SIMULATION OF SEMICONDUCTOR DEVICES AND PROCESSES Vol. 3
Edited by G. Baccarani, M. Rudan - Bologna (Italy) September 26-28, 1988 - Tecnoprint

3D AUTOMATIC COMPUTATION OF EXTRINSIC CAPACITANCE MATRIX

F.Hecht. A. Marrocco

I.N.R.I.A Domaine de Voluceau, Rocquencourt
B.P 105 78153 Le Chesnay CEDEX, FRANCE

SUMMARY

We describe here an engineerized package, named LGCAPA, including a specific language which allows us to describe in a natural way the problem under consideration. This package LGCAPA provides a method for the computation of the capacitance matrix in multilayer interconnection environment and in presence of different dielectric regions.

The approximation of the physical problem is done via finite element technique; the threedimensional mesh is constructed automatically from the data needed for problem description. The set of linear problems are solved numerically by conjugate gradient with SSOR preconditionning algorithm. The package has been developped on a workstation of apollo type and was supported by the CNET.

INTRODUCTION

It is well known that the delays introduced by the interconnections can limit dramatically the speed of integrated circuits. The aim of this paper is to present a program package named LGCAPA allowing the exact capacitance calculation with use of 3.D finite elements technique on real IC environment including several dielectric and metallization layers. In the speed range which is actually raised by Ga As or Silicon digital I.C's , the quasi TEM approximation will be assumed.

When the electrostatic potential V, is the unknown variable, the problem we have to solve in order to be able to determine the capacitance matrix are of Laplace type and do not present mathematical difficulties. The major preoccupation remain of practical order essentially due to the fact that we are in 3.D environment. Data pre-processing are not so easy, and the resulting linear system obtained from finite element approximation are very large : at the present time in the software we can choose between two techniques to solve linear system ;

a) Cholesky decomposition
b) SSOR-preconditionned conjugate gradient
this last technique seems to be more efficient for large systems. After a presentation of the problem we want to solve, we will focus on data preprocessing i.e, how is obtained the finite element structure (mesh + physical data) needed for the approximation of Laplace equation, from a problem description which is as close as possible to the successive masks definitions in technological processes.

FORMULATION OF THE PROBLEM

From [1,2] by exemple, we obtain for a system of n conductors Ω_i ,i=1,...,n, (see fig 1) the following definition for the capacitance matrix

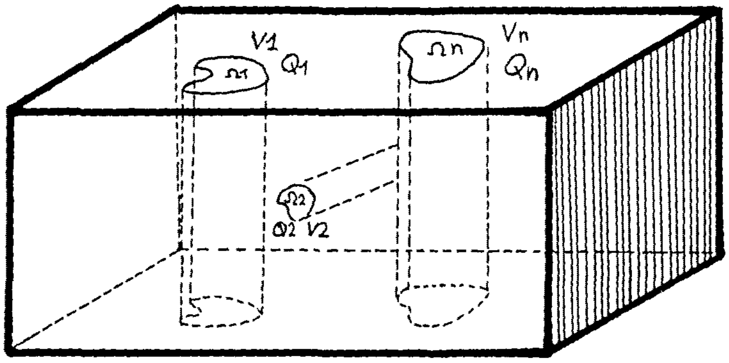

Fig: 1 - System of n conductors

- The conductor number 1 is at potential 1 and the others conductors at potential 0, then the corresponding charges are denoted by

(1) $C_{11}, C_{12}, \ldots, C_{1n}$

- If the conductor 1 is now at potential V_1 (the other conductors remaining at potential 0), from the linearity of equilibrium equations, the charges are now given by

(2) $C_{11} V_1, \ C_{12} V_1, \ldots, V_1 \ C_{1n} V_1$

- We exchange now the role of conductor 1 and 2, and if V_2 is the potential of conductor 2, the charges on conductor are given by

(3) $C_{21} V_2, C_{22} V_2, \ldots, C_{2n} V_2$

and so on, for the last conductor at potential V_n and the other ones at potential 0, the charges are given by

(4) $C_{n1} V_n$, $C_{n2} V_n$,..., $C_{nn} V_n$

- Finally, if V_i is the potential of the conductor number i, the charge Q_j on the conductor number j is given by

(5) $Q_j = \Sigma_{i=1,n} C_{ij} V_i$

The n x n matrix C_{ij} is the capacitance matrix, and we will see now how to compute this matrix by solving n linear problems ;

Given n conductors Ω_j at prescribed potential V_j, the potential distribution V in a domain Ω surronding these n conductors Ω_j ; (and not including their conductor) may be obtained by solving, the classical equation

(6) $- \nabla.(\varepsilon \nabla V) = 0$ on Ω , with boundary conditions on $\Gamma = \partial \Omega$

where ε is the permittivity and is a fonction of x, in general piecewise constant. The boundary conditions are of non-homogeneous Dirichlet type on boundary $\Gamma_j = \partial\Omega_j$ of the conductors, and are homogeous elsewhere, we denote by Γ_o the part of the boundary with zero Dirichlet condition and $\Gamma_n = \partial\Omega - \Gamma_o - \Gamma_1 ... - \Gamma_n$ the part of boundary with homogeneous Neumann condition.

Solving (6) with prescribed potential on conductors, we obtain the potential distribution V which allows us to compute the charge Q_j on the conductor j by

(7) $Q_j = \int_{\Gamma_j} \varepsilon \nabla V. \mathbf{n} \, ds$

where **n** is is the unit normal to the boundary Γ_j of conductor j.

For the solution of (6) by finite element, we introduce a weak formulation of the problem [3].

Let us define the fonctionnal space

(8) $W_o = \{v / v \in H^1(\Omega), v = 0 \text{ on } \Gamma_i \;\; i=0,...,n\}$

with $H^1(\Omega)$ the classical Sobolev space of $L^2(\Omega)$ functions with derivatives in $L^2(\Omega)$.

A weak formulation of the problem (6), is given by.

$$(9) \quad \begin{cases} \text{Find } V \text{ in } H^1(\Omega) \text{ such that} \\ V = V_i \text{ on } \Gamma_i \;\; i=0,...,n \text{ and} \\ \int_\Omega \varepsilon \nabla V. \nabla w \, dx = 0 \qquad \text{for all } w \text{ in } W_o \end{cases}$$

Taking into account the linearity of the problem (9) which give the potential distribution V, we can write

(10) $$V = \Sigma_{i=1,n} V_i \varphi_i$$

where φ_i is the solution of the problem.

(11) $$\begin{cases} \text{Find } \varphi_i \text{ in } H^1(\Omega) \text{ such that} \\ \varphi_i = \delta_{ij} \text{ on } \Gamma_j \quad i=0,\dots,n \\ \int_\Omega \varepsilon \nabla \varphi_i . \nabla w \, dx = 0 \qquad \text{for all } w \text{ in } W_o \end{cases}$$

where δ_{ij} is Kronecker symbol ($\delta_{ij} = 0$ if $i \neq j$ else $\delta_{ij} = 1$)

The formula (7) which give the charge Q_j on the conductor j may be written as

(12) $$Q_j = \int_{\Gamma_j} \varepsilon \nabla V . \mathbf{n} \, ds = \int_\Gamma \varepsilon \nabla V . n \, w_j \, ds$$

where w_j is a function taking δ_{jk} value on Γ_k (k=0,...,n).
By using Green's formula we can write

(13) $$\int_\Gamma \varepsilon \nabla V . \mathbf{n} \, w_j \, ds = \int_\Omega \varepsilon \nabla V . \nabla w_j \, dx + \int_\Omega \nabla.(\varepsilon \nabla V) \, w_j \, dx$$

but when solving (6) (or equivalently the weak form (9)), the second term in the right hand side is 0, so we obtain the relation valide for $w_j \in H^1(\Omega)$

(14) $$\int_\Gamma \varepsilon \nabla V . \mathbf{n} \, w_j \, ds = \int_\Omega \varepsilon \nabla V . \nabla w_j \, dx$$

the function φ_j solution of (11) satisfy the condition on w_j given in (12) and belongs to $H^1(\Omega)$ so that (14) is valid, we can write

(15) $$Qj = \int_\Gamma \varepsilon \nabla V . \mathbf{n} \, w_j \, ds = \int_\Omega \varepsilon \nabla V . \nabla \varphi_j \, dx$$

using (10) we obtain

(16) $$Qj = \int_\Omega \varepsilon \nabla V . \nabla \varphi_j \, dx = \Sigma_{i=1,n} V_i \int_\Omega \varepsilon \nabla \varphi_i . \nabla \varphi_j \, dx$$

so with (5) we obtain directly

(17) $$C_{ij} = \int_\Omega \varepsilon \nabla \varphi_i . \nabla \varphi_j \, dx$$

with φ_i solution of problem (11).

It is sufficient to solve the problem (11) for i=1,...,n to be able to compute the capacitance matrix C_{ij}.

We can remark that Γ_o introduced before is in general the connection to the ground (reference potential), but may be empty and this part of boundary can be considered as a conductor; the capacitace matrix will be then (n+1) x (n+1).

The computation of capacitance matrix is straightforward from (17) when using finite element for solving problem (11).

PROBLEM DESCRIPTION AND AUTOMATIC MESHING

The problem we have to solve is described via an oriented language. This language allows us :

- to define scalar variables
- to control classical loops by keywords as
  do enddo, if then elseif else endif, repeat,...
- to include data files
- to use fortran type expressions.

In addition to this a set of specific instructions are more problem-oriented and are used to define the problem (geometry + material property -see below-), to set parameters values in order to control the mesh refinement (in 2D and 3D), to call specific modules (2D_mesh_generator, 3D_mesh_generator, capacitance calculation,...)

The major restriction in geometrical description, at present time, is that any subdomain, or part of subdomain (conductor; dielectric) has to be defined as cylindrical region i.e. by giving the following data : a set of node coordinates $(x_i,y_i),i=1,N$ which describe a contour in (X,Y) plane, with two height (h_1,h_2) which localize this region in the Z-direction. These data are sufficient for the geometrical subdomain description in our restricted context; we have to complete these data by the relative permittivity if the subdomain is a dielectric or by a number (in order to give a serial number to each contact) if the subdomain represents a conductor.

The description (and construction) of the computational domain is temporally dependent in the following sense: starting with an initial cylindrical box (in general this initial box is air) each new subdomain (which is obviously included in the initial box) will erase the previous material definition by the current one.

Sample example of input file for geometry description

```
-- variables definition for parametrisation
-- ( lines beginning with -- are comment)
--
     x_min=-100 ; x_max=100;
     y_min=0.   ; y_max=100;
     h0=0.      ; htop =500;
     h1=200     ; h2=h1+1  ; h3=h1+0.5 ;
     thick=1.5  ; h4=h3+thick;

-- contour def
-- Initial box (filled with air eps=1              ---> fig 2a

contour eps=1 ,hmin=h0,hmax=htop,
    (x_min,y_min),(x_max,y_min),(x_max,y_max),(x_min,y_max)
-- first dielectric layer                          ---> fig 2a
contour eps=12 ,hmin=h0,hmax=h1 ,
    (x_min,y_min),(x_max,y_min),(x_max,y_max),(x_min,y_max)
-- second dielectric layer                         ---> fig 2b
contour eps=7 ,hmin=h1,hmax=h2 ,
```

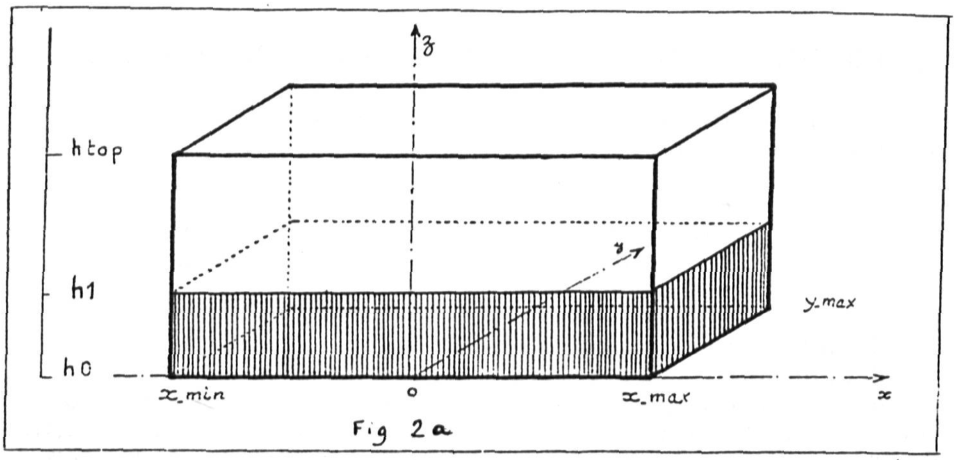

Fig 2a

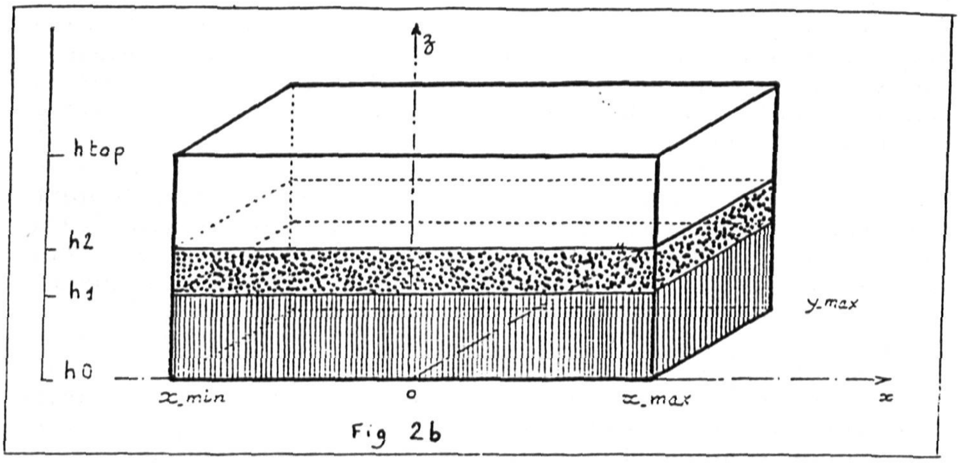

Fig 2b

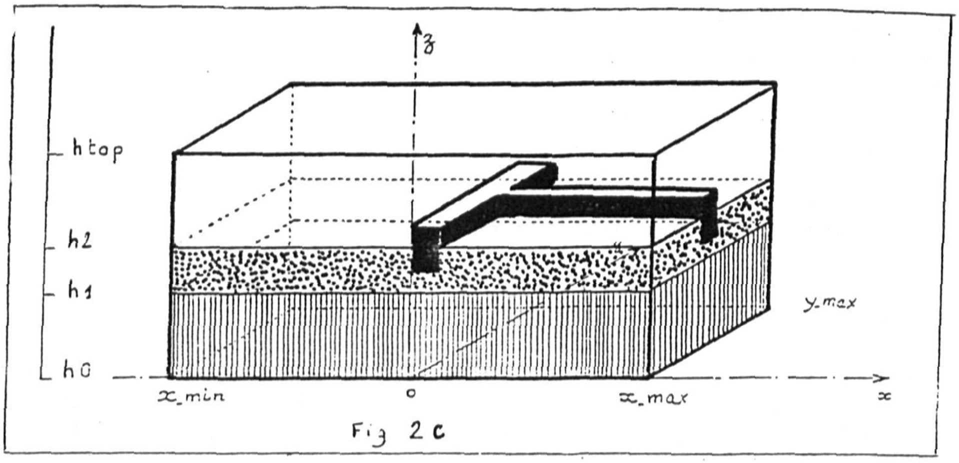

Fig 2c

Fig 2 : Domain construction

```
    (x_min,y_min),(x_max,y_min),(x_max,y_max),(x_min,y_max)
-- conductor inlayed in dielectric layer2          ---> fig 2c

-- definition of characteristic points of the contour
      x1= ( x_min + x_max ) /2;
      x2= x1 + thick;
      y1= ( y_min + y_max ) /2.   - thick*.5 ;
      y2= y1 + thick ;

contour  conducteur 1 , hmin=h3 , hmax=h4 ,
    (x1,y_min),(x2,y_min),(x2,y1),(x_max,y1),(x_max,y2),
    (x2,y2),    (x2,y_max),(x1,y_max) ;
```

The geometrical definition (including material property is ended by the following key-word **end_contours;** this key-word will activate a module which compute all the contours intersections (in X,Y plane) define 2D subdomains, and distribute points on these contours, i.e. this module will prepare the data needed for a 2D mesh generator. We can act on this 1.D mesh (if we want because default values are assumed), by the mean of three parameters **pasmin, pasmax, nbpas** which have then to be specified before the key-word **end_contours.**

The 2.D mesh generator is called by the key-word : **maillage2d**: two parameters **coef_mesh** and **puis_mesh** allows us to control the number of triangles generated and the depth of refiniment. The 2D mesh generator is the same as the one used in package TITAN [4].

The 3D mesh generator called by the key-word : **maillage3d** construct first a mesh with pentahedrons (triangular prisms) derived directly from the 2D (X,Y) triangular mesh by extension in Z-direction (layers in Z direction are automatically detected from the contours definitions). In a second step this mesh is transformed in a tetrahedral mesh by subdividing each pentahedron into 3 tetrahedrons. Then, the tetrahedreons inside the conductors are removed from the mesh (as they are not needed for potential calculation). Three parameters **pasminh, pasmaxh, nbpash** allows us to control the mesh refinements in the Z-direction (default values are assumed).

With the key-word **capa**, the electric potential distributions are computed and after the capacitance matrix is calculated according to (17). Differents parameters may be specified before acting the key-word **capa**, we can prescribe some boundary conditions, we can force some algorihmic variables (precision, maximum number of iterations in conjugate gradient) we can choose the technique for the solution of linear problem (**cholesky, gradient_conjugue** or **gradient_conjugue_ssor** (default choice)).

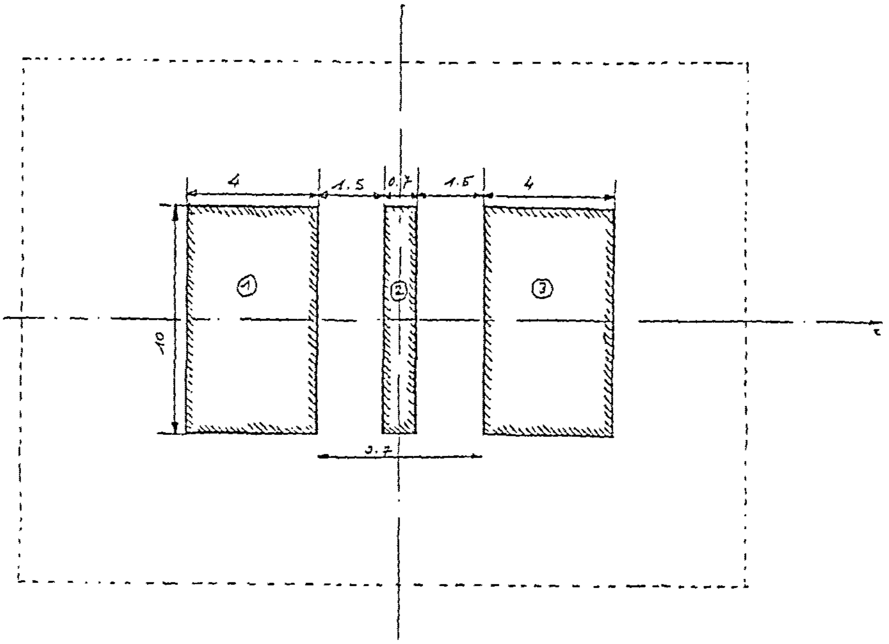

Fig 3 : EXAMPLE : TOP-VIEW (Masks)

EXAMPLE : CROSS-SECTION ALONG X

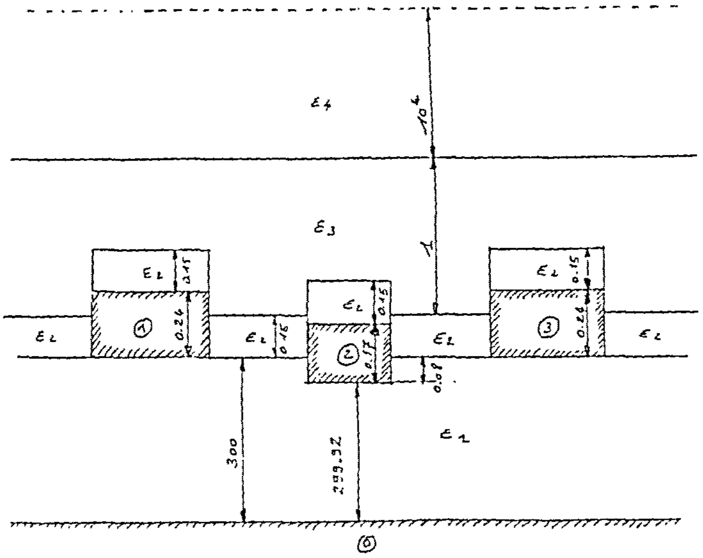

Fig 4 :

NUMERICAL APPLICATION

The numerical example is represented by the figure 3 and 4. Only half domain is needed for all potential calculations. The domain is composed with four different dielectrics (including air) and three contacts. the bottom is connected to the ground but is considered as a conductor for capacitance matrix computation (referenced as T).

The data for problem description are the following :

```
----------relative permittivity definition --------------
e1 = 12.  ;
e2 = 3.5  ;
e3 = 7.4  ;
e4 = 1.   ;
---------  heigths definition + typical points------------
h0 = 0  ;
h2 = 300 ;
h1 = h2-0.08 ;
h3 = h1 + 0.17 ;
h4 = h2 + 0.15 ;
h5 = h2 + 0.24 ;
h6 = h3 + 0.15 ;
h7 = h5 + 0.15 ;
h8 = h4 + 1    ;
h9 = 600       ;

x1 = 3.7/2;   x2=x1+4;     y1=0;     y2=10/2;

x_min = -100;
x_max =  100;
y_min = 0;
y_max = 100;

-- contours definition --
-------------------------

contour eps=e4,hmin= h0,hmax=h9,
     (x_min,y_min),(x_max,y_min),(x_max,y_max),(x_min,y_max);
contour eps=e1,hmin= h0,hmax=h2,
     (x_min,y_min),(x_max,y_min),(x_max,y_max),(x_min,y_max);
contour eps=e2,hmin= h2,hmax=h4,
     (x_min,y_min),(x_max,y_min),(x_max,y_max),(x_min,y_max);
contour eps=e3,hmin= h4,hmax=h8,
     (x_min,y_min),(x_max,y_min),(x_max,y_max),(x_min,y_max);

do i= 1,3,2
  contour conducteur i ,hmin = h2,hmax= h5,
       (x1,y1),(x2,y1),(x2,y2),(x1,y2);
  contour eps=e2       ,hmin = h5,hmax= h7,
       (x1,y1),(x2,y1),(x2,y2),(x1,y2);
  x1 = -x1; x2 = -x2;
enddo;
```

```
x0 = .7/2 ;
contour  conducteur  2  ,hmin  =  h1,hmax=  h3,
      (-x0,y1),(+x0,y1),(+x0,y2),(-x0,y2);
contour  eps=e2         ,hmin  =  h3,hmax=  h6  ,
      (-x0,y1),(+x0,y1),(+x0,y2),(-x0,y2);

pasmin  =  .7;  pasmax  =  1200/9;nbpas  =  5;
end_contours  ;
-- 2D mesh generator call ---
maillage2d;
-- 3D mesh generator call ---
pasminh  =   6.0E-02;   pasmaxh  =   300.0;   nbpash  =  1.70;
maillage3d  ;
-- capacitance matrix calculation ---
capa  ;
end
```

We can see on figure 5 and 6 the 2D mesh automatically generated and on figure 7 a view of the 3D mesh.

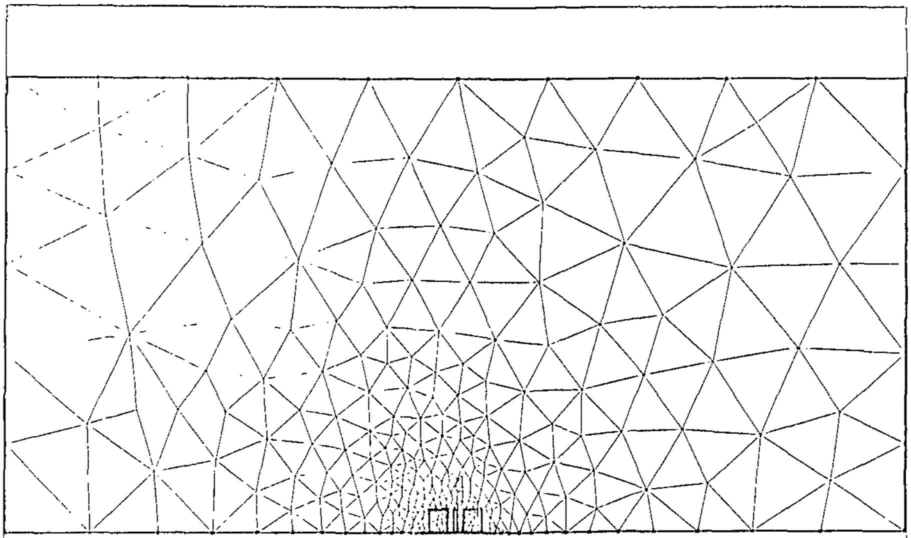
Fig 5 : 2D-Mesh

The capacitance matrix in femto Farads (for computational domain i.e. half domain) is given by

```
    Conductor 1   Conductor 2    Conductor 3      T

1     1.5348E-3   -5.4730E-4     -3.0920E-4    -6.7887E-4
2    -5.4730E-4    1.2669E-3     -5.3845E-4    -1.8129E-4
3    -3.0921E-4   -5.3845E-4      1.5319E-3    -6.8482E-4
T    -6.7881E-4   -1.8130E-4     -6.8472E-4     1.5401E-3
```

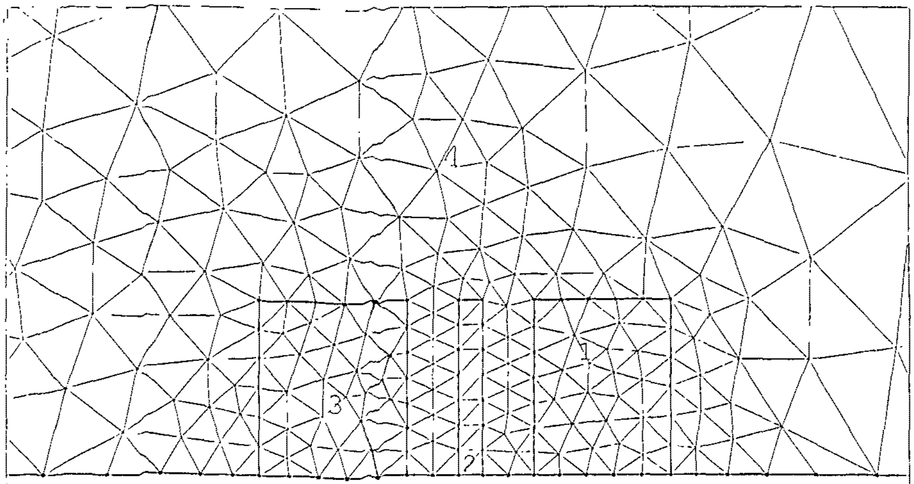

Fig 6 : Zoom on 2D. Mesh

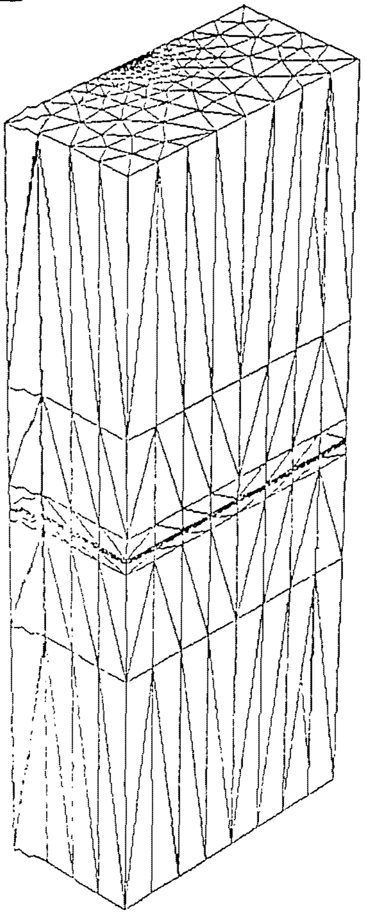

Fig 7 :

View of 3D. Mesh

Concerning the computing time on apollo workstations we have the following results:for a 3D mesh with 6356 nodes and 31896 tetrahedrons , the total CPU time was 5900 s on DN3000 apollo workstation, with 11156 nodes 57873 elements the total CPU time was 8596 s.

Different numerical tests have been done on inter-crossing lines, with different meshes in order to extract the variation coefficient versus the mesh size.A diagonal coefficient is a decreasing function with respect to the mesh size and tends rapidly to a constant value.An off diagonal element is an encreasing function and tends able to a limits when the mesh size decrease.

Remarks

In order to takes into account in a better way of infinite domain, several techniques are possible, first by considering purely fictitious medias surrounding a real one, medias with relative permittivity <<1. , (in this case nothing has to be changed in the module LGCAPA), the computational domain is artificially extended by this trick,secondly with more sophisticated techniques as use of infinite elements [5], mapped infinite elements [6], ballooning techniques [7], by coupling classical finite elements with boundary elements techniques [8].

REFERENCES

[1] **G.Bruhat** : Electricite ,*Masson et Cie Editeurs, 1963.*

[2] **R.Dautray,J.L.Lions**: Analyse mathématique et calcul numérique pour les sciences et les techniques (t 1) *Masson 1984.*

[3] **P.G.Ciarlet**:The finite element method for elliptic problems. North Holland 1978.

[4] **A.Gerodolle, S.Martin, A.Marrocco**:Finite element method applied to 2D MOS process simulation and defect diffusion: program TITAN. *Proceeding Nasecode IV Conference ,Dublin 1985.*

[5] **P.Bettess**: Infinite elements. *Int. Journal for numerical methods in engineering Vol 11, 53-64 ,(1977)*

[6] **O.C.Zienckiewicz, C.Emson, P.Bettess**: A novel infinite element. *Int. Journal for numerical methods in engineering Vol 19,393-404, (1983).*

[7] **P.P.Silvester, D.A.Lowther, C.J.Carpenter, E.A.Wyatt**: Exterior finite elements for 2-dimensional field problems with open boundaries. *Proc I.E.E. Vol 124, n° 12, dec 1977.*

[8] **O.C.Zienckewicz, D.W.Kelly, P.Bettess**: The coupling of finite element method and boundary solution procedures. *Int. Journal for numerical methods in engineering Vol 11,355-375 , (1977)*