The CURRY algorithm

S. Polak, W. Schilders, M. Driessen

Nederlandse Philips Bedrijven B.V.
Corporate ISA/CAD Centre, Mathematical Software Group
5600 MD Eindhoven, The Netherlands

ABSTRACT

This paper describes the algorithmic choices in the program CURRY. CURRY is a program package for 2D on state time dependent semiconductor device modelling.

1. INTRODUCTION

In the following the algorithmic choices made for the program package CURRY are described. This package can be used for the simulation of 2D transient semiconductor device modelling problems.
It is a package of similar structure to SEMMY (Polak et al., [8]) and MAGGY (Polak et al., [9]).
Here we only give a compact presentation of the total algorithm; an extensive discussion can be found in Polak et al., [12]. In order to design an algorithm, the following range of topics must be considered:

- choice of unknowns
- spatial discretisation
- time integration
- nonlinear solving
- linear solving.

As an example we show a MOSFET, for which we calculate the transient behaviour.

Throughout the paper the equations and parameters involved in device simulation are supposed to be known.
The equations have the basic form:

$$\operatorname{div} \varepsilon E = \rho$$
$$q\, \partial p/\partial t = -\operatorname{div} J_p - qR$$
$$q\, \partial n/\partial t = \operatorname{div} J_n - qR$$

where $E = -\operatorname{grad} \psi$ is the electric field, ε the permittivity, ρ the space charge density, q the elementary charge and J_p, J_n the current densities of holes and electrons respectively.

2. CHOICE OF UNKNOWNS

From a physical point of view the variables ψ, p and n are the most logical choices. However, for computational reasons other choices are often advantageous. The most used possibilities are shown in the following table.

1	2	3	4	5	6
ψ	ψ	ψ	ψ	ψ	ψ
p	$\bar{p} = p/n_i$	$\sigma_p = \log \bar{p}$	$\phi_p = \sigma_p + \psi$	$\phi_p = \exp(\phi_p)$	$\bar{p} = \exp(-\psi)\Phi_p$ $= \exp(\phi_p - \psi)$
n	$\bar{n} = n/n_i$	$\sigma_n = \log \bar{n}$	$\phi_n = \psi - \sigma_n$	$\phi_n = \exp(-\phi_n)$	$\bar{n} = \exp(\psi)\Phi_n$ $= \exp(\psi - \phi_n)$

Table 1. Choices of unknowns

The unknowns may be ranked according to their "exponential behaviour". Then σ is the "friendliest" and Φ the "worst". However, the character of the equations in σ is extremely nonlinear.
The character of the equations in Φ is, from a mathematical point of view, "friendlier". The ϕ_p continuity equation, e.g. is monotonic in Φ_p (see Polak et al., [12]). Basically there are two classes of unknowns, on the one hand σ, ϕ on the other hand (p, n), Φ where the first are less varying in general than the second. However we would like to choose one of the operators of the second class for the nonlinear equation solving.
So preferred choices of unknowns and equations do not go together. We shall see how they are combined in the CURRY algorithm nevertheless.

3. SPATIAL DISCRETISATION

For the spatial discretisation a mesh-box-grid pair is constructed in the following way.
The mesh consists of quadrilaterals and triangles. Then the boxgrid is constructed from the centroids ($^1/4$ of the sum of the vertices for quadrilaterals, $^1/3$ of the sum of the vertices for triangles) and the midpoints of the sides as shown in figure 1.

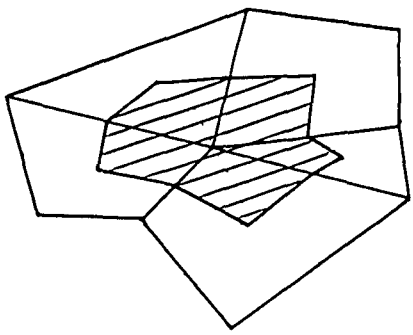

Fig. 1 Construction of mesh-box-grid

On the meshes unknown functions (e.g. ψ, σ_p, σ_n) are approximated in the FEM manner, by linear basis functions on the triangles and by isoparametic bilinear functions on the quadrilaterals. Then the equations, which are all of the type div. T= h, are replaced for each box by using a Green's theorem giving $\int_{\delta B} T.dn = \iint_{B} h$

where as usual B is the boxsurface and δB the boxperimeter. Then a quadrature is used to replace the left hand side by

$$\sum_{i=1}^{m} T_i.dn_i ,$$

and the right hand side by h_0.meas (B) where h_0 is evaluated in the only mesh point inside the box. For the continuity equations some form of fitting (see e.g. Doolan et al., [2], Polak et al., [11]) has to be used to prevent spurious oscillations. We use the fitting described in Polak et al., [11] (see also Polak et al., [12]), so in each quadrature point J is decomposed in $J_{E//}$ and $J_{E\perp}$, the components parallel and perpendicular to the field direction. Then the parallel component is fitted.

So e.g. the discrete J_p^h in a quadrature point is defined by

$$J_p^h = \mu \ (\text{grad } p_h + p_h \, \text{grad } \psi_h)$$

$$+ \ (\gamma(L) - 1)(\text{grad } \psi_h, \text{grad } p_h)(\text{grad } \psi_h / \ ||\text{grad } \psi_h \ ||^2)$$

where p_h and ψ_h are the (usual FEM) approximations,

$$\gamma(L) = L\coth(L) \quad \text{with } L = .5\sqrt{(\psi_v^2 \ ||v||^2 + \psi_w^2 \ ||w||^2)} \ ,$$

the isoparametic transformation given by $x(s, t)$, $y(s, t)$

$$v = (\partial x / \partial s, \ \partial y / \partial s) \ , \quad w = (\partial x / \partial t, \ \partial y / \partial t)$$

and ψ_v, ψ_w the v and w derivatives of ψ.

In the case of a quadrilateral mesh (no triangles) this scheme gives a nine point difference formula. There are two major criteria involved in choosing a discretisation for these equations: - stability
- local current continuity

The stability is obtained in an optimal way in some sense by the particular fitting procedure as is discussed in Polak et al., [11]. The local current continuity is guaranteed by the use of the Green's theorem. However, the usual way of applying this theorem by constructing boxes from midperpendiculars should be avoided because it gives excessive geometrical restrictions (Mock, [6]).
Also, in these conference proceedings, Van Welij ([15]) discusses a finite element method which can be viewed as a generalisation of the Scharfetter-Gummel method, both for 2- and 3-dimensional problems. In this method, the current density is uniquely defined. Also, it reduces to the standard finite element method whenever the electric field is zero.

4. TIME DISCRETISATION

For the time discretisation we compared three methods, backward Euler, the scheme presented in Bank et al., [1], here called the Bell scheme and a variant of Gear's method (see Hindmarsch, [4]). We also compared the choice of variables p and σ.
The conclusion was that Gear in σ is superior to the other schemes.

A typical comparison for a 1D diode is shown in the following table. The diode is switched from thermal equilibrium to 0.8 V in 10^{-9} secs. The table is a comparison at $t = 10^{-9}$.

METHOD	# DEC.PL.	# STEPS	NEWT.IT.
Bell	2.66	63	660
Gear	2.36	119	205
Gear( )	3.57	95	141

TABLE 2. Comparison of Bell, Gear and Gear ( ) when switching from thermal equilibrium to 0.8 V forward

5. NONLINEAR SOLVING

The discretised problem gives a finite set of nonlinear equations. To solve this set we use a combination of a number of algorithms:

- continuation
- Gauss-Seidel (Gummel)
- Newton + correction transformation
- subset solving

In this context continuation simply means increasing or decreasing the applied potentials from a situation with a known solution to a "nearby" situation with an unknown solution.
At present we use neither a predictor (see Polak et al., [9]) nor a sophisticated step strategy. The reason for this is that predictors sofar have been based on "polynomial reasoning" and disturb rather than predict the next solution.
Only the simplest step strategy is reasonably robust. However, we plan to investigate predictors again in the near future because we believe them to be essential.
Gummel's successive substitution is very well known so we do not describe it here. We only use it either where it converges fast (e.g. low currents) or to obtain an initial solution for the Newton-Raphson iterations. We always finally use the latter to obtain sufficient accuracy. Our use of the Newton-Raphson process in CURRY is special and very important for the computing time.
Here we come back to the choice of variables discussed in one of the previous sections.

Suppose we have a set of equations

$$\begin{matrix} F_1 & (u_1, \ldots, u_n) = 0 \\ \vdots & \vdots \\ F_n & (u_1, \ldots, u_n) = 0 \end{matrix}$$

with a Jacobian matrix J_n. Suppose further $u_i = u_i(v_i, \ldots, v_n)$. Then (under suitable nonsingularity conditions) $J_n = J_v \cdot J_{vu}$ where J_{vu} is the Jacobian matrix of $\partial u_i / \partial v_j$ components. This implies that a Newton correction vector dv for the v variables can be obtained from a Newton correction du for the u variables by a matrix multiplication. The matrix J_{vu} for e.g. the transformation from the Newton correction vector $d_{\sigma p}$ (of nodal values) to the Newton correction vector dp is a diagonal matrix. In general the matrix J_{vu} for any pair of choices for the unknown functions from table one is at most bidiagonal. So for the semiconductor problem it is a simple transformation from the vector of nodal Newton correction for one variable choice to the vector of nodal Newton corrections for another choice.

We use this to form triples, e.g. (σ, p, σ) where we first compute the Newton correction $d\sigma$, then from that the Newton correction dp and from this the correction $D\sigma$ for σ (that is not a Newton correction) defining the same change in p as dp.

The advantage is that we have a Newton process for the ϕ- operator, that is the "most linear" with only calculations in σ the "least exponentally behaved" variables. Of course, we do not explicitly compute the intermediate dp corrections. This implies that for the triple (σ, Φ, σ) we have

$$D\sigma = \log(1 + d\phi + d\psi) - d\psi \ .$$

Typically for a one dimensional transistor we may find the following table.

	without correct. transformation		with correction transformation	
BIAS	# NEWT.	CPU	# NEWT.	CPU
V_{be}= 0,V_{bc}= 1	15	6.43	5	3.26
V_{be}= 0.8,V_{bc}= 1	56	25.04	18	4.05
V_{be}= 0.9,V_{bc}= 1	38	23.28	23	7.38
Total	109	54.75	46	14.69

TABLE 3. Comparison of 1-D transistor simulations with and without correction transformation

SubSetSolving has been discussed in some detail in Polak et al., [10] and Schilders et al., [13]. The basic idea is to reduce the set of equations and unknowns as much as possible during the Newton process by omitting those with small enough corrections. Then we go back to the full set repeating this until in one reduction all corrections are small enough. The algorithm more precisely has the following form.

```
set initial guess u, tolerance C, T = 0, R = S
while T ≠ R
do R = S
    while R ≠ 0
    do S1 = R, S2 = S\R
        calculate du, u = u + du
        T = R
        R = { i ∈ R | |dui| > C }
     od
 od
```

We have used this algorithm for all our analyses the last years and found that where ever we performed a comparison the use of SubSetSolving was faster.
At present we are investigating the effect of the choice of variables on the SubSetSolving. So far we have used ϕ for the SubSetSolving.

However, we suspect that σ and p will give more benefit of the SubSetSolving, because charge neutral regions will have a zero Newton correction.

6. LINEAR EQUATION SOLVING

The linear equation solving has been extensively reported at the last conference in this series by Den Heyer ([3]).
We use a preconditioned CGS-method, for solving linear systems of the form Bx = d, where B is a non-symmetric matrix. The CGS-method is a variant of the bi-conjugate gradient method; see, e.g., Sonneveld ([14]). The algorithm is as follows:

Start: $y = 0$; $r_0 = b$; $p_0 = e_0 = r_0$; $k = 0$.
While ($||r_k|| >$ tolerance)
do

$$\alpha_k = (r_0, Br_k)/(r_0, B^2p)$$

$$y_{k+1} = y_k + \alpha_k(2e_k - Bp)$$

$$r_{k+1} = r_k - \alpha_k \times B(2e_k - Bp)$$

$$\beta_k = (r_0, Br_{k+1})/(r_0, Br_k)$$

$$e_{k+1} = r_{k+1} + \beta_k(e_k - \alpha_k \times Bp_k)$$

$$p_{k+1} = e_{k+1} + \beta_k(e_k - \alpha_k Bp_k + \beta_k p_k)$$

$$k = k + 1$$

od

The preconditioning is based on an idea of Meyerink ([5]). It consists of a nested tridiagonal block decomposition where the outer decompositions are incomplete.
In the inner decomposition small blocks of 3 by 3 matrices are found that are inverted exactly. To be more specific:
assume that

$$B = \begin{pmatrix} D_1 & U_1 & & & \\ L_2 & D_2 & U_2 & & \\ & \ddots & \ddots & \ddots & \\ & & \ddots & \ddots & U_{n-1} \\ & & & L_n & D_n \end{pmatrix}$$

Then: $B = (L + D')(D')^{-1}(D' + U)$
where D' is a block diagonal matrix with:

$$D'_1 = D_1$$

$$D'_j = D_j - L_j D_{j-1} U_{j-1}, \quad j = 2, \ldots, n.$$

However, the diagonal blocks D'_j of D' are not tridiagonal. Meijerink ([5]) proposed an incomplete factorization. In this decomposition, the matrix D' is replaced by Δ_1 with

$$\Delta_1 = D_1$$

$$\Delta_j = D_j - \text{trid}\,(L_j(\Delta_{j-1})^{-1} U_{j-1}), \quad j = 2, \ldots, n.$$

Then the block diagonal matrix Δ has the same sparsity patterns as D. Since L_j and U_{j-1} are (at most) tridiagonal, we only need to calculate the main diagonal and some codiagonals of $(\Delta_{j-1})^{-1}$. This is straightforward since Δ_{j-1} is tridiagonal:

$$\Delta_{j-1} = (\Lambda_{j-1} + I)\,\Theta_{j-1}\,(I + \Omega_{j-1})$$

Using the above decomposition, CGS is then applied to the following preconditioned system:

$$(I + \Omega)(L + \Delta)^{-1} B(\Delta + U)^{-1} (\Lambda + I)\,\Theta y$$

$$= (I + \Omega)(L + \Delta)^{-1} d.$$

7. EXAMPLE

Some examples illustrating the behaviour of the numerical methods are given in Polak et al. ([12]). Here we present a time dependent problem. We consider a MOSFET transistor, for which the doping profile is displayed in Fig. 2.

We first calculate the steady-state solution for the bias condition with 10V on the gate and 0V on the other contacts. Then, within 1ns, we switch the gate voltage to -5V. A transient simulation is then performed.

In Table 4 we display the process when using Gear's method with ψ, σ_p, σ_n as the unknowns. In Figs. 2 - 5 we show plots of the electron concentration at different time levels, whereas Figs. 6 - 9 contain the modulus of the electron current density.

TIME	# STEPS	ORDER	I-drain (n/cm)
1 ns	77	1	8.033(-1)
2	118	3	1.802(-1)
5	127	2	1.590(-1)
10	133	2	1.470(-1)
20	137	2	1.346(-1)
50	146	2	1.159(-1)
100	152	3	1.070(-1)
200	159	2	1.016(-1)
500	165	1	8.731(-2)
1 µs	171	2	6.665(-2)
2	179	2	4.315(-2)
5	190	2	1.987(-2)
10	196	2	1.003(-2)
20	202	2	4.865(-3)
50	209	2	1.795(-3)
100	214	2	8.120(-4)

TABLE 4. Results of transient simulation

REFERENCES

[1] R.E. Bank, W.M. Coughran, Jr., W. Fichtner and D.J. Rose, 'Computational aspects of semiconductor device simulation', Numerical Analysis Manuscript 85-3, AT&T Bell Laboratories (1985)

[2] E.P. Doolan, J.J.H. Miller and W.H.A. Schilders, Uniform numerical methods for problems with initial and boundary layers, Boole Press, Dublin (1980)

[3] C. den Heijer, 'Preconditioned iterative methods for non-symmetric linear systems', Proc. Int. Conf. Simulation of Semiconductor Devices and Processes, Pineridge Press, Swansea, 267-285 (1984)

[4] A.C. Hindmarsch, Ordinary differential equation solver, UCID-30001 Rev. 3, Lawrence Livermore Lab. (1974)

[5] J.A. Meijerink, 'Iterative methods for the solution of linear equations based on incomplete factorization of the matrix', Publ. 643, Shell, Rijswijk, The Netherlands (1983)

[6] M.S. Mock, 'Quadrilateral elements and the Scharfetter-Gummel method', Proc. Int. Conf. on Simul. of Semic.Dev. and Proc., Swansea 1984, 298-309, Pineridge Press, Swansea (1984)

[7] S.J. Polak, A. de Beer, A. Wachters and J.S. van Welij, 'MAGGY2, a program package for two-dimensional magnetostatic problems', Int. J. Num. Meth. Eng. 15, 113-127 (1980)

[8] S.J. Polak, A. Wachters, Th. Beelen and P.W. Hemker, 'A mesh-parameter-continuation method', Elliptic problem solvers, ed. M.H. Schultz, Academic Press, 383-390 (1981)

[9] S.J. Polak, A. Wachters, H.M.J. Vaes, A. de Beer and C. den Heijer, 'A continuation method for the calculation of electrostatic potentials in semiconductors', Proc. NASECODE I Conf., 149-175, Boole Press, Dublin (1979)

[10] S.J. Polak, W.H.A. Schilders, C. den Heijer, A.J.H. Wachters and H.M. Vaes, 'Automatic problemsize reduction for on-state semiconductor problems', SIAM J. Sci. Stat. Comput. 4, no. 3, 452-461 (1983)

[11] S. Polak, W. Schilders and A. Wachters, 'Box schemes for the semiconductor continuity equation', Coll. topics in appl. num. anal., ed. J.G. Verwer, Centre for Mathematics and Computer Science, Amsterdam, 313-324 (1984)

[12] S.J. Polak, C. den Heijer, W.H.A. Schilders and P. Markowich, 'Semiconductor device modelling from the numerical point of view', Int. J. Num. Meth. Eng. (to appear)

[13] W. Schilders, S. Polak and C. den Heijer, 'A comparison of subset solving algorithms', Proc. NASECODE III Conf., ed. J.J.H. Miller, Boole Press, Dublin, 258-264 (1983)

[14] P. Sonneveld, 'CGS, a fast Lanczos-type solver for nonsymmetric linear systems', Rept. 84-16, Delft University, Dept. of Math., Delft, The Netherlands (1984)

[15] J.S. van Welij, 'Element shape functions matching tangential components on edges', these conference proceedings

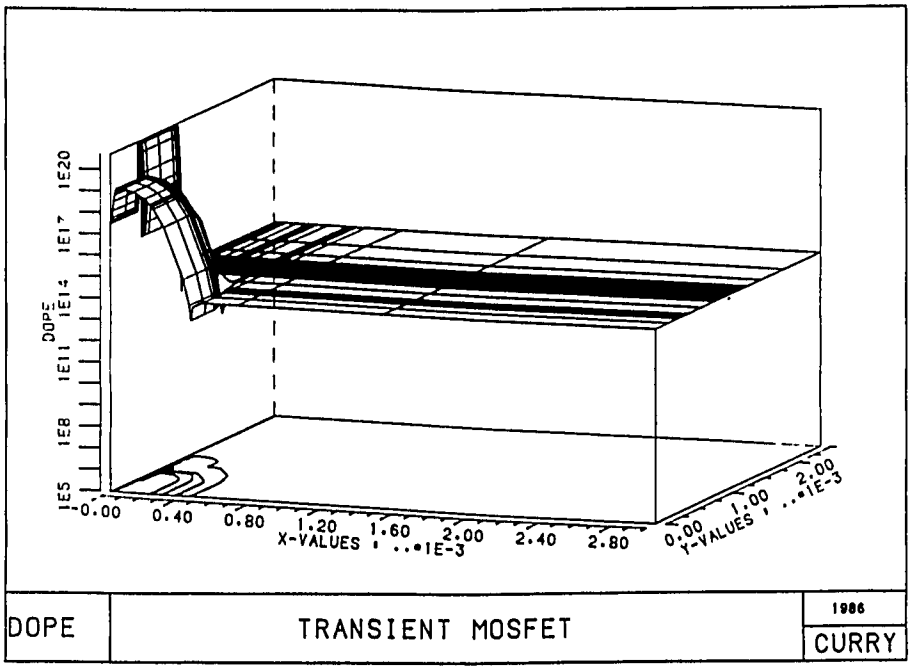

Figure 2

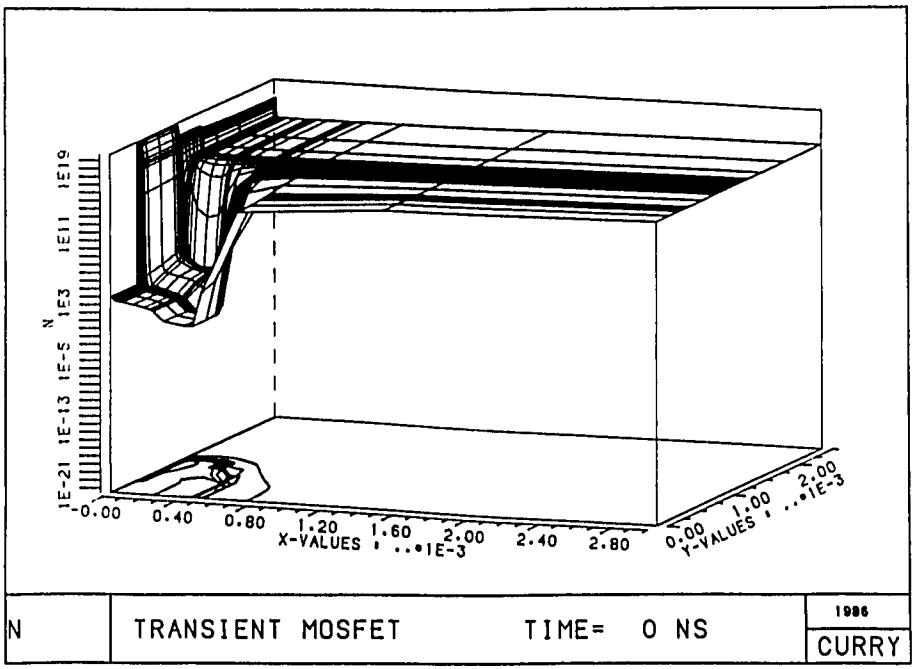

Figure 3

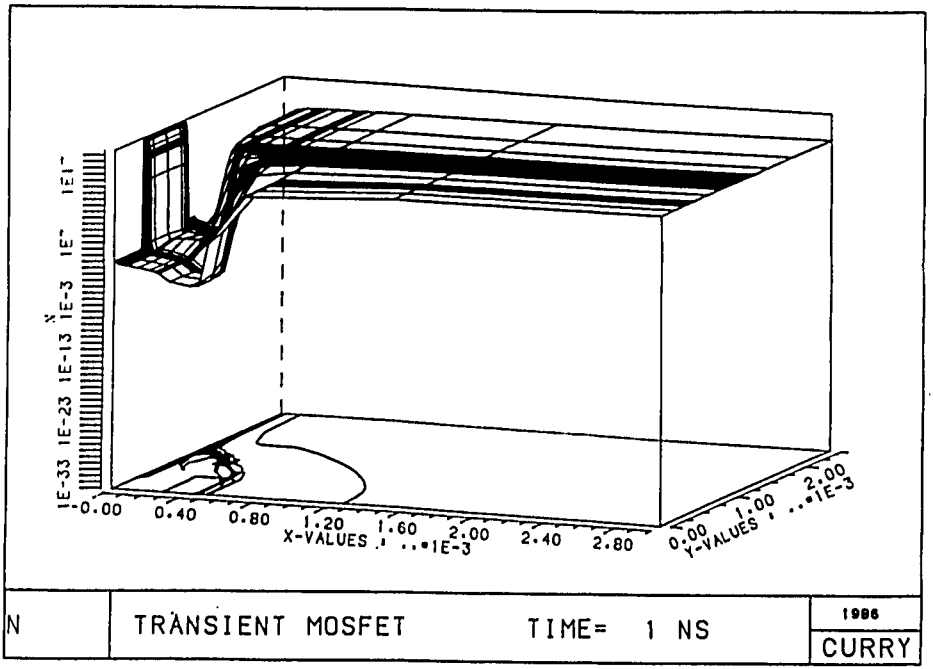

Figure 4

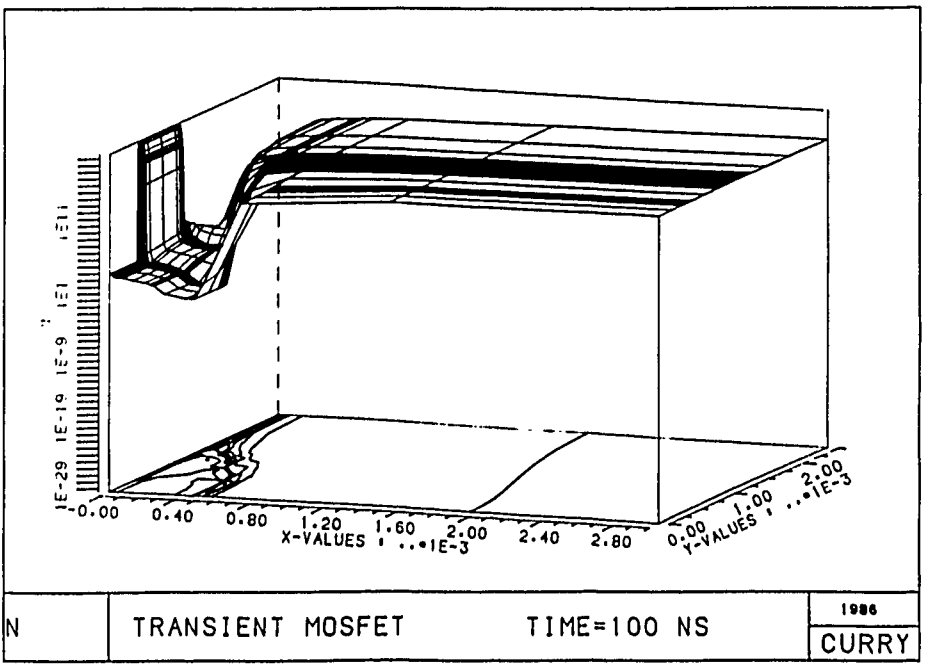

Figure 5

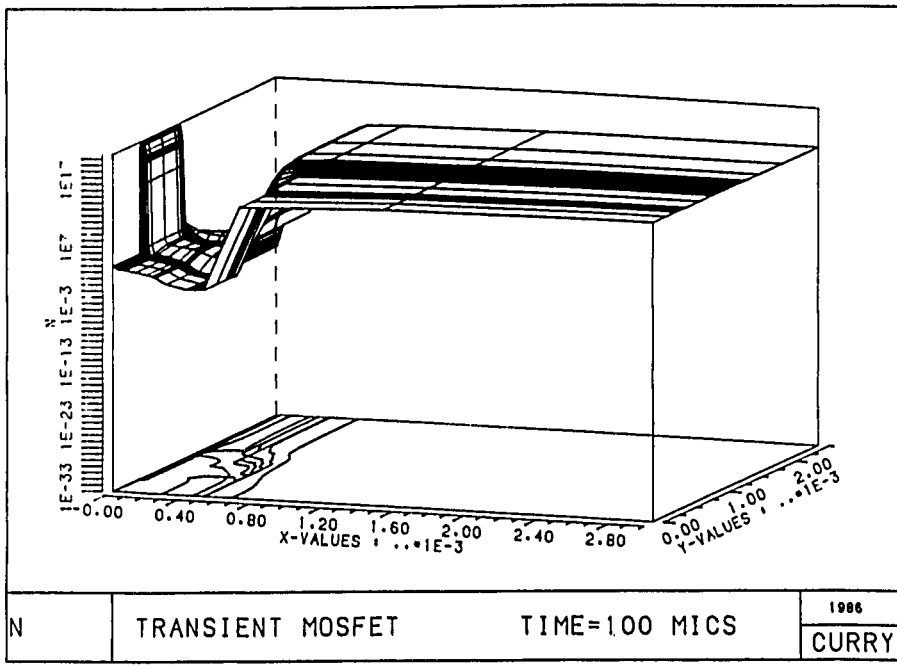

Figure 6

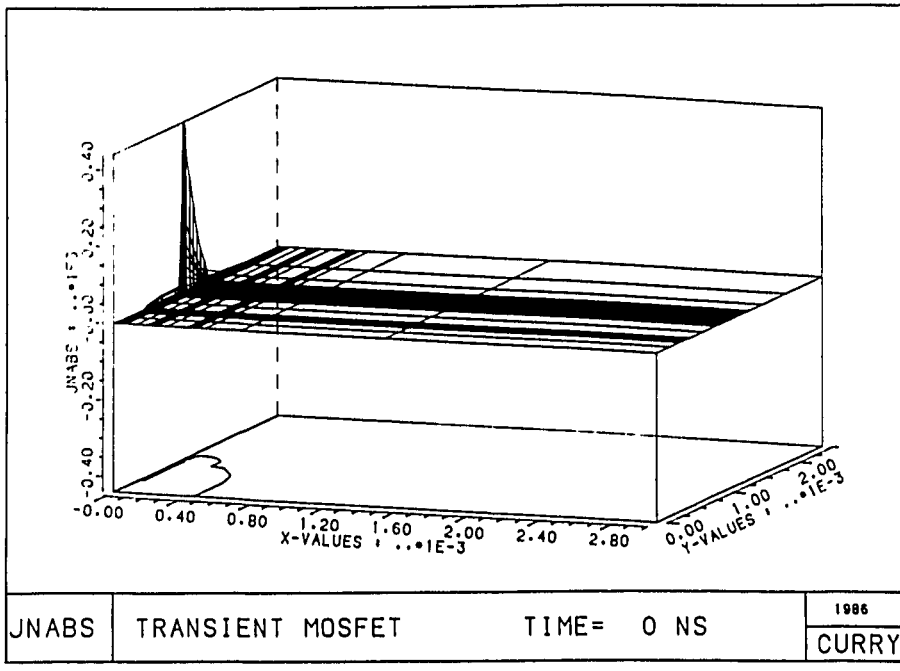

Figure 7

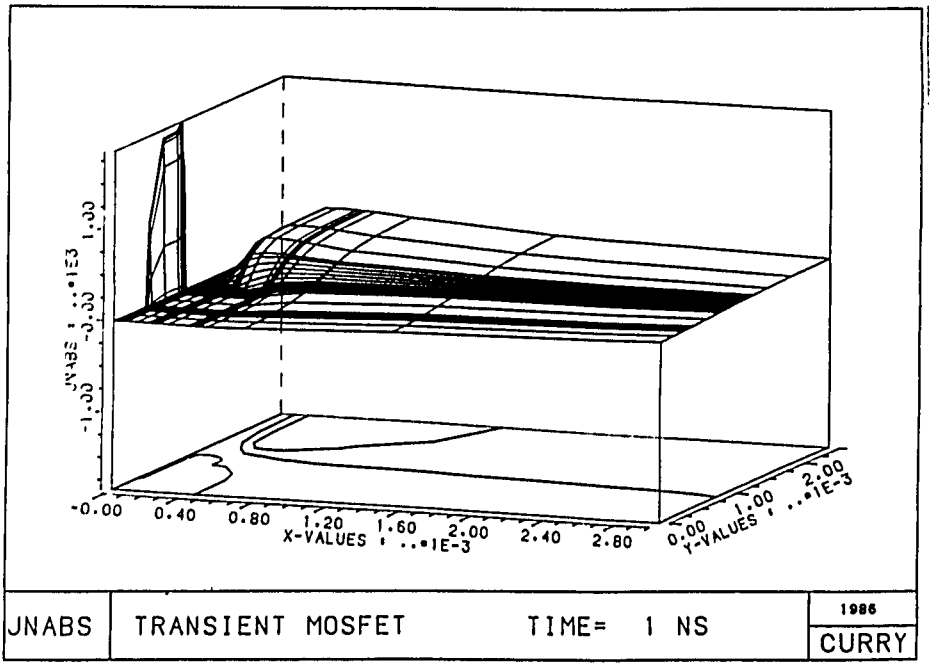

Figure 8

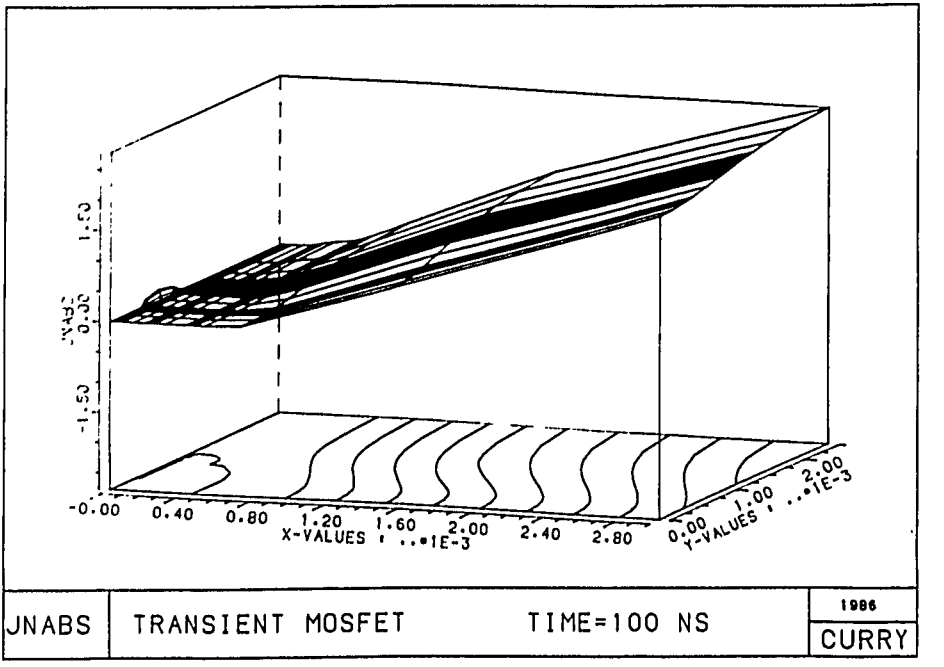

Figure 9