NBS SPECIAL PUBLICATION 400-57

U.S. DEPARTMENT OF COMMERCE / National Bureau of Standards

Semiconductor Measurement Technology:

DISTRIB I, An Impurity Redistribution Computer Program

NATIONAL BUREAU OF STANDARDS

The National Bureau of Standards[1] was established by an act of Congress March 3, 1901. The Bureau's overall goal is to strengthen and advance the Nation's science and technology and facilitate their effective application for public benefit. To this end, the Bureau conducts research and provides: (1) a basis for the Nation's physical measurement system, (2) scientific and technological services for industry and government, (3) a technical basis for equity in trade, and (4) technical services to promote public safety. The Bureau's technical work is performed by the National Measurement Laboratory, the National Engineering Laboratory, and the Institute for Computer Sciences and Technology.

THE NATIONAL MEASUREMENT LABORATORY provides the national system of physical and chemical and materials measurement; coordinates the system with measurement systems of other nations and furnishes essential services leading to accurate and uniform physical and chemical measurement throughout the Nation's scientific community, industry, and commerce; conducts materials research leading to improved methods of measurement, standards, and data on the properties of materials needed by industry, commerce, educational institutions, and Government; provides advisory and research services to other Government Agencies; develops, produces, and distributes Standard Reference Materials; and provides calibration services. The Laboratory consists of the following centers:

Absolute Physical Quantities[2] — Radiation Research — Thermodynamics and Molecular Science — Analytical Chemistry — Materials Science.

THE NATIONAL ENGINEERING LABORATORY provides technology and technical services to users in the public and private sectors to address national needs and to solve national problems in the public interest; conducts research in engineering and applied science in support of objectives in these efforts; builds and maintains competence in the necessary disciplines required to carry out this research and technical service; develops engineering data and measurement capabilities; provides engineering measurement traceability services; develops test methods and proposes engineering standards and code changes; develops and proposes new engineering practices; and develops and improves mechanisms to transfer results of its research to the utlimate user. The Laboratory consists of the following centers:

Applied Mathematics — Electronics and Electrical Engineering[2] — Mechanical Engineering and Process Technology[2] — Building Technology — Fire Research — Consumer Product Technology — Field Methods.

THE INSTITUTE FOR COMPUTER SCIENCES AND TECHNOLOGY conducts research and provides scientific and technical services to aid Federal Agencies in the selection, acquisition, application, and use of computer technology to improve effectiveness and economy in Government operations in accordance with Public Law 89-306 (40 U.S.C. 759), relevant Executive Orders, and other directives; carries out this mission by managing the Federal Information Processing Standards Program, developing Federal ADP standards guidelines, and managing Federal participation in ADP voluntary standardization activities; provides scientific and technological advisory services and assistance to Federal Agencies; and provides the technical foundation for computer-related policies of the Federal Government. The Institute consists of the following divisions:

Systems and Software — Computer Systems Engineering — Information Technology.

[1]Headquarters and Laboratories at Gaithersburg, Maryland, unless otherwise noted; mailing address Washington,D.C. 20234.
[2]Some divisions within the center are located at Boulder, Colorado, 80303.

The National Bureau of Standards was reorganized, effective April 9, 1978.

Semiconductor Measurement Technology:

DISTRIB I, An Impurity Redistribution Computer Program

David Gilsinn

Institute for Computer Sciences
and Technology
National Bureau of Standards
Washington, D.C. 20234

and

Richard Kraft

Center for Applied Mathematics
National Engineering Laboratory
National Bureau of Standards
Washington, D.C. 20234

This activity was supported by
The Defense Advanced Research Projects Agency
1400 Wilson Boulevard
Arlington, Va 22209

U.S. DEPARTMENT OF COMMERCE, Juanita M. Kreps, Secretary

Jordan J. Baruch, Assistant Secretary for Science and Technology

NATIONAL BUREAU OF STANDARDS, Ernest Ambler, Director

Issued February 1979

Library of Congress Catalog Card Number: 79-600002

National Bureau of Standards Special Publication 400-57
Nat. Bur. Stand. (U.S.), Spec. Publ. 400-57, 130 pages (Feb. 1979)
CODEN: XNBSAV

U.S. GOVERNMENT PRINTING OFFICE
WASHINGTON: 1979

For sale by the Superintendent of Documents, U.S. Government Printing Office, Washington, D.C. 20402
Stock No 003-003-02030-3 Price $3
(Add 25 percent additional for other than U.S. mailing).

PREFACE

This work was conducted as part of the Semiconductor Technology Program at the National Bureau of Standards and was funded by the Defense Advanced Research Projects Agency (Order No. 2397, Program Code 5D10).

The authors wish to thank Sally Peavy for permission to include in Appendix 2 the documentation of her plot routine, W. R. Thurber for the experimental data shown in figure 11, and also acknowledge the efforts of M. G. Buehler in the formulation of the redistribution problem; finally, the authors wish to thank Bill Keery for his editorial aid and Jane Walters for the typing and layout work.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

Semiconductor Measurement Technology:

DISTRIB I, An Impurity Redistribution Computer Program

by

David Gilsinn and Richard Kraft

This report provides documentation of a computer program which calculates the redistribution of impurities in silicon during a single oxidation step. The documentation provides: (1) a physical and mathematical description of the redistribution process, (2) a detailed description of the discretization of the appropriate partial differential equations, and (3) a complete description of the FORTRAN program for computing the solution.

Key Words: Diffusion; electronic technology; impurity distribution; material transport; segregation; semiconductor technology.

1. INTRODUCTION

1.1. Main Objectives of the Documentation

The principal objectives of this documentation are:

(1) to give a detailed derivation of the algebraic equations that are used in the computer program,

(2) to give in detail, a step-by-step description of the way the computation in each computational block of the program is carried out, and

(3) to provide a simple example illustrating how to use the program.

The format of this documentation is modeled after the partial differential equation computer program format description in [9] in the FORTRAN IV language.

In fabrication processes redistribution of impurities takes place in multiple oxidation and diffusion steps (each step being characterized by constant process parameters). DISTRIB is aimed at simulating this complete process; however, due to the complexity of the total problem, it was decided to only document a fundamental oxidation step here. This explains the appellation DISTRIB VERSION 1.

The next section introduces the user to DISTRIB VERSION 1.

1.2. Tips on Where to Locate the Key Parts of This Documentation

Question 1: What should I do if I have a question about this program that is not answered in the documentation?

Answer 1: Call Richard Kraft, NBS, Applied Mathematics Division or Electron Devices Division, (301) 921-3621.

Question 2: Where is the physical problem explained?

Answer 2: In section 2.1.

Question 3: Where are the partial differential equations listed in summary form?

Answer 3: Equations (2.13) to (2.19).

Question 4: Where are the computational procedures outlined?

Answer 4: In section 3.1.

Question 5: Where are the discretized equations derived?

Answer 5: In chapters 5 and 6.

Question 6: What machine has the program run on?

Answer 6: UNIVAC 1108.

Question 7: Where are the flow diagrams?

Answer 7: In chapter 8.

Question 8: Where are the detailed explanations of the computational steps?

Answer 8: In chapter 7.

Question 9: Where are the input and output and a description of how to work a simple problem?

Answer 9: In chapter 9.

Question 10: Where is the listing of the computational blocks?

Answer 10: In chapter 11.

Question 11: Suppose I want to know where to go in the documentation to learn about some quantity in the listing?

Answer 11: Go to the glossary in chapter 12.

Question 12: What is the basis for the program's validity and what are its limitations?

Answer 12: See chapter 10.

Question 13: What about the "deck cards" and the statements in the listings that are not documented in VERSION 1?

Answer 13: Completely ignore them. Everything needed to understand VERSION 1 is explained in this documentation and is in the computational blocks (see listing, chapter 11).

2. DESCRIPTION OF THE PROBLEM

2.1. Physical and Mathematical Description of the Problem

The problem considered here [4] deals with the redistribution of dopant impurities found in silicon as the silicon is oxidized and converted into silicon dioxide. For the sake of specificity, we consider the impurity to be boron and have studied the following ideal situation.

Suppose the entire half space lying to the right of the x-z plane, see figure 1a, to be occupied by silicon while the region on the left-hand side of this plane is filled with oxygen at some elevated temperature. Also, suppose that the silicon is doped with boron and that the density distribution of this impurity is constant in planes parallel to the x-z plane so the distribution is essentially only a function of the y-dimension.

The time evolution of an initially uniform boron distribution (for appropriate conditions at the oxygen-oxide interface) is illustrated in figure 1b. When the oxygen contacts the silicon, it begins to react to form silicon dioxide, and a sharply defined moving boundary which separates the pure silicon from that previously converted into silicon dioxide begins to move into the pure silicon. The position of this moving boundary is known empirically, see [1] and section 4.2, and is denoted by

$$y = y_o(t), \text{ for } t \geq 0 \tag{2.1}$$

where t is time.

Inside the silicon and oxide, the transport of boron is governed by ordinary Fickian diffusion but with different diffusion coefficients D_1 and D_2 in the oxide and silicon, respectively.

The more complicated physical processes take place at the moving oxide-silicon interface. The boron has a preference to be dissolved in the oxide rather than the silicon. Hence, as the moving boundary moves into the silicon, the boron's preference to be in the oxide depletes the boron density in the vicinity of the moving boundary. The preference of the boron to be in the oxide rather than the silicon is quantitatively expressed by requiring that the moving boundary segregation condition be satisfied.

$$C_2(y_o(t),t) = m\, C_1(y_o(t),t) \tag{2.2}$$

where m is a given constant (the segregation coefficient) and eq (2.2) must be satisfied for all time $t > 0$. In eq (2.2) $C_1(y,t)$ and $C_2(y,t)$ represent the boron densities in the oxide and silicon, respectively.

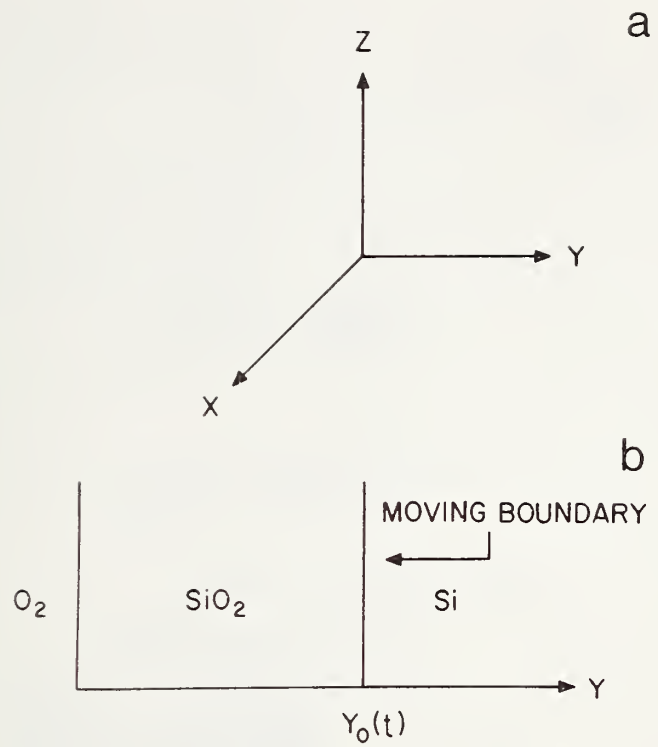


Figure 1. The problem geometry.

In addition to the condition eq (2.2) holding at the moving boundary, it is also assumed that there is conservation of boron atoms. Mathematically, this condition is expressed by the requirement that

$$D_1 \partial_y C_1 \Big|_{y=y_o(t)} + (\alpha^{-1}-1)\dot{y}_o C_1 \Big|_{y=y_o(t)} = D_2 \partial_y C_2 \Big|_{y=y_o(t)} . \qquad (2.3)$$

In order to understand the origin of the convective flux $(\alpha^{-1}-1)\dot{y}_o C_1$, it is necessary first to understand a second complicated physical effect occurring at the moving oxide-silicon boundary. When silicon is converted into oxide, its volume expands by a factor α^{-1}, where $\alpha \approx 0.45$. In the model chosen here [4], it is assumed that all the oxidation of the silicon takes place entirely at the moving oxide-silicon interface. In this case, the expansion tends to push the previously formed oxide backwards at a certain velocity into the oxygen, creating another moving boundary between the oxygen and oxide, see figure 2. The velocity with which the oxide is being convected backwards is assumed to set up a convective drift transport in the oxide's boron atoms. The magnitude of the velocity of this convective flux of boron atoms is determined in the following manner.

Figure 2. A hypothetical redistribution process.

Consider a thin slab with width Δy of pure silicon located exactly at the moving oxide-silicon interface at some time t_o just before it is converted into oxide. After some time Δt, the moving boundary has traveled from one side of this thin slab to the other

$$\Delta y = y_o(t_o + \Delta t) - y_o(t_o) \ , \tag{2.4}$$

converting it into a thin slab of pure oxide of width $\alpha^{-1}\Delta y$. The difference between this new larger width and the original width is assumed to be the distance that the previously formed oxide (i.e., prior to t_o) is moved backward. Therefore, this rate is

$$\lim_{\Delta t \to 0} \frac{\alpha^{-1}\Delta y - \Delta y}{\Delta t} = (\alpha^{-1}-1) \left.\frac{dy_o}{dt}\right|_{t=t_o} \tag{2.5}$$

and the backward velocity of the oxide is $-(\alpha^{-1}-1)\dot{y}_o$ where $\dot{y}_o = \dfrac{dy_o}{dt}$.

Thus, the full equation describing the transport of the boron in the oxide is

$$\partial_t C_1(y,t) = D_1 \partial_y^2 C_1 + (\alpha^{-1}-1)\dot{y}_o(t)\partial_y C_1 \ , \tag{2.6}$$

where the last term in eq (2.6) represents the convective boron transport.

By introducing a coordinate system (see figure 3), moving with the origin at the oxygen-oxide interface,

$$z = y + (\alpha^{-1}-1)y_o(t)$$

or

$$z = y + z_o(t) - y_o(t) \ , \tag{2.7}$$

where

$$z_o(t) = \alpha^{-1} y_o(t) \ , \tag{2.8}$$

eq (2.6) simplifies to

$$\partial_t C_1 = D_1 \partial_z^2 C_1 \text{ for } 0 \leq z \leq z_o(t) \ . \tag{2.9}$$

Moreover, in the z-frame there is no longer a moving boundary at the oxygen-oxide interface to contend with.

The moving boundary conservation condition in the z- and y-frame is

$$D_1 \partial_z C_1(z_o,t) + \dot{z}_o C_1(z_o,t) = D_2 \partial_y C_2(y_o,t) + \dot{y}_o C_2(y_o,t) \ . \tag{2.10}$$

In this model we assume the transport of boron across the oxygen-oxide interface to be governed by

$$D_1 \partial_z C_1 = C_p \ (C_1(0,t) - C_{out}) \ , \tag{2.11}$$

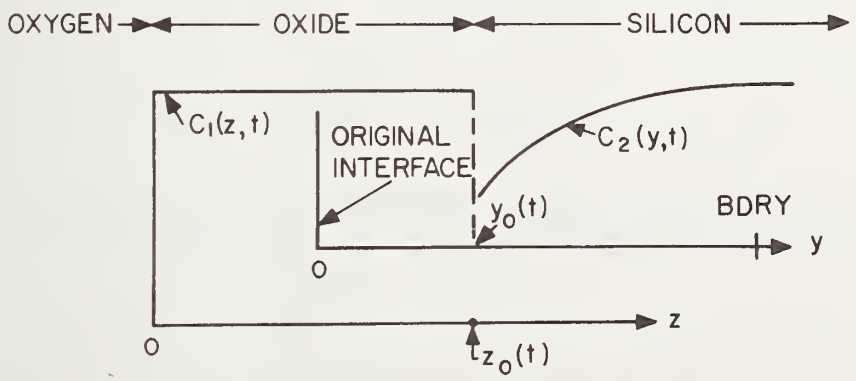


Figure 3. The moving z-coordinate frame.

where C_p and C_{out} are assumed to be known constants. Finally, at $y = \infty$ we assume the concentration to be stationary and equal to a "bulk" value C_B:

$$C_2(\infty,t) = C_B \ . \tag{2.11'}$$

Since we intend to solve the above equations on the computer, it is mandatory to truncate the infinite y-domain by selecting some point y = BDRY far from the y-origin and requiring there that

$$C_2(BDRY,t) = C_B \ . \tag{2.12}$$

Finally, at time t = 0 it is assumed that the silicon slab is completely unoxidized [$y_o(0) = 0$] and that the concentration has a step function distribution:

$$\begin{aligned} C_2(y,0) &= C_{max} \qquad y \leq NH \\ C_2(y,0) &= C_B \qquad NH < y \leq BDRY \end{aligned} \tag{2.13}$$

where C_{max}, NH > 0 are parameters.

In summary, we will solve eq (2.13) and

$$\partial_t C_1 = D_1 \partial_z{}^2 C_1 \text{ for } 0 \leq z \leq z_o(t) \ , \tag{2.14}$$

$$\partial_t C_2 = D_2 \partial_y{}^2 C_2 \text{ for } y_o(t) \leq y \leq BDRY \ , \tag{2.15}$$

$$D_1 \partial_z C_1 + \dot{z}_o C_1 \Big|_{z=z_o(t)} = D_2 \partial_y C_2 + \dot{y}_o C_2 \Big|_{y=y_o(t)} \ , \tag{2.16}$$

$$C_2(y_o,t) = mC_1(z_o,t) \ , \tag{2.17}$$

$$D_1 \partial_z C_1 - C_p(C_1 - C_{out}) \Big|_{z=0} = 0 \ , \text{ and} \tag{2.18}$$

$$C_2(BDRY,t) = C_B \ , \tag{2.19}$$

where the moving boundary $y_o(t)$ or $z_o(t) = \alpha^{-1} y_o(t)$ is a specified monotonic curve, see section 4.2.

2.2. The Partial Differential Equations in Their Equivalent Integral Form

The usual procedure for the numerical solution of partial differential equations like eqs (2.13) to (2.19) comes through solving the system of algebraic equations that results from replacing the partial derivatives in these equations by their approximating difference quotients. When this standard procedure was tried (see [10-13] for other approaches), it did not lead to a very accurate solution. The discretization of the following system of equations equivalent to the partial differential equations and which involves integrals in place of partial derivatives does lead to a system of algebraic equations that can be solved readily to yield an accurate solution of the original system.

The system of equivalent equations is:

$$\frac{d}{dt}\int_{z_A}^{z_B} C_1(z,t)\,dz = -D_1\partial_z C_1\Big|_{z=z_A} + D_1\partial_z C_1\Big|_{z=z_B} \tag{2.20}$$

for every pair of points $z_A < z_B$ such that $0 \le z_A,\ z_B \le z_o(t)$,

$$\frac{d}{dt}\int_{y_A}^{y_B} C_2(y,t)\,dy = -D_2\partial_y C_2\Big|_{y=y_A} + D_2\partial_y C_2\Big|_{y=y_B} \tag{2.21}$$

for every pair of points $y_A < y_B$ such that $y_o(t) \le y_A,\ y_B \le \text{BDRY}$,

$$\frac{d}{dt}\int_{z_A}^{z_o(t)} C_1(z,t)\,dz + \int_{y_o(t)}^{y_B} C_2(y,t)\,dy = -D_1\partial_z C_1\Big|_{z=z_A} + D_2\partial_y C_2\Big|_{y=y_B}\ , \tag{2.22}$$

$$C_2(y_o,t) = m\,C_1(z_o,t)\ , \tag{2.23}$$

$$D_1\partial_z C_1 - C_p(C_1(0,t) - C_{out}) = 0\ , \tag{2.24}$$

$$C_2(\text{BDRY},t) = C_B\ , \text{ and} \tag{2.25}$$

$$C_2(y,0) = C_o(y)\ , \tag{2.26}$$

where

$$C_o(y) = \begin{cases} C_{max} & y \leq NH \\ C_B & NH < y \leq BDRY \end{cases} .$$

To see that eq (2.14) is equivalent to eq (2.20), first carry out the differentiation on the left-hand side of eq (2.20) under the integral sign and substitute for $\partial_t C_1$ by employing eq (2.14). One gets

$$\frac{d}{dt}\int_{z_A}^{z_B} C_1(z,t)dz = \int_{z_A}^{z_B} D_1 \partial_z{}^2 C_1 dz \tag{2.27}$$

and

$$\int_{z_A}^{z_B} D_1 \partial_z{}^2 C_1 dz = -D_1 \partial_z C_1 \Big|_{z=z_A} + D_1 \partial_z C_1 \Big|_{z=z_B} \tag{2.28}$$

yielding the right-hand side of eq (2.20).

By reversing these steps starting with the right-hand side of eq (2.20), one finds

$$\int_{z_A}^{z_B} (\partial_t C_1 - D_1 \partial_z{}^2 C_1)dz = 0 \tag{2.29}$$

for every $0 \leq z_A < z_B \leq z_o(t)$. Therefore, the integral of eq (2.29) must vanish (it is assumed to be continuous) and eq (2.14) results. By using trivial modifications of these same techniques, the remaining equations can be shown to be equivalent to eqs (2.13) to (2.19).

3. THE COMPUTATIONAL PROCEDURE

3.1. General Description of the Computational Procedure

The numerical approach to the solution of the partial differential equations in the integral form, eqs (2.20) to (2.26), replaces the problem of finding the continuous functions $C_1(z,t)$, $C_2(y,t)$ with the problem of determining the functions $C_1(z, t_{MM})$, $C_2(y, t_{MM})$ which are only defined at discrete points on a finite difference grid. The time coordinates of these grid points are specified through the definition of a sequence of discrete times t_{MM} for MM = 1, ..., MME where MME is some given positive integer and the first discrete time, t_1, is zero. The spatial coordinates of the "fixed" grid points (see sec. 4.1 and fig. 4 for further details) are

$$z = (J - 1)\Delta z \text{ for } J = 1,2,\ldots ,$$
$$y = (I - 1)\Delta y \text{ for } I = 1,\ldots,N2 , \qquad (3.1)$$

where N2 is some given integer index. The mesh widths Δz, Δy are also given quantities whose magnitudes like those of MME and N2 need not be considered now. In addition to the fixed grid points in eq (3.1), there are two grid points located at the moving boundary (see sec. 2.1). One is at the moving boundary bordering the z-region and the second is at the moving boundary bordering the y-region, see figure 4. The space and time coordinates of these two moving boundary grid points are, respectively, $(z_o(t_{MM}),t_{MM})$ and $(y_o(t_{MM}),t_{MM})$. In addition to seeking the concentrations at the fixed grid points $C_1(J,t_{MM})$, $C_2(I,t_{MM})$, we also desire to find the concentrations at the two moving boundary grid points,

$$C_1(z_o(t_{MM}),t_{MM}),\ C_2(y_o(t_{MM}),t_{MM}) \text{ for MM=1,..., MME.}$$

Because of the initial condition eq (2.26), these concentrations are known at time $t_1 = 0$.

Let us assume that we have already determined the concentrations at the fixed and moving grid points up to some arbitrary time t_{MM}; then we will proceed to describe how these same concentrations at the next time $t_{MM} + \Delta t = t_{MM+1}$ (where $\Delta t \equiv t_{MM+1} - t_{MM}$) can be found by performing the operations in the following three steps.

Step #1: Discretize the equations.

By discretizing the equivalent description of the partial differential equations in eqs (2.20) to (2.26), we derive a system of linear algebraic tridiagonal equations that couple the concentrations at time t_{MM} with those at t_{MM+1}. Specifically, with each of the fixed grid points in the z-region, we derive and associate an equation of the form

$$Z_1C1(J - 1,TTMMP1) + Z_2C1(J,TTMMP1) + Z_3C1(J + 1,TTMMP1) = Z_4 \qquad (3.2)$$

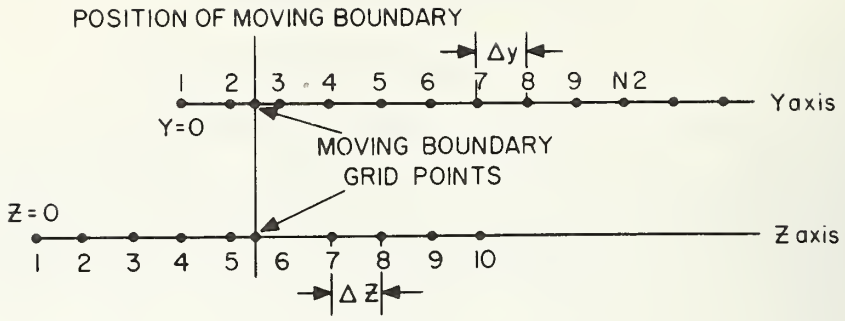


Figure 4. Finite difference grids with $\alpha = 1/3$ and NZO = 3.

Table 1: FORTRAN Version of Symbols in Eqs (3.2) to (3.4) in the Y- and Z-Regions.

$C_1((P-1)\Delta z, t_{MM+1})$	$\equiv$ C1(P,TTMMP1)
$C_2((P-1)\Delta y, t_{MM+1})$	$\equiv$ C2(P,TTMMP1)
t_{MM+1}	$\equiv$ TTMMP1
$C_1(z_o(t_{MM+1}), t_{MM+1})$	$\equiv$ CB(1,TTMMP1)
$C_2(y_o(t_{MM+1}), t_{MM+1})$	$\equiv$ CB(2,TTMMP1)

for J = 1,..., NL and where the FORTRAN notation is defined in table 1. Similarly, with each of the fixed grid points in the y-region, we derive and associate an equation of the form

$$Y_1C2(I-1,TTMMP1) + Y_2C2(I,TTMMP1) + Y_3C2(I+1,TTMMP1) = Y_4 \quad (3.3)$$

where I = NR,..., N2 and table 1 contains the FORTRAN symbol definitions. The coefficients $Z_1,\ldots,Z_4$, $Y_1,\ldots,Y_4$ are composed of the mesh width quantities Δt, Δz, Δy and hence are known. In addition, the quantities Z_4, Y_4 contain, respectively, $C_1(z,t_{MM})$ and $C_2(y,t_{MM})$ and therefore these equations couple the concentrations at the two successive time steps t_{MM} and t_{MM+1}.

At the moving boundary grid point in the z-region, there is an additional equation called the preliminary equation (for reasons explained in step #3) which expresses the value of $C_1(z_o(t_{MM+1}), t_{MM+1})$ in terms of the concentrations at the neighboring grid points, at time t_{MM+1},

$$DM1\ CB(1,TTMMP1) = DM2\ C1(NL,TTMMP1)$$
$$+ DM3\ C2(NR,TTMMP1) + DM4\ C1(NL - 1,TTMMP1) \quad (3.4)$$
$$+ DM5\ C2(NR + 1,TTMMP1) + DM6$$

where the FORTRAN symbols are defined in table 1. In this equation DM1,...,DM6 are composed of known mesh quantities and DM6 contains concentrations, at time t_{MM} (at the grid points coincident and adjacent to the moving boundary grid points), which are of course assumed to be known.

Step #2: Put the tridiagonal equations into "P,Q" form.

By algebraically manipulating eq (3.2), it is possible to put it in the form

$$C1(J,TTMMP1) = P1(J)\ C1(J + 1,TTMMP1) + Q1(J) \quad (3.5)$$

for J = 1,..., NL - 1 and

$$C1(NL,TTMMP1) = PA(1)\ CB(1,TTMMP1) + QA(1)\ . \quad (3.6)$$

By similar manipulations, eq (3.3) can be put into the form

$$C2(I,TTMMP1) = P2(I)\ C2(I - 1,TTMMP1) + Q2(I) \quad (3.7)$$

for I = NR + 1,...,N2 and

$$C2(NR,TTMMP1) = PA(2)\ CB(2,TTMMP1) + QA(2)\ . \quad (3.8)$$

The "P, Q" coefficients in these equations, i.e., P1(J), Q1(J), P2(I), Q2(I), PA(L), QA(L) for L = 1,2 are algebraic combinations of the quantities $Y_1,\ldots, Y_4, Z_1,\ldots, Z_4$ and are hence known.

Step #3: Determination of the concentrations at time TTMMP1 = t_{MM+1}.

The first important use of eqs (3.5) to (3.8) is that it is possible using these equations with J = NL - 1 in eq (3.5) and I = NR + 1 in eq (3.7) to eliminate with the help of eq (2.23) all the unknown concentrations on the right-hand side of the preliminary eq (3.4). Thus, we obtain the value of the concentration at the moving boundary in the z-region in terms of known quantities,

$$CB(1,TTMMP1) = \text{"Some function of } (DM1,\ldots,DM6, Z_1,\ldots,Y_4)\text{"}\ . \quad (3.9)$$

Since eq (3.4) leads directly in this way to what will turn out to be the most important concentration at time TTMMP1 in the sense that with this value of CB(1,TTMMP1) all the other concentrations are simply calculated, we call eq (3.4) the preliminary equation for distinction pur-

poses. From this result and the segregation boundary condition eq (2.23) with SEG ≡ m, we can find DB(2,TTMMP1).

$$CB(2,TTMMP1) = SEG \cdot CB(1,TTMMP1) \ . \qquad (3.10)$$

By using these two concentrations at the moving boundary and back substituting in eqs (3.6) and (3.8), we can find C1(NL,TTMMP1) and C2(NR,TTMMPL).

Finally, by using these two quantities and by back substituting in eq (3.5) starting with J = NL - 1 and by back substituting in eq (3.7) beginning with I = NR + 1, we can successively generate all the concentrations at the fixed grid points at time t_{MM+1} = TTMMP1.

By repeating steps one through three starting at time $t_1 = 0$ when the concentrations are known in the y-region (there is no z-region since the moving boundary starts at z = 0 at time t = 0), all the concentrations at all the grid points at the discrete times t_{MM}, MM = 1,...,MME can be found.

3.2. Short Guide to the Following Chapters

In this documentation, the derivation of the tridiagonal linear eqs (3.2), (3.3), and (3.4) is carried out in Chapter 5. The manipulation of these equations into the form eqs (3.5) to (3.9) is carried through in Chapter 6. The computational procedure is described in detail in Chapter 7, and Chapter 8 contains flow diagrams of the listing in Chapter 11. Chapter 9 contains a worked example and a compact and precise description of the input and output for the program. Chapter 10 describes the validation and limitations of the program, and Chapters 12 and 13 contain a glossary for FORTRAN symbols and references. The appendices collect finite difference approximations used in Chapter 5 and an "in-house" plot documentation.

The next chapter collects material needed in Chapter 5 and gives a detailed description of the finite difference grid, moving boundary motion, equation dimensionalization, and plotting description.

4. DESCRIPTION OF AUXILIARY COMPUTATIONAL PROCEDURES

4.1. A Detailed Description of the Finite Difference Grid

In this section we describe more fully the finite difference grid already introduced in section 3.1, see eq (3.1). We (see fig. 4) begin by describing how the magnitudes of the mesh widths Δy, Δz, Δt and the number of grid points, N2, on the y-axis are determined.

The mesh width Δy is calculated in DISTRIB by dividing the program input quantity BDRY [see above eq (2.12)] by an integer input NYO, i.e.,

$$\Delta y = \frac{\text{BDRY}}{\text{NYO}} . \tag{4.1}$$

Consequently, placing the grid point index 1 at y = 0 and calling the index of the y-grid point at y = BDRY, N2, implies that

$$\text{N2} = \text{NYO} + 1 . \tag{4.2}$$

In order to understand the motivation for the way we define the magnitude of the mesh width Δz, imagine the width of an interval of size Δy of silicon after it has been converted into oxide. It swells to a size [see sec. 2.1] $\alpha^{-1}\Delta y$. Since in DISTRIB α is an unrestricted input and is possibly very small, this latter quantity, $\alpha^{-1}\Delta y$, could be very large. We want to choose the magnitude of Δz in such a way that there are enough Δz intervals in this width to describe any concentration variation in it. Therefore, we introduce another program input integer NZO and use it to define

$$\Delta z = \alpha^{-1}\Delta y/\text{NZO} \tag{4.3}$$

in such a way that $\alpha^{-1}/\text{NZO} \gtrsim 1$ so that Δz and Δy are approximately the same magnitude. Another very important advantage of this choice of Δz is that it implies that when the moving boundary sweeps through a mesh width of magnitude Δy in the y-region NZO mesh widths of magnitude Δz are swept out in the z-region.

In order to explain the reasoning behind the definition of Δt, we remark that the computer program's numerical stability seems to be affected if the moving boundary moves too rapidly between successive grid points. Therefore, Δt is defined in such a way that the moving boundary does not travel more than a fraction WFRC (real program input) of a Δz mesh width between the times t = 0 and t = Δt. Thus, we solve the equation

$$\text{WFRC} \cdot \Delta z = z_o(\Delta t) - z_o(0) \tag{4.4}$$

for Δt and then define Δt as this solution. This is how Δt for the first iteration (see sec. 3.1) when $t_1 = 0$ and $z_o(0) = 0$ is chosen. In computational blocks #16 and #20 (see sec. 7.1), it is explained how Δt is increased during later iterations of the main loop 78 in DISTRIB.

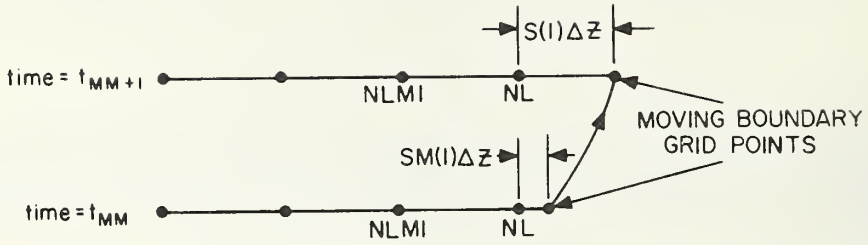


Figure 5: Motion of the moving boundary in the z-region.

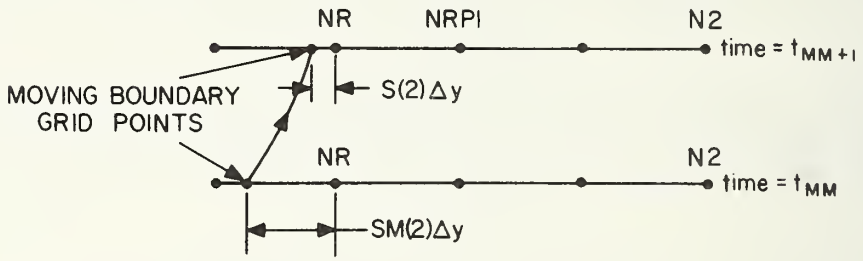


Figure 6. Motion of the moving boundary in the y-region.

Before leaving the description of the finite difference grid, it is important and convenient to introduce special names for certain indices of special grid points. Figure 5 shows the motion of the moving boundary starting at time t_{MM} and ending at time t_{MM+1} in the z-region. We denote the index of the z-grid point which is at or immediately to the left of the moving boundary at time t_{MM} by NL and the next grid point to its left by NLM1 (= NL - 1). Similarly (see fig. 6), the index of the first grid point in the y-region greater than the position of the moving boundary at time t_{MM} is denoted by NR, and the index of the next point to its right is NRP1 (= NR + 1). Of course, the values of NL and NR change with t_{MM}.

4.2. The Description of the Moving Boundary

In DISTRIB the user can choose between two different formulas for the moving boundary. These are the two motions related to the wet and dry oxidations in [1]. In both cases there are three parameters A, B, and τ (each with values that are temperature dependent) in the equations that describe the motion of the moving boundary.

Case I: Wet oxidation, $\tau = 0$.

The equation is [see 1, eq (13)]

$$z_o = \frac{A}{2}\left[1 + \frac{4tB}{A^2}\right]^{1/2} - 1 \text{ for } t > 0 . \qquad (4.5)$$

The equation expressing t in terms of z_o is

$$t = (z_o^2 + Az_o)/B \ . \tag{4.6}$$

Table 1 in [1] gives the values of A and B for different temperatures.

Case II: Dry oxidation, $\tau > 0$.

For times $t \geq \tau$ [1]

$$z_o = \frac{A}{2}\left[1 + \frac{4tB}{A^2}\right]^{1/2} - 1 \text{ for } t \geq \tau \ . \tag{4.7}$$

When $t \leq \tau$, we use [1]

$$z_o = \frac{ZCRIT}{\tau} \cdot t \ , \tag{4.8}$$

where

$$ZCRIT = CNSTX = \frac{A}{2}\left[1 + \frac{4tB}{A^2}\right]^{1/2} - 1 \ . \tag{4.9}$$

CNSTX is a duplicate name for ZCRIT. Table II in [1] gives the values of A, B, and τ for different temperatures.

The FORTRAN symbols for the constants in the previous equations are given in table 2.

Table 2: FORTRAN Version of Symbols Used in the Moving Boundary Equation and Finite Difference Grid.

A = CNSTA
B = CNSTB
.5A = CNA
τ = CNTAU = TLIN
ZCRIT = CNSTX
Δz = DZ
Δy = DY
Δt = DT
α = ALPHA

4.3. Dimensionalization

Although the physical input in DISTRIB is required to be in units of micrometers and hours, it is computationally more efficient to solve the partial differential equations with the physical quantities expressed in other units. For the kinds of problems of interest in impurity redistribution, a convenient basic length is BLTH = 0.02 μm. Given the basic length unit, a basic time is frequently introduced in diffusion theory by defining

$$\mathrm{BTM} = \frac{\mathrm{BLTH}^2}{D_o} \text{ hours} \tag{4.10}$$

where D_o is some diffusion coefficient in the problem. In our program, D_o is chosen to be D_2, the diffusion coefficient in silicon. When these new units are used to dimensionalize the diffusion equation in the silicon, one finds the new equation has a diffusion constant of unity. That is, making a change of independent variables

$$t' = \frac{t}{\mathrm{BTM}},\ y' = \frac{y}{\mathrm{BLTH}},\ \text{and}\ z' = \frac{z}{\mathrm{BLTH}}$$

transforms eq (2.14) into

$$\partial_{t'} C_2 = \partial^2_{y'} C_2$$

and eq (2.15) into

$$\partial_{t'} C_1 = \frac{D_1}{D_2} \partial^2_{z'} C_1 \ .$$

The rest of the dimensionalizations carried out in block #4 of section 7.1 are standard conversions from micrometers and hours to the new units BLTH and BTM.

4.4. The Description of the Way Arrays C1 and C2 Are Plotted

The plotting is really a peripheral item and not described in as much detail as substantive parts of the program. All the quantities undefined are defined or referenced in the glossary and their definition in the glossary makes their function clear. First, the way C1 is plotted is described. Let NC be the integer part of (NL - 1)/NMOD. When NC > 0, we plot the concentrations C1 and z-coordinate (in units of ZOP1) at every grid point K = V · NMOD + 1 for V = 1,..., NC. When NC = 0, we plot the concentrations at z = 0 and at the moving boundary z_o. The quantities X1S, Y1S determine the largest abscissa and ordinate on the graph of C1 versus z.

The graph of array C2 is plotted in the following way. All abscissa are in units of UN2 (which is essentially SCALE2, since y is still in units of BLTH). The first C2 concentration plotted is CB(2) at the moving boundary. The next concentration is C2(NR) with abscissa S(2)DY. Next, we plot ND - 2 concentrations with the indices KS = NR + (V - 2) NMOD1 for V = 3,..., ND. Next, we pass these grid points through a filter which eliminates those with an abscissa greater than X2S/2 = X2SCT. The quantities X2S and Y2S determine the largest abscissa and ordinate that appear on the C2 versus y plot. Note: The specific scales used as the ordinates and abscissa for these plots were chosen in order to make the resultant graphs identical in size to those obtained from the equipment which experimentally determined the concentration data. This also accounts for the elimination of points beyond X2S/2. Both conditions may be changed to accommodate the user.

The smallest abscissa and ordinate are determined in both graphs by (0,0).

The plot subroutine is an NBS "house program" that is documented in Appendix 2.

This plotting procedure has worked well in all cases, the only adjustments needing to be made being in NMOD and NMOD1 (see table 20 in sec. 9.22).

5. DERIVATION OF THE DISCRETE ALGEBRAIC EQUATIONS

5.1. The General Prescription for Obtaining the Tridiagonal Equation Associated with a Fixed Grid Point

This section, which should be read casually, gives a rough description of the precise derivations in the following sections.

Each particular tridiagonal equation in eqs (3.2), (3.3) [and (3.4) also] associated with some particular grid point is derived by discretizing some integral form of the conservation of number eqs (2.20) to (2.22) over the "flux cell" (to be described below) that contains that particular grid point. The flux cells are intervals that arise by partitioning the z- and y-regions in a certain way. That is, both the z-region and the y-region are broken up into nonoverlapping intervals called flux cells or concentration cells in such a way that the intervals completely cover the regions $[0, z_o(t)]$ and $[y_o(t), BDRY]$ at every time t; and each grid point is contained in at most one cell except for the grid points with indices NL and NR. Each of these grid points is contained in both the irregular sized cell with which it is associated and the "moving boundary cell"; these statements will be explained more completely later. The two boundaries of each of the concentration cells are called the flux boundaries of the cell. We interpret the concentration, $C1(J,TTMMP1) = C1((J - 1)\Delta z, t_{MM+1})$, at some grid point J to represent the average number density in the cell at time t_{MM+1}.

The flux boundaries for the cells in the y-region are shown in figure 7 by the vertical lines located halfway between successive grid points. Notice that all the cells in the y-region that contain fixed grid points (see sec. 3.1) are of width Δy except the irregular one of width $0.5\ \Delta y$ that contains and leads to the tridiagonal algebraic equation associated with the grid point with index NR at $(NR - 1)\Delta y$. This cell has one flux boundary at $(NR - 1)\Delta y$ and the other is at $(NR - 1/2)\Delta y$. Similar remarks apply to figure 8.

The significance of the flux cell containing some grid point, with index I in the y-region (see fig. 7) is that it leads to a linear algebraic equation containing the concentration C2(I,TTMMP1) as an unknown when the conservation eq (2.21) defined on this cell is discretized.

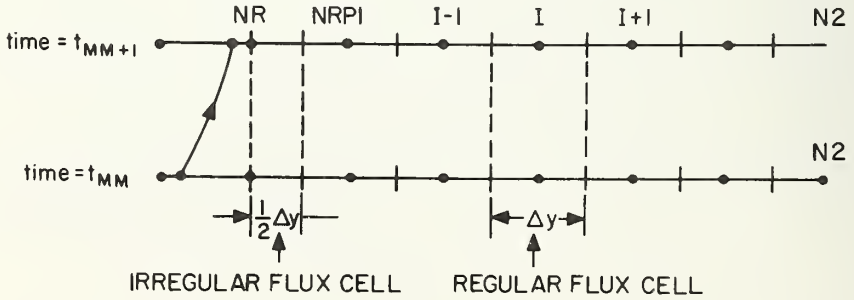


Figure 7: Flux cells and flux boundaries in the y-region for fixed grid points.

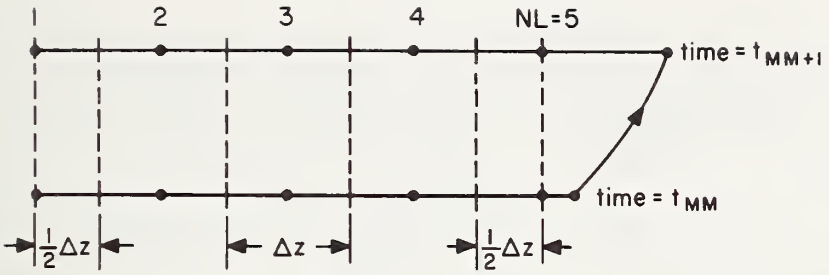


Figure 8. Flux cells for fixed grid points in the z-region when NL > 2.

In particular, let y_A and y_B in eq (2.21) be at $(I - 3/2)\Delta y$ and $(I - 1/2)\Delta y$, respectively. Make now the following approximations in eq (2.21) (with $t = t_{MM+1}$):

$$\frac{d}{dt}\int_{y_A}^{y_B} C_2(y,t)dy \approx \int_{y_A}^{y_B} [C_2(y,t_{MM+1}) - C_2(y,t_{MM})]dy/\Delta t \ ,$$

$$\int_{y_A}^{y_B} C_2(y,t_{MM+1})dy \approx C_2((I - 1/2)\Delta y,t_{MM+1})\Delta y \ ,$$

$$\int_{y_A}^{y_B} C_2(y,t_{MM})dy \approx C_2((I - 1)\Delta y,t_{MM})\Delta y \ ,$$

$$D_2\partial_y C_2\Big|_{y=y_A} \approx D_2[C_2((I - 1)\Delta y,t_{MM+1}) - C_2((I - 2)\Delta y,t_{MM+1})] \ , \text{ and}$$

$$D_2\partial_y C_2\Big|_{y=y_B} \approx D_2[C_2(I\Delta y, t_{MM+1}) - C_2((I - 1)\Delta y, t_{MM+1})] \ .$$

Then eq (2.21) becomes, up to terms of higher order, the linear tridiagonal equation in eq (3.4) associated with the grid point with index I,

$$C_2((I - 2)\Delta y,t_{MM+1}) - \left[2 + \frac{\Delta y^2}{D_2\Delta t}\right] C_2((I - 1)\Delta y,t_{MM+1}) + C_2(I\Delta y,t_{MM+1}) = -\frac{\Delta y^2}{D_2\Delta t} C_2((I - 1)\Delta y,t_{MM}) \ .$$

The important fact this example illustrates is that, by evaluating the fluxes in eq (2.21) at the flux boundaries of the flux cell and evaluating the integrals by a quadrature approximation with a "node" located at the grid point in the flux cell, one ends up with a tridiagonal (i.e., an equation involving the concentrations at three consecutive grid points) equation for the unknown concentration at time t_{MM+1} at that grid point. Thus, with the help of these flux cells one can in a natural way associate with each unknown at a fixed grid point an algebraic equation.

The flux cells of the points in the z-region depend on the values of NL. There are three cases depending on whether NL = 1, NL = 2, or NL > 2. The cell configuration for the case NL > 2 can be visualized from figure 8. In the case NL = 2 (see fig. 9), there are just two irregular cells each of width 0.5 Δz. The flux cell associated with the grid point with index NL has its right side flux boundary located coincident with the grid point. The flux cell associated with the first z-grid point with index 1 has its left-hand flux boundary coincident with the first grid point. These facts also hold when NL > 1, but in this case there are regular grid cells, i.e., with width Δz in between the irregular ones at the outer boundaries. The case NL = 1 is special because there are no grid points in the z-region between the moving and fixed boundaries.

Finally, we point out that for all values of NL the conservation formula eq (2.22) (conservation across the moving boundary) is discretized to derive the preliminary eq (3.4), and in this derivation the flux cell is as shown in figure 10 with the flux boundaries at $z_A = (NL - 1)\Delta z$ and $y_B = (NR - 1)\Delta y$. The trapezoidal rule is used to evaluate the integrals in eq (2.22) with the nodes at NL, NR and the moving boundary grid points at $S(1)\Delta z + (NL - 1)\Delta z$ and $(NR - 1)\Delta y - S(2)\Delta y$, see figures 5 and 6.

These different cases, which arise because we have flux cells with different widths and with different orientation of grid points relative to the flux boundaries for different values of NL, explain the slight variations, classified in tables 3 and 15, in the way the tridiagonal algebraic equations and their "P,Q" coefficients (see sec. 3.1) are derived in the following sections.

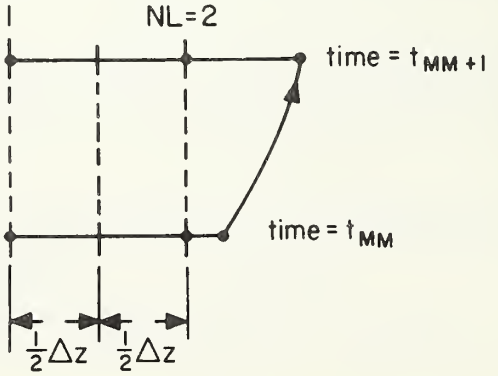


Figure 9. Flux cells for fixed grid points in the z-region when NL = 2.

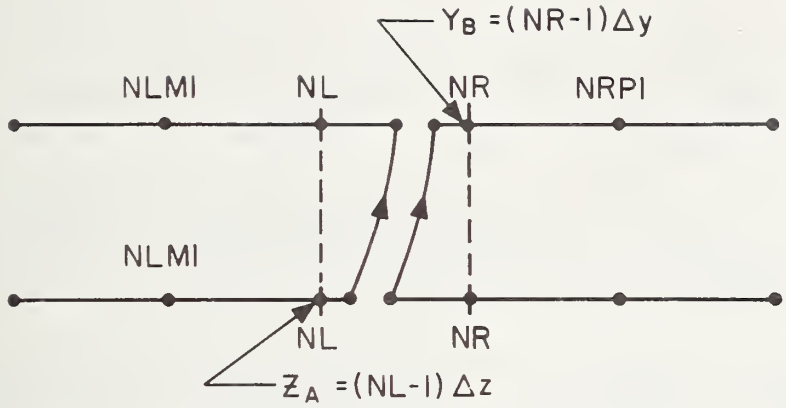


Figure 10. Schematic description of the flux cell associated with the discretization of the conservation equation across the moving boundary.

Table 3: Guide to the Sections Where the Tridiagonal Algebraic Equations Associated With the Grid Points are Described and Derived.

Case I:	The moving boundary has not reached the second z-grid point or NL < 2.
	(a) The index of the z-grid point is 1 (see sec. 5.9)
	(b) The index of the y-grid point is greater than NR (see sec. 5.3)
	(c) The index of the y-grid point is NR (see sec. 5.5)
	(d) The grid point is at the moving boundary (see sec. 5.7)
Case II:	The moving boundary has reached the second z-grid point or NL $\geq$ 2.
	(a) The index of the z-grid point is 1 (see sec. 5.8)
	(b) The index of the z-grid point ranges from 2 to NL - 1 (see sec. 5.2)
	(c) The index of the z-grid point is NL (see sec. 5.4)
	(d) The grid points are at the moving boundary (see sec. 5.6)
	(e) The index of the y-grid point is greater than NR (see sec. 5.3)
	(f) The index of the y-grid point is NR (see sec. 5.5)

5.2. The Index of the Z-Grid Point Ranges from 2 to NL - 1 When NL > 2.

In this case we suppose the index, J + 1, is such that 2 < J + 1 < NL - 1, and begin with the formula, see eq (2.20)

$$\frac{d}{dt}\int_{z_A}^{z_B} C_1(z,t)dz = - D_1\partial_z C_1\Big|_{z=z_A} + D_1\partial_z C_1\Big|_{z=z_B} . \qquad (5.1)$$

For the z-grid point, J + 1, we assume z_A and z_B "the flux boundaries" to be at $(J - 1/2)\Delta z$, $(J + 1/2)\Delta z$, respectively. We speak of these grid points as regular grid points because the distance between the cell flux boundaries is Δz.

The left-hand side of eq (5.1) is discretized by employing the backward time difference formula (A7) in Appendix 1 between the times $t = t_{MM}$ and $t = t_{MM} + \Delta t$. Here, Δt is some small positive time increment that sometimes depends on the index MM. After the derivative has been replaced by the backward difference quotient, the integrals are approximated by the midpoint quadrature rule (A2).

Finally, after further algebraic manipulation we have

$$\frac{d}{dt}\int_{z_A}^{z_B} C_1(z,t)dz = \frac{\Delta z}{\Delta t}\,[C_1(J\Delta z, t_{MM} + \Delta t) - C_1(J\Delta z, t_{MM})] + O(\Delta z^2 + \Delta t^2)/\Delta t . \qquad (5.2)$$

Since all the expressions in eq (5.1) are assumed to be evaluated at $t = t_{MM} + \Delta t$, the approximation of the derivatives on the right-hand side of eq (5.1) gives

$$\partial_z C_1\Big|_{z=z_A} = [C_1(J\Delta z, t_{MM} + \Delta t) - C_1((J - 1)\Delta z, t_{MM} + \Delta t)]/\Delta z + O(\Delta z) \qquad (5.3)$$

and

$$\partial_z C_1\Big|_{z=z_B} = [C_1((J + 1)\Delta z, t_{MM} + \Delta t) - C_1(J\Delta z, t_{MM} + \Delta t)]/\Delta z + O(\Delta z) . \qquad (5.4)$$

By substituting eqs (5.2) to (5.4) into eq (5.1) and further rearranging, we obtain up to higher order terms the discrete version of eq (5.1) in tridiagonal form,

$$D_1 C_1((J - 1)\Delta z, t_{MM} + \Delta t) - (2 \cdot D + \Delta z^2/\Delta t)\, C_1(J\Delta z, t_{MM} + \Delta t) + D_1 C_1((J + 1)\Delta z, t_{MM} + \Delta t) = \frac{-(\Delta z)^2}{\Delta t}\, C_1(J\Delta z, t_{MM}) . \qquad (5.5)$$

After introducing the FORTRAN quantities in table 4, eq (5.5) becomes

$$DF1 \cdot C1(J,TTMMP1) - (2 \cdot DF1 + DZ**2/DT) \cdot C1(J+1, TTMMP1) + DF1 \cdot C1(J+2),TTMMP1) = \frac{-(DZ)**2}{DT} \cdot C1(J+1,TTMM) \quad (5.6)$$

Table 4: FORTRAN Version of Quantities in Eq (5.5).

D_1	DF1
$C_1(,)$	C1(,)
t_{MM}	TTMM
$t_{MM} + \Delta t$	TTMMP1
Δt	DT
Δz	DZ
$C_1(J\Delta z, t_{MM})$	C1(J + 1,TTMM)
$C_1(J\Delta z, t_{MM} + \Delta t)$	C1(J + 1, TTMMP1)

In the FORTRAN program, the following intermediate quantities in table 5 have been introduced for convenience and computational efficiency.

Table 5: Intermediate Quantities Used in Eq (5.7).

$\frac{DT}{DZ^2}$	RA1
DF1	A2L
$-(2 \cdot DFL + (RA1)^{-1})$	A2M
DF1	A2R
$-(RA1)^{-1}$	A1M
AF(J + 1)	A1M · C1(J + 1,TTMM)

In terms of these intermediate quantities, eq (5.6) can be written in the compact tridiagonal FORTRAN form

$$A2L \cdot C1(J,TTMMP1) + A2M \cdot C1(J+1,TTMMP1) + A2R \cdot C1(J+2,TTMMP1) = AF(J+1) \; . \quad (5.7)$$

As has been explained in more detail in section 3.1, the quantities A2L,..., AF(J) are used in computing the values of the arrays P1, Q1, and D1. Specifically, the quantities A2L,..., A2R, A1M are computed in

subroutine INTER. The subroutine INTER is called by subroutine TRIDNL when this latter subroutine is calculating the arrays P1, Q1. The array AF(J) is computed in block #25 of the main program, see section 7.1.

SUMMARY

Schematic summary of the derivation of the tridiagonal FORTRAN equation for z-grid points with indices between 1 and NL:

$$(5.1) \longrightarrow (5.5) \longrightarrow (5.6) \longrightarrow (5.7) \ .$$

Summary notes:

(1) Quantities in table 5 except for AF are computed in subroutine INTER.

(2) The array AF(J) is computed in block #25 of the main program.

5.3. The Index of the Y-Grid Point is Greater than NR

The derivation of the tridiagonal FORTRAN form of the algebraic equations associated with these grid points is almost entirely analogous to the derivation of the equations in subsection 5.2 once the substitutions z for y, D_1 for D_2, C_1 for C_2, etc., are made. However, there is one exception; in the y-region we have divided the equivalent of eq (5.5) by D_2 so that the equivalent of that equation is

$$C_2((J-1)\Delta y, t_{MM} + \Delta t) - \left[2 + \frac{\Delta y^2}{D_2 \Delta t}\right] C_2(J\Delta y, t_{MM} + \Delta t) + C_2((J+1)\Delta y, t_{MM} + \Delta t) = \frac{\Delta y^2}{D_2 \Delta t} C_2(J\Delta y, t_{MM}) \ . \tag{5.8}$$

Table 6 is the equivalent of table 4.

Table 6: FORTRAN Version of Quantities in Eq (5.8).

D_2 = DF2
Δy = DY
$C_2(\cdot,\cdot)$ = C2(·,·)
Δz = DZ
$C_2(J\Delta y, t_{MM})$ = C2(J + 1,TTMM)
$C_2(J\Delta y, t_{MM} + \Delta t)$ = C2(J + 1,TTMMP1)

When the quantities in table 6 are introduced into eq (5.8), it becomes

$$C1(J,TTMMP1) - \left[2 + \frac{DY**2}{DF2 \cdot DT}\right] C2(J+1,TTMMP1) + C2(J+2,TTMMP1) = \frac{DY**2}{DF2 \cdot DT} C2(J+1,TTMM) \quad . \tag{5.9}$$

This form of the equation is further simplified by introducing the intermediate quantities in table 7.

Table 7: Intermediate Quantities Used in Eq (5.10).

$\frac{DT}{DY^2} = RA2$
$1 = B2L$
$-(2 + RA2^{-1}/DF2) = B2M$
$1 = B2R$
$-RA2^{-1}/DF2 = B1M$
$BF(J+1) = B1M \cdot C2(J+1,TTMMP1)$

In terms of the intermediate quantities in table 7, eq (5.9) assumes the form

$$B2L\ C2(J, TTMMP1) + B2M\ C2(J+1, TTMMP1) + B2R\ C2(J+2, TTMMP1) = BF(J+1) \quad . \tag{5.10}$$

The quantities RA2, B2L,...,B1M are computed in subroutine INTER. This subroutine is called by subroutine TRIDNL and in this routine the quantities RA2,...,B1M are used to compute arrays P2, D2. The quantity BF(J) is computed in block #26 of the main program.

SUMMARY

Schematic summary of the derivation of the tridiagonal FORTRAN equations for y-grid points with indices greater than NR:

$$(5.8) \longrightarrow (5.9) \longrightarrow (5.10) \quad .$$

Summary notes:

(1) Quantities in table 7 except for BF are calculated in subroutine INTER.

(2) The array BF(J) is computed in block #26 of the main program.

5.4. The Index of the Z-Grid Point is NL and NL > 1

In this subsection we derive the FORTRAN tridiagonal form of the equation associated with the z-grid point with index NL which is located at $z = (NL - 1)\Delta z$.

The derivation begins with the integral conservation equation

$$\frac{d}{dt}\int_{z_A}^{z_B} C_1(z,t)dz = -D_1\partial_z C_1\Big|_{z=z_A} + D_1\partial_z C_1\Big|_{z=z_B} \tag{5.11}$$

In this equation, $z_B = (NL - 1)\Delta z$, $z_A = (NL - 3/2)\Delta z$ are the "flux boundaries"; notice the flux boundary at z_B coincides with the location of the grid point associated with the cell (see fig. 9). Also, the cell width, the distance between the flux boundaries, is $\Delta z/2$, and therefore this grid point is called an irregular grid point, see section 5.1.

By using the backward difference quotient approximation in eq (A7) to approximate the time derivative on the left-hand side of eq (5.11) between the times t_{MM} and $t_{MM} + \Delta t$ and the right end point quadrature approximation in eq (A3) to approximate the resulting integrals, the left-hand side of eq (5.11) becomes equivalent to

$$\frac{d}{dt}\int_{(NL-3/2)\Delta z}^{(NL-1)\Delta z} C_1(z,t)dz = \frac{\Delta z}{2\Delta t}\Big\{C_1[(NL - 1)\Delta z,\ t_{MM} + \Delta t] - C_1[(NL - 1)\Delta z,\ t_{MM}] + O(\Delta t + \Delta z^2/\Delta t)\Big\}. \tag{5.12}$$

The spatial derivative on the right-hand side of eq (5.11) at $z = z_A$ is approximated by the central difference approximation

$$\partial_z C_1\Big|_{z=z_A} = \frac{1}{\Delta z}\Big\{C_1[(NL -1)\Delta z,\ t_{MM} + \Delta t] - C_1[(NL - 2)\Delta z,\ t_{MM} + \Delta t]\Big\} + O(\Delta z) . \tag{5.13}$$

To approximate the derivative at $z = z_B$ on the right-hand side of eq (5.11), we use the three point derivative approximation given in eq (A9). Setting in that formula

$$Xl = z_o(t_{MM} + \Delta t)$$
$$X = (NL - 1)\Delta z = z_B$$
$$X_2 = (NL - 2)\Delta z$$
$$h = \Delta z$$

we obtain the sought-after result

$$\partial_z C_1\Big|_{z=z_B} = \frac{1}{\Delta z}\left\{-\frac{S(1)}{1 + S(1)} C_1[(NL - 2)\Delta z,\ t_{MM} + \Delta t]\right.$$
$$+ [(S(1) - 1)/S(1)]\ C_1[NL - 1)\Delta z,\ t_{MM} + \Delta t] \qquad (5.14)$$
$$\left. + \frac{1}{S(1)\ [1 + S(1)]} C_1[z_o(t_{MM} + \Delta t),\ t_{MM} + \Delta t]\right\} + O(\Delta z^2)\ ,$$

where

$$S(1) = [z_o(t_{MM} + \Delta t) - (NL - 1)\Delta z]/\Delta z\ . \qquad (5.15)$$

When eqs (5.12), (5.13), and (5.14) have been substituted into eq (5.11) and the expression simplified, it becomes up to terms of higher order

$$-\frac{D_1 S(1)}{1 + S(1)} C_1[(NL - 2)\Delta z,\ t_{MM} + \Delta t]$$
$$+\left[\frac{D_1 \cdot S(1)}{R(1)} + D_1\right] C_1[(NL - 1)\Delta z,\ t_{MM} + \Delta t] \qquad (5.16)$$
$$-\frac{D_1}{(1 + S(1))} C_1[z_o(t_{MM} + \Delta t),\ t_{MM} + \Delta t]$$
$$= \frac{S(1)\ C_1[(NL - 1)\Delta z,\ t_{MM}]}{R(1)}\ ,$$

where

$$R(1) = \frac{2\Delta t}{\Delta z^2}\ . \qquad (5.16')$$

After introducing the FORTRAN notation in table 8 into eq (5.16), it becomes

$$- \frac{DF1}{1 + S(1)} Cl(NL - 1, TTMMP1) + \frac{DF1}{R(1)} + \frac{DF1}{S(1)} Cl(NL, TTMMP1)$$

$$- \frac{DF1}{S(1)\ (1 + S(1))} CB(1) = \frac{Cl(NL, TTMM)}{R(1)} \ . \qquad (5.17)$$

Table 8: FORTRAN Version of Quantities in Eq (5.16).

D_1 = DF1
t_{MM} = TTMM
$t_{MM} + \Delta t$ = TTMMP1
Δt = DT
Δz = DZ
$R(1) = \frac{Z \cdot DT}{DZ**2}$
$C_1((NL - 2)\Delta z, t_{MM} + \Delta t) = Cl(NL - 1, TTMMP1)$
$C_1((NL - 1)\Delta z, t_{MM} + \Delta t) = Cl(NL, TTMMP1)$
$C_1((NL - 1)\Delta z, t_{MM}) = Cl(NL, TTMM)$
$C_1(z_o(t_{MM} + \Delta t), t_{MM} + \Delta t) = CB(1)$

Equation (5.17) will be used to obtain the "P, Q" quantities PA(1), QA(1) for z-grid point NL, i.e., quantities such that

$$Cl(NL, TTMMP1) = PA(1) \cdot CB(1) + QA(1) \ , \qquad (5.18)$$

in section 6.6.

SUMMARY

Schematic summary of the derivation of the tridiagonal FORTRAN equation for the z-grid point with index NL when NL $\geq$ 2,

$$(5.11) \longrightarrow (5.16) \longrightarrow (5.17) \ .$$

Summary note:

> Equation (5.17) is used in section 6.6 to obtain quantities PA(1), QA(1).

5.5. The Index of the Y-Grid Point is NR

The derivation of the tridiagonal FORTRAN formula for this grid point is analogous to the derivation in section 5.4 for the irregular z-grid

point at NL so the user should refer to that section if more details concerning the following derivation are desired.

Again, we start with the integral conservation equation,

$$\frac{d}{dt}\int_{y_A}^{y_B} C_2(y,t)dy = -D_2\partial_y C_2\Big|_{y=y_A} + D_2\partial_y C_2\Big|_{y=y_B} , \tag{5.19}$$

where $y_A = (NR - 1)\Delta y$ and $y_B = (NR - \frac{1}{2})\Delta y$.

By using the same approximation procedure that led from eq (5.11) to eq (5.12) (except for using the left end point quadrature approximation), we find

$$\frac{d}{dt}\int_{y_A}^{y_B} C_2(y,t)dy = \frac{\Delta y}{2\Delta t}\Big\{C_2[(NR - 1)\Delta y,\ t_{MM} + \Delta t] - C_2[(NR - 1)\Delta y,\ t_{MM}]\Big\} + O\left(\frac{\Delta t^2 + \Delta y^2}{\Delta t}\right). \tag{5.20}$$

The spatial derivative on the right-hand side of eq (5.19) at $y = y_B$ is approximated by the central difference approximation (see eq (A8), with $h = \Delta y/2$)

$$\partial_y C_2\Big|_{y=y_B} = \frac{1}{\Delta y}\Big\{C_2(NR\Delta y,\ t_{MM} + \Delta t) - C_2[(NR - 1)\Delta y,\ t_{MM} + \Delta t]\Big\} + O(\Delta y) . \tag{5.21}$$

The spatial derivative at $y = y_A$ on the right-hand side of eq (5.19) is approximated by using the three point formula eq (A9) with:

$$X1 = NR\Delta y$$
$$X = (NR - 1)\Delta y$$
$$X_2 = y_o(t_{MM} + \Delta t) .$$

We find

$$\partial_y C_2\Big|_{y=y_A} = \frac{1}{\Delta y}\Big\{\frac{S(2)}{1 + S(2)} C_2((NR)\Delta y,\ t_{MM} + \Delta t) + \frac{(1 - S(2))}{S(2)} C_2((NR - 1)\Delta y,\ t_{MM} + \Delta t) - \frac{1}{S(2)(1 + S(2))} C_2(y_o(t_{MM} + \Delta t),\ t_{MM} + \Delta t)\Big\} + O(\Delta y^3) , \tag{5.22}$$

where

$$S(2) = -[y_o(t_{MM} + \Delta t) - (NR - 1)\Delta y]/\Delta y \ . \tag{5.23}$$

When eqs (5.20), (5.21), and (5.22) have been substituted into eq (5.19) and the resulting expression is simplified, it becomes

$$\begin{aligned} &- \frac{1}{1 + S(2)} C_2[y_o(t_{MM} + \Delta t),\ t_{MM} + \Delta t] \\ &+ \left(\frac{S(2)}{R(2)} + 1\right) C_2[(NR - 1)\Delta y,\ t_{MM} + \Delta t] \\ &- \frac{S(2)}{1 + S(2)} C_2(NR\Delta y,\ t_{MM} + \Delta t) = \frac{S(2)}{R(2)} C_2[(NR - 1)\Delta y,\ t_{MM} + \Delta t] \ , \end{aligned} \tag{5.24}$$

where

$$R(2) = \frac{2\ D_2 \Delta t}{\Delta y^2} \ . \tag{5.25}$$

After introducing the FORTRAN notation in table 9 into eq (5.24), it becomes

$$\begin{aligned} &- \frac{1}{1 + S(2)} CB(2,\ TTMMP1) + \left(\frac{S(2)}{R(2)} + 1\right) C2(NR,\ TTMMP1) \\ &- \frac{S(2)}{1 + S(2)} C2(NR + 1,\ TTMMP1) = \frac{S(2)}{R(2)} C2(NR,\ TTMM) \ . \end{aligned} \tag{5.26}$$

Table 9: FORTRAN Version of Quantities in Eq (5.24).

D_2 = DF2
t_{MM} = TTMM
$t_{MM} + \Delta t$ = TTMMP1
Δt = DT
Δy = DY
$R(2) = \frac{2 \cdot DF2 \cdot DT}{DY^{**}}$
$C_2(y_o(t_{MM} + \Delta t),\ t_{MM} + \Delta t)$ = CB(2)
$C_2((NR - 1)\Delta y,\ t_{MM} + \Delta t)$ = C2(NR, TTMMP1)
$C_2((NR - 1)\Delta y,\ t_{MM})$ = C2(NR, TTMM)

Equation (5.26) will be utilized later in section 6.9 to find the quantities PA(2), QA(2) such that

$$C2(NR, TTMMP1) = PA(2)\ CB(2) + QA(2) \qquad (5.26')$$

Schematic summary of the derivation of the tridiagonal FORTRAN equation for the y-grid point NR,

$$(5.19) \longrightarrow (5.24) \longrightarrow (5.26)$$

Summary note:

> Equation (5.26) is used in section 6.9 to determine the quantities PA(2), QA(2) associated with grid point NR.

5.6. The Grid Points Lie at the Moving Boundary and NL > 1 (The Preliminary Equation)

The main equation associated with the two unknowns $C_1[z_o(t_{MM} + \Delta t), t_{MM} + \Delta t]$, $C_2[y_o(t_{MM} + \Delta t), t_{MM} + \Delta t]$ located at the grid points which form the interior boundary of the z- and y-regions, respectively, is derived from the integral form of the conservation of number boundary condition eq (2.22), namely

$$\frac{d}{dt}\int_{z_A}^{z_o(t_{MM}+\Delta t)} C_1(z,t)dz + \int_{y_o(t_{MM}+\Delta t)}^{y_B} C_2(y,t)dy$$

$$= - D_1 \partial_z C_1\Big|_{z=z_A} + D_2 \partial_y C_2\Big|_{y=y_B} \qquad (5.27)$$

where $z_A = (NL - 1)\Delta z$ and $y_B = (NR - 1)\Delta y$.

Furthermore, it is assumed that the moving boundary $z_o(t)$ is such that $(NL - 1)\Delta z \leq z_o(t) \leq NL\Delta z$, for all times t between t_{MM} and $t_{MM} + \Delta t$. Similarly, for times in this same interval the moving boundary $y_o(t)$ is assumed to satisfy

$$(NR - 2)\Delta y \leq y_o(t) \leq (NR - 1)\Delta y\ .$$

With these restrictions, eq (5.27) is equivalent to the differential boundary condition eq (2.16).

Before proceeding with the discretization of eq (5.27), it is convenient to introduce immediately the FORTRAN notation in table 10.

Table 10: FORTRAN Symbols Used in Eq (5.28).

$C_1((NL - 1)\Delta z, t_{MM} + \Delta t)$	C1(NL, TTMMP1)
$C_1((NL - 1)\Delta z, t_{MM})$	C1(NL, TTMM)
$C_2((NR - 1)\Delta y, t_{MM} + \Delta t)$	C2(NR, TTMMP1)
$C_2((NR - 1)\Delta y, t_{MM})$	C2(NR, TTMM)
$C_1(z_o(t_{MM} + \Delta t), t_{MM} + \Delta t)$	CB(1, TTMMP1)
$C_2(y_o(t_{MM} + \Delta t), t_{MM} + \Delta t)$	CB(2, TTMMP1)
$C_1(z_o(t_{MM}), t_{MM})$	CB(1, TTMM)
$C_2(y_o(t_{MM}), t_{MM})$	CB(2, TTMM)

By using the backward difference quotient approximation in eq (A7) to approximate the time derivatives on the left-hand side of eq (5.27) and using the trapezoidal rule to approximate the integrals in the resulting expression, we find (using the notation introduced in table 10) that the left-hand side of eq (5.27) is equivalent to

$$\begin{aligned} &\frac{S(1)\Delta z}{2\Delta t}\,C1(NL, TTMMP1) + CB(1, TTMMP1) + \frac{S(2)\Delta y}{2\Delta t}\,CB(2, TTMMP1) \\ &\quad + C2(NR, TTMMP1) - \frac{SM(1)\Delta z}{2\Delta t}\,C1(NL, TTMM) \\ &\quad + CB(1, TTMM) - \frac{SM(2)\Delta y}{2\Delta t}\,CB(2, TTMM) + C2(NR, TTMM) \\ &\quad + \text{higher order terms in } \Delta y, \Delta z, \Delta t\ , \end{aligned} \tag{5.28}$$

where

$$\begin{aligned} S(1) &= [z_o(t_{MM} + \Delta t) - (NL - 1)\Delta z]/\Delta z \\ SM(1) &= [z_o(t_{MM}) - (NL - 1)\Delta z]/\Delta z \\ S(2) &= [(NR - 1)\Delta y - y_o(t_{MM} + \Delta t)]/y \\ SM(2) &= [(NR - 1)\Delta y - y_o(t_{MM})]/\Delta y\ . \end{aligned} \tag{5.29}$$

These are consistent with previous definitions of S(1) and S(2).

When eqs (5.14) and (5.22) (re-expressed in the FORTRAN notation of table 10) are employed to approximate the spatial derivatives on the right-hand side of eq (5.27) and these approximations along with eq (5.28) are substituted into eq (5.27), there results after simplification and up to terms of higher order

$$\frac{S(1)}{R(1)}\,C1(NL,\ TTMMP1) + CB(1,\ TTMMP1) + \frac{R \cdot S(2)}{R(2)}\,CB(2,\ TTMMP1)$$

$$+ C2(NR,\ TTMMP1) - \frac{SM(1)}{R(1)}\,C1(NL,\ TTMM) + CB(1,\ TTMM)$$

$$- \frac{R \cdot SM(2)}{R(2)}\,CB(2,\ TTMM) + C2(NR,\ TTMM)$$

$$= -D_1 \left\{ - \frac{S(1)}{1 + S(1)}\,C1(NL - 1,\ TTMMP1) \right.$$

$$+ (S(1) - 1)\,C1(NL,\ TTMMP1) \qquad (5.30)$$

$$\left. + \frac{1}{S(1)(1 + S(1))}\,CB(1,\ TTMMP1) \right\}$$

$$+ R \left\{ - \frac{1}{S(2)(1 + S(2))}\,CB(2,\ TTMMP1) \right.$$

$$+ \frac{1 - S(2)}{S(2)}\,C2(NR,\ TTMMP1)$$

$$\left. + \frac{S(2)}{1 + S(2)}\,C2(NR + 1,\ TTMMP1) \right\} ,$$

where

$$R = \frac{RM}{DR}$$

$$RM = \frac{\Delta z}{\Delta y} \qquad (5.31)$$

$$DR = \frac{1}{D_2} .$$

In order to eliminate the division by S(2) (which could have the value zero) in the last term on the right-hand side of eq (5.30), we eliminate that term by substituting from the relation

$$- \frac{1}{S(2)(1 + S(2))}\,CB(2,\ TTMMP1) + \frac{1 - S(2)}{S(2)}\,C2(NR,\ TTMMP1)$$

$$+ \frac{S(2)}{1 + S(2)}\,C2(NR,\ TTMMP1) \qquad (5.32)$$

$$= C2(NR + 1,\ TTMMP1) - C2(NR,\ TTMMP1)$$

$$+ \frac{1}{R(2)}\,[-C2(NR,\ TTMMP1) + C2(NR,\ TTMM)] .$$

This latter equation is just the formula [equivalent before simplification to eq (5.24)] that results when eqs (5.20), (5.21), and (5.22) are substituted into eq (5.19) and the FORTRAN notation of table 9 introduced.

If the terms in the brackets on the right-hand side of eq (5.30) are eliminated by substitution from eq (5.32) and if the resulting equation is multiplied through by S(1) (to prevent possible division by zero), then we obtain [after CB(2, TTMMP1) has been replaced by (m = SEG) · CB(1, TTMMP1) using boundary condition eq (2.23)]:

$$\left[\frac{S(1)^2}{R(1)} + \frac{D_1}{1 + S(1)} + \frac{SEG \cdot S(1) \cdot S(2)}{R(2)}\right] CB(1, TTMMP1)$$

$$= \left[-\frac{S(1)^2}{R(1)} + (1 - S(1)) \cdot D_1\right] C1(NL, TTMMP1)$$

$$+ \left[-\frac{R \cdot S(1) \cdot S(2)}{R(2)} - R \cdot S(1)\left(1 + \frac{1}{R(2)}\right)\right] C2(NR, TTMMP1)$$

$$+ \frac{S(1)^2 \cdot D_1}{1 + S(1)} C1(NL - 1, TTMMP1) + R \cdot S(1)\, C2(NR + 1, TTMMP1)$$

$$+ \frac{S(1) \cdot SM(1)}{R(1)}\left[C1(NL, TTMM) + CB(1, TTMM)\right] \qquad (5.33)$$

$$+ \frac{R \cdot SM(2) \cdot SM(1)}{R(2)} CB(2, TTMM)$$

$$+ S(1) \cdot \left(\frac{R \cdot SM(2)}{R(2)} + \frac{R}{R(2)}\right) C2(NR, TTMM) \ .$$

Further FORTRAN notation in table 11 can be introduced into eq (5.33) and by using intermediate variables, it can be more compactly rewritten:

$$DM1 \cdot CB(1, TTMMP1) = DM2 \cdot C1(NL, TTMMP1) + DM3 \cdot C2(NR, TTMMP1)$$
$$+ DM4 \cdot C1(NL - 1, TTMMP1) + DM5 \cdot C2(NR + 1, TTMMP1) + DM6 \ , \qquad (5.34)$$

where

$$DM1 = \frac{S(1)**2}{R(1)} + \frac{DF1}{1 + S(1)} + \frac{SEG \cdot RT \cdot S(1) \cdot S(2)}{R(2)} \ ,$$

$$DM2 = -\frac{S(1)**2}{R(1)} + DF1 \cdot (1 - S(1)) \ ,$$

$$DM3 = -\frac{RT \cdot S(1) \cdot S(2)}{R(2)} - RT \cdot S(1) \cdot \left(1 + \frac{1}{R(2)}\right) \ ,$$

$$DM4 = \frac{DF1 \cdot S(1)**2}{1 + S(1)} \ , \qquad (5.34')$$

$$DM5 = RT \cdot S(1) \text{ , and}$$

$$DM6 = \frac{S(1) \cdot SM(1)}{R(1)} C1(NL, TTMM) + CB(1, TTMM)$$

$$+ \frac{RT \cdot SM(2) \cdot S(1)}{R(2)} CB(2, TTMM)$$

$$+ S(1) \cdot \frac{RT \cdot SM(2) - RT}{R(2)} \cdot C2(NR, TTMM) \text{ ,}$$

and where

$$R = RT = \frac{DF2 \cdot DZ}{DY} \text{ ,}$$

$$R(1) = \frac{2 \cdot DT}{DZ**2} \text{ ,}$$

$$R(2) = \frac{2 \cdot DF2 \cdot DT}{DY**2} \text{ ,}$$

$$S(1) = (ZOP1 - (NL - 1) \cdot DZ)/DZ \text{ ,} \qquad (5.35)$$

$$S(2) = ((NR - 1)DY - YOP1)/DY \text{ ,}$$

$$SM(2) = ((NR - 1)DY - y_o(t_{MM}))/DY \text{ , and}$$

$$SM(1) = (z_o(t_{MM}) - (NL - 1)DZ)/DZ \text{ .}$$

Table 11: FORTRAN Version of Symbols in Eq (5.33).

Δt	DT
Δy	DY
Δz	DZ
$z_o(t_{MM} + \Delta t)$	ZOP1
$y_o(t_{MM} + \Delta t)$	YOP1
D_1	DF1
D_2	DF2

SUMMARY

Schematic summary of the derivation of the FORTRAN equation for the z-grid point at the moving boundary, i.e., at $(z_o(t_{MM} + \Delta t)$, when NL > 1:

$$(5.27) \longrightarrow (5.33) \longrightarrow (5.34) \text{ and } (5.34') \text{ .}$$

Summary notes:

(1) All the concentrations on the right-hand side of eq (5.34), i.e., C1(NL, TTMMP1),...,C2(NR + 1, TTMMP1) will be eliminated in terms of known quantities and thereby CB(1, TTMMP1) will be found. This is carried through in section 6.10.

(2) The concentrations in DM6 are known because they are evaluated at time TTMM.

5.7. The Grid Point Is at the Moving Boundary and NL = 1

The derivation of the preliminary equations for $C1(z_o(t_{MM} + \Delta t), t_{MM} + \Delta t)$ is very similar to the derivation of the preliminary equation for the same quantity in the previous section, so if the user desires more details than are given below, he should refer to that section.

We again begin with the conservation eq (5.27). However, instead of approximating the spatial derivative at $z = z_A$ by eq (5.14), we use boundary condition eq (2.24). Because of this change, we find in place of eq (5.30)

$$\frac{S(1)}{R(1)}\,[C1(NL, TTMMP1) + CB(1, TTMMP1)]$$

$$+ \frac{R \cdot S(2)}{R(2)}\,[CB(2, TTMMP1) + C2(NR, TTMMP1)]$$

$$- \frac{SM(1)}{R(1)}\,[C1(NL, TTMM) + CB(1, NR, TTMM)]$$

$$- \frac{R \cdot SM(2)}{R(2)}\,[CB(2, TTMM) + C2(NR, TTMM)] \qquad (5.36)$$

$$= -\,[+\,C_p\,\Delta z(C1(1, TTMMP1) - C_{out})]$$

$$+ R\left[-\frac{1}{S(2)(1 + S(2))}\,CB(2, TTMMP1) + \frac{1 - S(2)}{S(2)}\,C2(NR, TTMMP1)\right.$$

$$\left. + \frac{S(2)}{1 + S(2)}\,C2(NR + 1, TTMMP1)\right] .$$

When the term in the second parenthesis on the right-hand side of eq (5.36) is eliminated by using eq (5.32), there results

$$\frac{S(1)}{R(1)}\,[C1(NL, TTMMP1) + CB(1, TTMMP1)]$$

$$+ \frac{R \cdot S(2)}{R(2)}\,[CB(2, TTMMP1) + C2(NR, TTMMP1)]$$

$$- \frac{SM(1)}{R(1)} [Cl(NL, TTMM) + CB(1, TTMM)]$$

$$- \frac{R \cdot SM(2)}{R(2)} [CB(2, TTMM) + C2(NR, TTMM)] \qquad (5.37)$$

$$= -[C_p \Delta z(Cl(1, TTMMP1) - C_{out})]$$

$$+ R [C2(NR + 1, TTMMP1) - C2(NR, TTMMP1)]$$

$$+ \frac{R}{R(2)} [-C2(NR, TTMMP1) + C2(NR, TTMM)] .$$

By rearranging terms and introducing the FORTRAN symbols in eq (5.35) into eq (5.37), we get

$$DM1 \cdot CB(1, TTMMP1) = DM2 \cdot Cl(NL, TTMMP1) + DM3 \cdot C2(NR, TTMMP1)$$

$$+ DM4 \cdot Cl(NL - 1, TTMMP1) \qquad (5.38)$$

$$+ DM5 \cdot C2(NR + 1, TTMMP1) + DM6 ,$$

where

$$DM1 = \frac{S(1)}{R(1)} + \frac{RT \cdot S(2) \cdot SEG}{R(2)} ,$$

$$DM2 = - \frac{S(1)}{R(1)} + CONPRO \cdot DZ ,$$

$$DM3 = - \frac{RS(2)}{R(2)} - R \left(1 + \frac{1}{R(2)}\right) , \qquad (5.38')$$

$$DM4 = 0 ,$$

$$DM5 = RT , \text{ and}$$

$$DM6 = \frac{SM(1)}{R(1)} [Cl(NL, TTMM) + CB(1, TTMM)]$$

$$+ \frac{RT \cdot SM(2)}{R(2)} [CB(2, TTMM)]$$

$$+ \frac{RT \cdot SM(2) - RT}{R(2)} [C2(NR, TTMM)]$$

$$+ DZ \cdot CONPRO \cdot CONOUT ,$$

and where the FORTRAN symbols above are the same as in eq (5.35) with the addition of the ones in table 12.

Table 12: Additional FORTRAN Symbols for Eq (5.38).

C_p	= CONPRO
C_{out}	= CONOUT

Notice that the quantities DM3, DM5, and DM6 [with DZ · CONPRO · CONOUT added to it in eq (5.34′)] are equal to their counterparts in eq (5.38′) when S(1) = 1. Hence, by setting S(1) to 1 when we are computing these quantities, i.e., in the case NL = 1, we can use the same FORTRAN expression as is used to compute these quantities in eq (5.34′) when NL > 1.

Of course in computing in this way we must leave the S(1) in DM1 and DM2 of eq (5.38′) unchanged. This can be done, when NL = 1, by introducing the quantity SAVE which is equal to S(1) and replacing S(1) where it appears in DM1 and DM2 by the symbol SAVE.

SUMMARY

Schematic summary for obtaining the preliminary equation for the z-grid point at the moving boundary when NL = 1.

$$(5.27) \longrightarrow (5.37) \longrightarrow (5.38) \text{ and } (5.38') \ .$$

Summary notes:

(1) Since the coefficients of the concentrations on the right-hand side of eq (5.38) are the same as those on the right-hand side of eq (5.34), the concentrations from both of these preliminary equations can be eliminated by the same procedure which will be described in section 6.10.

(2) The actual computation of the quantities in eq (5.38′) are carried through as described above in computing block #4 of subroutine CENTER, see section 7.2.

5.8. The Index of the Z-Grid Point is 1 and NL > 1

The derivation of the tridiagonal algebraic equation that is associated with the z-grid point 1 (when the moving boundary has reached the second z-grid point) begins with the conservation of number eq (2.20)

$$\frac{d}{dt}\int_{z_A}^{z_B} C_1(z,t)dz = -D_1\partial_z C_1\Big|_{z=z_A} + D_1\partial_z C_1\Big|_{z=z_B} \qquad (5.39)$$

where $z_A = 0$ and $z_B = \Delta z/2$ denote the positions of the flux boundaries.

After approximating the time derivative on the left-hand side of eq (5.38) by the back difference quotient approximation eq (A7) and the integrals by quadrature formulas with a node at the left-hand end point, we find

$$\frac{d}{dt}\int_0^{z_o(t_{MM}+\Delta t)} C_1(z,t)dz = \frac{1}{\frac{2\Delta t}{\Delta z}} [C_1(0 \cdot \Delta z,\ t_{MM} + \Delta t) - C_1(0 \cdot \Delta z,\ t_{MM})] \tag{5.40}$$

$$+ 0(\Delta z^2 + \Delta t^2/\Delta t) \ .$$

By substituting from eq (2.24),

$$C_p(Cl(0 \cdot \Delta z,\ t_{MM} + \Delta t) - C_{out})$$

for $D_1 \partial_z C_1 \Big|_{z=z_A}$ on the right-hand side of eq (5.39) and then

$$D_1(C_1(\Delta z,\ t_{MM} + \Delta t) - C_1(0 \cdot \Delta z,\ t_{MM} + \Delta t))$$

for $D_1 \partial_z C_1 \Big|_{z=z_B}$ on the right-hand side of eq (5.39), that equation

yields together with eq (5.40) after simplification

$$\left[\left(\frac{2\Delta t}{\Delta z^2}\right)^{-1} + \Delta z \cdot C_p + D_1\right] C_1(0 \cdot \Delta z,\ t_{MM} + \Delta t) = D_1 C_1(\Delta z,\ t_{MM} + \Delta t) \tag{5.41}$$

$$+ \frac{C_1(\Delta z,\ t_{MM})}{\frac{2 \cdot \Delta t}{\Delta z^2}} + \Delta z \cdot C_p \cdot C_{out} \ .$$

After introducing into eq (5.41) the FORTRAN notation in table 13, that equation becomes

$$\left[\left(\frac{2DT}{DZ^2}\right)^{-1} + DZ \cdot CONPRO + DF1\right] Cl(y,\ TTMMP1) = DF1 \cdot Cl(2,\ TTMMP1) \tag{5.42}$$

$$+ \frac{Dl(1,\ TTMM)}{\frac{2DT}{DZ**2}} + DZ \cdot CONPRO \cdot CONOUT \ .$$

Table 13: FORTRAN Version of Quantities in Eq (5.41).

Δt	DT
Δz	DZ
C_p	CONPRO
C_{out}	CONOUT
D_1	DF1
$C_1(0 \cdot \Delta z, t_{MM} + \Delta t)$	C1(1, TTMMP1)
$C_1(0 \cdot \Delta z, t_{MM})$	C1(1, TTMM)
$C_2(\Delta z, t_{MM} + \Delta t)$	C2(2, TTMMP1)

Equation (5.42) is the FORTRAN tridiagonal equation which will give rise to the "P, Q" quantities associated with the z-grid point 1 and unknown C1(1, TTMMP1). This is carried through in section 6.4.

SUMMARY

Schematic summary of the derivation of the FORTRAN tridiagonal equation for the z-grid point 1 when the moving boundary is beyond z-grid point 2.

$$(5.39) \longrightarrow (5.41) \longrightarrow (5.42).$$

Summary note:

> The FORTRAN eq (5.42) leads to values of P1(1), Q1(1) that are computed in computational block #24 of the main program. See section 6.4 for the derivation of these "P, Q" quantities.

5.9. The Index of the Z-Grid Point is 1 and NL = 1

This equation is different from all the others in that it is not developed from an integral conservation equation. It is basically derived from combining eq (2.14) with a Taylor series expansion.

From a spatial Taylor's formula expansion of $C_1(z, t_{MM} + \Delta t)$, we have

$$C_1(z_o(t_{MM} + \Delta t)) = C_1(0, t_{MM} + \Delta t) + \partial_z C_1\Big|_{z=0} z_o(t_{MM} + \Delta t) + 1/2\ \partial C_1\Big|_{z=0} (z_o(t_{MM} + \Delta t))^2 + O(\Delta z^3)\ . \tag{5.43}$$

By defining

$$S(1) = z_o(t_{MM} + \Delta t)/\Delta z$$

and substituting this together with (from eq (2.14))

$$\partial_z{}^2 C_1 = \partial_t C_1 / D_1$$

into eq (5.43), it becomes after rearranging and up to higher order terms

$$D_1 C_1(z_o(t_{MM} + \Delta t),\ t_{MM} + \Delta t) = D_1 C_1(0,\ t_{MM} + \Delta t) + D_1 \partial_z C_1\Big|_{z=0} (S(1)\Delta z) + 1/2\ \partial_t C_1\Big|_{z=0} (S(1) \cdot \Delta z)^2 \ . \tag{5.44}$$

When the backward difference approximation,

$$\partial_t C_1 = (C_1(0,\ t_{MM} + \Delta t) - C_1(0,\ t_{MM}))/\Delta t + 0(\Delta t) \ ,$$

and the expression from eq (2.24),

$$D_1 \partial_z C_1\Big|_{z=0} = C_p(C_1(0,\ t_{MM} + \Delta t) - C_{out}) \ , \tag{5.45}$$

are substituted into eq (5.44), it becomes, after rearrangement (ignoring higher order terms),

$$\left[D_1 + \frac{(S(1)\Delta z)^2}{\Delta t \cdot 2} + S(1) \cdot \Delta z \cdot C_p\right] C_1(0,\ t_{MM} + \Delta t) = D_1 C_1(z_o(t_{MM} + \Delta t),\ t_{MM} + \Delta t) + C_1(0,\ t_{MM}) \cdot \frac{(S(1) \cdot \Delta z)^2}{2\Delta t} + S(1) \cdot \Delta z \cdot C_p \cdot C_{out} \ . \tag{5.46}$$

By transferring the symbols in eq (5.46) through the correspondences in table 14, that equation becomes

$$DF1 + \frac{(S(1) \cdot DZ)**2}{2 \cdot DT} + S(1) \cdot DZ \cdot CONPRO \cdot C1(1,\ TTMMP1) = DF1\ CB(1,\ TTMMP1) + C1(1,\ TTMM)\ \frac{(S(1) \cdot DZ)**2}{2DT} + S(1) \cdot DZ \cdot CONPRO \cdot CONOUT \ . \tag{5.47}$$

Table 14: FORTRAN Version of Quantities in Eq (5.46).

D_1	DF1
Δz	DZ
Δt	DT
$C_1(0, t_{MM} + \Delta t)$	C1(1, TTMMP1)
$C_1(0, t_{MM})$	C1(1, TTMM)
$C_1(z_o(t_{MM} + \Delta t), t_{MM} + \Delta t)$	CB(1, TTMMP1)
C_p	CONPRO
C_{out}	CONOUT

FORTRAN tridiagonal eq (5.47) will lead to the quantities P1(1), Q1(1) in section 6.3. These are the "P, Q" associated with z-grid point 1 when the moving boundary has not yet reached the second z-grid point.

Notice that when $t_{MM} = 0$ the quantity C1(1,0) is unknown, i.e., it is not given as data in the problem. Thus, the equation we use in place of eq (5.46) for the first iteration MM = 1 when $t_{MM} = 0$ is

$$(DF1 + S(1) \cdot DZ \cdot CONPRO) \cdot C1(1, TTMMP1) = DF1 \cdot CB(1, TTMMP1) + S(1) \cdot DZ \cdot CONPRO \cdot CONOUT \ . \tag{5.48}$$

This is the equation we would have arrived at if we had truncated the Taylor's formula in eq (5.43) to first order terms and proceeded just as above, and eq (5.48) is consistent with, i.e., the same as, the formula we would get by approximating eq (5.45) directly.

SUMMARY

Schematic summary of the derivation of the FORTRAN tridiagonal equation associated with the z-grid point 1 when NL = 1,

(5.43) $\longrightarrow$ (5.45) $\longrightarrow$ (5.47) or (5.48) (use (5.48) when $t_{MM} = 0$).

Summary note:

The "P, Q" terms that result from eq (5.47) or eq (5.48) are derived in sections 6.2 and 6.3.

6. METHOD OF SOLVING THE ALGEBRAIC EQUATIONS

6.1. The Derivation of the "Double Sweep" "P, Q" Coefficients Associated with the Grid Points

In this section we put each of the tridiagonal FORTRAN equations associated with the fixed grid points in "double sweep" form. Specifically, we manipulate these tridiagonal FORTRAN eqs (3.2) and (3.3) into the forms

$$C1(J, TTMMP1) = P1(J)\ C1(J + 1, TTMMP1) + Q1(J) \qquad (6.1)$$

for J = 1,...,NL - 1, and

$$C1(NL, TTMMP1) = PA(1)\ CB(1, TTMMP1) + QA(1)\ . \qquad (6.2)$$

Also, at the fixed grid points in the y-region,

$$C2(J, TTMMP1) = P2(J)\ C1(J + 1, TTMMP1) + Q2(J) \qquad (6.3)$$

for J = NR + 1,...,N2, and

$$C2(NR, TTMMP1) = PA(2)\ CB(2, TTMMP1) + QA(2)\ . \qquad (6.4)$$

Once again the specific composition of the double sweep "P, Q" factors associated with a grid point depends on the location of the grid point and the position of the moving boundary at time TTMMP1. The following table 15 completely classifies all the cases that arise. Of course, because the "P, Q" coefficients for a grid point are obtained from the tridiagonal equation associated with that grid point, this table is the same as table 3 except for Case I(e) and Case II(g) which were not included in table 3 (since they are so trivial).

Table 15: Guide to the Subsections Where the "P, Q" Coefficients Associated With Each of the Fixed Grid Points Are Derived and Described.

Case I: The index NL equals 1:

(a) The first z-grid point, see sections 6.2 and 6.3.
(b) The index of the y-grid point is greater than NR, see section 6.8.
(c) The index of the y-grid point is NR, see section 6.9.
(d) The grid point lies at the moving boundary, see section 6.10.
(e) The largest y-grid point, which has index N2 and is located at y = BDRY = NYO • DY, see section 6.7.

Case II: The index NL is greater than 1.

(a) The index of the z-grid point is 1, see section 6.4.
(b) The index of the z-grid points ranges from 2 to NL - 1, see section 6.8.
(c) The index of the z-grid point is NL, see section 6.6.
(d) The grid point is at the moving boundary, see section 6.10.
(e) The index of the y-grid point is greater than NR, see section 6.8.
(f) The index of the y-grid point is NR, see section 6.9.
(g) The index of the y-grid point is N2 and is at y = BDRY, see section 6.7.

6.2. The First Z-Grid Point When NL = 1 and t_{MM} = TTMM = 0

After dividing both sides of eq (5.48) by (DF1 + S(1) · DZ · CONRPO), we get

$$C1(1, TTMMP1) = P1(1) \cdot CB(1, TTMMP1) + Q1(1) \tag{6.5}$$

where

$$\begin{aligned} P1(1) &= DM \cdot DF1 \\ Q1(1) &= DM \cdot (C1(1) \cdot DM3 + DM1 \cdot CONPRO \cdot CONOUT) \\ DM1 &= S(1) \cdot DZ \\ DM2 &= 0 \\ DM3 &= DM2 \\ DM &= (DF1 + DM2 + DM1 \cdot CONPRO)^{-1} . \end{aligned} \tag{6.5'}$$

The introduction of the zero quantities DM2, DM3 into the above equations enables us to use below in section 6.3 the same expression DM but with DM2, DM3 $\neq$ 0.

SUMMARY

Equations (6.5′) are calculated in computational block #23 of the main program (see sec. 7.1).

6.3. The First Z-Grid Point When NL = 1 and t_{MM} = TTMM > 0

When eq (5.47) is divided through by $DF1 + \frac{(S(1) \cdot DZ)^{**2}}{2\ DT} + S(1) \cdot DZ \cdot CONPRO$, there results

$$C1(1, TTMMP1) = P1(1)\ CB(1, TTMMP1) + Q1(1) \tag{6.6}$$

with

$$
\begin{aligned}
&Pl(l) = DM \cdot DFl \\
&Ql(l) = DM \cdot (Cl(l, TTMM) \cdot DM3 + DMl \cdot CONPRO \cdot CONOUT) \\
&DM = (DFl + DM2 + DMl \cdot CONPRO)^{-1} \\
&DMl = S(l) \cdot DZ \\
&DM2 = (DMl)^2/2 \cdot DT \\
&DM3 = DM2 \ .
\end{aligned}
\tag{6.6'}
$$

SUMMARY

Note that all quantities in eq (6.6′) are the same as in eq (6.5′) except for DM2 and DM3. The calculation of the quantities in eq (6.6′) takes place in computational block #23 of the main program (see sec. 7.1).

6.4. The Index of the Z-Grid Point is 1 and NL > 1

After dividing eq (5.42) by $\frac{(DZ)**2}{2 \cdot DT} + DZ \cdot CONPRO + DFl)$, it becomes

$$Cl(1, TTMMPl) = Pl(1) \cdot Cl(2, TTMMPl) + Ql(1) \tag{6.7}$$

where

$$
\begin{aligned}
&Pl(l) = DFl \cdot DM \\
&Ql(l) = DM\ Cl(l, TTMM) \cdot \frac{DZ**2}{2 \cdot DT} + DZ \cdot CONPRO \cdot CONOUT \\
&DM = \frac{DZ**2}{2 \cdot DT} + DZ \cdot CONPRO + DFl \ .
\end{aligned}
\tag{6.7'}
$$

SUMMARY

The quantities in eq (6.7′) are computed in computational block #24 of the main program (see sec. 7.1).

6.5. The Index of the Z-Grid Point is Between 1 and NL - 1 and NL > 2

We begin with eq (5.7) which holds for the indices J = 2,...,NL - 1. Assume that for one of the indices in this range

$$Cl(J - 1, TTMMPl) = Pl(J - 1) \cdot Cl(J, TTMMPl) + Ql(J - 1) \ . \tag{6.8}$$

After substituting this value of Cl(J - 1, TTMMPl) into the left-hand side of eq (5.7) and rearranging the terms, there results

$$Cl(J, TTMMPl) = Pl(J) \cdot Cl(J + 1, TTMMPl) + Ql(J) \tag{6.9}$$

where

$$
\begin{aligned}
P1(J) &= -A2R/D1(J) \\
Q1(J) &= (AF(J) - Q1(J-1)A2L)/D1(J) \\
D1(J) &= A2M = P1(J-1) \cdot A2L \ .
\end{aligned}
\tag{6.9'}
$$

Notice that eq (6.8) with J = 2 holds because of eq (6.7). Hence, by induction we get eqs (6.9) and (6.9′) for J = 2,...,NL - 1.

SUMMARY

Summary notes:

(1) The arrays P1(J), D1(J) are computed in computational block #3 of subroutine TRIDNL.

(2) The quantities A2L,...,A1M (see table 5) are computed in subroutine INTER which is called by subroutine TRIDNL when P1(J), D1(J) are being computed.

(3) The arrays AF(J), Q1(J) are calculated in computational block #25 of the main program.

6.6 The Index of the Z-Grid Point is NL and NL > 1

When eq (5.17) is multiplied through by S(1) (since S(1) may be zero, this prevents possible division by zero) and the terms are transposed, there results

$$
\begin{aligned}
DM1\ C1(NL, TTMMP1) &= DM2\ CB(1, TTMMP1) \\
&\quad + DM3\ C1(NL-1, TTMMP1) + DM4 \ ,
\end{aligned}
\tag{6.10}
$$

with

$$
\begin{aligned}
DM &= 1 + S(1) \\
DM &= DF1 + \frac{S(1)}{R(1)} \\
DM2 &= DF1/(1 + S(1)) = DF1/DM \\
DM3 &= DF1 \cdot S(1)/(1 + S(1)) = DF1 \cdot DM2 \\
DM4 &= \frac{S(1)}{R(1)}\ C1\ (NL, TTMM) \ .
\end{aligned}
\tag{6.10'}
$$

We also have from eq (6.9) with J = NL - 1,

$$
C1(NL-1, TTMMP1) = P(1)\ C1(NL, TTMMP1) + Q(1) \tag{6.11}
$$

where we have used the abbreviations

$$
\begin{aligned}
P(1) &= P1(NL-1) \\
Q(1) &= Q1(NL-1) \ .
\end{aligned}
\tag{6.11'}
$$

When eq (6.11) is substituted into the right-hand side of eq (6.10) and the result is simplified, we find

$$C1(NL, TTMMP1) = PA(1)\ CB(1, TTMMP1) + QA(1) \tag{6.12}$$

where

$$\begin{aligned} PA(1) &= DM2/(DM1 - DM3 \cdot P(1)) \\ QA(1) &= (DM4 + Q(1) \cdot DM3)/(DM1 - DM3 \cdot P(1)) \ . \end{aligned} \tag{6.12'}$$

SUMMARY

The quantities in eqs (6.10′) and (6.12′) are evaluated in computational block #3 of subroutine CENTER.

6.7. The Index of the Y-Grid Point is N2 = NYO + 1 and is located at y = BDRY

The boundary condition for C2(,) at the boundary point y = BDRY is (see eq (2.12))

$$C2(N2, TTMMP1) = CBULK \ . \tag{6.13}$$

We rewrite this simple equation for use later on in the more convenient form

$$C2(N2, TTMMP1) = P2(N2)\ C2(N2 - 1, TTMMP1) = Q2(N2) \ , \tag{6.14}$$

if we assume (and we do)

$$\begin{aligned} P2(N2) &= 0 \\ Q2(N2) &= CBULK \ . \end{aligned} \tag{6.14'}$$

SUMMARY

The quantities in eq (6.14′) are calculated in computational block #13 of the main program. These quantities are used in block #26 of the main program (see sec. 7.1) and block #4 of subroutine INTER (see sec. 7.3).

6.8. The Index of the Y-Grid Point is Between NR and N2

We start with tridiagonal eq (5.10) which holds for J = NR + 1,..., N2 - 1.

Assume for one of the indices in this range that

$$C2(J + 1, TTMMP1) = P2(J + 1)\ C2(J, TTMMP1) + Q2(J + 1) \ . \tag{6.15}$$

After substituting this value of C2(J + 1, TTMMP1) into the left-hand side of eq (5.10), with J in place of J - 1, and rearranging terms, it becomes for J = NR + 1,..., N2 - 1

$$C2(J, TTMMP1) = P2(J)\ C2(J - 1, TTMMP1) + Q2(J) \tag{6.16}$$

where

$$\begin{aligned} P2(J) &= -\frac{1}{D2(J)} \\ Q2(J) &= (BF(J) - Q2(J + 1)\ B2R)/D2(J) \\ D2(J) &= B2M + P2(J + 1)B2R \end{aligned} \tag{6.16'}$$

or, since B2R = 1 from table 7,

$$D2(J) = B2M + P2(J + 1) \ .$$

Notice that eq (6.15) with J = N2 - 1 holds because of eq (6.14). Hence, by induction we get eq (6.16) and eq (6.16′) for J = NR + 1,...,N2 - 1.

SUMMARY

Summary notes:

(1) The arrays P2(J), D2(J) are calculated in computational block #4 of subroutine TRIDNL.

(2) The quantities B2M, B1M (the other B's equal 1) are computed in subroutine INTER which is called by subroutine TRIDNL when P2(J), D2(J) are computed.

(3) The arrays Q2(J), BF(J) are calculated in computational block #26 of the main program.

6.9. The Y-Grid Point has Index NR

After transposing terms in the tridiagonal FORTRAN eq (5.26), it takes the form

$$\begin{aligned} DM1\ C2(NR, TTMMP1) = {} & DM2\ CB(2, TTMMP1) \\ & + DM3\ C2(NR + 1, TTMMP1) + DM4 \end{aligned} \tag{6.17}$$

where

$$\begin{aligned} DM1 &= 1 + \frac{S(2)}{R(2)} \\ DM2 &= \frac{1}{1 + S(2)} \\ DM3 &= S(2)/(1 + S(2)) \\ DM4 &= \frac{S(2)}{R(2)}\ C2(NR, TTMM) \ . \end{aligned} \tag{6.17'}$$

We also have from eq (6.16) with J = NR + 1

$$C2(NR + 1, TTMMP1) = P(2) \cdot C2(NR, TTMMP1) + Q(2) \qquad (6.18)$$

where we have used the abbreviations

$$\begin{aligned} P(2) &= P2(NR + 1) \\ Q(2) &= Q2(NR + 1) \ . \end{aligned} \qquad (6.18')$$

When eq (6.18) is substituted into the right-hand side of eq (6.17) and the result is simplified, we find

$$C2(NR, TTMMP1) = PA(2)\ CB(2, TTMMP1) + QA(2) \qquad (6.19)$$

where

$$\begin{aligned} PA(2) &= DM2/(DM1 - DM3 \cdot P(2)) \\ QA(2) &= (DM4 + Q(2) \cdot DM3)/(DM1 - DM3 \cdot P(2)) \ . \end{aligned} \qquad (6.19')$$

SUMMARY

The quantities eqs (6.17′) and (6.19′) are evaluated in computational block #3 of subroutine CENTER.

6.10. The Determination of CB(1, TTMMP1) in Terms of Known Quantities

The values of the "P, Q" coefficients in the eqs (6.1) through (6.4) are determined by the coefficients in the tridiagonal eqs (3.2) and (3.3) and these in turn are composed of known mesh parameters, material parameters and concentrations evaluated at time TTMM. Hence, by using eqs (6.1) through (6.4), we can eliminate the concentrations evaluated at time TTMMP1 that occur on the right-hand side of the "preliminary" eq (5.34) and thereby determine the value of CB(1, TTMMP1) entirely in known quantities.

More precisely, substituting eqs (6.11) and (6.18) into the right-hand side of the "preliminary equation," i.e., eq (5.34) for NL > 1 (and eq (5.38) for NL = 1), we get

$$\begin{aligned} DM1\ CB(1, TTMMP1) &= DM8 \cdot C1(NL, TTMMP1) \\ &\quad + DM9\ C2(NR, TTMMP1) + DM10 \end{aligned} \qquad (6.20)$$

where

$$\begin{aligned} DM8 &= DM2 + DM4 \cdot P(1) \\ DM9 &= DM3 + DM5 \cdot P(2) \\ DM10 &= DM6 + Q(1) \cdot DM4 + Q(2) \cdot DM5 \ . \end{aligned} \qquad (6.20')$$

Finally eliminating the concentrations at the grid point NL and NR on the right-hand side of eq (6.20) by use of eqs (6.12) and (6.19) leads to the

expression for CB(1, TTMMP1) in terms of known quantities

$$CB(1, TTMMP1) = \frac{DM11}{DM12} , \tag{6.21}$$

with

$$\begin{aligned} DM12 &= DM1 - DM8 \cdot PA(1) - SEG \cdot DM9 \cdot PA(2) \\ DM11 &= DM10 + QA(1) \cdot DM8 + QA(2) \cdot DM9 , \end{aligned} \tag{6.21'}$$

and where

DM1
DM2
DM3
DM4
DM5
DM6

are given in eq (5.39) if NL = 1
eq (5.34) if NL > 1

and furthermore,

PA(2), QA(2) are given in eq (6.19′)

P(2), Q(2) are given in eq (6.18′)

PA(1), QA(1) are given in eq (6.12′)

P(1), Q(1) are given in eq (6.11′)

R(1) is given in eq (5.16′)

R(2) is given in eq (5.35)

RT = R is given in eq (5.31) (6.22)

SUMMARY

The quantities in eqs (6.21), (6.21′) and all the quantities listed below eq (6.21′) are computed in computational block #4 of subroutine CENTER to find the value of CB(1, TTMMP1).

7. DETAILED DESCRIPTION OF THE PROGRAM

7.1. Block-by-Block Detailed Description of the Calculational Procedure in the Main Program

As is shown in the program listing (Chapter 11), the overall calculations in DISTRIB: VERSION 1 are broken down into a set of consecutive blocks. The computations in each of these blocks have a unified purpose which is described with pertinent comments for each of these blocks #1 through #36 in this chapter. In Chapter 8, the logic in these blocks is flow charted. The precise calculations carried through in each of these blocks are shown in the program listing in Chapter 11, and the material in the following blocks is intended to illuminate the contents by organizing and elaborating on the material in the listing rather than to redescribe the contents in a different language.

IMPORTANT: Quantities with names which appear in the program listing but do not appear in the glossary are not needed or used in DISTRIB: VERSION 1 and therefore should be ignored, i.e., all material in the listing that is not in a block or glossary.

Block #1: Input Declaration, Type Statements, Dimensionalize Arrays and Common Blocks

The arrays SRL( ) and YRL( ) are declared real because of certain constraints in the plotting routines where these arrays are employed. The fact that they are dimensionalized by 200 means that only 200 data points can be plotted in a single call of the plot routine.

Arrays that have an element associated with each grid point, e.g., C2(J) (see glossary) are dimensionalized by 2000. This means the maximum number of grid points is 2000 and the maximum number of mesh widths in either region must be less than 2000.

Since the number of cells of the array C1( ) actually used is, at time TTMMP1, equal to NR · NZO, the input quantities NZO and TFINAL (which together with the equation of the moving boundary determine the largest possible value NR takes) must be constrained so that the product NR · NZO is always less than the dimension of C1( ).

Many of the arrays are dimensionalized in common blocks because they are also used in subroutines.

Block #2: Input the Proper Problem and Material Parameters with Accompanying Format Declarations

The definition of each of the input quantities is to be found in the glossary, where references are given to places in the main body of this documentation which contain more elaborate descriptions of these quantities. The list of input and output is given in section 9.2 in a systematic fashion and in classified form.

Real input is read under format 4 and real output printed under format 5. Integer input and output are under format 1.

A table listing the input data is found in section 9.2 along with ranges of values for the input parameters that have worked well in trial runs. The units for the input data are micrometers, hours. The problem with CONOUT = 0 is homogeneous in the concentrations, and therefore the input concentrations are normalized to CBULK = 1. By homogeneous is meant the value of $C_i(x,t)$ for CBULK = A is equal to $C_i(x,t)$ [for CBULK = 12 times A.

Block #3: Store Input in Terms of Their Original Dimension Units

In the next block certain input parameters are dimensionalized (rescaled). Consequently, these input quantities are stored in their original units in the array PARAM so they can be printed out later in block #34 in their original units. Also, before its defining quantities are changed, the quantity SCALE2 (in micrometers which is used to scale the distance along the y-coordinate system, i.e., in the silicon, in the plots that are printed in block #34 is calculated and stored for later output. The reason SCALE2 does not appear in the plot routine is that its role there is played by UN2 which represents the same physical quantity as SCALE2 but is in units of BLTH rather than micrometers. The fact that DY, YOP1, etc., when they appear in the plot routines are in BLTH units requires the use of UN2 instead of SCALE2. A detailed description of the dimensionalization procedure is given in section 4.3.

Block #4: Redimension the Input Data by Introducing Computationally Efficient Units

The dimensionalization carried out here is described in section 4.3 in detail. Note that, in Version 1, DFC is set equal to 1 in a declare card. Thus, during the calculations, DF2 will have the value 1 and DF1 will be the ratio DF1/DF2. The computational formula for BTM is given in eq (4.10).

Block #5: Compute the Values of the Grid Mesh Widths

This is the step described in section 4.1 where DY and DZ, the mesh widths in the y- and z-regions, respectively, are calculated. The computational formula for DY is eq (4.1) and eq (4.3) for DZ.

Block #6: Compute the Value of ZCRIT = CNSTX.

In this step CNSTX, which is another name for ZCRIT, is calculated, see section 4.2. The quantities CNA and CNB are just two intermediate variables used in computing DNSTX and ZOP1, ZOP2 later on.

Block #7: Compute the Time Increment DT for the First Iteration of the Main Loop 78

In this block the main purpose is to compute the initial value of Δt (= DT). Recall, from section 4.2, that in the "wet oxidation" Case I the time TLIN = CNTAU = $\tau = 0$, so the motion of the moving boundary is always parabolic. In this case DT is defined as the time it takes the moving boundary to move the distance DM = WFRC · DZ, having started at Z = 0. The number WFRC must be positive, real, and less than 1, for the program to be stable. In the case of a "dry oxidation," Case II in section 4.2 which starts at Z = 0 (in VERSION 1) TLIN = CNTAU = $\tau > 0$, the boundary moves linearly for the time period CNTAU with a speed SLOPE = ZCRIT/CNTAU. Hence, the time DT the moving boundary needs to transverse the distance DM (where DM is as above) is DM/SLOPE. Values of WFRC that have been successfully tested range between 0.01 and 0.33.

We have required TLIN to be less than or equal to 10^{-9} rather than zero in order to distinguish between the wet and dry oxidation cases. This does not seem now to be necessary, but we have left this as it is because in all practical cases TLIN is always much greater than 10^{-9} whenever it is greater than zero, so no harm is done.

The quantity DTS is used to store the quantity DT for recall at a later step, see computational block #17.

When CNTAU = $\tau = 0$, the quantities ZCRIT = CNSTX and SLOPE which are not used but still appear in later formulas are assigned the values 0 and 1, respectively, which do not foul up these formulas, for example by causing division by zero.

In case $\tau > 0$ (i.e., dry oxidation), WFRC must certainly be chosen so that DM < ZCRIT for the above prescription for DT to make sense. In all the cases we have run, this has been naturally satisfied when data in the literature are satisfied.

Block #8: Computation and Storage of Initial Value DT in Hours

The quantity DT just calculated in the previous block is multiplied by BTM (the unit in which time has been rescaled) to determine its magnitude in units of hours. It is stored in the array PARAM for printout in block #34.

Block #9: Calculation of Initial Values of NL, NR and Value of N2 for the First Iteration of the Main Loop 78

The quantity N1 is another name for the index, NL, of the first grid point with a z-coordinate less than or equal to the z-coordinate of the moving boundary at the beginning time of the main iteration loop 78. Because YO1, ZO1 are zero in Version 1, N1 has the value 1, as it should since at this point in the program the moving boundary is at z = 0 (recall we are computing here NL for the first iteration of loop 78) NR

(which is the index of the first grid point that has a y-coordinate greater than the y-coordinate of the moving boundary at the start of the iterations in loop 78) has the value 2.

The quantity N2 is the index of the last grid point on the y-axis and since NYO is the number of grid mesh widths in the interval (0, BDRY) (see below eq (2.11) for the definition of BDRY) N2 = NYO + 1 (see computational formula eq (4.2)).

Block #10: Calculation of Initial Values of C2( )

The initial values of array C2( ) at all the grid points are determined from the prescription in eq (2.13) of C2(y,0). Because of this prescription, the grid points from NR through NR + NH (NH is an input integer) are assigned the value CMAX (input value) while all the following grid points are assigned the value CBULK (input value).

Since in VERSION 1 no oxide is initially present, there is no necessity to assign array C1( ).

Notice that C2(1) is essentially assigned through the assignment of CB(2). This is because CB(2) denotes by definition the concentration in the silicon at the position of the moving boundary and initially the moving boundary is located at the position of the first y-grid point.

Block #11: Calculation of the Value of TTMM for MM = 1

This is the step in which TTMM, the time at the beginning of the MMth iteration of loop 78, for the first iteration is calculated. Since ZO1 in VERSION 1 has been assigned the value zero, in a declaration card, this quantity obviously has the correct value which is zero.

Block #12: Calculation of the Initial Values of SM( )

The quantities SM(1), SM(2) are, respectively, the distance between grid point NL and ZOP1 (the position of the moving boundary in the z-coordinate system at time TTMMP1) and the grid point NR (for more details see figs. 5 and 6).

The quantity FNL represents the number of mesh widths between z-grid point 1 and z-grid point NL, and FNR is the number of mesh widths between y-grid point NR and y-grid point N2 (the last grid point on the y-axis in the domain of the silicon).

Since ZO1 and YO1 are declared zero in VERSION 1, SM(1) = 0 and SM(2) = 1 as they should be when we are calculating these quantities for the initial iteration of loop 78.

Block #13: Calculation of the "P, Q" Coefficients for Grid Point N2

As has been explained in section 6.7, the satisfaction of the boundary condition C2(N2) = CBULK (see eq (2.12)) at y = BDRY is equivalent (in the way this problem is being solved) to assuming P2(N2), Q2(N2) are assigned in such a way that C2(N2) = P2(N2) C2(N2 - 1) + Q2(N2) [see eqs (6.14) and (6.14′)].

The values of P2(N2), Q2(N2) are used in subroutine TRIDNL.

Block #14: Beginning of the Main Loop which Calculates Concentrations at Time = TTMMP1

At the beginning of this loop, 78, the quantities in the arrays C1( ), C2( ) representing the concentrations in the z- and y-regions at each grid point have their values at time TTMM. At the end of the MMth iteration, these same quantities will have their proper values at time TTMMP1.

Before the main loop begins, certain starting values for quantities used in the loop are set. The definitions and use of these quantities ZOPS, TJ, etc., are explained in later steps.

The overall manner in which the new concentrations are calculated is explained in section 3.

Block #15: Setting the Print-out Control Parameter

The integer KW controls print-out in block #30. This data will be printed out after the first iteration, then skip KPRINT - 1 iterations of loop and print out the data again, and so on.

Block #16: Increasing the Distance that the Moving Boundary Travels in an Iteration of Loop 78

During the MMth iteration of loop 78, the quantity ZOACEL measures that fraction of the mesh width DZ that the moving boundary transversed during the previous iteration of loop 78. (If MM = 1, ZOACEL = 0 since ZOP1 = 0. and ZOPS = 0. In this case, see block #14.) If ZOACEL is less than 0.2 and KPOW (defined below) is greater than 100, then the magnitude to DT is increased by 0.05 in order to accelerate the distance the boundary moves during the MMth iteration of loop 78. This acceleration is put into the algorithm in order to decrease the number of iterations that are required in order to compute the solution to a problem in which the moving boundary moves a certain specified distance.

Of course, subroutine TRIDNL, which computes and stores quantities that depend on DT, has to be recomputed and the new value of DT is placed in DTS. When leaving the loop in which DT has been increased, the value of the control parameter KPOW is increased by 100 so that DT will not be increased again for at least another 100 iterations of loop 78.

The quantities 0.2 and 100 mentioned above are, of course, somewhat arbitrary, but the above values have worked satisfactorily in a great number of trial runs.

Block #17: Adjustments Required When Moving Boundary Reached New Z-Grid Point on Previous Iteration of Loop 78.

The quantity TJ will be computed in a later step, block #20, in such a way that when and only when the moving boundary reached a new z-grid point during the previous step will TJ have a value greater than or equal to 1. Hence, assuming TJ $\geq$ 1, NL, which is the first grid point less than or equal to the position of the moving boundary at the time when MMth iteration of loop 78 starts, has to be increased in value by one. The quantity FNL (the number of mesh widths to the left of NL) also must be increased by 1 and NLM1 = NL - 1, NLM2 = NL - 2, increased accordingly.

Also, C1 at the new grid point, NL, must be defined.

Furthermore, since DT has been adjusted (see block #20) during the previous iteration of loop 78, this quantity is reset to the value it had during the (MM - 2)th iteration, specifically DTS. Also, TRIDNL, the subroutine computing the arrays P1, Q1, etc., depends on DT (besides a new P1, D1 being added in the y-region) and hence needs to be recomputed.

The value SM(1) (see secs. 5.6 and 5.7 for its meaning) must be set equal to zero because during the (MM - 1)th iteration the moving boundary reached a new grid point in the z-region.

All these changes are effected when TJ > 1. Of course, in the contrary case the implication is that the moving boundary had not yet reached a new z-grid point and the above changes are not required.

Block #18: Computing the Time and Position of the Moving Boundaries at the End of the MMth Iteration of Loop 78

In this step we are simply computing what the value of the time will be at the end of the MMth iteration of loop 78. Since at the beginning of the loop, t has the value TTMM by definition and since time is made to increase by DT during the loop, the time at the end of the loop TTMMP1 is given by TTMM + DT. After this value of TTMMP1 is computed, the location of the moving boundary in the z- and y-regions, respectively, ZOP1, YOP1 can be computed through the use of formulas in section 4.2 and eq (2.8). Remember, ZW and TW equal zero in VERSION 1.

Block #19· Adjustments Required When Moving Boundary Reached New Y-Grid Point on Previous Iteration of Loop 78

The quantity SES is the fraction which the signed distance between NR and YOP1 is of the mesh width DY = ΔY. If YOP1 is beyond the y-grid point NR (during the MMth iteration of loop 78), then SES is negative

and NR, FNR, and NRP1 = NR + 1 have to be increased by 1. When SES is negative at the beginning of the MMth loop, it means (see the next block) the moving boundary is located at time TTMM on a grid point, and hence SM(2) should be set equal to 1.

Block #20: If the Moving Boundary Moves for a Time 2DT, Will It Reach a New Z-Grid Point?

During the MMth iteration of loop 78, we look at where $z_o(\ )$ is at time TTMMP2 = TTMMP1 + DT. This position is denoted as ZOP2. Then we compute TJ which is that fraction the signed distance between grid point NL and ZOP1 is of the mesh width ΔZ = DZ. If this fraction is less than one, it means that during the (MM + 1)th iteration of loop 78 the moving boundary will not reach a new z-grid point, and we proceed with the MMth iteration. On the other hand, if TJ is greater than or equal to 1, we know that the moving boundary will reach or cross grid point NL + 1 on the next iteration (MM + 1). Therefore, we decide to change the value of DT so that while still in the MMth iteration of loop 78, z_o travels all the way to the next grid point. At the time TTMMP1 when this happens, i.e., z_o(TTMMP1) = (FNL + 1), DZ is calculated (as shown in the listing) for the two cases CNTAU = 0 or CNTAU > 0. (Remember that ZW = TW = 0 in VERSION 1.) Finally, we calculate the new value of DT that will take the moving boundary $z_o(\ )$, starting at time TTMM, to its new position. Lastly, we recompute TRIDNL because the arrays P1, P2, etc., computed there depend on the value of DT.

Block #21: Calculation of the Parameters S(1), S(2) Needed for Subroutine CENTER

The quantities FNL, FNR, etc., have now been reset (if it was necessary). These quantities S(1) and S(2) (see secs. 5.4 to 5.7 for details) are used in computing the coefficients of the unknowns in the algebraic equations that will be solved later in subroutine CENTER.

Block #22: For the First Iteration of Loop 78, Call TRIDNL

In the first iteration of loop 78, the arrays P2, D2 that are needed later and which are calculated in subroutine TRIDNL have not been calculated before in the program.

Notice that whenever a new NL occurs, and hence the quantities in TRIDNL need to be updated, this is done in step #20.

Block #23: Calculating of P1(1), Q1(1) When the Moving Boundary z_o Has not Yet Reached Grid Point Two

In the case NL = 1, the moving boundary at time TTMMP1 has still not reached the second z-grid point. Therefore, P1(1), Q1(1) must be calculated using formulas in sections 6.2 and 6.3, specifically eqs (6.5′) when TTMM = 0 and (6.6′) when TTMM > 0.

Of course, DM, DM1, etc., are just intermediate variables.

For the computational significance of P1(1), Q1(1), see again sections 6.2 and 6.3. P1(1) is not calculated in subroutine TRIDNL like the rest of its elements, and Q1(1) is calculated here rather than with the other elements of Q1 in block #25.

Block #24: Computation of P1(1), Q1(1) When z_o Has Moved Beyond the First Grid Mesh Widths

In this case (NL > 1), the moving boundary has traveled through the first z-mesh width and P1(1), Q1(1) are to be calculated by the formulas in section 6.4, specifically eq (6.7′). P1(1) is not calculated in subroutine TRIDNL like the rest of its elements, and Q1(1) is calculated here rather than with the other elements of Q1 in block #25.

Block #25: Calculation of Arrays AF( ), Q1( ) When There are Regular Points in the Z-Region

When NL > 2, there are grid points with cell widths (see sec. 5.1) of magnitude DZ in the z-region, and the arrays AF and Q1 associated with these points are therefore computed. [This amounts to saying that the arrays AF( ), Q1( ) that are associated with these points are calculated by using the formulas in table 5.2 and eq (6.9′).]

For the significance of the array Q1 in the computational procedure, see chapter 3. The array AF plays the role of the quantity Z_4 in eq (3.2); see also eq (5.7). Array AF is used to compute array D1 in subroutine TRIDNL. The array Q1, which is composed of AF and D1, is used in block #29.

Block #26: Calculation of Arrays BF and Q2 Associated with the Grid Points in the Y-Region

The array BF is used to compute Q2 later in this block.

For further elucidation of the computational significance of these arrays, see the references to BF in section 5.3, table 7, and eq (5.10). See section 6.8 and eqs (6.16) and (6.16′) for the significance and definition of Q2(J). Notice P2(N2), Q2(N2) are calculated in block #13. Arrays P2, Q2 are used in block #28.

Block #27: Calculation of CB(1), CB(2), C1(NL), C2(NR) at Time TTMMP1

The quantities above at the end of the MMth iteration of loop 78 (i.e., at time TTMMP1) are computed in subroutine CENTER which is documented in detailed block form in section 7.2.

Block #28: Computation of C2( ) at Time TTMMP1

By making use of C2(NR) that was computed in subroutine CENTER in block #27, we can carry out the backward sweep; see steps #2 and #3 in chapter 3 that generate C2(J) for J = NR + 1 to N2 - 1.

Block #29: Computation of Array C1( ) at Time TTMMP1

In this case NL = 1, the value of C1(1), which is the only grid point besides CB(1) in the y-region, is already known, since it has been computed in subroutine CENTER as C1(NL). However, when NL is greater than one, the calculation of C1(J) for J = 1, NL - 1 must be completed by a backward sweep, as described in section 3.1, steps #2 and #3.

Block #30: Print-out Concentrations When KW = 0

The integer KW is zero when (MM - 1) is divisible by KPRINT.

The concentrations are printed under a time TTMMP1 which is in units of BTM (see sec. 4.3). The user is advised not to print out every concentration in each region because there are so many, especially in the y-region (i.e., the silicon).

Also printed out are the values of the concentrations at the boundary and C1(NL). (Note that C2(NR) will always be printed with the other concentrations in the y-region.)

Block #31: Reset Quantities in Preparation for Next Iteration of Loop 78

Unless a new grid point is reached by the moving boundary in the next iteration, the new values of SM(1), SM(2), TTMM are as given (see figs. 5 and 6). If in the next iteration of loop 78 a new grid point is reached, the values are corrected appropriately at that time, in blocks #17 and #19.

Block #32: Test to Determine Whether TTMMP1 Has Reached TFINAL

The quantity TFINAL (INPUT) is the time we want to stop the problem and TIMEND is the value of TFINAL in units of BTM which is the unit of TTMMP1. Hence, as long as TIMEND is greater than TTMMP1, we proceed to the next iteration of loop 78. If MME is not large enough to reach TFINAL, a diagnostic to this effect is printed before control shifts to statement 79 and then the end card.

Block #33: Shift Program to Statement 800 in VERSION 1

The time is greater than or equal to TFINAL and in VERSION 1 the value of JSTOP is set equal to 1 in a data card. Hence, control sends the program to statement 800 where printing and plotting occur prior to the end of the program.

Block #34: Print-out of Data and Plot of Concentrations at the Final Value of TTMMP1

The detailed description of the plotting subroutine is given in Appendix 2 and section 4.4.

Block #35: Redimensionalize Parameters to Original Units

Parameters such as ZOP1 and TTMMP1 (at the last iteration of loop 78 that occurs) are changed into micrometer and hour units and printed out with other pertinent parameters.

Block #36: Control in VERSION 1 Sends Program to Its End

The integers JSTOP and JSAVE have the values one and zero (see below data cards) in VERSION 1; consequently, in this case the program is sent to the end.

If TIMEND has not been reached by TTMMP1 before the number of iterations MME of loop 78 is completed, a diagnostic message to this effect is printed out.

Block #37: This Block Contains Statement 79 and Ends Program.

7.2. Block-by-Block Description of Subroutine CENTER

This subroutine, which is called in computational block #27 of the main program, is where the quantities CB(1), C1(NL), CB(2), C2(NR) are evaluated at time TTMMP1.

Block #0. Insert Input Through COMMON

The following input for the subroutine passes in through COMMON: S( ), SM( ), N2M1, N2, NLM1, NL, NR, C1( ), C2( ), CB( ), DF1, DF2, CONPRO, CONOUT, DT, DZ, DY, P1, P2, Q1, Q2.

The output is fed to the main program through COMMON.

Block #1: Computation of the Quantities P(2), Q(2)

These quantities defined in eqs (6.18) and (6.18′) are used to determine PA(2), QA(2) in computational block #3.

Block #2: Compute P(1), Q(1)

These quantities [see eqs (6.11) and (6.11′)] are used when NL > 1 to determine PA(1), QA(1). When NL = 1, the role played by PA(1), QA(1) in linking the value of CB(1) to the value of C1(1) is played by P1(1), Q1(1), which are computed in block #23 of the main program. Therefore, at the end of block #3, we set PA(1) = P1(1), QA(1) = Q1(1) when NL = 1.

Block #3: Computation of PA(L), QA(L) for L = 1,2

The computation formula for these quantities when L = 1 is described in eqs (6.10′), (6.11′), and (6.12′). [See eqs (5.16′) and (5.25) for the definition of R(1) and R(2); also see tables 7 and 8.] The computation formulas of the above quantities in the case L = 2 are described

in eqs (6.17′), (6.18′), and (6.19′). The significance of PA(L), QA(L) is seen from eqs (5.26′) and (5.18) or eqs (6.12) and (6.19).

Block #4: Computation of CB(1), CB(2) at Time TTMMP1 in Terms of Known Quantities

The concentration at the moving boundary in the z-region CB(1, TTMMP1) = CB(1) is computed by using formula eqs (6.21) and (6.21′) and the auxiliary eq (6.22). To find CB(2) = CB(2, TTMMP1), we just use the boundary condition in eq (2.23). For the significance and explanation of how SAVE is used, see the end of section 5.7, before SUMMARY.

Block #5: Calculation of C1(NL), C2(NR) at Time TTMMP1

We introduce the abbreviations

$$C(1) = C1(NL, TTMMP1) = C1(NL)$$

$$C(2) = C2(NR, TTMMP1) = C2(NR)$$

and compute these desired quantities from CB(1), CB(2), PA(L), QA(L) known from blocks #1 to #4 by employing eqs (6.12) and (6.19).

7.3. Block-by-Block Description of Subroutine TRIDNL

The input for this subroutine is the output of subroutine INTER, DT, DZ, DY, DF1, DF2, and CONPRO. The arrays P1, D1, P2, D2 in contrast to the Q1, Q2 arrays do not depend on concentrations. The output of this subroutine is used both in the main program blocks #25, #26, #28, and #29 and subroutine CENTER (see blocks #1 and #2). The output of the subroutine is P1(J), D1(J) for J = 2, NL - 1 and P2(J), D2(J) for J = NR + 1, ..., N2 - 1.

Block #1: Call Subroutine INTER

By calling this subroutine, we input A2L, ..., A1M, B2L, ..., B1M through COMMON. These quantities are used in blocks #3 and #4 here.

Block #2: Calculation of P1(1) when NL > 1

When NL > 2, this value of P1(1) is used in the next block. Calculation formulas of the above quantities are found in eqs (6.7) and (6.7′). (This is a duplication of P1(1) calculated in block #24 of the main program and is not necessary in VERSION 1.)

Block #3: Calculation of Arrays P1(J), Q1(J)

The calculation formulas for these quantities when J = 2, NL - 1 is eq (6.9′). These quantities are used in blocks #25 and #29 of the main program.

Block #4: Calculation of Arrays P2(J), D2(J)

The calculation formulas for these arrays when J = NR + 1, ..., N2 - 1 are given in eq (6.16′). Notice that P2(N2) is calculated in block #13 of the main program and used here.

7.4. Description of Subroutine INTER

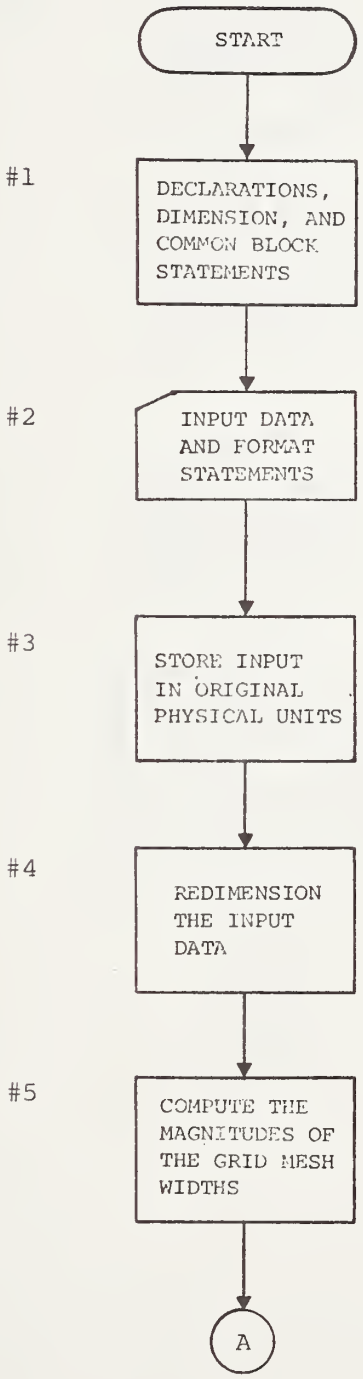
This subroutine is so trivial we only describe it briefly.

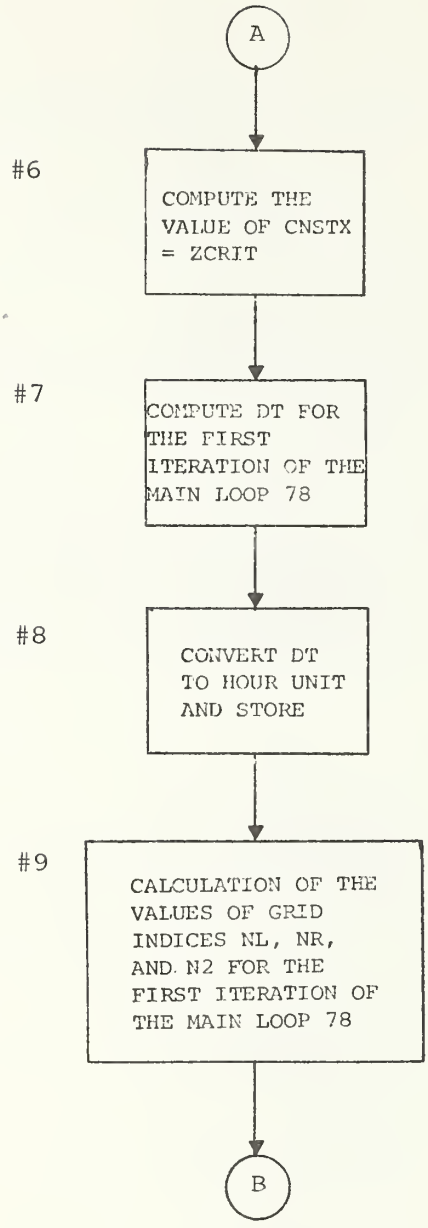
The routine uses the value of DT, DZ, DY, DF1, DF2 to compute the value of A2L, A2M, A2R, B2L, B2M, B2R, A1M, B1M for use in subroutine TRIDNL (see blocks #3 and #4).

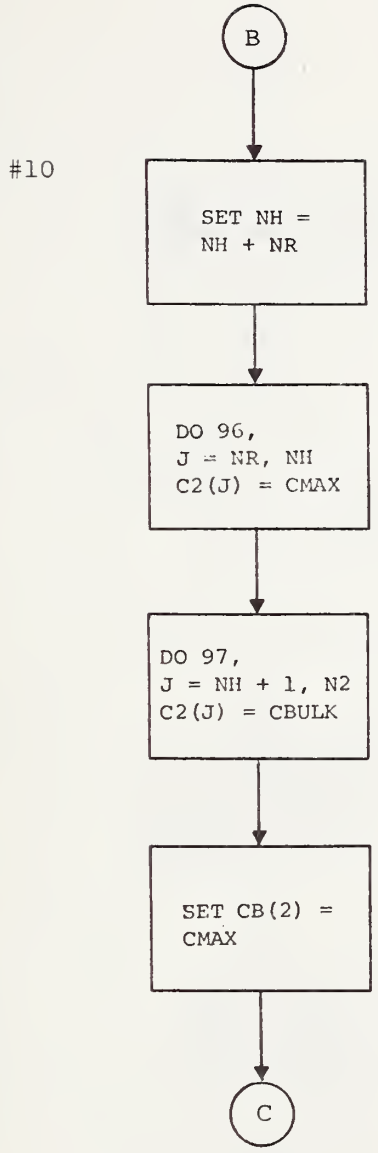
Since the values of A2L, ..., B1M depend on DT, every time this quantity changes these quantities need to be recomputed. This explains why subroutine TRIDNL is called whenever DT changes in the main program, i.e., because TRIDNL calls INTER (see block #1 in TRIDNL) and thereby corrects the values of A2L, ..., B1M for the new DT and, consequently, simultaneously corrects the values of the arrays P1, D1, P2, D2 for the values of the new DT.

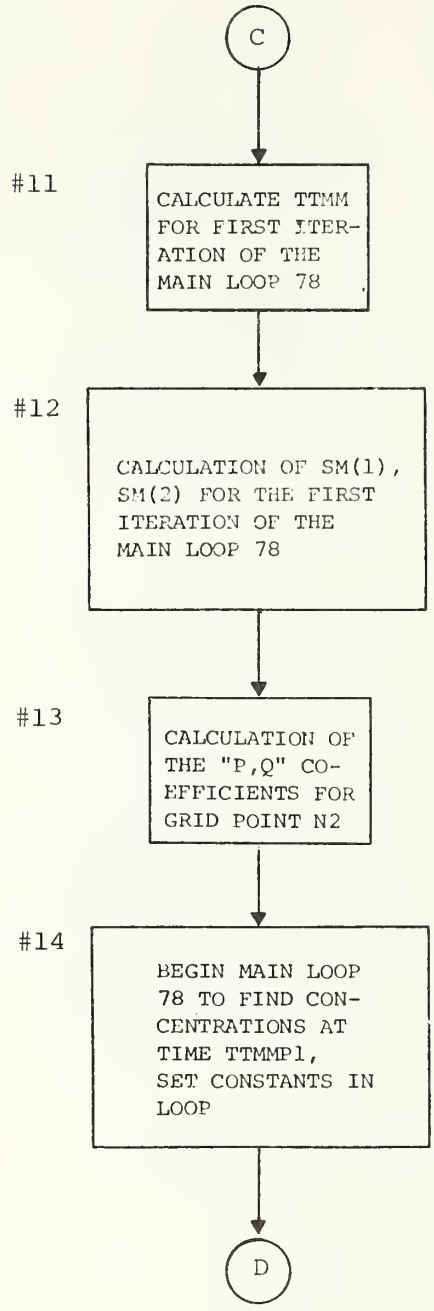
8. FLOW DIAGRAMS

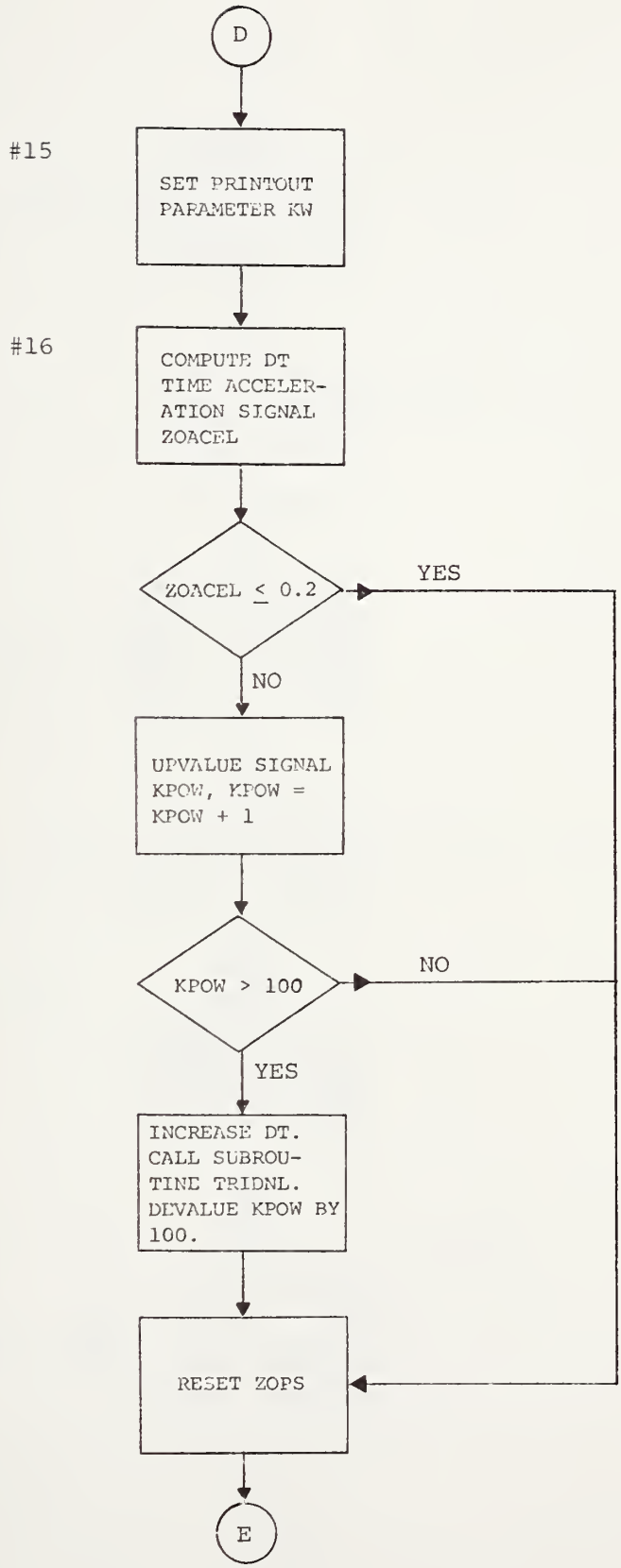
8.1. Flow Diagram of Main Program in DISTRIB

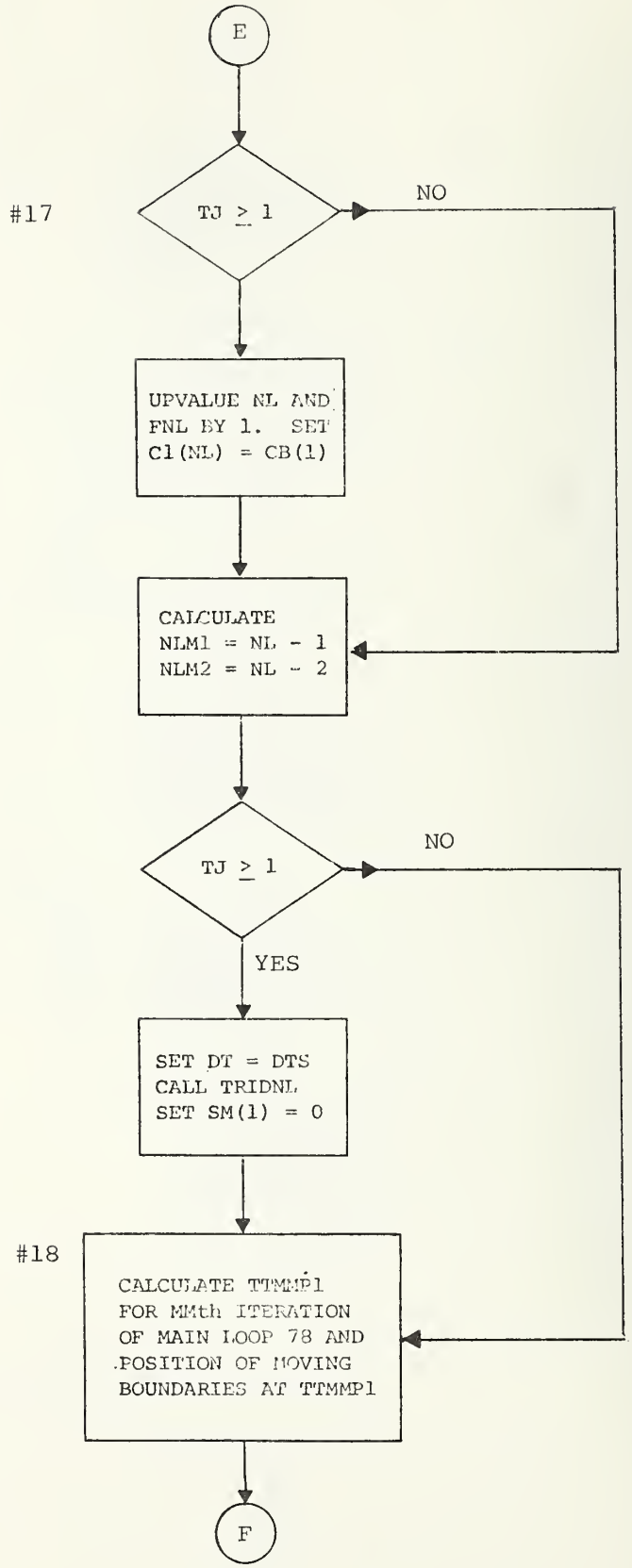


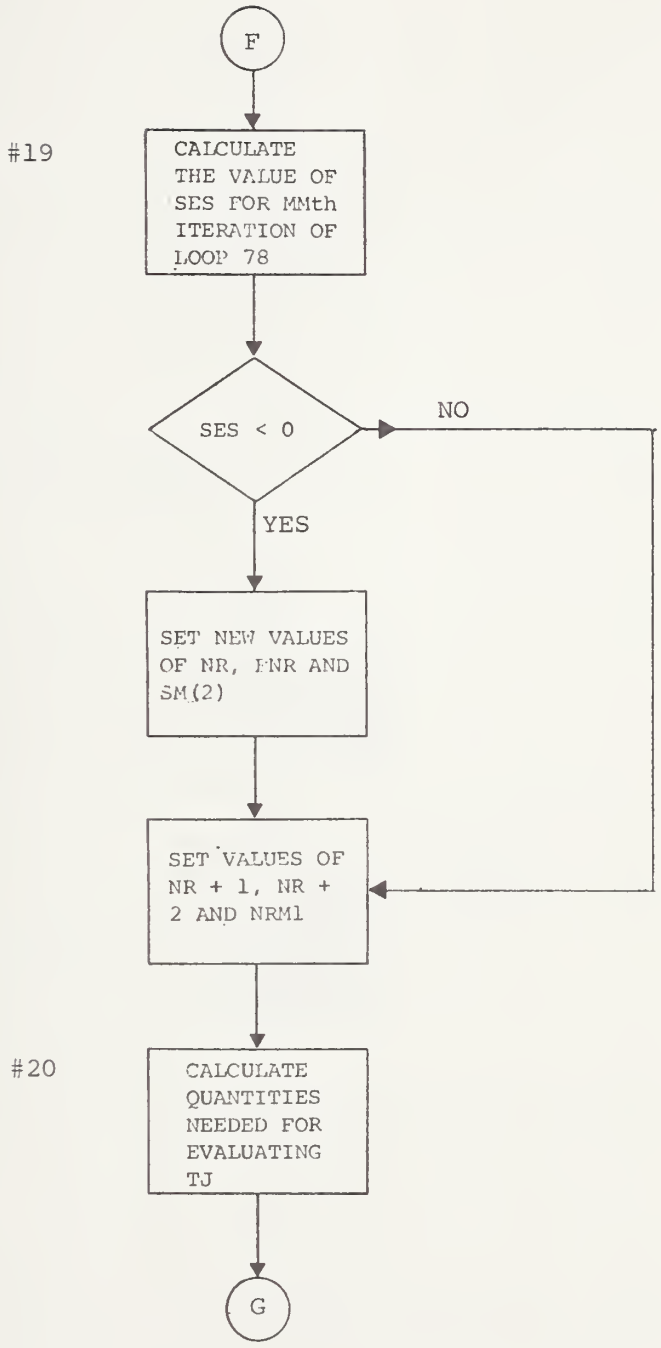

A
#6
COMPUTE THE VALUE OF CNSTX = ZCRIT
#7
COMPUTE DT FOR THE FIRST ITERATION OF THE MAIN LOOP 78
#8
CONVERT DT TO HOUR UNIT AND STORE
#9
CALCULATION OF THE VALUES OF GRID INDICES NL, NR, AND N2 FOR THE FIRST ITERATION OF THE MAIN LOOP 78
B

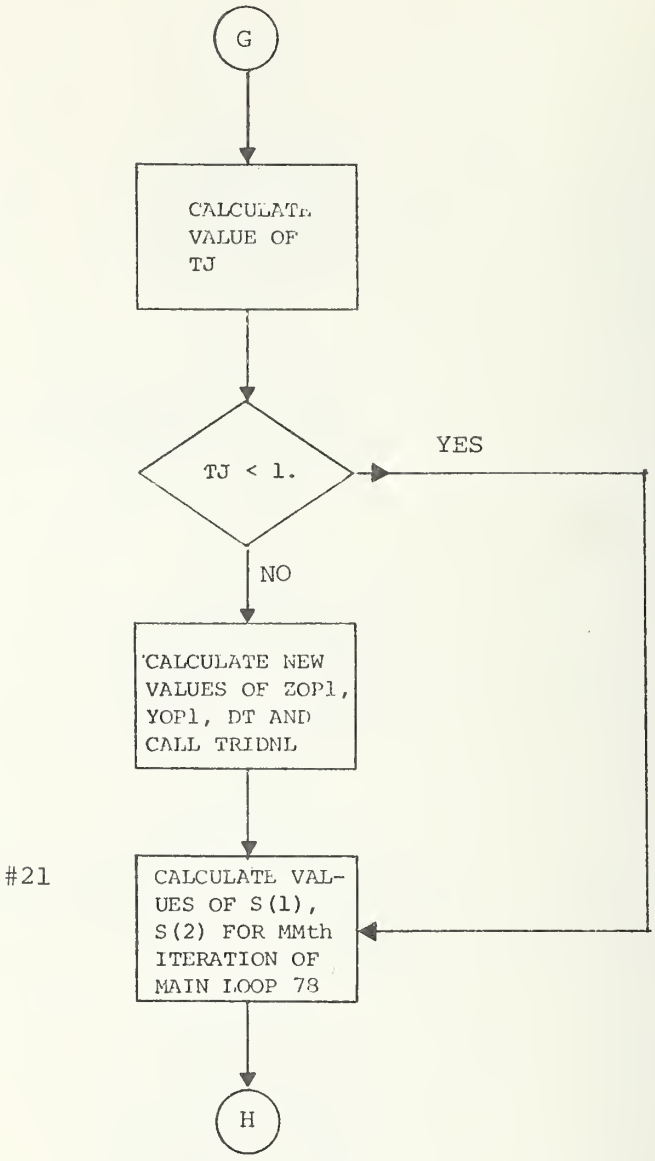

B
#10
SET NH =
NH + NR
DO 96,
J = NR, NH
C2(J) = CMAX
DO 97,
J = NH + 1, N2
C2(J) = CBULK
SET CB(2) =
CMAX
C

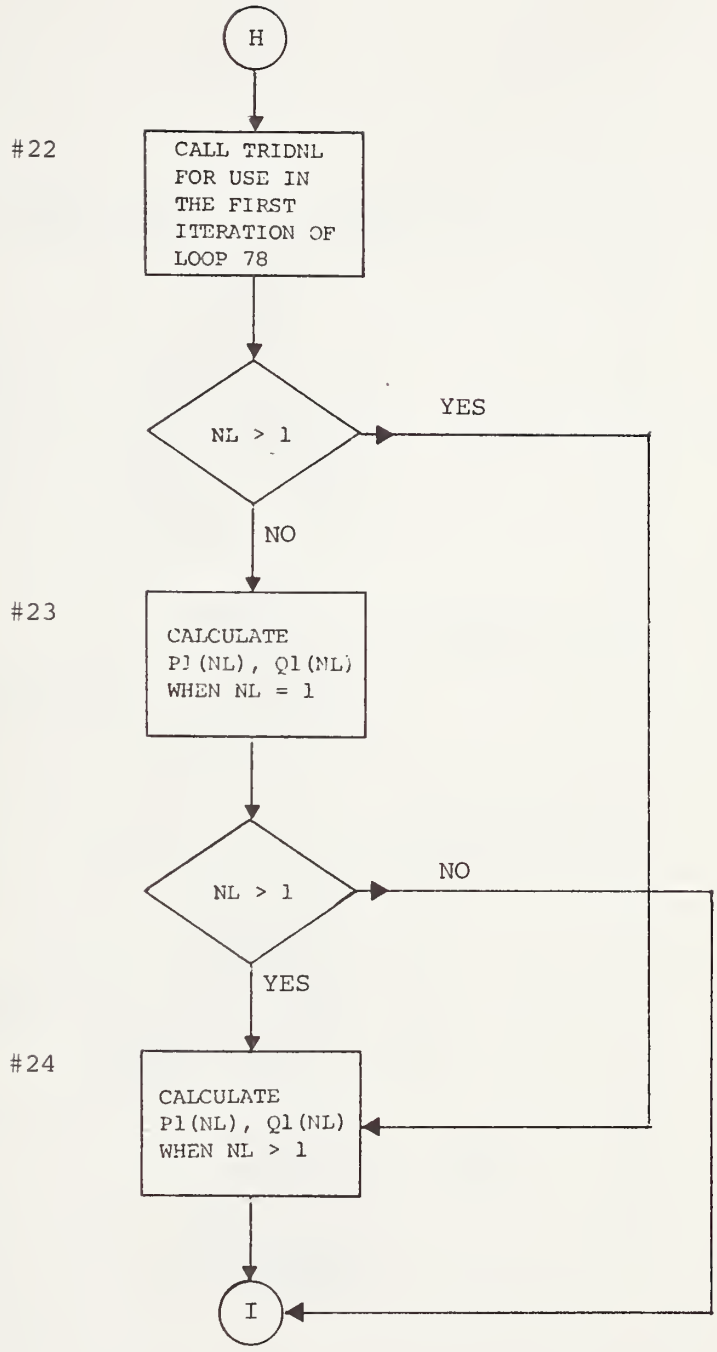

C
#11
CALCULATE TTMM FOR FIRST ITER-ATION OF THE MAIN LOOP 78
#12
CALCULATION OF SM(1), SM(2) FOR THE FIRST ITERATION OF THE MAIN LOOP 78
#13
CALCULATION OF THE "P,Q" CO-EFFICIENTS FOR GRID POINT N2
#14
BEGIN MAIN LOOP 78 TO FIND CON-CENTRATIONS AT TIME TTMMP1, SET CONSTANTS IN LOOP
D

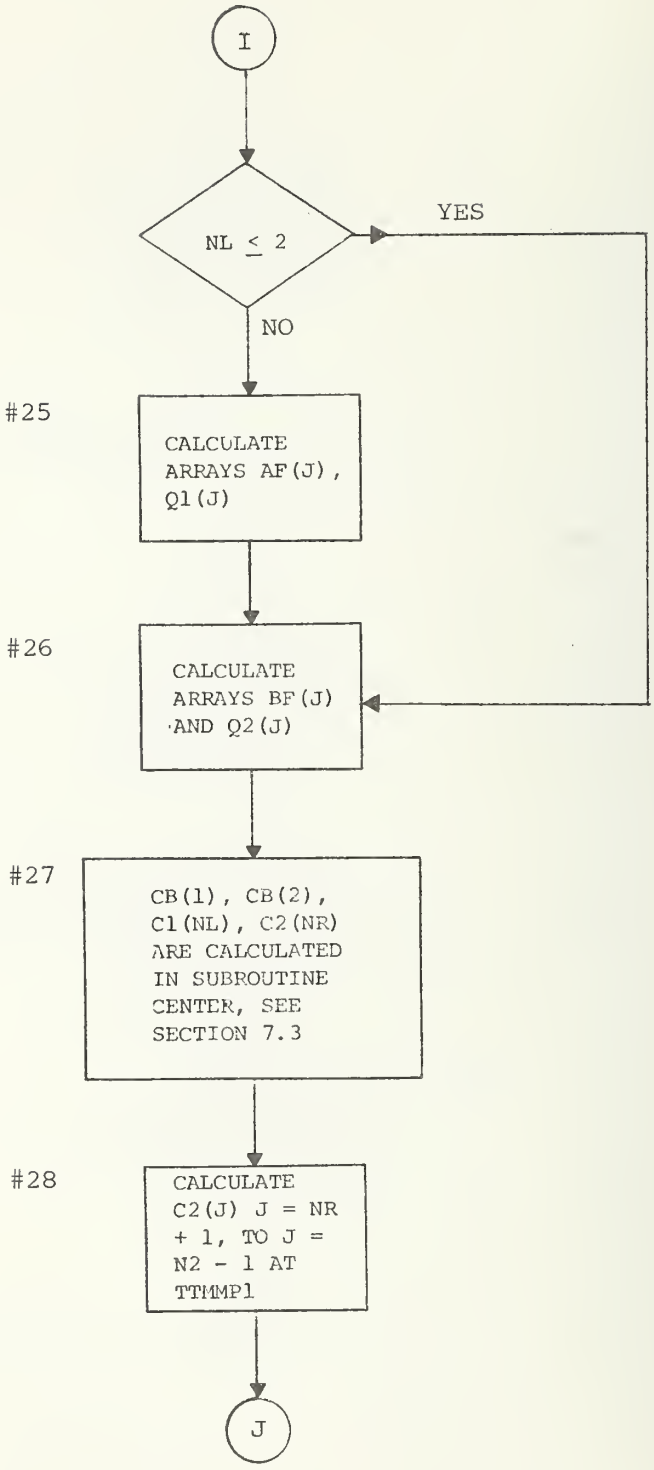

D
#15
SET PRINTOUT PARAMETER KW
#16
COMPUTE DT TIME ACCELER-ATION SIGNAL ZOACEL
ZOACEL ≤ 0.2
YES
NO
UPVALUE SIGNAL KPOW, KPOW = KPOW + 1
KPOW > 100
NO
YES
INCREASE DT. CALL SUBROU-TINE TRIDNL. DEVALUE KPOW BY 100.
RESET ZOPS
E

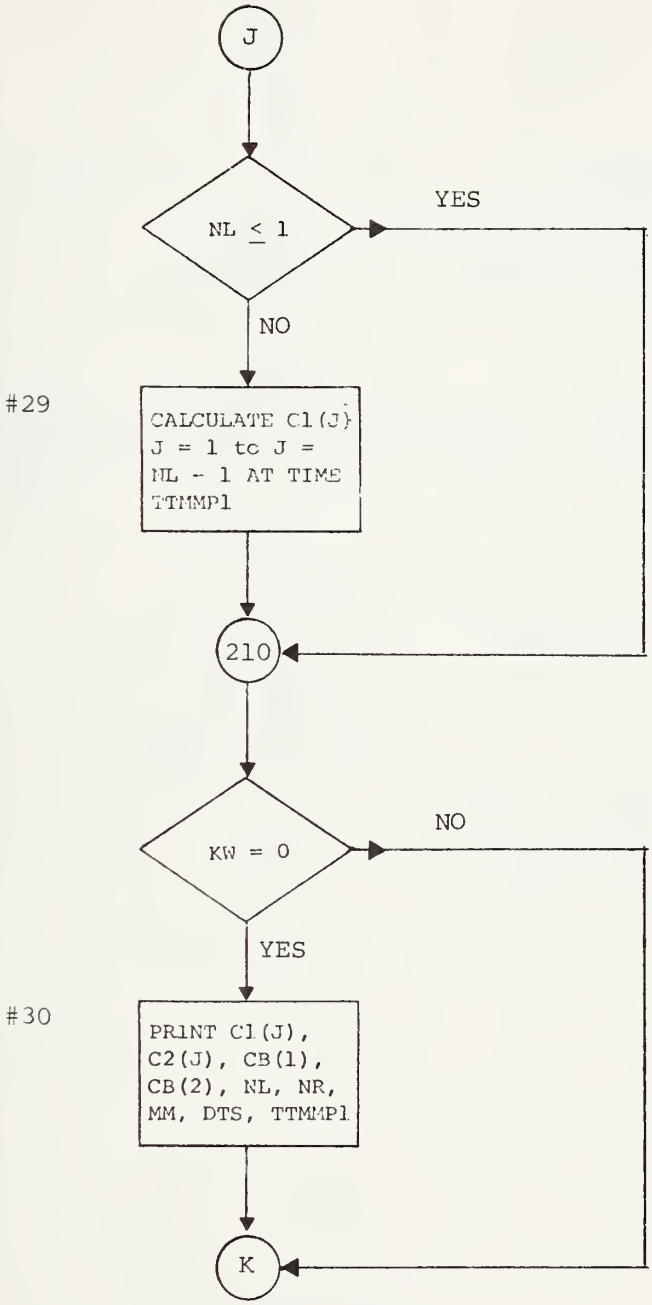

E
#17
TJ ≥ 1
NO
UPVALUE NL AND FNL BY 1. SET C1(NL) = CB(1)
CALCULATE NLM1 = NL - 1 NLM2 = NL - 2
TJ ≥ 1
NO
YES
SET DT = DTS CALL TRIDNL SET SM(1) = 0
#18
CALCULATE TTMMP1 FOR MMth ITERATION OF MAIN LOOP 78 AND POSITION OF MOVING BOUNDARIES AT TTMMP1
F

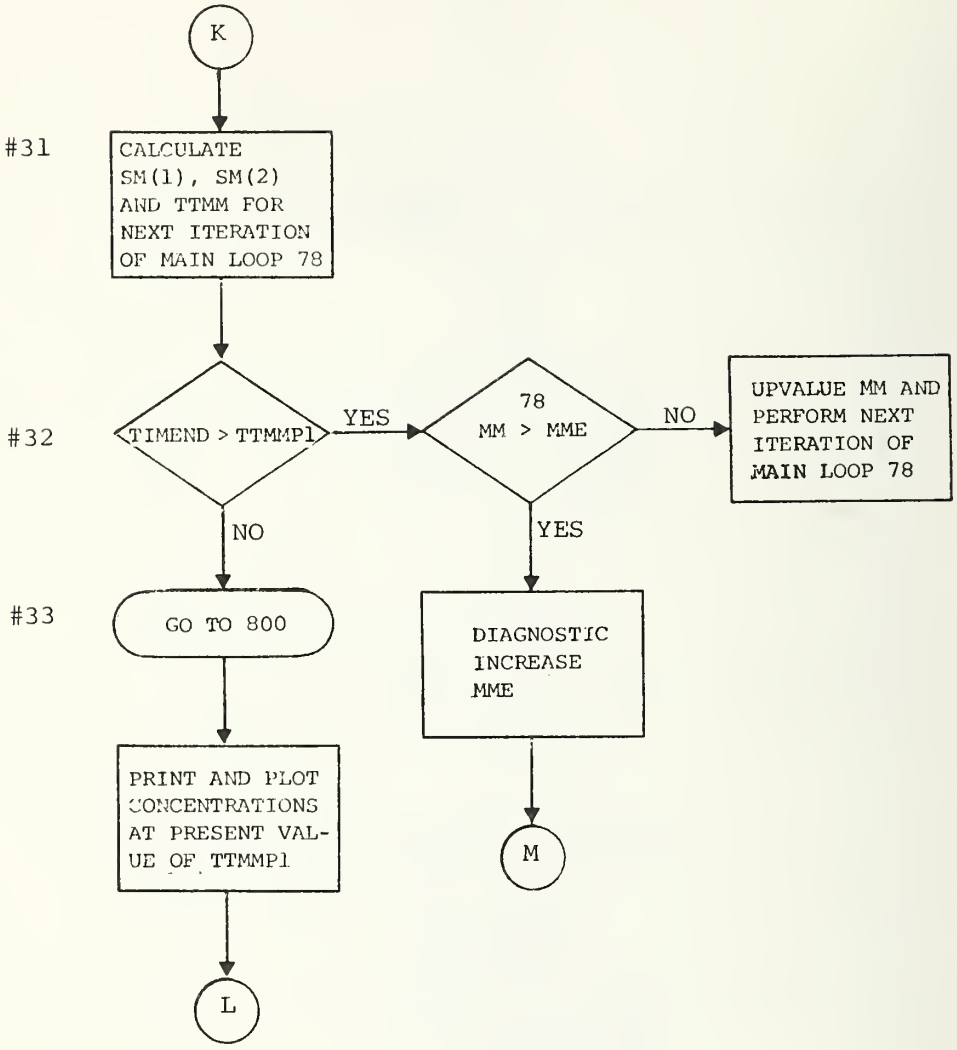

F
#19
CALCULATE THE VALUE OF SES FOR MMth ITERATION OF LOOP 78
SES < 0
NO
YES
SET NEW VALUES OF NR, NNR AND SM(2)
SET VALUES OF NR + 1, NR + 2 AND NRM1
#20
CALCULATE QUANTITIES NEEDED FOR EVALUATING TJ
G

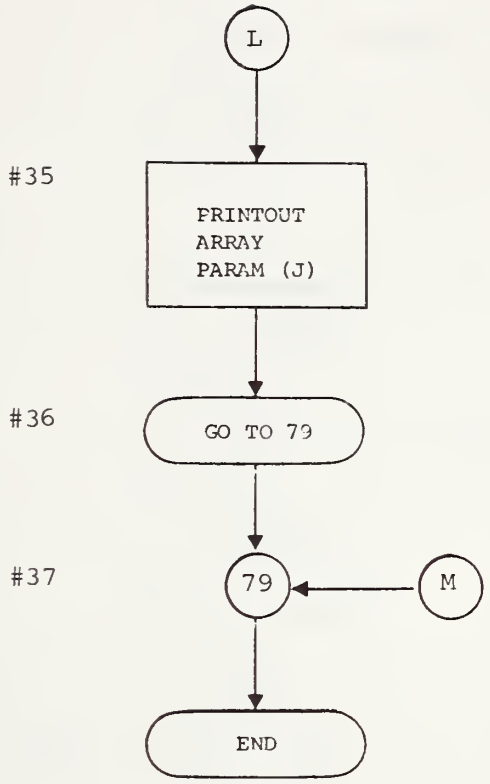

G
CALCULATE VALUE OF TJ
TJ < 1.
YES
NO
CALCULATE NEW VALUES OF ZOP1, YOP1, DT AND CALL TRIDNL
#21
CALCULATE VALUES OF S(1), S(2) FOR MMth ITERATION OF MAIN LOOP 78
H


H
#22
CALL TRIDNL FOR USE IN THE FIRST ITERATION OF LOOP 78
NL > 1
YES
NO
#23
CALCULATE PJ(NL), Q1(NL) WHEN NL = 1
NL > 1
NO
YES
#24
CALCULATE P1(NL), Q1(NL) WHEN NL > 1
I

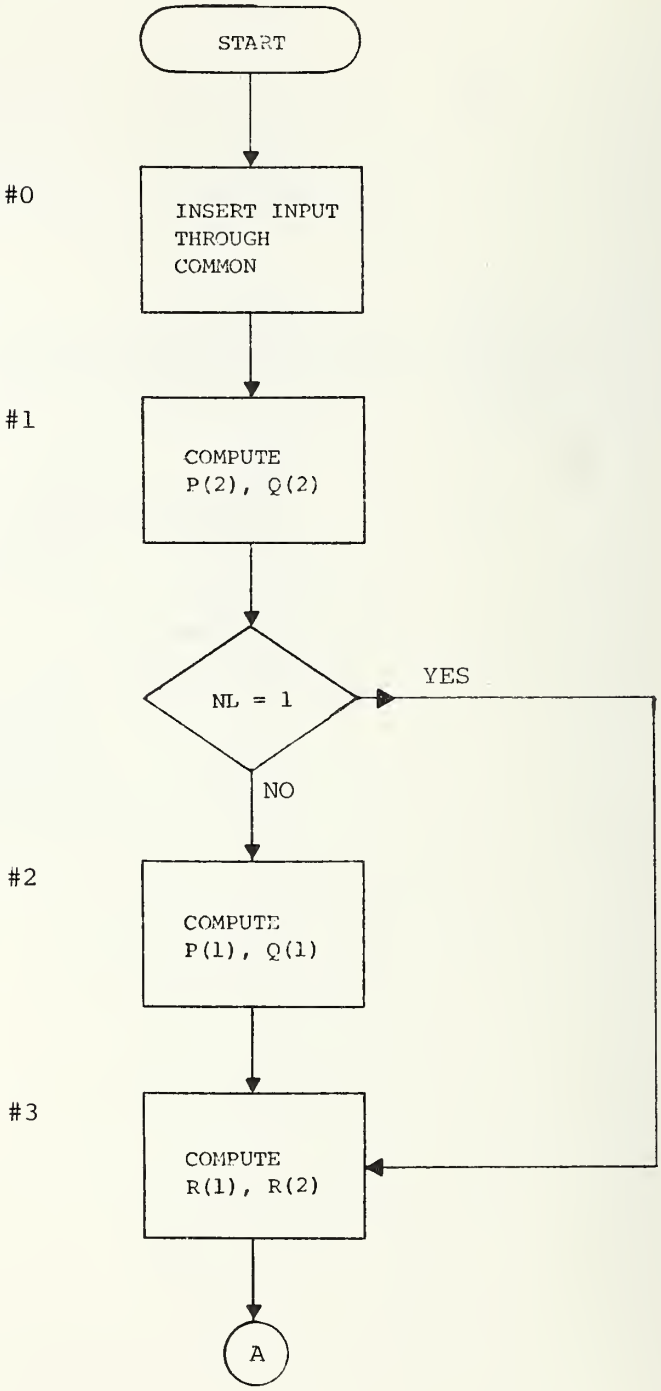

I
NL ≤ 2
YES
NO
#25
CALCULATE ARRAYS AF(J), Q1(J)
#26
CALCULATE ARRAYS BF(J) AND Q2(J)
#27
CB(1), CB(2), C1(NL), C2(NR) ARE CALCULATED IN SUBROUTINE CENTER, SEE SECTION 7.3
#28
CALCULATE C2(J) J = NR + 1, TO J = N2 - 1 AT TTMMP1
J

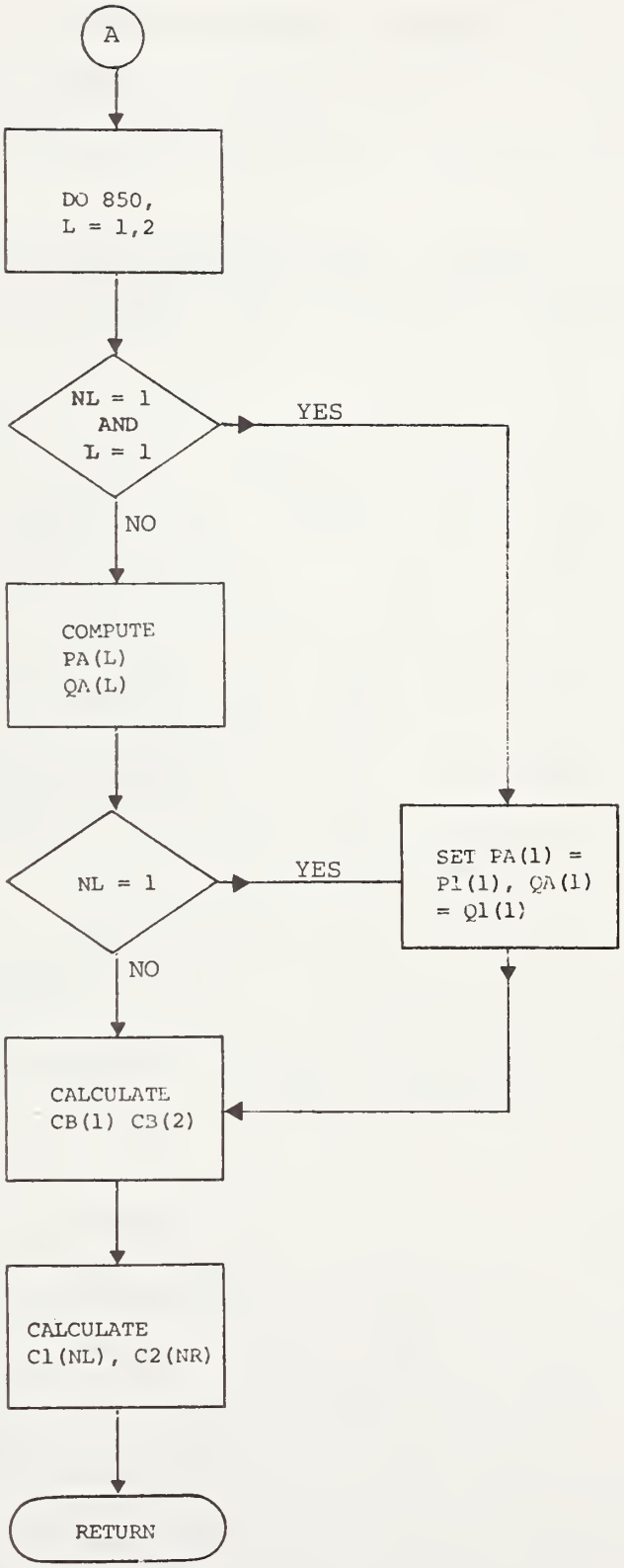

J
NL ≤ 1
YES
NO
#29
CALCULATE C1(J) J = 1 to J = NL - 1 AT TIME TTMMP1
210
KW = 0
NO
YES
#30
PRINT C1(J), C2(J), CB(1), CB(2), NL, NR, MM, DTS, TTMMP1
K


K
#31
CALCULATE SM(1), SM(2) AND TTMM FOR NEXT ITERATION OF MAIN LOOP 78
#32
TIMEND > TTMMP1
YES
78
MM > MME
NO
UPVALUE MM AND PERFORM NEXT ITERATION OF MAIN LOOP 78
NO
YES
#33
GO TO 800
DIAGNOSTIC
INCREASE
MME
PRINT AND PLOT CONCENTRATIONS AT PRESENT VALUE OF TTMMP1
M
L

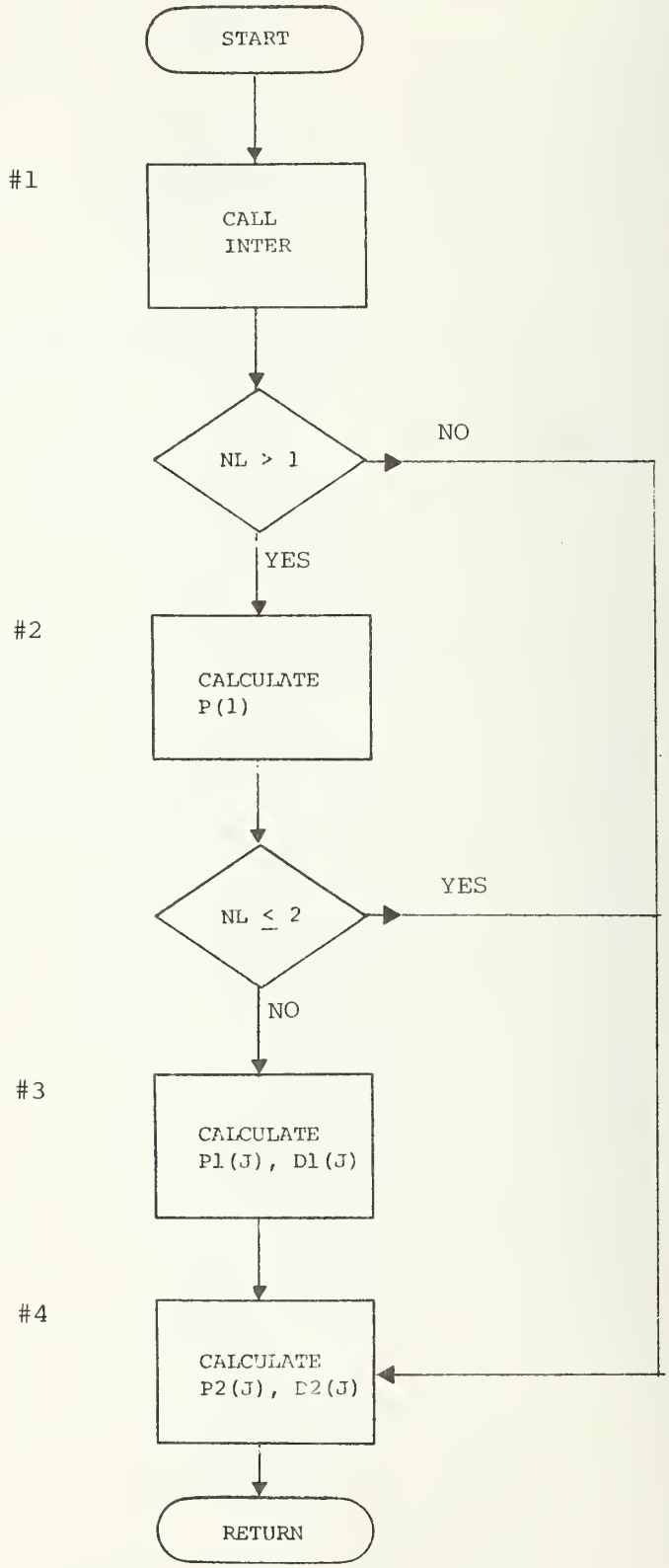

L
#35
PRINTOUT
ARRAY
PARAM (J)
#36
GO TO 79
#37
79
M
END

8.2. Flow Diagram for Subroutine CENTER




A
DO 850,
L = 1,2
NL = 1
AND
L = 1
YES
NO
COMPUTE
PA(L)
QA(L)
NL = 1
YES
SET PA(1) =
P1(1), QA(1)
= Q1(1)
NO
#4
CALCULATE
CB(1) CB(2)
#5
CALCULATE
C1(NL), C2(NR)
RETURN

8.3. Flow Diagram for Subroutine TRIDNL.



9. HOW TO USE DISTRIB

9.1. A Brief Description of DISTRIB: VERSION 1

The intent of this section is to give a quick general introduction to DISTRIB: VERSION 1.

The meaning of the undefined terms in this chapter is explained in the glossary, chapter 12. A more detailed description of the physical aspects of the redistribution problem is given in chapter 2. Chapter 3 contains a more detailed description of the overall computational procedure.

DISTRIB: VERSION 1 calculates a redistributed initial impurity profile in a crystal wafer that has undergone an oxidation at a constant temperature for an input time t = TFINAL. The impurity concentration in the oxide C_1 (z,t) is calculated at time t = TFINAL. Here, z denotes (fig. 3) the distance measured into the oxide from the interface between the ambient oxygen and the oxide at time t = TFINAL. (The z-coordinate system changes with time.) Also calculated at time = TFINAL are the impurity concentrations in the silicon $C_2(y, t)$ where y represents (fig. 3) distance into the oxide-silicon composite measured from the position of the oxygen-silicon interface at time t = 0. At this initial time, the silicon is completely unoxidized. The principal output quantities are the concentrations $C1(L) = C_1((L - 1)DZ, TFINAL)$ for L = 1, ..., NL where NL represents the index of the first grid point to the left of the position of the moving oxide-silicon interfacial boundary, $Z_o(t)$, at t = TFINAL (fig. 3). Also computed are $C2(K) = C_2((K - 1)$ DY, TFINAL) for K = NR + 1, ..., N2 where NR is the index of the first grid point to the right of the moving boundary, $y_o(t)$, at time t = TFINAL (fig. 3) and N2 represents the number of grid points in the original silicon wafer. The quantities DZ and DY above are uniform grid widths in the z- and y-coordinate systems. Besides the concentrations C1( ) and C2( ), DISTRIB computes $CB(1) = C_1(Z_o(TFINAL))$ and $CB(2) = C_2(Y_o(TFINAL))$, the impurity concentrations in the oxide and silicon at the moving oxide-silicon interface at time TFINAL. All of these concentrations are also plotted.

9.2. Definitions and Other Information Related to Input and Output

In section 9.21 the real input quantities are briefly defined. A table follows the definitions giving further information about the previously defined input, such as: the computational block number in the listing that contains the read and format cards, format information, data sources, and numerical ranges for which the parameters have been successfully tested, etc. This same procedure is used to describe the integer input in section 9.22. The same procedure is also used to describe the real and integer output in sections 9.23 and 9.24. Consult the glossary for information concerning undefined quantities and places in the documentation where more thorough definitions are given. All physical quantities must be entered in units of hours and micrometers.

9.2.1. Real Input

The real input is defined in table 16 in the same order they are read (see computational block #2 in the listing, chapter 11, for the read and format card).

Table 16: Brief Definitions of the Real Input

	Card 1
CBULK	The program handles automatically piece-wise constant initial concentrations in the silicon. The grid points in the silicon beyond the input grid point NR are assigned the value CBULK representing the concentration in the bulk of the silicon.
CMAX	The grid points 1 through NH are assigned initial concentrations of magnitude CMAX.
ALPHA	A given volume of silicon swells to a volume $(ALPHA)^{-1}$ times (the original amount of silicon at the oxide-oxygen interface).
TFINAL	The total time that the wafer is to undergo redistribution.
SEG	The redistribution coefficient.
	Card 2
DF1	The diffusion coefficient of boron in the oxide.
DF2	The diffusion coefficient of boron in the silicon.
WFRC	The fraction of a mesh width distance, DZ, that the moving boundary moves in the first iteration of loop 78, i.e., in the first time step.
BDRY	A positive number representing the width of the wafer.
CONPRO	A proportionality constant representing the evaporation rate of boron from the oxide into the oxygen ambient.
	Card 3
CONOUT	A constant representing the concentration of boron in the oxygen at the oxygen-oxide interface. (In our testing CONOUT was always set equal to zero.)
CNSTA, CNSTB, CNTAU	Constants appearing in the defining equation of the moving boundary, section 4.2.
Y1S	Same definition as Y2S, etc., below.
	Card 4
Y2S, X2S, X1S	Quantities that are used to scale the concentrations that are outputted in the plotting routines, section 4.4, and computational block #34.

Runs of DISTRIB have been successfully completed using data in the ranges shown in table 17. Sources for the data and formal information are also given.

Table 17: Real Input (4 FORMAT (5 F 10.5)) (see block #2 in listing, chapter 11). Physical quantities are assumed to be in units of micrometers.

VARIABLE	RANGE (Note 4)
(1) CBULK	1. (Note 4)
(2) CMAX	1. (Not tested with other values)
(3) ALPHA	0.44
(4) TFINAL	0.1 to 4.
(5) SEG	0.01 to 10.
(6) DF1	See [4], [6], [7], [8]
(7) DF2	0.00029·DF1 (See [7] and Note 2)
(8) WFRC	0.01 to 0.33 (Note 3)
(9) BDRY	3. to 8.
(10) CONPRO	0.1 to 1000.
(11) CONOUT	All runs have been with zero because of our experimental conditions.
(12) CNSTA	See tables in [1]
(13) CNSTB	"
(14) CNTAU	"
(15) Y1S	5.
(16) Y2S	2.
(17) X2S	4.
(18) X1S	2.

Notes:
(1) The ranges in the subsections are values from successful runs and are not to be interpreted as precise limitations.
(2) This relationship is cited in [4].
(3) WFRC must be chosen small enough so that DM = WRRC*DZ < ZCRIT. For an explanation of this restriction, see the comment at the end of block #7 of the main program in chapter 7.
(4) The problem with CONOUT = 0. is homogeneous in the concentrations and therefore all runs have been made with CBULK1.

9.2.2. Integer Input

Table 18 contains brief definitions of DISTRIB's integer input.

Table 18: Brief Definition of Integer Input

Card 5	
NZO	A parameter used to determine the magnitude of DZ, eq (4.3).
MME	The concentrations are calculated at times t_1, ..., t_{MME}. MME must be large enough for t_{MME} > TFINAL or the program will stop before calculating the concentrations at TFINAL. (See table and sec. 9.3 for guidance in choice of MME.)
ND	This integer determines how many grid points in the silicon adjacent to the moving boundary will be plotted.
NYO	The number of mesh widths of length DY that equal BDRY.
NH	The grid point on the y-axis at which the initial boron concentration changes from CMAX to CBULK, eq (2.10) and section 7.2, block #10.
KPRINT	The concentrations and a few other integer parameters such as MM, NL, NR will be printed out, whenever MM = 0 modulo (KPRINT).

The range of values of integer data successfully calculated in DISTRIB is given in table 19.

Table 19: Integer Input Information
Integer Input (1 FORMAT (6I6)) (see block #2 of listing)

VARIABLE	RANGE
(1) NZO	2. to 8.
(2) MME	up to 4000
(3) ND	20. to 30.
(4) NYO	100 to 800
(5) NH	10. to 20.

Certain constants that can be used to control various aspects of how DISTRIB runs but do not have to be changed from run to run (and consequently, are not entered formally) are listed in the next table.

Table 20: Constants Used in Program But Not Made Part of Input.

(1) 0.2 and 100 used in block #16 in the listing of the main program (see sec. 7.1., block #16).

(2) NMOD and NMOD1 (sec. 4.4.) and block #34 of listing of the main program. These quantities need to be changed frequently to get plotted points nicely spaced. We recommend starting with NMOD = 8, NMOD1 = 4.

(3) BLTH (sec. 4.3). The value 0.02 micrometers has worked all the time.

(4) In printing out C1, C2 in blocks #30 and #34, certain indices are skipped. Of course, this is something one changes from run to run routinely, in order to find the values at the grid points that interests one.

9.2.3. The Real Output of DISTRIB

Table 21: Brief Definitions of the Real Output. See Tables 24 and 25.

TTMMP1	DISTRIB calculates concentrations in the oxide and silicon for a sequence of iterations of loop 78 with indices MM = 1, ..., MME. The time at the end of the MMth iteration of this loop is TTMMP1 (in units of BTM).
CB(1), CB(2)	The concentrations of boron in the oxide and silicon at the moving boundary.
C1(J) C2(J)	These are the concentrations in the oxide and silicon at the Jth grid point (for the specific values of J see below) at time TTMMP1.
C1(1)	The concentration at the oxide-oxygen interface.
	See table 26.
PARAM SEG ALPHA TFINAL ZOCR SCALE2	PARAM is an array that contains material constants (diffusion constants, etc.) in the original units of micrometers and hours. SEG is the redistribution (≡ segregation) coefficient. The swelling constant ALPHA represents the reciprocal of the factor that a given volume of silicon swells when converted to oxide at the moving boundary. TFINAL is the length of time of oxidation. ZOCR is the position of the moving boundary in the z-coordinate frame in micrometers. SCALE2 is a scale factor used in the plots.

Further specific information about the previously defined output is given in table 22.

Table 22: Real Output (5 FORMAT (7F14.8)) (see blocks #30 and #34).

Whenever MM = 1 modulo (KPRINT), we print out

(1) C1(J) for J = 1, NL, 15.

(2) C2(J) for J = NR, N2, 20.

(3) TTMMP1 (in units of BTM).

(4) C1(1), CB(1), CB(2), and DTS (in BTM units).

In addition, when TIMEND < TTMMP1, we print out

(5) Array PARAM, SEG, ALPHA, TFINAL, ZOCR, and SCALE2 (all in micrometers and hours).

9.2.4. The Integer Output

Table 23: Definitions of Integer Output. See tables 24 and 25.

NL, NR	NL is the index of the grid point in the oxide that is nearest the moving boundary grid point. NR is the grid point in the silicon that is nearest the grid point at the moving boundary.
MM	The index of loop 78 that calculates the solution at time t = TTMMP1 from the known solution at time t = TTMM. See table 26.
NZO MME ND NYO NH KPRINT	These quantities were defined in table 18.

9.3. A Simple Example

In this section we illustrate how DISTRIB: VERSION 1 is used to compute the redistribution of an initially uniform distribution of boron under wet oxidation conditions. The material constants used in this sample calculation were chosen to obtain the best fit to the experimental data shown in figure 13. For further comments about the proper values for these material constants, see the end of this section.

The first card in computational block #2 reads the real input under format four. (This format card is also in computational block #2.) Since the initial boron distribution is uniform, the value of CMAX which is the concentration at the first NH grid points in the silicon (y-region) is equal to CBULK which represents the initial concentration in the

bulk of the silicon and is the value assigned in the program to the grid points NH + 1, NH + 2, ..., N2. Because the partial differential equations are homogeneous in the concentrations when CONOUT (sec. 2.1) is zero, as in this example, we normalize the input quantities CMAX, CBULK by the value of CBULK (atoms per cubic micrometer); therefore, these two input quantities have the value 1 and the output concentrations of these quantities should be multiplied through by CBULK to obtain the true concentrations. Once again, the glossary is for answering these questions. The next parameter read is the swelling ratio ALPHA (sec. 2.1) and it is assigned the value 0.44 [4].

We desire to find the concentration of the redistributed profile after TFINAL = 0.3 h. When TTMMP1 (T_{MMP1}) is greater than TFINAL, the program's control is shifted to the output printing blocks #34 and #35. The segregation coefficient SEG (dimensionless) is chosen to be 0.2, in order to obtain the fit shown in figure 13. The diffusion coefficients DF1, DF2 for the boron in the oxide and silicon, respectively, were specifically assigned the values 0.001 and 0.133 in order to obtain the fit shown in figure 13, and these values are within the range of the reported values for these quantities (see the end of this section). (Note that the ratios of the diffusion constants used in the program, see [7], were relaxed in order to obtain the fit in this example.) The meaning of the real grid input parameter WFRC = 0.5 is explained in section 7.1 in the material concerned with block #7.

The meaning of the next parameters read, CONPRO (micrometers per hour) and CONOUT (atoms per cubic micrometer), are related to the flux of boron between the ambient and oxide (sec. 2.1, eq (2.11)). As mentioned already, CONOUT = 0 in this example and CONPRO is chosen to have the value 1; other calculations show that results are not sensitive to the choice of CONPRO including the value zero. The next three parameters read, CNSTA (micrometers), CNSTB (square micrometers), CNTAU (hours), relate to the moving boundary motion (sec. 4.2, eq (4.5)). The wet oxidation is assumed to take place at 1100°C and therefore from table 1 in [1] CNTAU(τ) = 0.0, CNSTA(A) = 0.11, and CNSTB(B) = 0.51. The real input read next, Y1S, Y2S, X2S, and X1S (sec. 4.4.), are parameters that assign maximum abscissa and ordinates that are plotted in regions one and two, i.e., the oxide and silicon regions, respectively, in the graphical output in computational block #34. The values Y1S = 5, Y2S = 2, X2S = 4, and X1S = 2 have always yielded good graphical output, and we recommend their use.

The integer input for DISTRIB is read following the real input in computational block #2 under format number 1 (also in computational block #2). The integer NZO = 3, whose significance is explained in section 4.1 below, eq (4.2), is read first. The quantity MME = 1500 represents the number of iterations we *a priori* assign to the loop 78 (computational block #14). In this example, only 595 iterations of loop 78 occur before T_{MMP1} > TFINAL = 0.3 h and the program stops after printing the output in computational blocks #34 and #35. Integer ND represents the number of grid points in silicon to the right of the moving boundary point in

the silicon that are plotted in computational block #34 (see DO LOOP 234). The integer NYO = 350 is read next and its function in defining DY(Δy) is explained by eq (4.1). The integer NH = 12 (sec. 2.1, eq (2.13)) signifies that the initial concentration value CMAX is assigned to the first twelve grid points in the silicon region at time zero. Since in this example CMAX = CBULK, it makes no difference what the value of NH is except NH should be less than NYO in order for the problem to make sense. The print-out parameter KPRINT = 1000 signals (in block #30) the program to print out C1( ), C2( ), the concentrations in the oxide and the silicon, respectively, at the end of the iterations (1 + L*KPRINT) of loop 78 where L = 0, 1, ..., etc. Since the largest value of MM is, for this example, 595, the only values of arrays C1( ), C2( ) printed out are those at the end of the first iteration of loop 78.

These are certain constant parameters that are not read into the program. These are described in section 9.2 (table 20).

For various literature references for material constants, we recommend [1,3,4,5,6,7,9,12,13] in the supplemental bibliography in chapter 13.

The output of DISTRIB is listed and described in section 9.2. The print-out in block #30 is effected after every MM = (1 + L·KPRINT)th iteration (for L = 0, 1, 2, ...) of loop 78, which is the loop computing the concentrations at the successive times t_{MM}. Since KPRINT = 1000 for this example and only 595 iterations of loop 78 are required to compute the concentrations at time t_{MM} = TFINAL = 0.3 h in this example, the only concentrations printed, under block #30, are those after the first iteration and are shown in table 24. From this table and computational block #30, we see printed out the values of

C1(J), J = 1, NL, 15

C2(J), J = NR, N2, 20.

Since, at time t_1 = 0.0389 (BTM units), there is only one fixed grid point in the oxide, the only value in the oxide that is printed is the grid point at the oxide-oxygen interface. However, between the values of C1( ) and C2( ) that we have printed out in table 24, there appear the values of boron concentration at the moving boundary grid points in the oxide and silicon CB(1), CB(2). Also printed in the same row with these quantities are DTS (in BTM units) and the mesh width lengths in the oxide and silicon, respectively, DZ and DY in units of BLTH = 0.02 μm. In the final print-out (table 26), BTM = 0.0030 h is printed. Also, table 24 shows the last index MM of loop 78 and index NL of the first fixed grid point to the left of the moving boundary in the oxide. Finally, in the same line appears the index NR of the first grid point to the right of the moving boundary in the silicon.

The printout from block #34 is shown in table 25. The main output are the concentrations

C1(J), J = 1, NL, 15

C2(J), J = NR, N2, 20.

These are the concentrations at the first time that TTMMP1 (= 99.756). is greater than TFINAL (in BTM units). It is seen from table 25 that this occurs at the end of the MMth = 595 iteration. The meaning of the other output in table 25 is similar to the corresponding output in table 24 that has already been described (see also sec. 9.23).

The concentrations at time TTMMP1 = 99.756 (in BTM units) or TTMMP1 ≈ TFINAL = 0.3 h in the oxide are plotted in figure 11. The distances from the oxide-oxygen interface (scaled in ZOP1 units which is the length of the oxide region at time TTMMP1 ≈ TFINAL) are plotted on the x-axis. The concentrations at the positions of the first and (1 + L*NMOD)th grid points (for L = 1, 2, ...) are plotted on the y-axis, in this example, NMOD = 8. The value of ZOP1 (in micrometers) is computed in the last table (26). At the same time that the concentrations in the oxide are plotted in figure 11, the concentrations in the silicon are plotted in figure 12. The x-axis in this figure represents distance in the unit UN2 = 2SQRT(DF2·TTMMP1) (where TTMMP1 is very nearly equal to TFINAL) from the oxide-silicon moving boundary. The concentrations at the moving boundary (in the silicon) and at the location of the grid point with index NR are plotted. After this, the concentrations at the grid points with the successive locations (NR - 1) DZ + (J - 2)·NMOD1·DZ, for J = 3, ..., ND (with NMOD1 = 4 in this particular example) are plotted. The unit length of the abscissa, UN2 (micrometers), is computed in table 26 from computational block #35. In addition, in table 26 are shown all the input quantities in units of micrometers and hours as printed out in computational block #35. Besides the input data that are printed out, the quantities ZO(TFINAL) (this is the same as ZOP1, when TFINAL = TTMMP1) and 2·SQRT(DF2·TFINAL) in micrometers which, as we have already explained, are used as quantities to scale the x-axis in the oxide and silicon concentration plots in figures 11 and 12. Also printed is the value of BTM in hours; hence, the times in tables 24 and 25 which are in units of BTM, can be converted to hours, if so desired.

Table 24. Example of output of concentration in oxide and silicon at end of iteration MM = (1 + L*KPRINT) in loop 78 for L = 0.

```
C1(.) AT TIME       .038998095
   2.52280046
MM       NL     NR
    1     1      2
     C1(1)          CB(1)          CB(2)          DTS            DZ             DY
   2.52280046     3.88795223      .77759045      .03899809      .54112554      .71428571
C2(.) AT TIME       .038998095
    .98484473     1.00000000     1.00000000     1.00000000     1.00000000     1.00000000     1.00000000
   1.00000000     1.00000000     1.00000000     1.00000000     1.00000000     1.00000000     1.00000000
   1.00000000     1.00000000     1.00000000     1.00000000
```

Table 25. Example concentrations in oxide and silicon when TTMMP1 ≈ TFINAL.

```
C1(.) AT TIME     99.756364064
    .03302989      .87542717      .83500469      .82160333      .81678325      .81453516      .81331455
MM       NL     NR
  595     32     12
     C1(1)          CB(1)          CB(2)          DTS            DZ             DY
    .03302989      .81307112      .16261422      .28899809      .54112554      .71428571
C2(.) AT TIME     99.756364064
    .18953686      .84437386      .98733067      .99958049      .99999438      .99999997     1.00000000
   1.00000000     1.00000000     1.00000000     1.00000000     1.00000000     1.00000000     1.00000000
   1.00000000     1.00000000     1.00000000
```

Table 26. Listing of input physical data (micrometers, hours) and important computational data.

```
DATA FOR STEP NO       1
      DF1             DF2           CONPRO         CONOUT          CNSTA          CNSTB          CNTAU
    .00100000       .13300000     1.00000000      .00000000      .11000000      .51000000      .00000000
      SEG            ALPHA          WFRC           TFINAL      ZO(TFINAL)   2*SQRT(DF2*TFINAL)
    .20000000       .44000000      .05000000      .30000000      .34001236      .39949969
   CB(1)           CB(2)          FIRST+DT       REAL TIME       BDRY
    .81307112       .16261422      .00086917      .30001914     5.00000000
     CMAX           CBULK           Y1S            Y2S            X2S            X1S            BTM
   1.00000000     1.00000000     5.00000000     2.00000000     4.00000000     2.00000000     .00300752
  NZO   NPE   ND   NYO    NH   KPRINT
    3  1500   30   350    12   1000
```

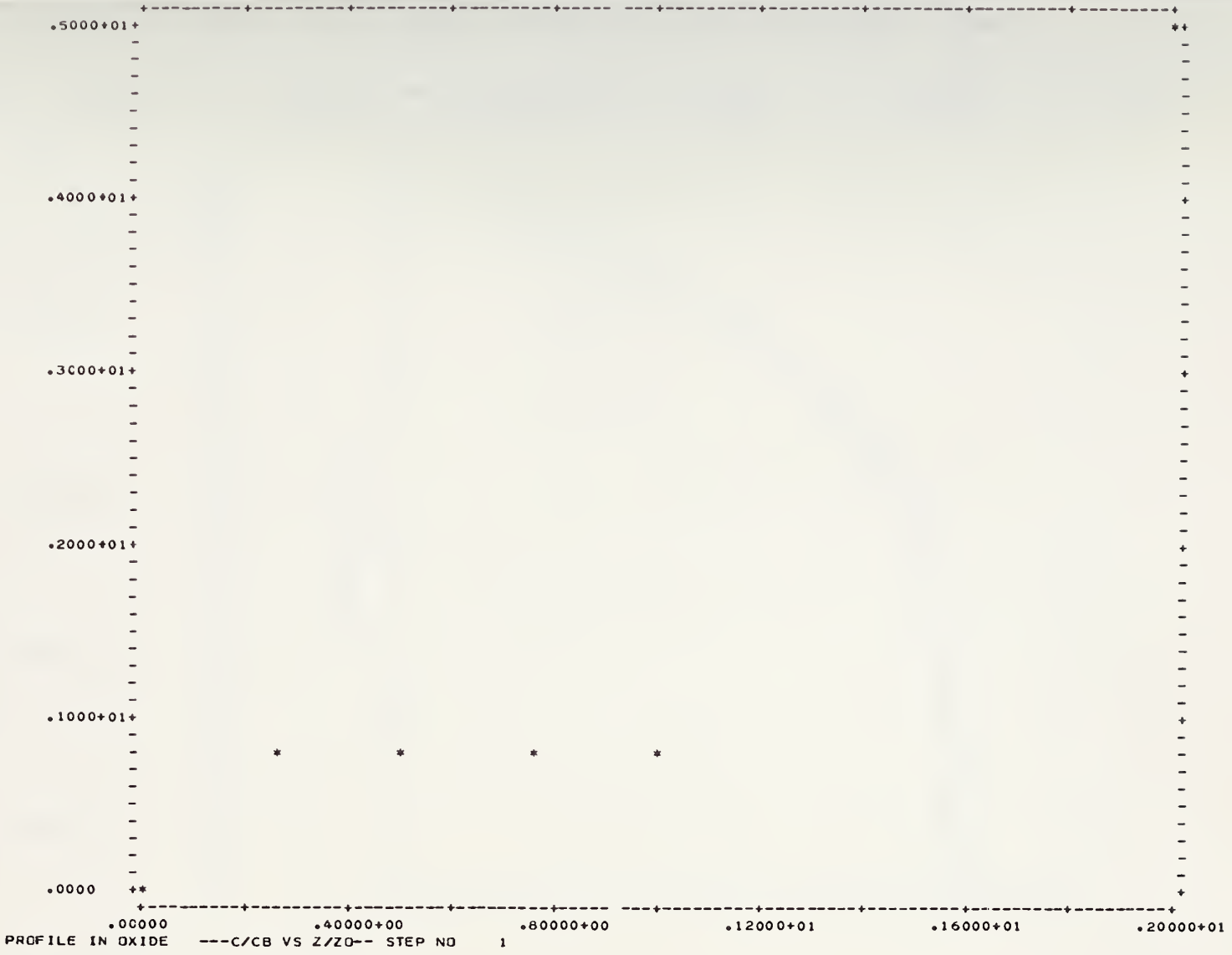


Figure 11. Plot of example concentrations in oxide when TTMMP1 ≈ TFINAL.

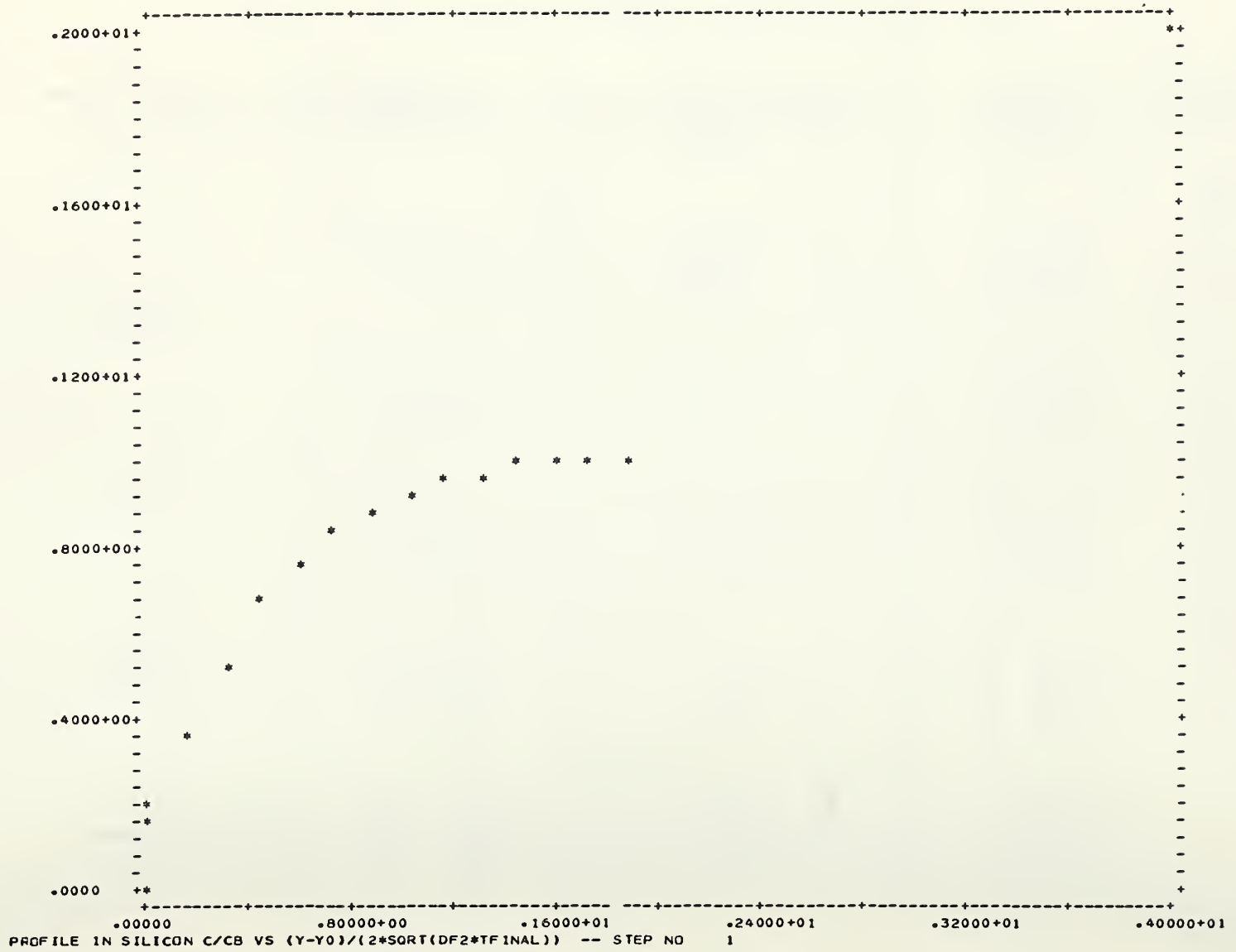


Figure 12. Plot of example concentrations in silicon when TTMMP1 ≈ TFINAL.

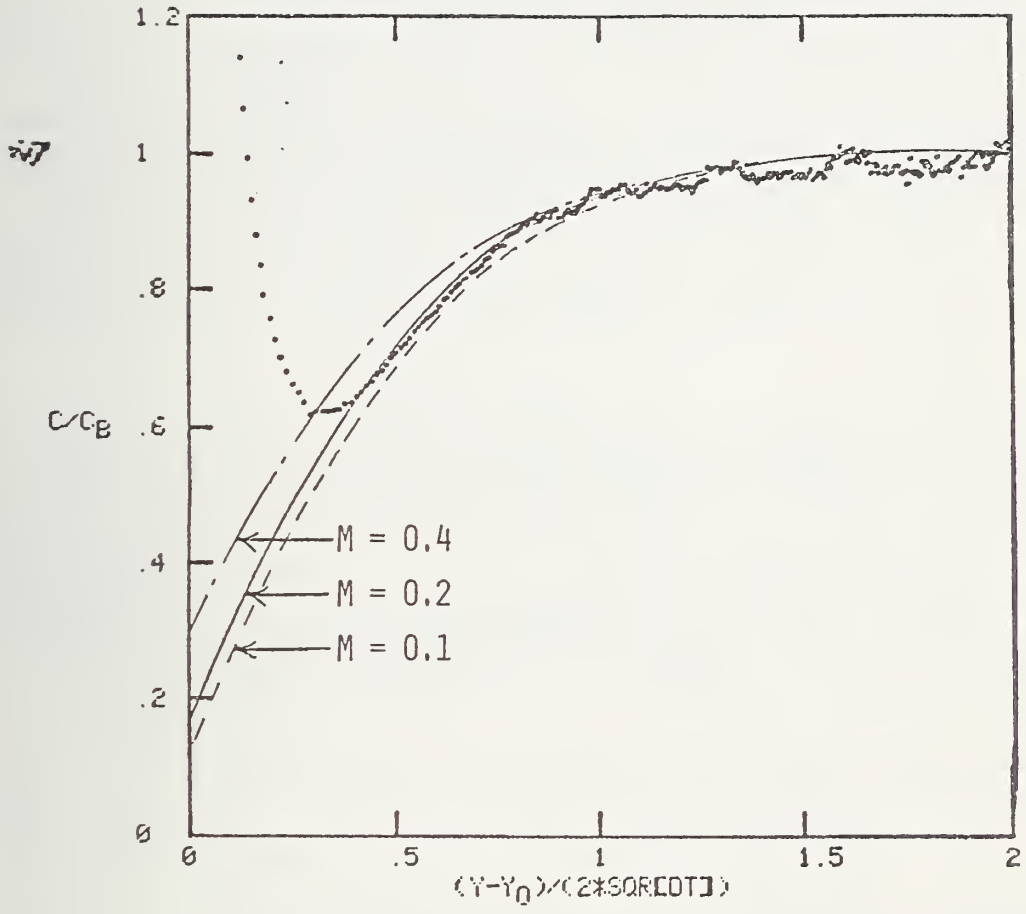


Figure 13. Comparison of three calculated impurity concentration distributions in silicon with experimentally determined concentration data.

10. VERIFICATION OF PROGRAM

10.1. Validation of the Program

A solution calculated by DISTRIB for a problem with the specified concentration $C_1(0,t) = C_o$ at the oxygen-oxide interface has been compared with an analytic similarity solution given in [4]. The calculated boundary concentrations $C_1(z_o(t), t)$ agree with the similarity solution [5] to within a few percent for a variety of problems with differing segregation constants. The calculated solutions also have qualitative properties possessed by the similarity solution such as moving boundary concentrations that are constant with time. There is also an analytic solution to special problems with zero flux at the oxygen-oxide interface. In this problem, the concentration in the oxide or z-region is constant and DISTRIB's calculated solution is constant also for these conditions.

10.2. Limitations of the Program

The previous remarks must all be qualified for very short times, up to $5\Delta t$ [5]. The difficulties in obtaining the solutions for the first few iterations are similar to those described in [2] for analogous problems. The problem with the inaccuracy of the first few iterations can be somewhat mollified, however, by choosing smaller and smaller values of WFRC. Our recommendation for obtaining a solution at small times is to calculate solutions using a series of decreasing values of WFRC and then use those solutions to extrapolate the solutions to time zero.

11.1. DISTRIB: VERSION 1 MAIN PROGRAM

```
C      START  COMPUTATION BLOCK NUMBER   1
       IMPLICIT DOUBLE PRECISION (A-H,O-Z)
       REAL  XRL(200)  ,YRL(200)
       DIMENSION TT(2000),ZO(2000),AF(2000),BF(2000),PARAM(21)
       DIMENSION  BT(20), DFA(20),DFB(20),CONPR(20),CNST1(20),CNST2(20),
      1 CNTA(20),TFINN(20),ZONEW(20),TNEW(20)
       COMMON/BL20/DT,DY,DZ,THETA
       COMMON/BL21/DF1,DF2
       COMMON/BL22/DZP,DYP
       COMMON/BL23/ B2M,B2R,B2L,A2L,A2M,A2R,A1L,A1M,A1R,B1L,B1M,B1R
       COMMON/BL24/C2L,C2M,C2R,C1L,C1M,C1R
       COMMON/BL25/P(2),Q(2),PA(2),QA(2),C(2),CR(2),R(2),S(2),SM(2)
       COMMON/BL26/P1(2000),P2(2000),Q1(2000),Q2(2000),D1(2000),D2(2000)
       COMMON/BL28/N2M1,NLM1,N2 , NRP1,NL,NR
       COMMON/BL29/C1(2000),C2(2000)
       COMMON /BL30 / CO,CK,TZERO,JEE,ALPHA,CAPA,SEG,CMVY
       COMMON /BL31 / DIST(100), SOL(100)
       COMMON/BL32/CONPRO,CONOUT
C      END    COMPUTATION BLOCK NUMBER   1
     C =========1=========2=========3=========4=========5=========6=========7=========8
C      START  COMPUTATION BLOCK NUMBER   2
       READ(5,4)CBULK,CMAX,ALPHA,TFINAL,SEG,DF1,DF2,WFRC,BDRY,CONPRO,
      1CONOUT,CNSTA,CNSTB,CNTAU,Y1S,Y2S,X2S,X1S
       READ(5,1)NZO,MME,ND,NYO,NH,KPRINT
 1     FORMAT (6I6)
 4     FORMAT (5F10.5)
 5     FORMAT (7F14.8)
 41    FORMAT ( 40H MM       NL     NR                                    )
 43    FORMAT (40H MME MUST BE INCREASED                                  )
  8    FORMAT (16H C1(.) AT TIME          ,1F14.9)
  9    FORMAT (16H C2(.) AT TIME          ,1F14.9)
 26    FORMAT( 1H1 )
  27   FORMAT(40H TLIN CNTAU TTMM SLOPEZCRIT CNSTX                )
  36   FORMAT ( 45H PROFILE IN OXIDE   ---C/CB VS Z/ZO-- STEP NO  ,1I5 )
 32    FORMAT  (1H1    )
  39   FORMAT ( 40H TIMEND TNEW(J) BTM  CNTAU TLIN BT(J)      )
  45   FORMAT  (101H      C1(1)         CB(1)         CR(2)            DTS
     1           DZ                 DY                                 )
C      END    COMPUTATION BLOCK NUMBER   2
  40   FORMAT ( 52H ZOP1,DF2,UN2,DY,DZ,ENC,(NL-1),TTMMP1,SCALE2,X2SCUT   )
  38   FORMAT (  19H DATA FOR STEP NO      ,1I5)
 28    FORMAT( 33H NZO MME ND NYO NH JSTOP                )
 29    FORMAT( 43H S(1) ST STM E1 E2 E3  ERROR                    )
  30   FORMAT(  33H LIST OF ZO(.)                        )
  31   FORMAT(  33H LIST OF TT(.)                        )
  33   FORMAT (100H       DF1             DF2          CONPRO         CONOUT
     1         CNSTA           CNSTB          CNTAU                 )
 35    FORMAT(  67H PROFILE IN SILICON C/CB VS (Y-YO)/(2*SQRT(DF2*TFINAL)
     1)  -- STEP NO   ,1I5   )
 20    FORMAT (21H LIST  AF(.)                )
 21    FORMAT (21H LIST  BF(.)                )
  22   FORMAT ( 30H COXIDE CBULKZO1 ALPHA TFINAL     )
  23   FORMAT (103H SEG PRINT DF1 DF2 WFRC BDRY  CMAX CONPRO CONOUT
     1 CNSTA CNSTB CNTAU  Y1S Y2S X2S                                  )
 24    FORMAT( 33H LIST N2 NH  N1 NR  NL                  )
 25    FORMAT( 33H LIST DZ DY  DT TT(1) SM(1) SM2)     )
 19    FORMAT (21H LIST  P1(.) Q2(.)          )
  18   FORMAT ( 33H LIST B2L B2R B2M B1L B1R B1M          )
```

```
  17  FORMAT ( 33H LIST A2L A2R A2M A1L A1R A1M          )
  15  FORMAT( 20H LIST OF P1(.),Q1(.) )
  14  FORMAT (60H VALUES OF RM RZ  RY B C F CMV1 CMV2 P1S P2S AT MM=
     1                  ,1I9 )
  13  FORMAT (61H VALUES OF FNL DZP DYP TES ZOP1 FNP  TT(MMP1) SES AT
     1  MM=        ,1I8)
  12  FORMAT (28H DIVIDE FAULT AT MOV. BDRY.          )
  11  FORMAT (28H DIVIDE FAULT AT LOOP 107            )
  10  FORMAT (28H DIVIDE FAULT AT LOOP 105            )
  34  FORMAT (101H      SEG          ALPHA         WFRC              TFINAL
     1     ZO(TFINAL)   2*SORT(DF2*TFINAL)                          )
   6  FORMAT ( 7F10.5 )
  37  FORMAT( 80H    CB(1)          CB(2)         FIRST+DT        REAL TIME
     1      BDRY                                                    )
  44  FORMAT (101H       CMAX          CBULK         Y1S               Y2S
     1       X2S            X1S            BTM                      )
  42  FORMAT ( 50H   NZO   MME   ND   NYO     NH  KPRINT                 )
C     DATA FIXED IN VERSION 1
      DATA   DFA(1),DFB(1), CONPR(1),CNST1(1),CNST2(1), CNTA(1),TFINN(1)
     1  /.0001D0,.0001D0, .01D0, .027D0,.1D0,.1D0,.01D0/
      TW=0.
      ZW=0.
      JSAVE=0
      JSTOP  =1
      ZO1=0.
      COXIDE=1.
C     JSTOP TELLS NUMBER OF  DRIVE IN STEPS
      JSTM1=JSTOP-1
    C ===============================================================
C     START  COMPUTATION BLOCK NUMBER   3
      SCALE2 = 2.* SQRT(DF2*TFINAL)
C     THESE MATERIAL PARAMETERS ARE RESCALED THUS WE HAVE TO STORE
C     THESE GIVEN PARAMETERS IN UNITS OF MICRONS AND HOURS
      PARAM(1)=DF1
      PARAM(2)=DF2
      PARAM(3)= CONPRO
      PARAM(4)=CONOUT
      PARAM(5)=CNSTA
      PARAM(6)=CNSTB
      PARAM(7)=CNTAU
C     END    COMPUTATION BLOCK NUMBER   3
    C ===============================================================
C     START  COMPUTATION BLOCK NUMBER   4
      BLTH= .02
C     DFC MODIFIES PROGRAM  TO DIMENENTIONALISE ABOUT DFC*DF2
      DFC=  1.
      DF2=DF2*DFC
      BTM=(BLTH**2)/DF2
      CONPRO=CONPRO*(BTM/BLTH)
      CNSTA=CNSTA/BLTH
      CNSTB=CNSTB/DF2
      CNTAU=CNTAU/BTM
      BDRY=BDRY*(1./BLTH)
      ZO1=ZO1*(1./BLTH)
      DF1=DF1/DF2
      DF2=1./DFC
      TIMEND=TFINAL*(1./BTM)
C     END    COMPUTATION BLOCK NUMBER   4
C     ZO1 IS POSITION ON Z-AXIS WHERE THE NON DEGENERATE (ZO(ZERO)
C     NOT EQUAL TO ZERO ) MOVING BOUNDARY BEGINS
C     ZCRIT IS THE POSITION ON THE Z-AXIIS WHERE THE MOVING BOUNDARY
C      CHANGES FROM LINEAR TO PARABOLIC
C     CALCULATE DY,DZ,DT AND AUXILLARY QUANTITIES
      YO1=ALPHA*ZO1
```

```
    C ==================================================================
C     START  COMPUTATION BLOCK NUMBER   5
      DY=BDRY/NYO
      FNZO=FLOAT(NZO)
      DZ=DY*(1./(ALPHA*FNZO))
C     END    COMPUTATION BLOCK NUMBER   5
    C ==========================================================
C     START  COMPUTATION BLOCK NUMBER   6
      CNA=.5*CNSTA
      CNB=CNSTB
C     TAKE OUT FOLLOWING CARD IF CNSTX IS KNOWN
      CNSTX=SQRT(CNA**2+CNB*CNTAU) -CNA
C     END    COMPUTATION BLOCK NUMBER   6
      CNTAU=CNSTX*(CNSTX+CNSTA) /CNSTB
    C ==========================================================
C     START  COMPUTATION BLOCK NUMBER   7
      DM=DZ*WFRC
C      INSERT IF MOVING BDRY IS      LINEAR AND THEN PARABOLIC
      TLIN=CNTAU
C   IF MOVING BOUNDARY STARTS AT Y'0. THESE CARDS TAKE CARE OF SITUATIO
C        N THAT LINEAR PART OF MOVING BOUNDARY IS NOT PRESENT
      IF( TLIN .LE. .00000001 )DT=(DM*(DM+CNSTA))/CNSTB
      IF( TLIN .LE. .00000001 )DTS=DT
      IF( TLIN .LE. .00000001 )ZCRIT=0.
      IF( TLIN .LE. .00000001 )SLOPE=1.
      IF(TLIN .LE.  .00000001 )GO TO 216
      ZCRIT=CNSTX
      SLOPE=ZCRIT/CNTAU
      DT=DM/SLOPE
      DTS=DT
  216 CONTINUE
C     END    COMPUTATION BLOCK NUMBER   7
    C ==========================================================
C     START  COMPUTATION BLOCK NUMBER   8
      PARAM(10)=DT*BTM
C     END    COMPUTATION BLOCK NUMBER   8
    C ==========================================================
C     START  COMPUTATION BLOCK NUMBER   9
C     CALCULATE INITIAL  INITIAL DISTRIBUTIONS
      DUM=YO1/DY
      NR=INT(DUM)+2
      N1=INT(ZO1/DZ)+1
      NL=N1
      N2=NYO+1
      N2M1=N2-1
C     END    COMPUTATION BLOCK NUMBER   9
      DO 98 J=1,N1
      C1(J)=COXIDE
   98 CONTINUE
      CB(1)=COXIDE
    C ==========================================================
C     START  COMPUTATION BLOCK NUMBER  10
      N1P1=N1+1
      NH=NH +NR
      NHP1=NH+1
      DO 96 J=NR  ,NH
      C2(J)=CMAX
   96 CONTINUE
      DO 97 J=NHP1,N2
      C2(J)=CBULK
   97 CONTINUE
      CB(2)=CMAX
C     END    COMPUTATION BLOCK NUMBER  10
```

```
C      START  COMPUTATION BLOCK NUMBER  11
       IF(Z01 .LE.  ZCRIT) TTMM = Z01/SLOPE
       IF(Z01 .GT.  ZCRIT) TTMM=  (Z01*(Z01+CNSTA))/CNSTB
C      END    COMPUTATION BLOCK NUMBER  11
     C =========1=======1======1=======1========1=======1======1=======
C      START  COMPUTATION BLOCK NUMBER  12
       FNL=FLOAT(NL)-1.
       FNR=FLOAT(NR)-1.
       SM(1)=(Z01-FNL*DZ)/DZ
       SM(2)=(FNR*DY-Y01)/DY
C      END    COMPUTATION BLOCK NUMBER  12
     C =========1=======1======1=======1========1=======1======1=======
C      START  COMPUTATION BLOCK NUMBER  13
C      MAIN LOOP
       P2(N2)=0.
       Q2(N2)=C2(N2)
C      END    COMPUTATION BLOCK NUMBER  13
     C =========1=======1======1=======1========1=======1======1=======
C      START  COMPUTATION BLOCK NUMBER  14
       ZOPS=0.
       ZOP1=0.
       TJ=0.
       NORUN=1
       KPOW=0
       JSAVE=0
       DO 78  MM=1,MME
C      END    COMPUTATION BLOCK NUMBER  14
     C =========1=======1======1=======1========1=======1======1=======
C      START  COMPUTATION BLOCK NUMBER  15
       KW=MOD(MM,KPRINT)
C      END    COMPUTATION BLOCK NUMBER  15
     C =========1=======1======1=======1========1=======1======1=======
C      START  COMPUTATION BLOCK NUMBER  16
C      SPEEDS UP MOVING BOUNDARY
       ZOACEL=(ZOP1-ZOPS)/DZ
       IF(ZOACEL .GE.  .2 ) GO TO 88
       KPOW=KPOW+1
       IF(KPOW .GT. 100 )DTS=DTS+.05
       IF(KPOW .GT. 100 )DT=DTS
       IF(KPOW .GT. 100 )CALL TRIDNL
       IF(KPOW .GT. 100 )KPOW=KPOW-100
 88    CONTINUE
C      ZOPS IS ZO(TTMM)
       ZOPS=ZOP1
C      END    COMPUTATION BLOCK NUMBER  16
     C =========1=======1======1=======1========1=======1======1=======
C      START  COMPUTATION BLOCK NUMBER  17
       IF(TJ .GE. 1.) NL=NL+1
       IF(TJ .GE. 1.)C1(NL)=CB(1)
       IF(TJ .GE. 1.)FNL=FNL+1.
       NLM1=NL-1
       NLM2=NL-2
       IF(TJ .GE. 1.)DT=DTS
       IF(TJ .GE. 1.)  CALL TRIDNL
       IF(TJ .GE.1. ) SM(1)=0.
C      END    COMPUTATION BLOCK NUMBER  17
     C =========1=======1======1=======1========1=======1======1=======
C      START  COMPUTATION BLOCK NUMBER  18
       MMP1=MM+1
       TTMMP1=TTMM+DT
       IF( TTMMP1  .LE. TLIN )  ZOP1 = ( TTMMP1 - TW)* SLOPE  +ZW
       IF(TTMMP1 .GT. TLIN)ZOP1=SQRT((CNA**2)+CNB*(TTMMP1-TW))-CNA+ZW
       YOP1=ALPHA*ZOP1
C      END    COMPUTATION BLOCK NUMBER  18
```

```
    C ========================================================================
C     START  COMPUTATION BLOCK NUMBER  19
C     CHECK IF NEW CELLS ARE ADDED
      SES=(FNR*DY-YOP1)/DY
      IF(SES .LT. 0.) NR=NR+1
      IF(SES .LT. 0.) FNR=FNR+1.
      IF(SES .LT. 0.)SM(2)=1.
      NRP1=NR+1
      NRP2=NR+2
      NRM1=NR-1
C     END    COMPUTATION BLOCK NUMBER  19
    C ========================================================================
C     START  COMPUTATION BLOCK NUMBER  20
      MMP2=MMP1+1
      TTMMP2=TTMMP1+DT
      IF( TTMMP2 .LE. TLIN ) ZOP2=(TTMMP2 -TW )* SLOPE  + ZW
      IF( TTMMP2 .GT. TLIN ) ZOP2=SQRT((CNA**2)+CNB*(TTMMP2-TW))-CNA+ZW
      TJ=(ZOP2-FNL*DZ)/DZ
      IF(TJ .LT. 1.) GO TO 219
      ZOP1=(FNL+1.)*DZ
      IF( ZOP1 .LE. ZCRIT)TTMMP1=(ZOP1-ZW)/SLOPE + TW
      IF( ZOP1 .GT. ZCRIT)TTMMP1=((ZOP1-ZW)*( ZOP1-ZW+CNSTA))/CNSTB +TW
      DT=TTMMP1-TTMM
      CALL TRIDNL
      YOP1=ALPHA*ZOP1
 219  CONTINUE
C     END    COMPUTATION BLOCK NUMBER  20
C     CONSERVATION OF NUMBER CALCULATION
      S1M=C1(NL)*DZ*.5+(C1(NL)+CB(1))*SM(1)*.5*DZ
      IF(NL .EQ. 1) S1M=(C1(1)+ CB(1))*SM(1)*.5*DZ
      IF(NL .GE. 2)S1M=S1M+.5*DZ*C1(1)
      IF(NL .LE. 2) GO TO 222
      DO 111 J=2,NLM1
      S1M=S1M+DZ*C1(J)
 111  CONTINUE
 222  S2M=C2(NR) *DY*.5+(C2(NR)+CB(2))* SM(2)*.5*DY
      DO 112  J=NRP1,N2M1
      S2M=S2M+C2(J)*DY
 112  CONTINUE
    C ========================================================================
C     START  COMPUTATION BLOCK NUMBER  21
      S(1)=(ZOP1-FNL*DZ)/DZ
      S(2)=(FNR*DY-YOP1)/DY
C     END    COMPUTATION BLOCK NUMBER  21
    C ========================================================================
C     START  COMPUTATION BLOCK NUMBER  22
      IF(MM .EQ. 1) CALL TRIDNL
C     END    COMPUTATION BLOCK NUMBER  22
    C ========================================================================
C     START  COMPUTATION BLOCK NUMBER  23
      IF(NL.GT. 1) GO TO 223
      IF(NL .EQ. 1)DM1=S(1)*DZ
      IF(NL .EQ. 1)DM2=DM1*DM1/(2.*DT)
      IF(NL .EQ. 1 .AND.MM .EQ.  1) DM2=0.
      IF(NL .EQ. 1)DM3=DM2
      IF(NL .EQ. 1)DM=1./( DF1+DM2+DM1*CONPRO)
      IF(NL .EQ. 1 )P1(1)=DM*DF1
      IF(NL .EQ. 1)Q1(1)=DM*(C1(1)*DM3+DM1*CONPRO*CONOUT)
C     END    COMPUTATION BLOCK NUMBER  23
```

```
C ===========================================================================
C     START  COMPUTATION BLOCK NUMBER  24
 223  IF(NL .GT. 1 )DM=1./(DZ*DZ/(2.*DT) +DZ* CONPRO+DF1 )
      IF(NL .GT. 1 )P1(1)=DF1*DM
        IF(NL .GT. 1 )Q1(1)=(+C1(1)*(DZ**2)/(2.*DT)+DZ*CONPRO*CONOUT)*DM
C     END    COMPUTATION BLOCK NUMBER  24
C ===================================================================
C     START  COMPUTATION BLOCK NUMBER  25
      IF(NL .LE. 2) GO TO 213
      DO 203 J=2,NLM1
      AF(J)=A1M*C1(J)
 203  CONTINUE
      DO 205  J=2,NLM1
      JM1=J-1
      Q1(J)=(AF(J)-Q1(JM1)*A2L)/D1(J)
 205  CONTINUE
C     END    COMPUTATION BLOCK NUMBER  25
      IF(MM .EQ. KD) WRITE( 6,15)
      IF(MM .EQ.KD) WRITE(6,5)(P1(J),J=1,NLM1,1)
      IF(MM .EQ.KD)WRITE(6,5) (Q1(J),J=1,NLM1,1)
      IF( MM .EQ. KD )  WRITE(6,20)
      IF(MM .EQ. KD)WRITE(6,5) (AF(J) ,J=2,NLM1,1)
C ===================================================================
C     START  COMPUTATION BLOCK NUMBER  26
 213  CONTINUE
      DO 204  J=NRP1,N2M1
      BF(J)=B1M*C2(J)
 204  CONTINUE
      NE=N2-NRP1
      DO 207 J=1,NE
      K=N2-J
      KP1=K+1
      Q2(K)=(BF(K)-Q2(KP1))/D2(K)
 207  CONTINUE
C     END    COMPUTATION BLOCK NUMBER  26
C ===================================================================
C     START  COMPUTATION BLOCK NUMBER  27
      CALL CENTER
C     END    COMPUTATION BLOCK NUMBER  27
C ===================================================================
C     START  COMPUTATION BLOCK NUMBER  28
      DO 209 J=NRP1,N2M1
      JM1=J-1
      C2(J)=P2(J)*C2(JM1)+Q2(J)
 209  CONTINUE
C     END    COMPUTATION BLOCK NUMBER  28
C ===================================================================
C     START  COMPUTATION BLOCK NUMBER  29
      IF(NL .LE. 1) GO TO 215
      DO 210 K=1,NLM1
      J=NL-K
      JP1=J+1
      C1(J)=P1(J)*C1(JP1)+Q1(J)
 210  CONTINUE
 215  CONTINUE
C     END    COMPUTATION BLOCK NUMBER  29
C ===================================================================
C     START  COMPUTATION BLOCK NUMBER  30
      IF(KW .EQ. 0  .OR.  KW  .GT. 1)  GO TO 799
      WRITE(6,32)
      IF(KW .EQ. 1) WRITE(6,8) TTMMP1
      IF(KW  .EQ. 1  )WRITE(6,5) (C1(J) ,J=1,NL,15 )
      WRITE(6,41)
      IF(KW .EQ. 1) WRITE(6,1) MM,NL,NR
```

```
      WRITE(6,45)
      WRITE(6,5) C1(1),CB(1),CB(2),DTS,DZ,DY
      IF(KW .EQ. 1) WRITE(6,9) TTMMP1
      IF(KW .EQ. 1)WRITE(6,5) (C2(J),J=NR  ,N2,20)
  799 CONTINUE
C     END    COMPUTATION BLOCK NUMBER  30
    C ==================================================================
C     START  COMPUTATION BLOCK NUMBER  31
      SM(1)=S(1)
      SM(2)=S(2)
      TTMM=TTMMP1
C     END    COMPUTATION BLOCK NUMBER  31
C     SHIFT TO NEW MOVING BOUNDARY AND CALC  NEW PARAMETERS
    C ==========================================================
C     START  COMPUTATION BLOCK NUMBER  32
      IF (  TIMEND .GT. TTMMP1 ) GO TO 78
C     END    COMPUTATION BLOCK NUMBER  32
    C ==========================================================
C     START  COMPUTATION BLOCK NUMBER  33
      IF(JSTOP .EQ. 1 ) GO TO 800
C     END    COMPUTATION BLOCK NUMBER  33
                MTIME=MTIME+1
                TT(MTIME)=TTMMP1
                ZO(MTIME)=ZOP1
C     CALCULATE CONERVED QUANTITIES AND GRAPH OF STANDARD S DIMENSION
      IF(KW .LT.  1) GO TO 800
      AF(1)=YOP1
      DM=S(2)
      AF(2)=AF(1)+DM *DY
      BF(1)=CB(2)
      BF(2)=C(2)
      DO 803 J=3,ND
      AF(J)=AF(2)+(J-2.)*DY*10.
      K=NR  +(J-2)*10
      BF(J)=C2(K)
  803 CONTINUE
      DO 250 J=1,ND
      XRL(J)=AF(J)
      YRL(J)=BF(J)
  250 CONTINUE
      WRITE(6,26)
      CALL PLOT  (ND,XPL,YRL)
C     CONSERVATION OF NUMBER CALCULATION
      S1 =C1(NL)*DZ*.5+(C1(NL)+CB(1))* S(1)*.5*DZ
      IF(NL .EQ. 1) S1  =(C1(1)+CB(1))*S(1)*.5 *DZ
      IF(NL .GE. 2)S1 =S1 +.5*DZ*C1(1)
      IF(NL .LE. 2) GO TO 221
      DO 113 J=2,NLM1
      S1 =S1 +DZ*C1(J)
  113 CONTINUE
  221 S2 =C2(NR) *DY*.5+(C2(NR)+CB(2))*  S(2)*.5*DY
      DO 114  J=NRP1,N2M1
      S2 =S2 +C2(J)*DY
  114 CONTINUE
      STM= S1M+S2M
      ST=S1+S2
      E1=ST-STM
      RY=(DF2*DT)/(DY**2)
      E2=-DT*(CONPRO)*(C1(1)-CONOUT)
      E3=RY*(C2(N2)-C2(N2M1))*DY
      ERROR=E1-(E2+E3)
```

```
C ==========================================================
C     START  COMPUTATION BLOCK NUMBER  34
  800 CONTINUE
      WRITE(6,26)
                    WRITE(6,8) TTMMP1
                      WRITE(6,5) (C1(J) ,J=1,NL, 5 )
      WRITE(6,41)
                    WRITE(6,1) MM,NL,NR
      WRITE(6,45)
      WRITE(6,5) C1(1),CB(1),CB(2),DTS,DZ,DY
                    WRITE(6,9) TTMMP1
                   WRITE(6,5) (C2(J),J=NR  ,N2,20)
      NMOD=8
      NMOD1=4
      UN1=1./ZOP1
      UN2=2.*SQRT(DF2*TTMMP1)
      UN2=1./UN2
      EZ=DZ*UN1
      NLP1=NL+1
      NLP2=NLP1+1
      NLP3=NLP2+1
      AF(1)=0.
      BF(1)=C1(1)
      NC=INT((NL-1)/NMOD)
      FNC=FLOAT(NC)
      IF(NC .LT.  1 ) GO TO 231
      DO 230 J=1,NC
      K=J*NMOD +1
      JP1=J+1
      BF(JP1)=C1(K)
      AF(JP1)=(K-1)*EZ
  230 CONTINUE
  231 NCP1=NC+2
      NCP2=NC+3
      NCP3=NC+4
      AF(NCP1)=ZOP1*UN1
      BF(NCP1)=CB(1)
      BF(NCP2)=Y1S
      AF(NCP2)=X1S
      AF(NCP3)=0.
      BF(NCP3)=0.
      DO 252 J=1,NCP3
      XRL(J)=AF(J)
      YRL(J)=BF(J)
  252 CONTINUE
      WRITE(6,26)
      CALL PLOT  (NCP3,XRL,YRL)
      WRITE(6,36)NORUN
      EY=DY*UN2
      AF(NLP2)=0.
      BF(NLP2)=CB(2)
      BF( NLP3)=C2(NR)
      AF(NLP3)=AF(NLP2)+S(2)*EY
      DO 232 J=3,ND
      K=NLP3+J-2
      KS=NR+(J-2)*NMOD1
      BF(K)=C2(KS)
      AF(K)=AF(NLP3)+(J-2)*EY*NMOD1
  232 CONTINUE
      DO 234 J=1,ND
      K=J+NLP1
      AF(J)=AF(K)
      BF(J)=BF(K)
```

```
  234 CONTINUE
      X2SCT=X2S/2.
      NDE=0
      DO 235 J=1,ND
      IF(AF(J) .LT.  X2SCT) NDE=NDE+1
 235  CONTINUE
      NDP1=NDE+1
      AF(NDP1)=X2S
      BF(NDP1)=Y2S
      NDP2=NDP1+1
      AF(NDP2)= 0.
      BF(NDP2)=0.
      DO 251 J=1,NDP2
      XRL(J)=AF(J)
      YRL(J)=BF(J)
  251 CONTINUE
      WRITE(6,26)
      CALL PLOT  (NDP2,XRL,YRL)
C     END    COMPUTATION BLOCK NUMBER  34
    C ==========================================================
C     START  COMPUTATION BLOCK NUMBER  35
      WRITE (6,35) NORUN
      PARAM(8)=CB(1)
      PARAM(9)=CB(2)
      PARAM(10)=DT*BTM
      PARAM(11)=TTMMP1*BTM
      PARAM(12)=BDRY*BLTH
      ZOCR=ZOP1*BLTH
      WRITE(6,32)
      WRITE(6,38)  NORUN
      WRITE(6,33)
      WRITE(6,5) (PARAM(J),J=1,7 )
      WRITE (6,34)
      WRITE(6,5) SEG,ALPHA,WFRC, TFINAL,ZOCR ,SCALE?
      WRITE(6,37)
      WRITE(6,5) (PARAM(J) , J=8,12 )
      PARAM(12)=CMAX
      PARAM(13)=CBULK
      PARAM(14)=Y1S
      PARAM(15)=Y2S
      PARAM(16)=X2S
      PARAM(17)=X1S
      PARAM(18)=BTM
      WRITE(6,44)
      WRITE(6,5) (PARAM(J), J=12,18  )
      WRITE(6,42)
      WRITE(6,1) NZO,MME ,ND ,NYO,NH ,KPRINT
C     END    COMPUTATION BLOCK NUMBER  35
    C ==========================================================
C     START  COMPUTATION BLOCK NUMBER  36
      JSAVE =JSAVE +1
      IF( JSTOP .EQ. JSAVE ) GO TO 79
C     END    COMPUTATION BLOCK NUMBER  36
      J = JSAVE
      DF2= DFB(J)*DFC
      BT (J) =( BLTH**2)/ DF2
      WRITE(6,5)DFB(1),DFC,DF2,BLTH,BT(1)
      TNEW(J)= TTMMP1* BTM/BT (J)
      CONPRO = CONPR (J)* ( BT (J)/ BLTH)
      CNSTA  = CNST1(J)/BLTH
      CNSTB  = CNST2(J)/DF2
      CNTAU  = CNTA (J)/BT (J)
      DF1 = DFA(J)/DF2
      DF2 = 1./DFC
```

```
      WRITE(6,5)DFB(1),DFC,DF2,BLTH,BT(1),DF1,DFA(1)
      CNA =  .5 * CNSTA
      CNB =   CNSTB
      CNSTX =  SQRT ( CNA**2 +CNB*CNTAU )-CNA
      TLIN = CNTAU  + TNEW(J)
C     CALCULATE DT SLOPE ZCRIT
      DM=DZ*WFRC
      IF( CNTAU  .LE.  .00000001)DT= (DM*(DM+ CNSTA) )/CNSTB
      IF( CNTAU  .LE.  .00000001)DTS =DT
      IF( CNTAU  .LE.  .00000001) ZCRIT = ZOP1
      IF( CNTAU  .LE.  .00000001) SLOPE =1.
      IF( CNTAU  .LE.  .00000001) GO TO 316
      ZCRIT = ZOP1 + CNSTX
      SLOPE =  CNSTX/CNTAU
      DT = DM/SLOPE
      DTS=DT
  316 CONTINUE
      ZONEW(J) = ZOP1
      TTMM= TNEW(J)
      NORUN=NORUN+1
      WRITE(6,27 )
      WRITE(6,5)TLIN,CNTAU,TTMM, SLOPE ,ZCRIT, CNSTX
      TW =TNEW(J)
      TIMEND = TFINN(J) * 1./BT (J)
      WRITE(6,39)
      WRITE(6,5) TIMEND,TNEW(J),BTM,CNTAU,TLIN,BT(J)
      ZW= ZONEW(J)
      SCALE2 = 2.* SQRT(DFB(J)*TFINN(J))
      BTM=BT(J)
      TFINAL=TFINN(J)
      CALL TRIDNL
      PARAM(1)=DFA(J)
      PARAM(2)=DFB(J)
      PARAM(3)= CONPR(J)
      PARAM(4)=CONOUT
      PARAM(5)=CNST1(J)
      PARAM(6)=CNST2(J)
      PARAM(7)=CNTA(J)
    C ==================================================================
    C START COMPUTATION BLOCK NUMBER 37
   78 CONTINUE
      IF(TIMEND .GT. TTMMP1) WRITE(6,43)
   79 CONTINUE
      STOP
      END
```

```
C      START  COMPUTATION BLOCK NUMBER    0
       IMPLICIT DOUBLE PRECISION (A-H,O-Z)
       COMMON/BL20/DT,DY,DZ,THETA
         COMMON/BL21/DF1,DF2
       COMMON/BL25/P(2),Q(2),PA(2),QA(2),C(2),CB(2),R(2),S(2),SM(2)
       COMMON/BL26/P1(2000),P2(2000),Q1(2000),Q2(2000),D1(2000),D2(2000)
       COMMON/BL28/N2M1,NLM1,N2 , NRP1,NL,NR
       COMMON/BL29/C1(2000),C2(2000)
       COMMON /BL30 / CO,CK,TZERO,JEE,ALPHA,CAPA,SEG,CMVY
       COMMON/BL32/CONPRO,CONOUT
C      END    COMPUTATION BLOCK NUMBER    0
 5     FORMAT (7F14.5)
 30    FORMAT( 43H S(1) SU SUM FL FR  ERR                          )
C      CONSERVE CALCULATION
       SUM=                DZ* (C1(NL)+CB(1))*.5 *SM(1) +DY*
      1SM(2)*(C2(NR) +CB(2))*.5+C2(NR)*DY*.5
     C ==================================================================
C      START  COMPUTATION BLOCK NUMBER    1
       P(2)=P2(NRP1)
       Q(2)=Q2(NRP1)
C      END    COMPUTATION BLOCK NUMBER    1
     C ==========================================================
C      START  COMPUTATION BLOCK NUMBER    2
       IF(NL .EQ. 1) GO TO 849
       P(1)=P1(NLM1)
       Q(1)=Q1(NLM1)
C      END    COMPUTATION BLOCK NUMBER    2
     C ==========================================================
C      START  COMPUTATION BLOCK NUMBER    3
 849   R(1)=    2.*DT/(DZ**2)
       R(2)=2.*DF2*DT/(DY**2)
       DO 850 L=1,2
       IF(NL .EQ. 1 .AND. L .EQ. 1 )GO TO 850
C      COMPUTE P AND Q FOR HALF CELL
       DM=1. +S(L)
       IF(L .EQ. 2 ) GO TO 847
       DM1=DF1+S(L)/R(L)
       DM2=DF1/DM
       DM3=DM2*S(L)
       IF(L .EQ. 1)DM4=(S(L)/R(L))*C1(NL)
       IF(L .EQ. 1) GO TO 848
  847  DM1=1.+S(L)/R(L)
       DM2=1./DM
       DM3=S(L)/DM
       IF(L .EQ. 2)DM4=(S(L)/R(L))*C2(NR)
  848  PA(L)=DM2/(DM1-DM3*P(L))
       QA(L)=(DM4+Q(L)*DM3)/(DM1-DM3*P(L))
  850  CONTINUE
       IF(NL .EQ. 1)PA(1)=P1(1)
       IF(NL .EQ. 1)QA(1)=Q1(1)
C      END    COMPUTATION BLOCK NUMBER    3
     C ==========================================================
C      START  COMPUTATION BLOCK NUMBER    4
C      CALCULATION  OF BOUNDARY DENSITIES
       IF(NL .EQ. 1)SAVE= S(1)
       IF(NL .EQ. 1)S(1)=1.
```

```
      DM=1.+S(1)
      D11=S(1)**2
      D12=S(1)*S(2)
      RT=(DZ*DF2)/(DY)
      DM1=D11/R(1)+       (SEG*RT*D12)/R(2) +DF1/DM
      IF(NL .EQ. 1)DM1=SAVE/R(1)+(SEG*RT*S(2))/R(2)
      DM2=-D11/R(1)+(1.-S(1))*DF1
      IF(NL .EQ. 1)DM2=-(SAVE/R(1)+DZ*CONPRO)
      DM3=-RT*D12/R(2)-RT*S(1)*(1.+1./R(2))
      DM4=D11*DF1/DM
      IF(NL .EQ. 1)DM4=0.
      DM5=RT*S(1)
      DM6=S(1)*SM(1)*(C1(NL)+CB(1))/R(1)+RT*SM(2)*S(1)*CB(2)/R(2)
     1 +S(1)*(RT*SM(2)/R(2)+RT/R(2) )*C2(NR)
      IF(NL .EQ. 1)DM6=DM6+DZ*CONPRO*CONOUT
      IF(NL .EQ. 1)S(1)=SAVE
      DM8=DM2+DM4*P(1)
      DM9=DM3+DM5*P(2)
      DM10=DM6+Q(1)*DM4+DM5*Q(2)
      DM11=DM10+QA(1)*DM8  +QA(2)*DM9
      DM12=DM1-DM8*PA(1)-SEG*DM9*PA(2)
      CB(1)=DM11/DM12
      CB(2)=SEG*CB(1)
C     END    COMPUTATION BLOCK NUMBER   4
    C ==========================================================================
C     START  COMPUTATION BLOCK NUMBER   5
      DO 855 L=1,2
      C(L)= PA(L)*CB(L)+QA(L)
 855  CONTINUE
      C1(NL)=C(1)
      C2(NR)=C(2)
C     END    COMPUTATION BLOCK NUMBER   5
C     CONSERVE CALCULATION
      SU=            DZ*S(1)*(C1(NL)+CB(1))*.5 + .5*  DY*
     1S(2)*(C2(NR)+CB(2))+C2(NR)*DY*.5
      IF(NL .GT.1)  FL=-R(1)*(-(S(1)**2)* C1(NLM1)/(S(1)+1.)+C1(NL)
     1*(S(1)-1.)+CB(1)/(S(1)+1. ) )*DZ *.5*DF1
      IF(NL .EQ.1)FL=-DT *CONPRO*(C1(1)-CONOUT)
      IF(NL .EQ. 1)FR=R(2)* (C2(NRP1)-C2(NR))*DY*.5
      IF(NL .EQ. 1)ERR=SU-SUM-(FL+FR)
      IF(NL .EQ. 1)GO TO 860
      FR=R(2)*S(1)*(C2(NRP1)-C2(NR)) *DY *.5
      ERR=S(1)*(SU-SUM)-(FL+FR)
 860  CONTINUE
      RETURN
      END
```

1.3. Subroutine TRIDNL

```
@FOR,IS TRIDNL
      SUBROUTINE TRIDNL
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      COMMON/BL20/DT,DY,DZ,THETA
      COMMON/BL21/DF1,DF2
      COMMON/BL23/ B2M,B2R,B2L,A2L,A2M,A2R,A1L,A1M,A1R,B1L,B1M,B1R
      COMMON/BL25/P(2),Q(2),PA(2),QA(2),C(2),CB(2),R(2),S(2),SM(2)
      COMMON/BL26/P1(2000),P2(2000),Q1(2000),Q2(2000),D1(2000),D2(2000)
      COMMON/BL28/N2M1,NLM1,N2 , NRP1,NL,NR
      COMMON/BL29/C1(2000),C2(2000)
      COMMON/BL32/CONPRO,CONOUT
 5    FORMAT (7F14.5)
C     CALC. P,S AND DUMMIES
C     START  COMPUTATION BLOCK NUMBER   1
      CALL INTER
    C ==========================================================================
C     END    COMPUTATION BLOCK NUMBER   1
      IF(NL .EQ. 1)DM1=S(1)*DZ
      IF(NL .EQ. 1)DM2=DM1*DM1/(2.*DT)
      IF(NL .EQ. 1)DM=1./( DF1+DM2+DM1*CONPRO)
      IF(NL .EQ. 1 )P1(1)=DM*DF1
C     START  COMPUTATION BLOCK NUMBER   2
      IF(NL .GT. 1 )DM=1./(DZ*DZ/(2.*DT) +DZ* CONPRO+DF1 )
      IF(NL .GT. 1 )P1(1)=DF1*DM
    C ==========================================================================
C     END    COMPUTATION BLOCK NUMBER   2
C     START  COMPUTATION BLOCK NUMBER   3
      IF(NL .LE. 2 )GO TO 106
      DO 105 J= 2,NLM1
      JM1=J-1
      D1(J) =A2M+P1(JM1)*A2L
      P1(J)=-A2R/D1(J)
 105  CONTINUE
 106  CONTINUE
C     END    COMPUTATION BLOCK NUMBER   3
    C ==========================================================================
C     START  COMPUTATION BLOCK NUMBER   4
      DO 101 J= 1,N2M1
      K=N2-J
      KP1=K+1
      D2(K)=B2M+P2(KP1)
      P2(K)=-1./D2(K)
 101  CONTINUE
C     END    COMPUTATION BLOCK NUMBER   4
      RETURN
      END
```

11.4. Subroutine INTER

```
      SUBROUTINE INTER
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      COMMON/BL20/DT,DY,DZ,THETA
      COMMON/BL21/DF1,DF2
      COMMON/BL23/ B2M,B2R,B2L,A2L,A2M,A2R,A1L,A1M,A1R,B1L,B1M,B1R
    5 FORMAT (7F14.5)
      RA1=DT/(DZ**2)
      RA2=DT/(DY**2)
      A2L= DF1
      A2R=A2L
      A2M=-(2.*DF1+1./RA1)
      A1L=1.
      A1R= A1L
      A1M=-1./RA1
      B2L=1.
      B2R=B2L
      B2M=-(2.+1./(DF2*RA2) )
      B1L=1.
      B1R =B1L
      B1M = -1./(DF2*RA2)
      RETURN
      END
```

12. GLOSSARY

A2L, A2M, A2R, A1M - See eqs (5.6) and (5.7) and table 5.

AF - An array that stores the inhomogeneous terms in the tridiagonal equations for the z-region (table 5).

ALPHA - Swelling ratio α [eq (2.5)].

B2L, B2M, B2R, B1M - See eqs (5.10) and table 7.

BDRY - The point representing infinity in the y-region [eq (2.11′)].

BLTH - The unit being used to dimensionalize lengths (sec. 4.3).

BTM - The unit being used to dimensionalize time (sec. 4.3).

BF - An array that stores the inhomogeneous terms in tridiagonal equations in the y-region [eq (5.10) and table 7].

C_B, C_{max}, C_{out}, C_p - Material parameters [eqs (2.11), (2.12), and (2.13)].

C1( ), C2( ) - Arrays representing concentrations at the grid points in the z- and y-regions, respectively. At the beginning of an iteration of loop 78, C1( ) represents C_1(, TTMM), and at the end of an iteration, C1( ) represents C_1(, TTMMP1). Likewise, C2( ).

CB(1), CB(2) - These represent $C_i(z_o(\text{TTMMP1}), \text{TTMMP1})$, i = 1, 2, i.e., the concentrations in the oxide and silicon at the moving boundary.

CBULK - The concentration in the bulk of the silicon and at y = 0 [eqs (2.13) and (2.19), $C_B \equiv$ CBULK].

CENTER - A subroutine for calculating C1(NL), C2(NR), CB(1), CB(2) (secs. 7.2 and 8.2).

CMAX - The value of C2(y,0) for $y \leq$ NH in eq (2.13).

CNA - By definition, A/2.

CNB - Another name for CNSTB.

CNSTA, CNSTB - The constants A and B, respectively, in eqs (4.5) to (4.9) (table 2).

CNSTX - Defined in eq (4.9) and FORTRAN name for ZCRIT.

CNTAU - The constant τ in eqs (4.8) and (4.9).

CONOUT - A constant in eq (2.18) representing the boron concentration in the oxygen at the oxygen-oxide interface (sec. 5.7); CONOUT = C_{out}.

CONPRO - A proportionality constant in eq (2.18) representing the evaporation rate of boron from the oxide into the oxygen ambient (sec. 5.7); CONPRO = C_p.

D1 - An intermediate array used to compute the arrays Q1, P1 [eq (6.9′) and sec. 7.3, computational block #3].

D2 - An intermediate array used to compute arrays P2, Q2 [sec. 7.3, block #4, and eq (6.16′)].

DF1 - The diffusion coefficient D_1 in the oxide or z-region.

DF2 - The diffusion coefficient D_2 in the silicon or y-region.

DFC - A normalizing factor set equal to one in VERSION 1.

DM1 through DM4 - For these quantities used in computational block #3 of subroutine CENTER, see eqs (6.10′) and (6.17′).

DM1 through DM10 - For these quantities used in computational block #4 of subroutine CENTER, see eqs (6.20), (6.21), and (6.22).

DT - The time mesh width Δt (sec. 4.4).

DTS - A value of DT that is stored before the moving boundary z_o crosses a grid point (sec. 7.1, computational block #20).

DY - The mesh width Δy in the y-region (sec. 4.1).

DZ - The mesh width Δz in the z-region (sec. 4.1).

EY, EZ - Scaled value of DY, DZ used in plot output.

FNL - The real value of NL - 1.

FNR - The real value of NR - 1.

FNZO - The quantity NZO floated, i.e., made real.

JSAVE - A parameter used to shift control in the program (computational block #36, sec. 7.1). JSAVE = 0 in VERSION 1 to begin with.

JSTOP - In VERSION 1, JSTOP equals 1.

KPOW - A signal to increase the value of DT when KPOW = 100 and ZOACEL $\leq$ 0.2 (computational block #16, sec. 7.1).

KPRINT - The concentrations and certain other integer parameters are printed out when MM = 0 modulo (KPRINT).

LOOP 78 - Each iteration of this loop calculates the concentrations at one time, Δt. (Later see computational block #14, sec. 7.1).

m - Another symbol for segregation coefficient SEG [eq (2.2)].

MM - This is the "DO" index in loop 78.

MME - This integer specifies the number of iterations in loop 78 (computational block #14, sec. 7.1).

N1 - The value of NL at time t = 0 in VERSION 1 (computational block #9, sec. 7.1).

N2 - The number of fixed grid points in the y-region, N2 = NYO + 1 [computational block #9, sec. 7.1, and eq (4.2)].

ND - Tells how many y-grid points to the right of NR will be plotted in output graph (block #34, sec. 7.1).

NE - A loop limit index in block #26, sec. 7.1.

NH - The point on the y-axis where the initial concentrations in the silicon changes from CMAX to CBULK [eq (2.13) and sec. 7.1, computational block #10].

NL - See end of section 4.1.

NMOD - The z-grid points are plotted modulo NMOD (block #34, sec. 7.1).

NMOD1 - The y-grid points from NR to the right are plotted modulo NMOD1 (block #34, sec. 7.1).

NORUN - In VERSION 1. NORUN = 1.

NR - See end of section 4.1.

NYO - The number of mesh widths of length DY that cover (0, BDRY) (sec. 4.1).

NZO - A parameter used in the definition of DZ [eq (4.3)].

P1, P2 - These are the arrays whose functions are described in eqs (3.5) and (3.7). The quantities with the exception of P1(1) are computed in subroutine TRIDNL. See eqs (6.5′), (6.6′), and (6.7′) for P1(1) and (6.9′) for P1(J), J > 1. See computational blocks #23 and #24 for computation of P1(1). See eq (6.16′) for the derivation of (6.16).

P(1), P(2) - Alternate names for P1(NLM1), P2(NRP1) [see eq (6.18′)].

PA - Coefficient linking C1(NL) with CB(1) and C2(NR) with CB(2) calculated in subroutine CENTER, block #3 [eqs (6.19′) and (6.12′)]. Function of this array is explained in section 3.1, steps #2 and #3.

PARAM - An array containing various material input constants and computer program constants of possible interest that are program output (computational blocks #2 and #35, sec. 7.1.

PLOT - An NBS "in-house" plotting program available from NBS Applied Mathematics Division. Documented in Appendix 2.

Q1, Q2 - Everything to do with these arrays is explained in the same places as P1 and P2.

Q(1), Q(2) - Alternate names for Q1(NLM1), Q2(NRP1) [eq (6.18′)].

R - Another name for RT.

R(1), R(2) - These are quantities that are defined in eqs (5.35) and calculated in computational block #3 of subroutine CENTER. [See also eqs (5.25) and (5.16, 5.16′)].

RA1, RA2 - Combinations of mesh widths used in subroutine INTER (tables 5 and 7).

RT - This combination of mesh quantities is defined in eq (5.35) and appears also in eqs (5.38) and (5.34) (computational block #4 of subroutine CENTER).

S(1), S(2) - See eqs (5.35), (5.29), (5.22), (5.15) and figures 5 and 6. These quantities are computed in computational block #21, section 7.1.

SAVE - See computational block #4 of subroutine CENTER; see remarks at the end of section 5.9 before the SUMMARY.

SCALE2 - Essentially the same as UN2, a scale factor used in plotting. (See sec. 4.4).

SEG - The FORTRAN symbol for m, the segregation coefficient [eq (2.2)].

SES - A signal which when negative implies the moving boundary y_o reached a new y-grid point during the previous iteration of loop 78 (computational block #19, sec. 7.1).

SLOPE - The speed of the moving boundary in dry oxidations [eq (4.8)].

SM(1), SM(2) - See the same equations and figures cited in description of S(1), S(2). These quantities are computed in computational block #12 and #31, section 7.1.

TFINAL - This is the time in hours that is inputted into the program.

TIMEND - When TTMMP1 is greater than TIMEND (which is TFINAL in BTM units), the program stops and control is shifted to printing output (computational blocks #32 and #4, sec. 7.1).

TJ - A signal which, when its value is greater than or equal to 1, indicates that the moving boundary z_o has reached a new z-grid point during the previous iteration of loop 78.

TLIN - The quantity T in dry oxidations (table 4.1); also called CNTAU.

TRIDNL - The subroutine where arrays P1, D1 are calculated (secs. 7.3 and 8.3).

TTMM - The time at the beginning of the MMth iteration of loop 78 in units of BTM (computational block #18, sec. 7.1).

TTMMP1 - The time at the end of the MMth iteration of loop 78 (in BTM units).

TW - This quantity is always zero in VERSION 1.

UN1, UN2 - Scaling factors used in the plot output (computational block #34, sec. 7.1, and sec. 4.4).

WFRC - The fraction of DZ that the moving boundary z_o moves in the first iteration of loop 78 [eq (4.4) and computational block #7, sec. 7.1].

X1S, X2S - Quantities used in setting scales for plots of concentrations (computational block #34, sec. 7.1, and sec. 4.4).

X2SCT - A cutoff for points with large abscissas (sec. 4.4).

XRL - An array for abscissas used in plotting (sec. 3.4).

$y_o(t)$ - The position at time t in the y-coordinate frame, calculated by using eq (2.8).

Y1S, Y2S - Quantities used in setting scales for plots of concentrations (computational block #34, sec. 7.1, and sec. 4.4).

YO1 - Always zero in VERSION 1.

YOP1 - Position in y-region of moving boundary at time, TTMMP1 (computational block #18, sec. 7.1).

YRL - An array for ordinates used in plotting (sec. 3.4).

$z_o(t)$ - The position of the moving boundary in the z-coordinate frame at time t (sec. 4.2).

ZCRIT - Same as CNSTX.

ZO1 - Always zero in VERSION 1.

ZOACEL - When the moving boundary is moving slowly, this parameter is small and the value of DT is increased in order to speed up distance the boundary moves in a given iteration of loop 78 (computational block #16, sec. 7.1).

ZOCR - ZOP1 in units of micrometers (computational block #34, sec. 7.1).

ZOP1 - Position in z-region of moving boundary at time, TTMMP1 (computational block #18, sec. 7.1).

ZOP2 - See computational block #20, sec. 7.1.

ZOPS - See computational block #16, sec. 7.1.

ZW - Always zero in VERSION 1.

13. REFERENCES

13.1. Bibliography

1. Deal, B. E., and Grove, A. G., General Relationship for Thermal Oxidation of Silicon, *J. Appl. Phys.* 36, 3770-3778 (1965).

2. Fox, L., What Are the Best Numerical Methods in "Moving Boundaries in Heat Flow and Diffusion," J. R. Ockendon, Ed, pp. 210-241 (Claredon Press, Oxford, 1975).

3. Gelfand, I., and Lokutsievski, O. V., in Appendix 2 of Theory of Finite Difference Schemes (authored by Godunov, S. R., and Ryabenki, V. S.) (North Holland, 1964).

4. Grove, A. S., Leistiko, O., Jr., and Sah, C. T., Redistribution of Acceptor and Donor Impurities During Thermal Oxidation of Silicon, *J. Appl. Phys.* 35, 2695-2701 (1961).

5. Kraft, R., Finite Difference Techniques for Diffusion and Redistribution Problems with Segregation Type Boundary Conditions, *Proc. AICA Int. Sym.*, Bethlehem, Pennsylvania, June 17-19, 1975, pp. 328-333.

6. Kurtz, A., and Lee, R., Diffusion of Boron in Silicon, *J. Appl. Phys.* 31, 303-305 (1960).

7. Horiuchi, S., and Yamaguchi, S., Diffusion of Boron in Silicon Through Oxide Layer, *Jap. J. Appl. Phys.* 1, 314-323 (1962).

8. Prince, J. L., and Schwettman, F. N., Diffusion of Boron from Implanted Sources Under Oxidizing Conditions, *J. Electrochem. Soc.* 121, 705-710 (1974).

9. Ralston, A. V., and Wilf, H., Mathematical Methods for Digital Computers (John Wiley & Sons, New York, 1960).

10. Semiconductor Measurement Technology, W. M. Bullis, Ed, NBS Spec. Publ. 400-1, pp. 9-12 (March 1974).

11. Semiconductor Measurement Technology, W. M. Bullis, Ed, NBS Spec. Publ. 400-4, p. 8 (November 1974).

12. Semiconductor Measurement Technology, W. M. Bullis, Ed, NBS Spec. Publ. 400-17, pp. 11-13 (November 1975).

13. Semiconductor Measurement Technology, W. M. Bullis, Ed, NBS Spec. Publ. 400-19, p. 25 (April 1976).

13.2. Supplemental Bibliography

1. Allen, W., and Atkinson, P., Comparison of Models for Redistribution of Dopants in Silicon During Thermal Oxidation, *Solid-State Electronics* 16, 1283-1287 (1973).

2. Atalla, M., and Tamenbaum, E., Impurity Redistribution and Junction Formation in Silicon by Thermal Oxidation, *Bell System Tech. J.* 39, 933-946 (1960).

3. Fuller, C. S., and Ditzenberger, J. A., Diffusion of Donor and Acceptor Elements in Silicon, *J. Appl. Phys.* 27, 544-547 (1956).

4. Ghezzo, M., and Brown, D., Diffusivity Şummary of B, Ga, P, As and S_b in SiO_2, *J. Electrochem. Soc.* 120, 146-148 (1973)

5. Ghoshtoyore, R. N., Intrinsic Diffusion of Boron and Phosphorus in Silicon Free from Surface Effects, *Phys. Rev. B.* 3, 389-403 (1971).

6. Kato, T., and Nishi, Y., Redistribution of Diffused Boron in Silicon by Thermal Oxidation, *Jap. J. Appl. Phys.* 3, 377-383 (1964).

7. Kurtz, A., and Yee, R., Diffusion of Boron in Silicon, *J. Appl. Phys.* 31, 303-305 (1960).

8. Morgalit, S., Neugroschel, A., and Bar Lev, A., Redistribution of Boron in Silicon After Two Oxidation Steps in MOST Fabrication, *IEEE Trans. Electron Devices* ED-19, 861-868 (1972).

9. Prince, J. L., and Schwettmann, F. N., Diffusion of Boron from Implanted Sources Under Oxidizing Conditions, *J. Electrochem. Soc.* 121, 705-710 (1974).

10. Rodoni, M., Buneman, O., and Dutton, R., Boron Redistribution After Oxidation, Abstract #333 (Electrochemical Society, Fall 1976).

11. Shimakura, K., Suzuki, T., and Yadoviva, Y., Boron and Phosphorus Diffusion Through an SiO_2 Layer from a Doped Polycrystalline Si Source Under Various Drive-in Ambients, *Solid-State Electronics* 18, 991-997 (1975).

12. Williams, E. L., Boron Diffusion in Silicon, *J. Electrochem. Soc.* 108, 795-798 (1961).

13. Wilson, P. R., The Diffusion of Boron in the Si-SiO_2 System, *Solid-State Electronics* 15, 961-970 (1972).

Appendix 1

FORMULAS USED IN DERIVATION OF DISCRETE ALGEBRAIC EQUATIONS

We collect here for convenience some standard finite difference approximations used in the discretizations effected in chapter five. The derivations of these formulas can be found in a text book on numerical analysis.

1. TAYLOR'S SERIES

$$f(x+\Delta x) = f(x) + f'(x)\Delta x + \frac{f''(x)}{2}\Delta x^2 + \ldots + \frac{f^n(x)}{n!}\Delta x^n + \ldots \qquad (A1)$$

2. QUADRATURE APPROXIMATIONS

a. Midpoint rule

$$\int_a^b f(x)dx = f(\frac{a+b}{2})\ (b-a) + 0[(b-a)^3] \qquad (A2)$$

b. Right end point

$$\int_a^b f(x)dx = f(b)\Delta x + 0[(b-a)^2] \qquad (A3)$$

c. Left end point

$$\int_a^b f(x)dx = f(a)\ (b-a) + 0[(b-a)^2] \qquad (A4)$$

d. Trapezoid rule

$$\int_a^b f(x)dx = \frac{f(a)+f(b)}{2}\ (b-a) + 0[(b-a)^3] \qquad (A5)$$

3. DERIVATIVE APPROXIMATIONS

a. Forward differences

$$\frac{df(x)}{dx} = \frac{f(x+\Delta x) - f(x)}{\Delta x} + 0[\Delta x] \qquad (A6)$$

b. Backward differences

$$\frac{df(x)}{dx} = \frac{f(x) - f(x-\Delta x)}{\Delta x} + 0[\Delta x] \tag{A7}$$

c. Central differences

$$\frac{df(x)}{dx} = \frac{f(x+\Delta x)-f(x-\Delta x)}{2\Delta x} + 0[\Delta x^2] \tag{A8}$$

4. THREE-POINT APPROXIMATIONS

Let x_2, x, x_1 be three points such that $x_1 > x > x_2$; then a three-point approximation to the derivative of $t(x)$ at point x is

$$\frac{df}{dx} = -\frac{\Delta x_1 \, f(x-\Delta x_2)}{\Delta x_2 \, (\Delta x_1+\Delta x_2)} + \frac{(\Delta x_1-\Delta x_2) f(x)}{\Delta x_1 \, \Delta x_2} + \frac{\Delta x_2 \, f(x-\Delta x_1)}{\Delta x_1 \, (\Delta x_1+\Delta x_2)} \tag{A9}$$

where $\Delta x_1 \equiv x_1 - x$ and $\Delta x_2 \equiv x - x_2$.

Plotting Subroutines

B. L. Joiner and S. T. Peavy

Five FORTRAN subroutines for producing plots similar to the one shown in Figure 1 are now available on the NBS UNIVAC 1108 FASTRAND file named PLOTS.

Questions related to the operation of these subroutines may be directed to:

Brian L. Joiner or Sally T. Peavy
A337 Administration Bldg.
National Bureau of Standards
Washington, D. C. 20234
921-2315

Task numbers for the UNIVAC 1108 may be obtained from the

Computer Services Division
A238 Administration Bldg.
National Bureau of Standards
Washington, D. C. 20234
921-3364

General Remarks

Except as specified below, the subroutines automatically figure out limits for the plots based on the smallest and largest data points. A new page is not called by any of these subroutines. This allows the user to label the top of the page before calling for the plot. These subroutines do not change any of the values of the arguments. Plotting is done by repetitively searching the arrays rather than by sorting them.

The resolution of the plots is 51 characters high by 101 characters wide and each plot consumes 54 lines counting borders and scale labeling. The length and width of the actual plotting area are 8½ inches and 10 inches respectively, and the overall dimensions including borders and scale labeling are 9 inches and 11¼ inches respectively.

These five subroutines may be called in from FASTRAND by inserting the following pair of cards after the RUN card:

```
@ XQT  CUR
  INF  PLOTS
```

where @ means a 7 and 8 punched in card column one.

Subroutines

I. PLOT (N, X, Y)

Plots the data in one column versus that in another column.

N : The number of points to be plotted.

X : A vector (one dimensional array) containing the values of the abscissa.

Y : A vector containing the values of the ordinate.

Example of main program.

```
      DIMENSION  X(10), Y(10)
      DO  2  I = 1,5
      X(I) = I
2     Y(I) = SIN(X(I))
      N = 5
      CALL PLOT (N, X, Y)
      STOP
```

II. PLOTS (NARGS, X, Y, NRMX, NROW)

Plots the data in up to 5 pairs of columns. The symbols used are . * + , - respectively for the 5 curves.

NARGS : The number of curves to be plotted.

X : A matrix (two dimensional array) having up to 5 columns and containing the values of the abscissa.

Y : A matrix having the same dimensions as X and containing the values of the ordinate. The first column of Y is plotted versus the first column of X, etc.

NRMX : A vector (one dimensional array) containing the number of points to be plotted in each of the (5 or less) column pairs.

NROW : The number of rows specified for X and Y in their dimension statement(s) in the main program.

(An example is given on the following page.)

Example of main program.

```
      DIMENSION  X(85,4), Y(85,4), NRMX(4)
      DO  5  I = 1,20
      X(I,1) = I-10
 5    Y(I,1) = ABS(X(I,1))
      DO  10  I = 1,37
      X(I,2) = I-20
10    Y(I,2) = 20. - X(I,2)**2
      NARGS = 2
      NRMX (1) = 20
      NRMX (2) = 37
      NROW = 85
      CALL PLOTS (NARGS, X, Y, NRMX, NROW)
      STOP
```

III. PLOTL (N, X, Y, IT)

Plots the data in one column versus that in another column using the symbols specified in the IT column.

N : The number of points to be plotted.

X : A vector (one dimensional array) containing the values of the abscissa.

Y : A vector containing the values of the ordinate.

IT : A vector containing numbers between 1 and 26 which dictate the letter of the alphabet to be used as a plotting symbol.

number:	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
letter:	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O

16	17	18	19	20	21	22	23	24	25	26
P	Q	R	S	T	U	V	W	X	Y	Z

Example of main program.

```
      DIMENSION  X(100), Y(100), IT(100)
      DO  20  I = 1,5
      X(I) = I
      Y(I) = SIN(X(I))
20    IT(I) = 19
      DO  30  I = 6,10
      X(I) = I-5
      Y(I) = COS(X(I))
30    IT(I) = 3
      CALL PLOTL (10, X, Y, IT)
      STOP
```

IV. PLOTLA (N, X, Y, IT, YMINS, YMAXS)

This subroutine is the same as PLOTL except that the limits for the Y axis (ordinate) are specified in the call statement. If any points fall outside the specified limits, the limits are stretched so as to include all points.

Example of main program is same as for PLOTL except for call.

```
CALL PLOTLA (10, X, Y, IT, -1., 1.)
```

V. PLOTLF (N, X, Y, IT, XMIN, XMAX, YMIN, YMAX)

This subroutine is also similar to PLOTL except that here the limits for both X and Y are specified in the call statement. Any points falling outside the specified limits are omitted but a talley is kept and the number of offending points is printed below the plot.

Example of main program is same as for PLOTL except for call.

```
CALL PLOTLF (10, X, Y, IT, 0., 5., -1., 1.)
```

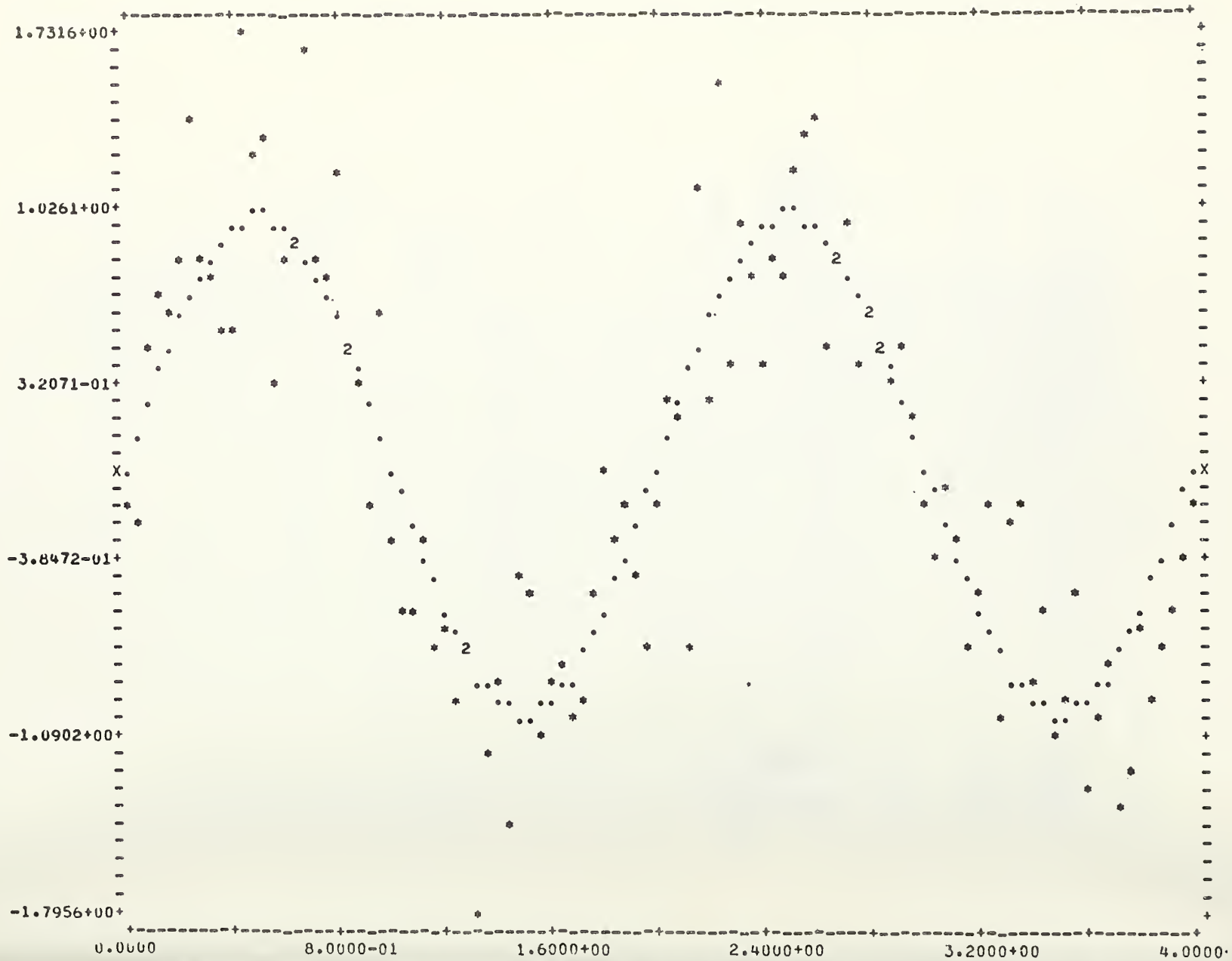

1.7316+00
1.0261+00
3.2071-01
-3.8472-01
-1.0902+00
-1.7956+00
0.0000
8.0000-01
1.6000+00
2.4000+00
3.2000+00
4.0000
X
2

Figure 1.

NBS-114A (REV. 9-78)

U.S. DEPT. OF COMM. BIBLIOGRAPHIC DATA SHEET	1. PUBLICATION OR REPORT NO. Spec. Publ. 400-57	2. Gov't. Accession No.	3. Recipient's Accession No.

4. TITLE AND SUBTITLE

Semiconductor Measurement Technology: DISTRIB I, An Impurity Redistribution Computer Program

5. Publication Date

February 1979

6. Performing Organization Code

7. AUTHOR(S)

David Gilsinn and Richard Kraft

8. Performing Organ. Report No.

9. PERFORMING ORGANIZATION NAME AND ADDRESS

NATIONAL BUREAU OF STANDARDS
DEPARTMENT OF COMMERCE
WASHINGTON, DC 20234

10. Project/Task/Work Unit No.

11. Contract/Grant No.

ARPA Order No. 2397

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

ARPA, 1400 Wilson Boulevard, Arlington, VA 22209

13. Type of Report & Period Covered

Final
July 1973 - Dec. 1975

14. Sponsoring Agency Code

15. SUPPLEMENTARY NOTES

Library of Congress Catalog Card Number: 79-600002

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

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

This report provides documentation of a computer program which calculates the redistribution of impurities in silicon during a single oxidation step. The documentation provides: (1) a physical and mathematical description of the redistribution process, (2) a detailed description of the discretization of the appropriate partial differential equations, and (3) a complete description of the FORTRAN program for computing the solution.

17. KEY WORDS *(six to twelve entries; alphabetical order; capitalize only the first letter of the first key word unless a proper name; separated by semicolons)*

Diffusion; electronic technology; impurity distribution; material transport; segregation; semiconductor technology.

18. AVAILABILITY

☒ Unlimited

☐ For Official Distribution. Do Not Release to NTIS

☐ Order From Sup. of Doc., U.S. Government Printing Office, Washington, DC 20402, SD Stock No. SN003-003-02030-3

☒ Order From National Technical Information Service (NTIS), Springfield, VA. 22161

19. SECURITY CLASS (THIS REPORT)

UNCLASSIFIED

21. NO. OF PRINTED PAGES

130

20. SECURITY CLASS (THIS PAGE)

UNCLASSIFIED

22. Price

$3.00

USCOMM-DC

*U.S. GOVERNMENT PRINTING OFFICE : 1979 0-281-067/6

NBS TECHNICAL PUBLICATIONS

PERIODICALS

JOURNAL OF RESEARCH—The Journal of Research of the National Bureau of Standards reports NBS research and development in those disciplines of the physical and engineering sciences in which the Bureau is active. These include physics, chemistry, engineering, mathematics, and computer sciences. Papers cover a broad range of subjects, with major emphasis on measurement methodology, and the basic technology underlying standardization. Also included from time to time are survey articles on topics closely related to the Bureau's technical and scientific programs. As a special service to subscribers each issue contains complete citations to all recent NBS publications in NBS and non-NBS media. Issued six times a year. Annual subscription: domestic $17.00; foreign $21.25. Single copy, $3.00 domestic; $3.75 foreign.

Note: The Journal was formerly published in two sections: Section A "Physics and Chemistry" and Section B "Mathematical Sciences."

DIMENSIONS/NBS

This monthly magazine is published to inform scientists, engineers, businessmen industry, teachers, students, and consumers of the latest advances in science and technology, with primary emphasis on the work at NBS. The magazine highlights and reviews such issues as energy research, fire protection, building technology, metric conversion, pollution abatement, health and safety, and consumer product performance. In addition, it reports the results of Bureau programs in measurement standards and techniques, properties of matter and materials, engineering standards and services, instrumentation, and automatic data processing.

Annual subscription: Domestic, $11.00; Foreign $13.75

NONPERIODICALS

Monographs—Major contributions to the technical literature on various subjects related to the Bureau's scientific and technical activities.

Handbooks—Recommended codes of engineering and industrial practice (including safety codes) developed in cooperation with interested industries, professional organizations, and regulatory bodies.

Special Publications—Include proceedings of conferences sponsored by NBS, NBS annual reports, and other special publications appropriate to this grouping such as wall charts, pocket cards, and bibliographies.

Applied Mathematics Series—Mathematical tables, manuals, and studies of special interest to physicists, engineers, chemists, biologists, mathematicians, computer programmers, and others engaged in scientific and technical work.

National Standard Reference Data Series—Provides quantitative data on the physical and chemical properties of materials, compiled from the world's literature and critically evaluated. Developed under a world-wide program coordinated by NBS. Program under authority of National Standard Data Act (Public Law 90-396).

NOTE: At present the principal publication outlet for these data is the Journal of Physical and Chemical Reference Data (JPCRD) published quarterly for NBS by the American Chemical Society (ACS) and the American Institute of Physics (AIP). Subscriptions, reprints, and supplements available from ACS, 1155 Sixteenth St. N.W., Wash., D.C. 20056.

Building Science Series—Disseminates technical information developed at the Bureau on building materials, components, systems, and whole structures. The series presents research results, test methods, and performance criteria related to the structural and environmental functions and the durability and safety characteristics of building elements and systems.

Technical Notes—Studies or reports which are complete in themselves but restrictive in their treatment of a subject. Analogous to monographs but not so comprehensive in scope or definitive in treatment of the subject area. Often serve as a vehicle for final reports of work performed at NBS under the sponsorship of other government agencies.

Voluntary Product Standards—Developed under procedures published by the Department of Commerce in Part 10, Title 15, of the Code of Federal Regulations. The purpose of the standards is to establish nationally recognized requirements for products, and to provide all concerned interests with a basis for common understanding of the characteristics of the products. NBS administers this program as a supplement to the activities of the private sector standardizing organizations.

Consumer Information Series—Practical information, based on NBS research and experience, covering areas of interest to the consumer. Easily understandable language and illustrations provide useful background knowledge for shopping in today's technological marketplace.

Order **above** *NBS publications from: Superintendent of Documents, Government Printing Office, Washington, D.C. 20402.*

Order **following** *NBS publications—NBSIR's and FIPS from the National Technical Information Services, Springfield, Va. 22161.*

Federal Information Processing Standards Publications (FIPS PUB)—Publications in this series collectively constitute the Federal Information Processing Standards Register. Register serves as the official source of information in the Federal Government regarding standards issued by NBS pursuant to the Federal Property and Administrative Services Act of 1949 as amended, Public Law 89-306 (79 Stat. 1127), and as implemented by Executive Order 11717 (38 FR 12315, dated May 11, 1973) and Part 6 of Title 15 CFR (Code of Federal Regulations).

NBS Interagency Reports (NBSIR)—A special series of interim or final reports on work performed by NBS for outside sponsors (both government and non-government). In general, initial distribution is handled by the sponsor; public distribution is by the National Technical Information Services (Springfield, Va. 22161) in paper copy or microfiche form.

BIBLIOGRAPHIC SUBSCRIPTION SERVICES

The following current-awareness and literature-survey bibliographies are issued periodically by the Bureau:

Cryogenic Data Center Current Awareness Service. A literature survey issued biweekly. Annual subscription: Domestic, $25.00; Foreign, $30.00.

Liquified Natural Gas. A literature survey issued quarterly. Annual subscription: $20.00.

Superconducting Devices and Materials. A literature survey issued quarterly. Annual subscription: $30.00. Send subscription orders and remittances for the preceding bibliographic services to National Bureau of Standards, Cryogenic Data Center (275.02) Boulder, Colorado 80302.

U.S. DEPARTMENT OF COMMERCE
National Bureau of Standards
Washington, D.C. 20234

OFFICIAL BUSINESS

Penalty for Private Use, $300

POSTAGE AND FEES PAID
U.S. DEPARTMENT OF COMMERCE
COM-215

SPECIAL FOURTH-CLASS RATE
BOOK