      PAC ,2
      STZ R+1000,2
      STZ DZ+1000,2
      TIX *-4,1,1
      NZT NTDG
      TRA BD
      LDQ =100
      MPY NTDG
      XCA
      PAX ,1
      PXA ,1
      PAC ,2
      STZ D-1,2
      TIX *-3,1,1
BR      CLA ACS
      STO CB1
      CLA IVG+1
      STO Y

```

```

      LXA IVG+4,1
      SXA I28,1
EXTY  TRA GOON+5
      RIS S3
      OSI S4
      CLA IVG+3
      SUB TEM+3
      TZE **2
      TRA RESET
      CLA IVG+1
      FAD DZ
      STO Y
      TRA EXT-3
FIR50  RIS S10
      CLA =2
      STO M2
      AXT **5,1
      SXA I27,1
      USI S12
      LLA =7
      TRA RET
      LXA =48,1
      OSI S7
TEST   SXA I23,1
      LXA I28,1
      PXA ,1
      SUB =1
      TZE **3
      STA I28
      TRA GOON
      AXT **3,1
      SXA I27,1
      TRA OUTPUT
      LXA IVG+4,1
      SXA I28,1
      CLA DZ
      TMI ENDR+2
      LDQ Y
      FMP =1.0000001E+0
      CAS IVG+2
      NOP
CNDR   TRA HALT2
      TRA GOON
      LDQ Y
      FMP =1.0000001E+0
      CAS IVG+2
      TRA GOON
      NOP
      TRA HALT2
GM8    LXA I23,1
      PXA ,1
      TZE **4
      SUB =1
      STA I23
      TRA GOON
      OFT S8
      TRA **4
      RIS S7
      OSI S8
      TRA GOON
      RIS S8
      ONT S10
      TRA SEC50
      TRA FIR50
GOON   CLA DZ
      FAD R
      FAD Y
      STO Y
      STQ R
      RIS S2
      OSI S1
      TRA BR+2

```

```

      CLA IVG+3
      FDP =5.0E+1
      STQ DZ
      CLA =1
      STO M2
      AXT **2,1
      TRA FIR50+4
      OFT S4
      TRA **4
      AXT **3,1
      SXA I27,1
      TRA OUTPUT
      OSI S7
      CLA BFF
      STO DR
      TRA GOON
      LLA BCS
      STN CB1
      CLA BCS+1
      STU CB2
      CLA BHH
      TRA FF-2
      CLA CCS
      STO CB1
      CLA CCS+1
      STO CB2
      CLA CCS+2
      STO CB3
      OSI S10
      CLA BGM8
      STO BR
      CLA =46
      STA I23
      TRA GOON
      CLA BTEST
      STO BR
      LLA IVG+3
      STO DZ
      OSI S9
      CLA =8
      AXT **4,1
      SXA I27,1
      OSI S12
      TRA RET
      TRA TEST
      BWW TRA WW
      BFF TRA FF
      BHH TRA HH
      BGM8 TRA GM8
      INTM OSI S2
      INTP PXA ,1
      PAC ,1
      ONT S1
      TRA INPSM
      OFT S3
      TRA INLZT
      CLA 2,4
      PAC ,2
      LDQ YNM2,1
      FMP CB3
      STO TEM
      LDQ YNM1,1
      STQ YNM2,1
      FMP CB2
      FAD TEM
      STO TEM
      LDQ Y,1
      STQ YNM1,1
      FMP CB1
      FAD TEM
      STO TEM
      LDQ TEM

```


DD FMP DZ,2
STO DZ,1
ONT S2
TRA *+5
CLS DZ,1
STO DZ,1
CC OSI S1
RIS S2
CLA 3,4
TMI *+3
TZE *+2
TRA 3,4
PXA ,4
SUB =1
PAX ,4
TRA CC+2
INLZT CLA 1,4
STO Y,1
TRA CC
CONM OSI S2
CONP PXA ,1
PAC ,1
ONT S1
TRA CC+1
OFT S3
TRA INLZT
CLA 2,4
PAC ,2
LDQ Y,1
TRA DD
MULM OSI S2
MULP PXA ,1
PAC ,1
ONT S1
TRA TT
OFT S3
TRA VV
CLA 3,4
PAC ,2
CLA Y,1
FSB YNM2,1
STO YNM2,1
FAD Y,1
XCA
FMP DZ,2
STO DZ,1
LDQ YNM2,1
FMP YNM1,1
FAD DZ,1
STO DZ,1
PXA ,4
SUB =1
PAX ,4
TRA DD+2
TT CLA 3,4
PAC ,2
CLA DZ,2
FAD YNM1,1
STO YNM1,1
CLA Y,1
STO YNM2,1
TRA EE-4
VV CLA 1,4
STO YNM1,1
CLA 2,4
STO Y,1
RIS S2
TRA CC+6
INPSM RIS S2
CLS 3,4
TMI EE
PAC ,2

CLA DZ,2
FAD R,1
STO R,1
PXA ,4
SUB =1
PAX ,4
TRA INPSM
EE TZE *-4
CLA Y,1
FAD R,1
STO Y,1
STQ R,1
TRA 3,4
ADDM OSI S2
ADDP PXA ,1
PAC ,1
ONT S1
TRA CC+1
STZ DZ,1
GG CLS 1,4
TMI LL
PAC ,2
CLA DZ,2
FAD DZ,1
STO DZ,1
PXA ,4
SUB =1
PAX ,4
TRA GG
LL TZE *-4
PXA ,4
ADD =2
PAX ,4
TRA DD+2
ABSM OSI S2
ABSP PXA ,1
PAC ,1
PXA ,4
ADD =1
PAX ,4
ONT S1
TRA CC+1
CLA 2,4
PAC ,2
CLA DZ,2
SSP
TRA DD+1
DSNM OSI S2
DSNP PXA ,1
PAC ,1
OFT S1
TRA *+4
PAX ,4
SUB =2
TRA EE-2
OFT S3
TRA MM
CLA 2,4
CAS Y,1
TRA ZERO
NOP
CLA Y,1
CAS 3,4
TRA ZERO
NOP
CLA 4,4
PAC ,2
CLA DZ,2
STO DZ,1
PXA ,4
SUB =2
TRA LL+3

ZERO STZ DZ,1
 TRA *-4
 MM CLA 1,4
 STO Y,1
 RIS S2
 TRA *-8
 EFGM OSI S2
 EFGP PXA ,1
 PAC ,1
 ONT S1
 TRA EE-4
 ONT S3
 TRA TLU
 CLA 2,4
 ALS 1
 SUB =2
 STO TEM+6
 ALS 18
 STU 180
 LDQ 166
 MPY =100
 XCA
 ADD TEM+6
 STO TEM+6
 AXT D,2
 PXA ,2
 ADD TEM+6
 STA 180
 CLA 180
 STO YNM2,1
 CLA 1,4
 STO Y,1
 TLU CLA YNM2,1
 STO 180
 SXA 131,4
 LXD 180,2
 CLA* 180
 CAS Y,1
 TRA FOUND
 TRA FNDEQU
 TIX *-4,2,2
 TRA HALT3
 FOUND LXD 180,4
 PXA ,4
 STO TEM+6
 PXA ,2
 SUB TEM+6
 TZE HALT3
 OSI S5
 PXA ,2
 STO TEM+6
 LXA 180,4
 PXA ,4
 SUB TEM+6
 SUB =2
 PAC ,4
 ONT S5
 TRA FNDEQU+2
 CLA 2,4
 FSB 0,4
 STO TEM+1
 CLA 3,4
 FSB 1,4
 FDP TEM+1
 LXA 131,2
 CLA 3,2
 ALS 21
 ARS 27
 SUB =043
 TZE PP
 STQ TEM+5
 CLA 4,4

FSB 2,4
 STO TEM+4
 FAD TEM+1
 STO TEM+1
 CLA 5,4
 FSB 3,4
 FDP TEM+4
 XCA
 FSR TEM+5
 FDP TEM+1
 STQ TEM+4
 CLA Y,1
 FSB 2,4
 XCA
 FMP TEM+4
 FAD TFM+5
 XCA
 STQ TEM+1
 CLA Y,1
 FSB 0,4
 XCA
 FMP TEM+1
 FAD 1,4
 OFT S3
 TRA +=5
 STO TEM+2
 FSB YNM1,1
 STO DZ,1
 CLA TEM+2
 STO YNM1,1
 CLA 166
 ADD =1
 STO 166
 LXA 131,2
 PXA ,2
 SUB =1
 PAX ,4
 TRA CC+1
 FNDEQU RIS S5
 TRA FOUND+7
 CLA 3,4
 TRA PP+6
 DELAYM OSI S2
 DELAYP PXA ,1
 PAC ,1
 CLA 166
 ADD =1
 STO 166
 ONT S3
 TRA NN
 OFT S1
 TRA CC+1
 LDQ 166
 MPY =100
 STQ TEM+6
 AXT D,2
 PXA ,2
 ADD TEM+6
 STA 165
 CLA 1,4
 ALS 18
 STO 165
 CLA 165
 STO Y,1
 STO YNM2,1
 TRA CC+1
 CLA Y,1
 PDX ,2
 OFT S1
 TRA KK
 SXA 155,4
 CLA 2,4

	PAC	0,4	T15	OCT	000004000000
	CLA	DZ,4	T16	OCT	000002000000
	OFT	S9	T17	OCT	000001000000
	TRA	++6	CONST	DEC	7,4,6,7,8,3,4,6
	FAD*	Y,1	CB1	DEC	0.0E0
	STO*	Y,1	CB2	DEC	0.0E0
	ONT	S8	CB3	DEC	0.0E0
	TRA	++8	ACS	DEC	1.0E0
	TRA	++2	BCS	DEC	15E-1,-5E-1
	STO*	Y,1	CCS	DEC	+1.9166667E0,-1.3333333E0,+4.1666667E-1
	TIX	++3,2,1	TEMP	DEC	0,0,0,0
	CLA	YNM2,1	TEM	DEC	0,0,0,0,0,0,0
	PDX	,2	S1	OCT	400000000000
	PXD	,2	S2	OCT	200000000000
	STD	Y,1	S3	OCT	100000000000
	LXA	I55,4	S4	OCT	040000000000
	TRA	CC+1	S5	OCT	020000000000
KK	OFT	S7	S6	OCT	010000000000
	TRA	CC+1	S7	OCT	004000000000
	OFT	S8	S8	OCT	002000000000
	TRA	++3	S9	OCT	001000000000
	CLA*	Y,1	S10	OCT	000400000000
	TRA	DD+1	S11	OCT	000200000000
	TIX	+-2,2,1	S12	OCT	000100000000
	TRA	CC+1	S13	OCT	000040000000
HALT1	TRA	++1	I23	NOP	
	OSI	0	I27	NOP	
	OFT	T15	I28	NOP	
	TRA	++8	I31	NOP	
	OFT	T16	I55	NOP	
	CLA	=0	I66	NOP	
	OFT	T17	I65	TIX	0,2,0
	LDQ	=0	I80	TIX	0,2,0
	RIS	T16	//	CONTRL	//
	RIS	T17	M2	COMMON	1
	TRA*	0	FMT	COMMON	13
	RIS	T15	N	COMMON	150
	CLA	=10	M5	COMMON	1
	TRA	++2	TITLE	COMMON	13
HALT3	CLA	=9	R	COMMON	1401
	OSI	S12	DZ	COMMON	1401
	AXT	++3,2	Y	COMMON	1401
	SXA	I27,2	YNM2	COMMON	1401
	TRA	RET	YNM1	COMMON	1401
	AXT	++3,2	D	COMMON	1500
	SXA	I27,2	I1	COMMON	1
	TRA	OUTPUT	I2	COMMON	1
	OSI	S13	M6	COMMON	1
	TRA	++2	M7	COMMON	1
HALT2	RIS	S13	X	COMMON	1
	RIS	S12	XNM1	COMMON	1
	CLA	YOUT	OUT	COMMON	48
	PAC	,2	YOUT	COMMON	48
	CLA	Y,2	NINT	COMMON	1
	STO	X	NCON	COMMON	1
	CLA	YNM1,2	NSUM	COMMON	1
	STO	XNM1	NREL	COMMON	1
	CLA	=3	NMUL	COMMON	1
	OFT	S13	NABS	COMMON	1
	TRA	++9	NTDG	COMMON	1
	CLA	=1	NEFG	COMMON	1
	OFT	S12	NOUT	COMMON	1
	TRA	++5	NCOMP	COMMON	1
	CLA	=5	IVG	COMMON	5
	OFT	S11	CODE	COMMON	7500
	TRA	++2	END		0
	CLA	=2			
	TRA	RET			
	OFT	S11			
	TRA	+-2			
	CLA	=4			
	TRA	+-4			