Electron Technology, 19, 1/2, pp. 59–73, Institute Technology, Warszawa 1986

ANALYSIS OF DIGITAL INTEGRATED CIRCUITS USING CHARGE CONSERVATION PRINCIPLE

BOGDAN M. WILAMOWSKI

Institute of Electronic Technology, Gdańsk Technical University
ul. Majakowskiego 11/12, 80-952 Gdańsk, Poland

DOUGLAS J. HAMILTON, ZBIGNIEW J. STASZAK

Department of Electrical Engineering, University of Arizona,
Tucson, Arizona 85721, USA

ANDRZEJ J. MAJEWSKI

Institute of Informatics, Gdańsk Technical University
ul. Majakowskiego 11/12, 80-952 Gdańsk, Poland

Received 16 January, 1986

SUMMARY

A VERY FAST and effective algorithm based on the charge conservation principle for the transient analysis of integrated circuits is presented. This algorithm uses the explicit method computation therefore time is directly proportional to the circuit complexity. For a medium-size MOS circuit the computing time is up to 100 times shorter than that required by the SPICE2 program. In the case of large circuits, even better results are expected.

1. INTRODUCTION

HISTORICALLY. CAD tools were first developed for linear circuits which required the solution of a set of linear equations. Then, more useful programs for transistor networks were developed in which nonlinear equations were linearized. The existing linear analysis programs were used many times in a Newton iterative procedure. Next, the transient analysis was introduced where nonlinear analysis was carried on for each time interval.

The majority of existing programs more or less follow the above procedure [1]. To improve the computing speed, various approaches are used, such as sparse matrix techniques, implicit integration methods, a sparse tableau analysis method

[2], a modified nodal analysis method [3], circuit decomposition and a modular approach. A notable exception is the waveform relaxation method [4], which also takes advantage of signal latency. It requires, however, a very large memory and, in the case of circuits with many feedback loops, its efficiency is rather poor. A method to compress storage data by one or two orders of magnitude has been also published [5].

In this paper, we describe a simple explicit method [8], in which the computing time for medium-size circuits is up to 100 times shorter than that required by the SPICE2 program [6, 7]. In the case of large circuits, even better results are expected. It should be pointed out that the SPICE2 program has been designed as a universal program for very accurate solutions but not as a very fast transient analysis program. It is used here to check accuracy of our solutions, and as a reference only for speed comparison.

The paper proceeds as follows. First we describe the algorithm to be used, which uses the unbalance of the conduction currents at each node to compute the charge stored on capacitances connected to each node. Then, the method of partitioning the charge among the capacitances is described in terms of the analysis of a capacitive network. Next, as an example, the transient response of a cascade of 4 bipolar transistor inverters is calculated. Finally, the transient responses of various CMOS circuits are calculated. In general, for this algorithm the computing time is directly proportional to circuit complexity.

2. PRINCIPLE OF THE ALGORITHM

At any node in the circuit consisting of nonlinear resistors, sources and capacitances, displacement current must flow if the algebraic sum of conduction currents flowing into a node is not zero. These displacement currents result from the charging of capacitances connected to the node. In general, a set of nonlinear differential equations has to be solved

$$\begin{gathered} I_{1j}(V)+C_{1j}(v_{1j})\,dv_{1j}/dt=0 \\ \cdots\cdots\cdots\cdots\cdots\cdots \\ I_{ij}(V)+C_{ij}(v_{ij})\,dv_{ij}/dt=0 \\ \cdots\cdots\cdots\cdots\cdots\cdots \\ I_{Nj}(V)+C_{Nj}(v_{Nj})\,dv_{Nj}/dt=0 \end{gathered} \tag{1}$$

where $I_{ij}(V)$ are the conduction currents flowing into node i; these may be nonlinear functions of several node voltages, $C_{ij}(v_{ij})$ are grounded $(i=j)$, and coupled $(i=j)$ capacitances; these may be nonlinear functions of voltage, v_{ij} are voltages betweeb nodes j and i, N is the total number of nodes.

If it is assumed that all voltages are known, the values of all conduction currents can be calculated. The net conduction current I_i folowing into a node for a time interval Δt produces a charge increment $\Delta Q_i = I_i\,\Delta t$ which is distributed among the capacitances connected to the node. In an integrated circuit the capacitances are nonlinear functions of node voltages of conduction currents; thus, if the node voltages are known, the capacitances can be calculated. Then, a network of capacitors of known capacitance can be analyzed with the previously calculated value of ΔQ_i at each node; from, this the corresponding Δv_i at each node can be determined. For the next time interval the node voltages are incremented by their appropriately Δv_i, New I_i, ΔQ_i and Δv_i are then computed.

The algorithm is thus implemented as follows:

1) If initial node voltages are not known, assume plausible values (perhaps 0) and compute the net conduction current at each node

$$I_i = \sum_{J=1}^{K} I_{ij}(V), \tag{2}$$

where the summation indicates that there are K conduction currents flowing into node i. Note, that these may be nonlinear functions of various node voltages. In the static case, the I_i are, of course, all zeros.

2) Calculate for each node the charge increment accumulated during Δt

$$\Delta Q_i = \int_{t}^{t+\Delta t} \Delta I_i(\tau)\, d\tau. \tag{3}$$

Since the functions $I_i(\tau)$ are generally unknown, and only discrete values for the previous time steps were calculated, the shape of this function can be predicted using zero, first or second order interpolations methods:

— with the zero order interpolation

$$\Delta Q_i = \Delta t\, I_i(t), \tag{4}$$

— with the first order interpolation

$$\Delta Q_i = \Delta t \cdot [1.5\, I_i(t) - 0.5\, I_i(t-\Delta t)], \tag{5}$$

— with the second order interpolation

$$\Delta Q_i = \Delta t\, [1.75\, I_i(t) - I_i(t-\Delta t) + 0.25\, I_i(t-2\Delta t)]. \tag{6}$$

In the last case, values from two previous time periods must be stored in memory. For the zero-order case, very short time steps must be used to achieve the required accuracy. As a compromise, the first-order case was used in examples to be given.

3) Using some appropriate interpolation method, predict the values of all nonlinear capacitances for the time $t+\Delta t$. For example, with the first order interpolation

$$C(t+\Delta t) = 2C(t) - C(t-\Delta t). \tag{7}$$

4) Compute the Δv_i at each node. Since the ΔQ_i are known, as are the capacitance values of all capacitors, the following set of equations representing the capacitor network must be solved

$$C_1 v_1 + \sum_{j=1}^{N} C_{1j}(v_j - v_1) = Q_1$$
$$\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots$$
$$C_i v_j + \sum_{j=1}^{N} C_{ij}(v_j - v_i) = Q_i \tag{8}$$
$$\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots$$
$$C_N + v_N + \sum_{j=1}^{N} C_{Nj}(v_j - v_N) = Q_N,$$

where C_i, C_{ij} are interpolated values of grounded and coupled capacitances, $\Delta r_i, \Delta v_j$ are increments of node voltages caused by the charge increment ΔQ_i.

Note, that if there are N nodes then Eq. (8) represents a system of N linear equations. Solution of these equations is discussed in the next section.

5) Compute the new values of node voltages

$$v_i(t+\Delta t) = v_i(t) + \Delta v_i. \tag{9}$$

In the implementation of the algorithm, the time period of interest, T, is divided into subintervals called external time steps. Data points are to be calculated at each external time step. The external time steps are divided into intervals called internal time step. An iteractive procedure using the internal time steps is carried out to obtain convergence for each external time step.

3. SOLUTION OF THE CAPACITIVE NETWORK

As was discussed in step 4) of the algorithms, to compute the v_i a network of capacitors must be analyzed. Such a network is characterized by a matrix equation in the general form

$$[C_{NN}][v_N] = [\Delta Q_N]. \tag{10}$$

Since the capacitance values for a given time interval are held constant, Eq. (10) is linear. Moreover, since the capacitive network is passive, the main diagonal of the C_{NN} matrix is always dominant. Therefore, a simple Gauss-Seidel iterative procedure can be applied, and convergence is quaranteed.

In examples that have been analyzed with the Gauss-Seidel procedure, convergence was typically obtained in 10 to 20 iteractions when the grounded capacitances were dominant. However, convergence is slow in cases where large capacitances between nodes occur. As an example, consider the circuit of Fig. 1. In this case, the capacitances between nodes are up to 10000 times greater than the grounded capacitances, and convergence was not obtained even after 100 iteractions; this is illustrated in Fig. 2.

A method producing a more rapid convergence assumes an exponential relationship for the node voltage increments

$$\Delta v_i(k+h) = \Delta v_i(k) \cdot [1 - \exp(-h \cdot \alpha_i)], \tag{11}$$

STASZAK

(8)

ances,
lQ_i.
linear

(9)

T, is
ulated
ervals
eps is

twork
natrix

(10)

stant,
main
Seidel

nver-
ances
ances
case,
nded
; this

ential

(11)

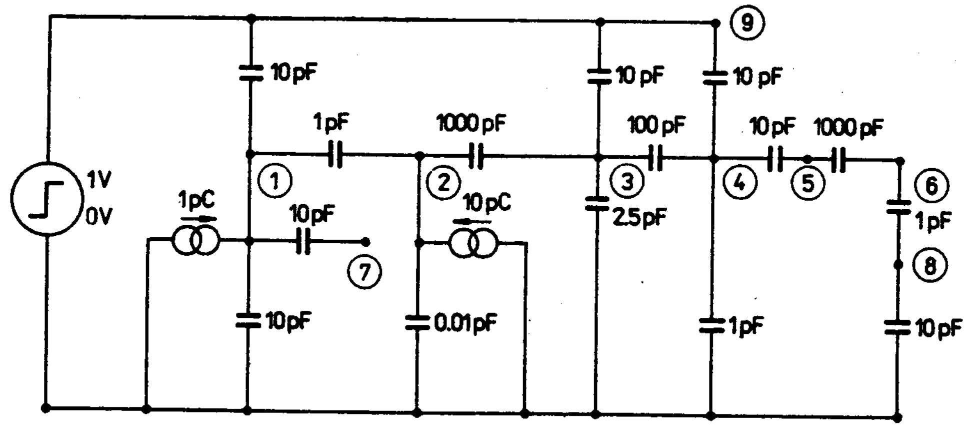

FIG. 1. Example of capacitive circuit activated by voltage source applied to node 9 and by injected charges into nodes 1 and 2.

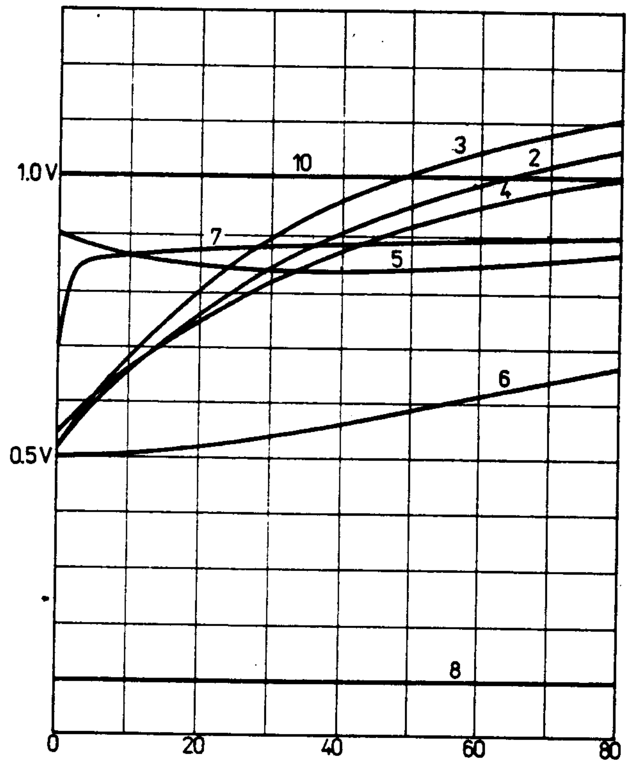

FIG. 2. Node voltage increments v_i for the circuit shown in Fig. 1 as a function of number of iteractions of a Gauss-Seidel algorithm.

where $\Delta v_i(k)$ is the voltage increment at node i for the k-th iteration, $\Delta v_i(k+h)$ is the voltage increment at node i for the $k+h$ iteration, α_i is the "attenuation constant" defined by

$$\alpha_i = 1/\ln[v_i(k-1)/v_i(k)]. \tag{12}$$

Therefore, after a few steps of using the standard Gauss-Seidel iterative procedure, new predicted values of node voltages can be computed as for example, the $k+100$ step. FIG. 3 shows that, even for networks with large capacitances between nodes (Fig. 1), rapid convergence is obtained.

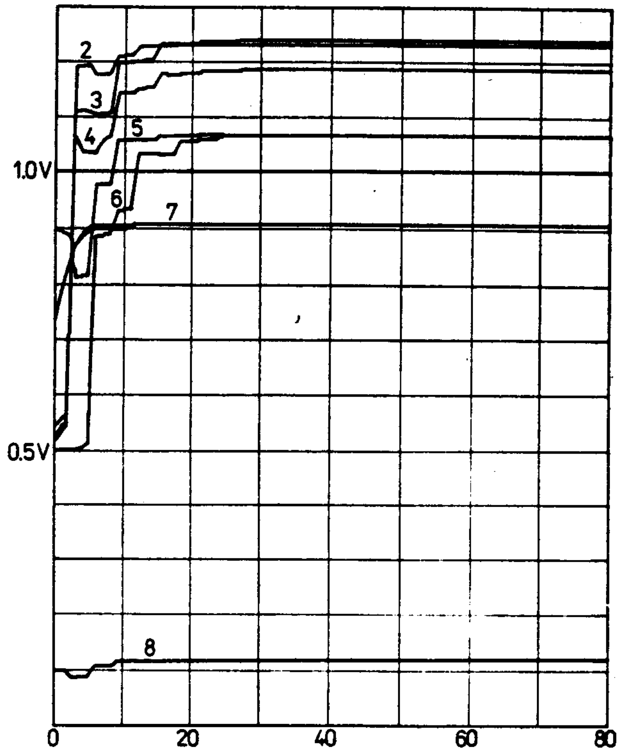

FIG. 3. Node voltage increments v_i for the circuit shown in Fig. 1 as a function of number of iteractions of a modified Gauss-Seidel algorithm.

4. SIMULATION OF BIPOLAR INTEGRATED CIRCUITS

To test the algorithm, a cascade of 4 bipolar inverter stages shown in FIG. 4 was analyzed. Such a saturating type of circuit has very nonlinear capacitances.

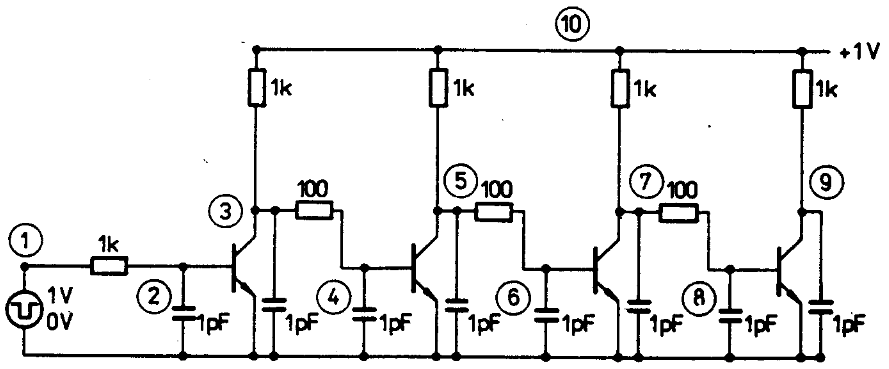

FIG. 4. Bipolar circuit analyzed with CHACO and SPICE2 programs.

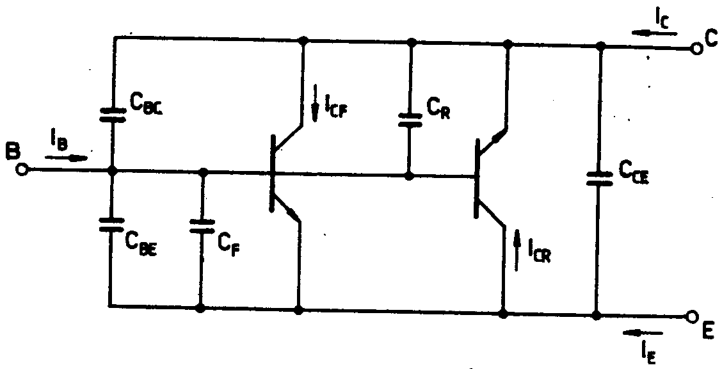

FIG. 5. Equivalent circuit for the bipolar *NPN* transistor model used in CHACO.

TABLE 1

Functional dependence of *NPN* bipolar transistor model

$I_{CF} = I_{SF} * [\exp(V_{BE}/V_T) - 1] * (1 + V_{CB}/V_F)$
$I_{CR} = I_{SR} * [\exp(V_{BC}/V_T) - 1] * (1 + V_{EB}/V_R]$
$I_B = I_{CF}/\beta_F + I_{CR}/\beta_R$
$I_C = I_{CF} - I_{CR} * (1 + 1/\beta_R)$
$I_E = I_{CR} - I_{CF} * (1 + 1/\beta_F)$
$C_{BE} = C_{BEO} * (V_{EB} + V_B) - 0.5$
$C_{BC} = C_{BCO} * (V_{CB} + V_B) - 0.5$
$C_F = I_{CF} * t_F/V_T$
$C_R = I_{CR} * t_R/V_T$

The equivalent circuit for the bipolar transistor model is shown in FIG. 5; functional dependencies of the nonlinearities are given in TABLE 1.

FIGURE 6 shows the computed waveforms for the node voltages for a time increment of 1 ns and total time period of 100 ns. The results from a SPICE2 analysis are shown in FIG. 7. Note that there are no significant differences between those figures. All computations were performed on a VAX-11/780 computer; for our algorithm the CPU time was 9.10 seconds while with SPICE2 it was 23.66 seconds. The times are not very different due to the fact the circuit contains large highly nonlinear capacitances which are not grounded and also static characteristic of bipolar transistors are very nonlinear in nature. Thus, the explicit algorithm does not have significant advantage over the implicit method used in SPICE2. In order to secure convergence for this particular circuit, the internal time step was 12 times smaller than the external time step of 1ns. In other words, the maximum possible internal time increment in computing the algorithm was 0.08 ns. For larger internal time steps, convergence was not obtained.

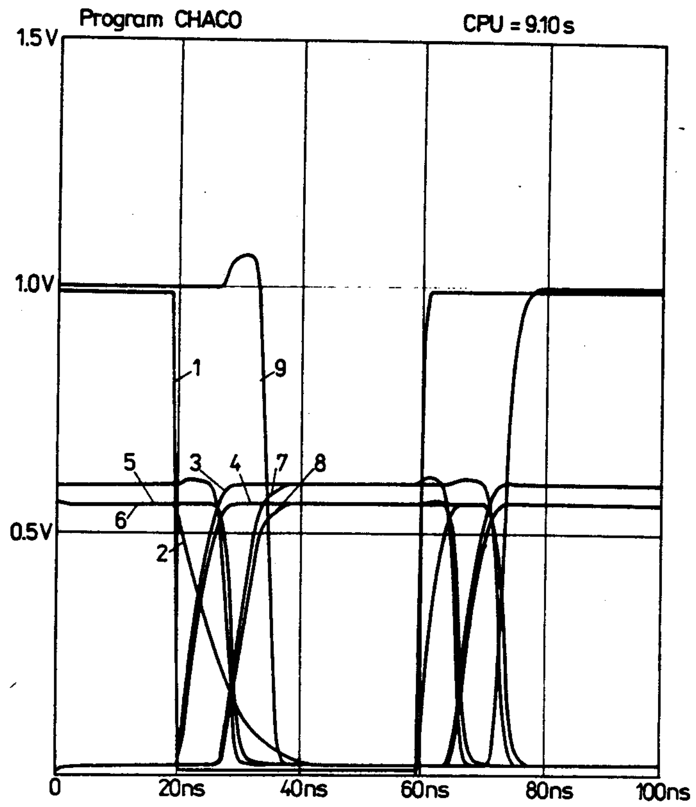

Fig. 6. Computed waveforms of the node voltages for the circuit of Fig. 4 obtained with program CHACO.

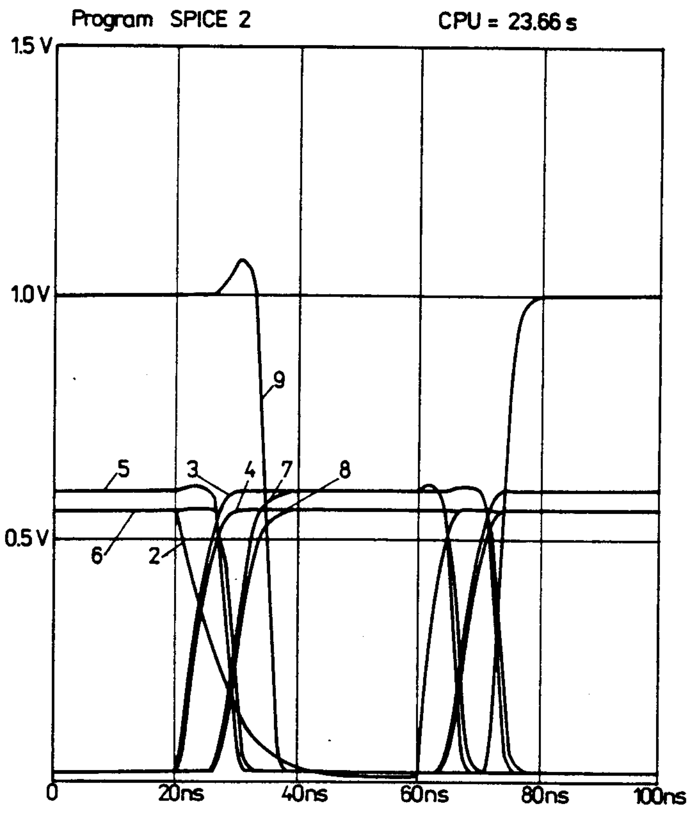

Fig. 7. Computed waveforms of the node voltages for the circuit of Fig. 4 obtained with program SPICE2.

5. SIMULATION OF MOS INTEGRATED CIRCUITS

The method described herein is more suitable for simulation of MOS circuits than bipolar circuits because the nonlinearities in the MOS type are less severe. A cascade of 7 CMOS inverters shown in FIG. 8 was analyzed. The device equivalent circuit is shown in FIG. 9, and the functional dependencies of the parameters are given in TABLE 2. This MOS transistor model includes channel length modulation, body effect due to substrate biasing, and subthreshold conduction as well as normal and inverted operation.

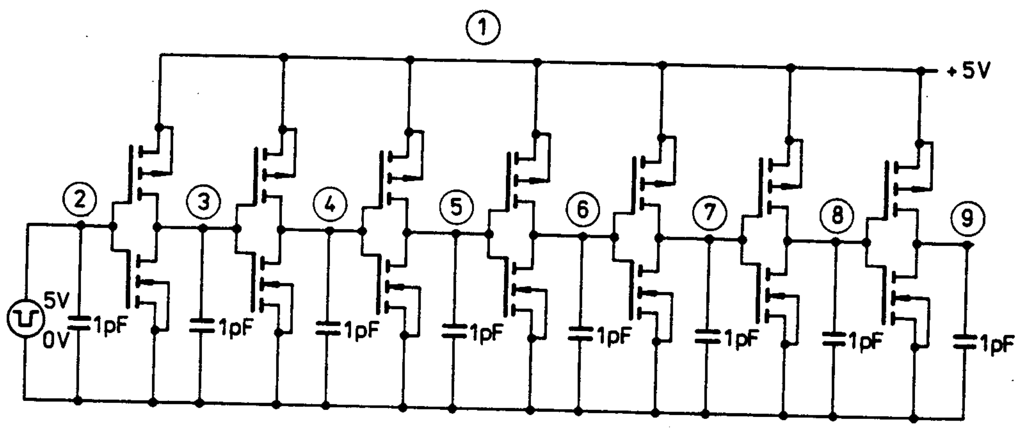

FIG. 8. Cascade of 7 CMOS inverters analyzed with CHACO and SPICE2 programs.

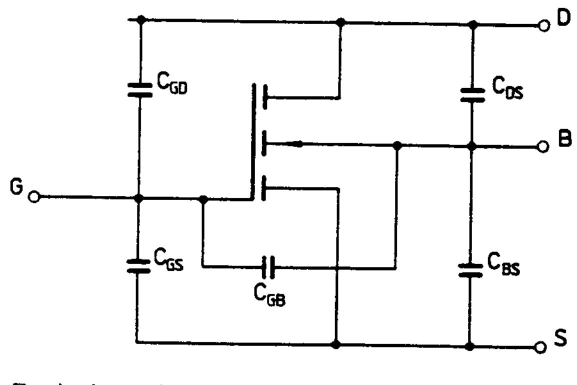

FIG. 9. Equivalent circuit for the N-channel MOS transistor model.

The computed node voltages of the circuit of FIG. 8 are shown in FIG. 10; and results obtained with SPICE2 are shown in FIG. 11. No significant differences are observed. For our method the CPU time was 1.99 seconds, while for SPICE2 it was 184.74 seconds. Since the MOS transistor parameters are inherently not as strongly nonlinear as those of bipolar transistor, an internal time step equal to the external time step (2ns) was used.

Comparison with the SPICE2 program was also performed for a cascade of 4 CMOS inverters and a cascade of 9 CMOS inverters. The CPU times for

TABLE 2

Functional dependence of N-channel MOS transistor parameters

$V_1 = \text{abs}(V_{DS})$
$V_2 = V_{SB}$; if $V_{DB} < V_{SB}$ then $V_2 = V_{DB}$
$V_{Th} = V_{Th0} + G * [(F - V_2)^{0.5} - F^{0.5}]$
$V_3 = V_{GS}$; if $V_{GD} > V_{GS}$ then $V_3 = V_{GD}$
$V_4 = V_3 - V_{Th}$
if $V_4 < Z$ then $V_4 = Z * \exp(V_4/Z - 1)$
$V_5 = V_4$; if $V_1 < V_4$ then $V_5 = V_1$
$I_D = B * (V_4 - 0.5 * V_5 * (1 + K * V_1)$
if $V_{DS} < 0$ then $I_D = -I_D$
$C_{SB} = C_{SB0} * (V_{BS} + F)^{0.5}$
$C_{DB} = C_{DB0} * (V_{BD} + F)^{0.5}$
$C_{GS} = C_{GS0}$; if $V_{GS} > V_{Th}$ then $C_{GS} = C_{GS0} + C_{GX}$
$C_{GD} = C_{GD0}$; if $V_{GD} > V_{Th}$ then $C_{GD} = C_{GD0} + C_{GX}$

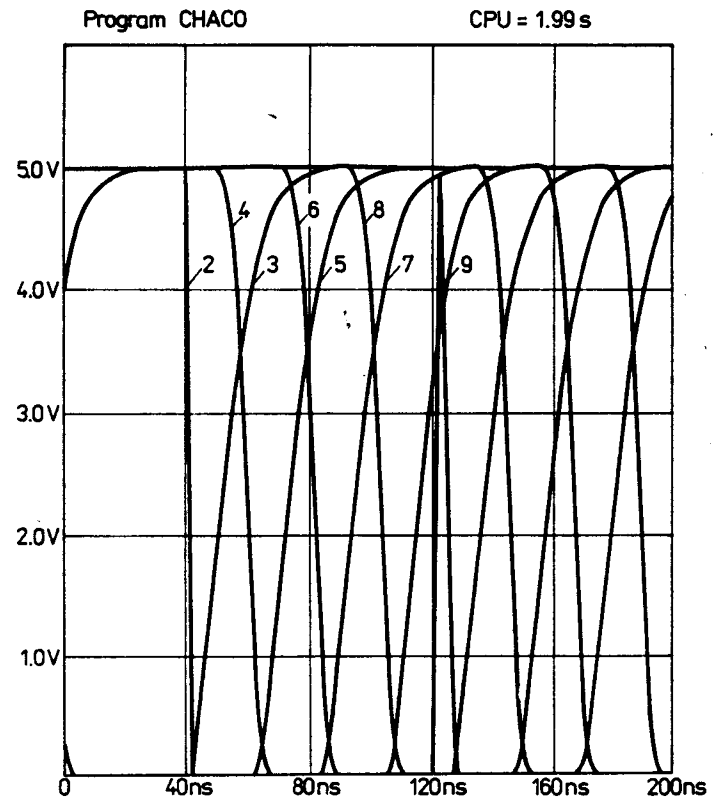

FIG. 10. Computed waveforms of the node voltages for the circuit of Fig. 8 obtained with program CHACO.

various circuits are summarized in TABLE 3. Note that for our method the CPU time is directly proportional to the number of nodes, while the SPICE2 it is approximated by

$$t_{CPU} \sim N^{1.4}. \tag{13}$$

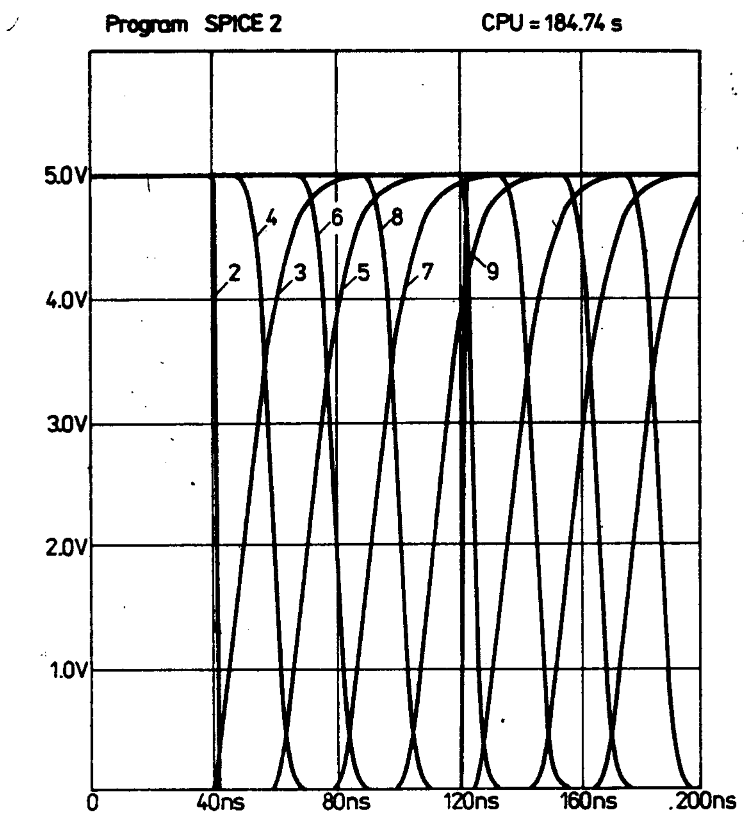

FIG. 11. Computed waveforms of the node voltages for the circuit of Fig. 8 obtained with program SPICE2.

TABLE 3

Comparison of the CPU times on VAX 11/780 computer for several circuits using CHACO and SPICE2 programs

CMOS Inverters	CHACO	SPICE2
4 stages 8 transistors 6 nodes	1.26s	79.18s
7 stages 14 transistors 9 nodes	1.99s	184.84s
9 stages 18 transistors 11 nodes	2.34s	266.99s
97 stages 194 transistors 99 nodes	23.04s	—

Thus, our method is attractive for large circuits. To illustrate this property, the relatively large circuit shown in FIG. 12 was analyzed; the results are shown in FIG. 13. No advantage of the inverter stage similarities has been taken, however. The CPU time for this circuit with 97 inverters (194 transistors) was only

23.04 seconds. It can thus, be predicted that a circuit of 2000 MOS transistor circuit will require about 230 seconds of CPU time, less than the time required by SPICE2 for the 20 transistor circuit previously analyzed. Circuits with transmission gates, with flip-flops and ring oscillators were also analyzed. Generally, CPU time increased proportionally with the number of circuit elements and number of nodes. Apart from this, the circuit structure itself had no effect on CPU time. Since static analysis does precede transient analysis, no problems with convergence occur. In the case of a program such as SPICE2, no static solutions may develop for circuits with a strong positive feedback or oscillatory nature. Our program will always give a solution if a small enough internal time step is chosen. Consider, for example, a 7 stage CMOS ring oscillator, where there are no static solutions. The oscillation will take place as shown in FIG. 14. A disadvantage of the program is that, if initial voltages are not properly chosen, some time is required before the circuit reaches state as can be seen in FIG. 10. If the choice of initial voltages is very poor, this time can be relatively long. This effect is illustrated in FIG. 15 where the transient response for a cascade of 7 inverters is shown, with all initial voltages were 2.5 V. The same time is required in an actual circuit to obtain steady state.

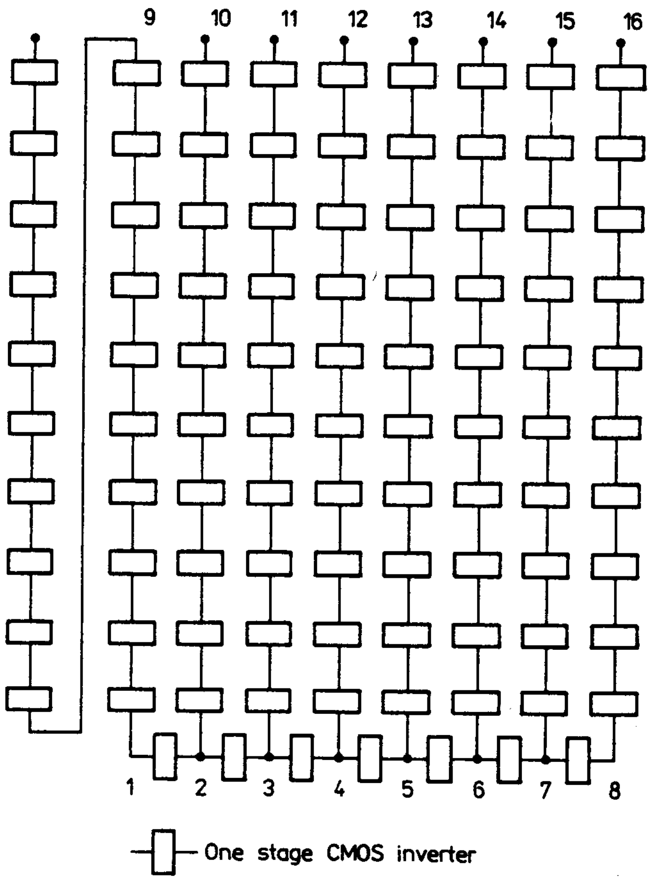

FIG. 12. Block diagram of the circuit containing 97 CMOS inverters analysis with CHACO program.

nsistor
quired
trans-
ıerally,
s and
ı CPU
; with
lutions
ıature.
e step
there
A dis-
hosen,
IG. 10.
long.
ascade
ime is

rogram.

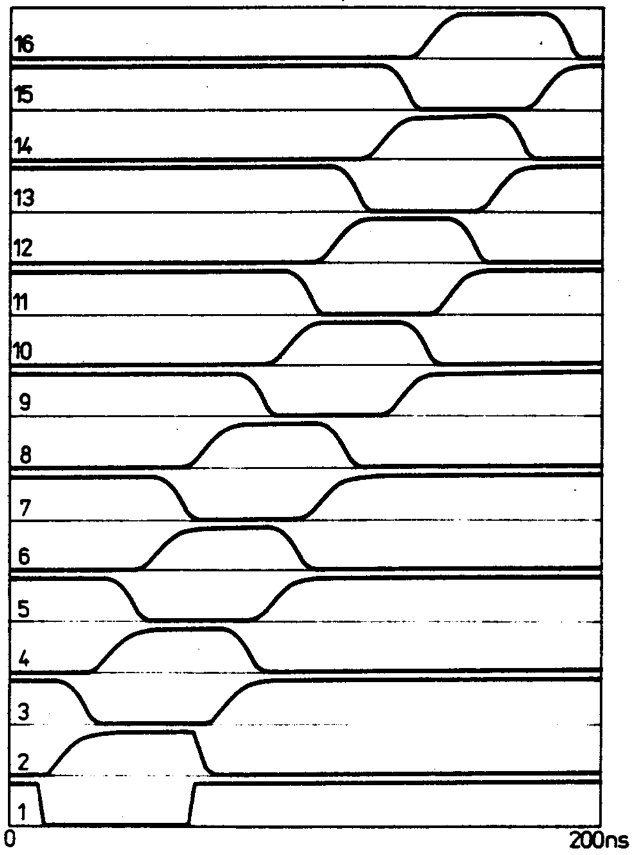

FIG. 13. Computed waveforms of some of node voltages for the circuit of Fig. 12.

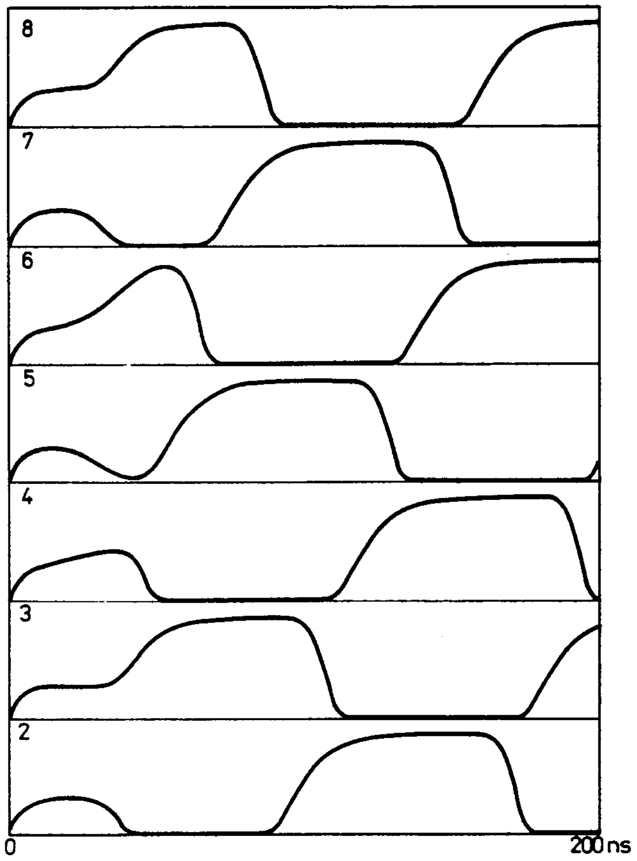

FIG. 14. Computed transient waveforms of the node voltages for the ring oscillator of 7 CMOS inverters, after power is switched "on", obtained with CHACO program.

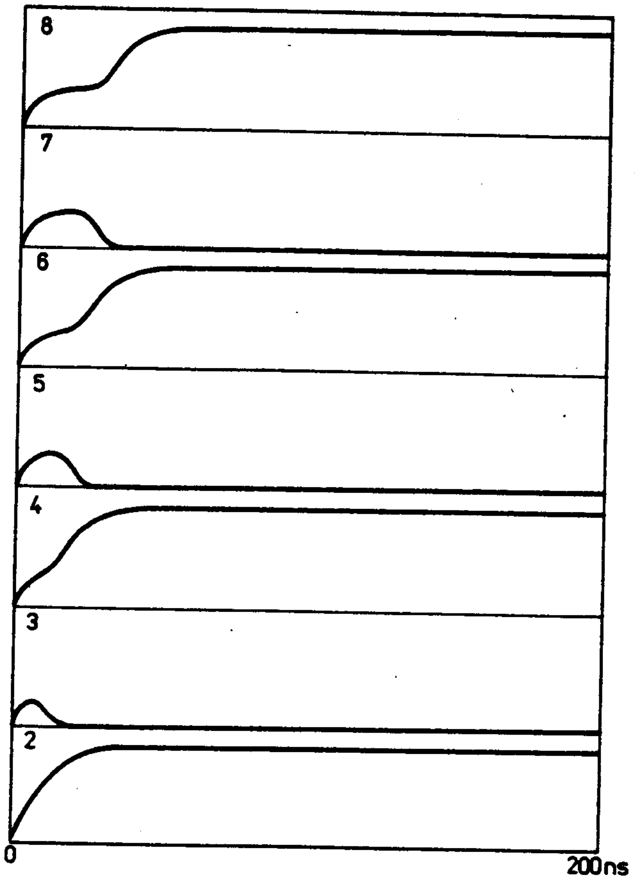

FIG. 15. Computed transient waveforms of node voltages, before circuit will reach steady-state, if all initial node voltages were setted to be 2.5 V.

6. CONCLUSION

The method we have described for the transient analysis of nonlinear networks of resistors, capacitors and sources requires a CPU time proportional to the circuit complexity in contrast to classical matrix methods CPU, where time is proportional to N^2 or more sophisticated approaches which show dependence on N^m where m is in the range of 1.2 to 1.5.

While the Gauss-Seidel iteration procedure used in this paper has the advantage of simplicity, it is not very suitable for circuits which have large inter-node capacitances. However, it should be noted that, for such cases, a significant reduction of computing time is possible.

In the digital integrated circuits the time constances of all nodes are of similar order (strongly related to the delay time of a single stage). Therefore proposed algorithm is fit very nicely for the digital *IC* analysis and simulation. In case of analog *ICs* the smallest time constant of whole circuit determines the internal time step and therefore in some circuits large number of time steps and long computing time can be required.

Further reduction of CPU time can be expected if means are used to take advantage of signal latency (temporary sparsity). The algorithm is so structured that it is relatively simple to omit computation at inactive nodes when this is warranted.

REFERENCES

1. A. E. RUEHLI, G. S. DITLOW, Circuit analysis, logic simulation and design verification for VLSI, January 1983, Proc. IEEE, **71**, 34–48.
2. G. D. HACHTEL, R. K. BRAYTON, F. GUSTAVSON, The sparse tableau approach to network analysis and design. IEEE Trans. Circuit Theory. January 1971, **CT-18**, 101–113.
3. C. HO, A. E. RUEHLI, P. A. BRENNAN, The modified nodal approach to network analysis, IEEE Trans. Circuits and Systems, June 1975, **CAS-22**, 504–509.
4. E. LELARASMEE, A. E. RUEHLI, A. L. SANGIOVANNI-VINCENTELLI, The waveform relaxation method for time-domain analysis of large-scale integrated circuits, IEEE Trans. CAD Integrated Circuit and Systems, July 1982, **CAD-1**, 131–145.
5. O. A. PALUSIŃSKI, M. W. GUARINI, Improving the efficiency of relaxation based circuit simulated via functional linearization and polynomial representation of solutions, IEEE Conf. on IC CAD, Santa clara, CA, September 12–15, 1983.
6. L. W. NAGEL, SPICE2: computer program to simulate semiconductor circuits, University of California. Berkeley. ERL Memo. ERL-M520. May 1975.
7. A. VLADIMIRESCU, K. ZHANG, A. R. NEWTON, D. O. PEDERSON, A. SANGIOVANNI-VINCENTELLI, SPICE Version 2G User's Guide, University of California, Berkeley, Tech. Memo, August 10, 1981.
8. B. M. WILAMOWSKI, D. J. HAMILTON, Z. J. STASZAK, Digital integrated circuit transient analysis program, Proc. Inter. Conf. "Modelling and simulation", Minneapolis, August 13–17, 1984, **2**, 99–112.

STRESZCZENIE

ANALIZA CYFROWYCH UKŁADÓW SCALONYCH PRZY ZASTOSOWANIU ZASADY ZACHOWANIA ŁADUNKU

Przedstawiono szybki i efektywny algorytm do analizy układów scalonych, oparty na zasadzie zachowania ładunku. Ten algorytm posługuje się jawną metodą obliczeń i dlatego czas jest wprost proporcjonalny do złożoności obwodu. Dla średniej wielkości obwodów MOS czas obliczeń jest blisko 100 razy krótszy, niż wymaga tego program SPICE2. W przypadku bardziej złożonych układów algorytm obliczeń może być bardziej efektywny.

РЕЗЮМЕ

АНАЛИЗ ЦИФРОВЫХ ИНТЕГРАЛЬНЫХ СХЕМ С ПРИМЕНЕНИЕМ ПРИНЦИПА СОХРАНЕНИЯ ЗАРЯДА

Представлен быстрый и эффективный алгоритм анализа интегральных схем использующий принцип сохранения заряда. В алгоритме применён явный метод рассчётов, благодаря чему время прямо пропорционально степени сложности схемы. Для схем МДП средней сложности время рассчётов почти в 100 раз короче чем в случае программы SPICE 2. Для более сложных схем алгоритм рассчётов может быть более эффективным.