THE WAVEFORM RELAXATION METHOD FOR TIME DOMAIN ANALYSIS OF LARGE SCALE INTEGRATED CIRCUITS

by

E. Lelarasmee, A. E. Ruehli and A. L. Sangiovanni-Vincentelli

Memorandum No. UCB/ERL M81/75

26 June 1981

THE WAVEFORM RELAXATION METHOD FOR TIME DOMAIN ANALYSIS OF LARGE SCALE INTEGRATED CIRCUITS

by

Ekachai Lelarasmee, Albert E. Ruehli and Alberto L. Sangiovanni-Vincentelli

Memorandum No. UCB/ERL M81/75

26 June 1981

ELECTRONICS RESEARCH LABORATORY

College of Engineering
University of California, Berkeley
94720

The Waveform Relaxation Method for Time Domain Analysis of Large Scale Integrated Circuits.[1]

Ekachai Lelarasmee, Albert E. Ruehli and Alberto L. Sangiovanni-Vincentelli

Abstract

The Wavefom Relaxation Method (WRM) is an iterative method for analyzing nonlinear dynamical systems in the time domain. The method, at each iteration, decomposes the system into several dynamical subsystems each of which is analyzed for the entire given time interval. Sufficient conditions for convergence of the WR method are proposed and examples in MOS digital integrated circuits are given to show that these conditions are very mild in practice. Theoretical and computational studies show the method to be efficient and reliable.

[1]This research was supported in part by the Joint Services Electronics Program Contract F49620-79-C-0178, Defense Advanced Research Projects Agency (DOD) Contract N00039-K-0251 and the National Science Foundation Grant ECS-79-13148.

E. Lelarasmee and A. L. Sangiovanni-Vincentelli are with the Department of Electrical Engineering and Computer Sciences, and the Electronics Research Laboratory, University of California, Berkeley, CA 94720.

A. E. Ruehli is with IBM T. J. Watson Research Center, Yorktown Heights, NY 10598.

I. Introduction.

The spectacular growth in the scale of integrated circuits being designed in the VLSI era has generated the need for new methods of circuit simulation. "Standard" circuit simulators, such as SPICE2 [1] and ASTAP [2], simply take too much CPU time and too much storage to analyze a VLSI circuit. These standard circuit simulators are essentially based on three techniques:

(i) Stiffly stable implicit integration methods, such as Backward Euler formula, for obtaining a system of nonlinear algebraic equations from the original system of nonlinear algebraic-differential equations describing the behavior of the circuit.

(ii) Newton-Raphson iteration to linearize the system of nonlinear algebraic equations of (i).

(iii) Sparse gaussian elimination to solve the system of linear algebraic equations of (ii).

New simulators, such as MOTIS [3], SPLICE [4,5], DIANA [6] and MACRO [7], in their quest for speed have rejected one or more of the principal features of standard simulators. In particular MOTIS and the timing simulation part of the mixed-mode simulator SPLICE use Backward Euler integration and relaxation techniques to decompose and solve the system of nonlinear algebraic equations, eliminating the need for sparse gaussian elimination. The decomposition achieved by relaxation allows the use of selective trace algorithms for exploiting the "latency" of VLSI circuits, a fundamental feature of SPLICE.

In MOTIS and SPLICE, the relaxation process is not carried out to convergence: only one "sweep" of relaxation is taken [3,4,5]. Therefore the numerical properties, such as stability and accuracy, of the Backward Euler integration no longer hold. In [8] these properties are examined. It turns out that the types of circuits which can be dealt with by these techniques are rather limited: MOS circuits with quasi-unidirectional device models and a grounded capacitor to every node. Moreover, in some cases, the step size has to be chosen very small to avoid instability and inaccuracy of the solution. The Waveform Relaxation Method (WRM) is introduced in this paper for the analysis of large scale circuits. The basic idea here is to apply relaxation directly to the system of nonlinear algebraic-differential equations describing the circuit. As a result, the system is decomposed into decoupled subsystems of algebraic-differential equations each of which can then be solved by means of standard techniques, i.e. stiffly stable integration method and Newton-Raphson iteration. The decomposition achieved allows the latency to be exploited in the most natural way.

This paper is organized as follows. In section 2, the method is presented and its basic features discussed. In section 3, some circuit examples are given to show how the method works on practical cases. In section 4, the convergence properties of the WRM are rigorously proven. In section 5, the method is specialized to the analysis of VLSI MOS circuits. Two WRM algorithms are described and their convergence properties are rigorously proven. In section 6, the computational aspects of WRM are studied from an experimental WRM circuit simulator, the results being compared to those obtained from a standard simulator, SPICE. Finally, section 7 contains discussions on how to further increase the computational effiency of WRM. ▪

II. Mathematical Formulation and the WRM Algorithm Model.

We consider dynamical systems which can be described by a system of mixed implicit algebraic-differential equations of the form:

$$F(\dot{y}(t), y(t), u(t)) = 0 \tag{2.1a}$$

$$E(y(0) - y_0) = 0 \tag{2.1b}$$

where $y(t) \in \mathbb{R}^p$ is the vector of unknown variables at time t, $\dot{y}(t) \in \mathbb{R}^p$ is the time derivative of y at time t, $u(t) \in \mathbb{R}^r$ is the vector of input variables at time t, $y_0 \in \mathbb{R}^p$ is the given initial value of y, $F : \mathbb{R}^p \times \mathbb{R}^p \times \mathbb{R}^r \to \mathbb{R}^p$ is a continuous function, and $E \in \mathbb{R}^{n \times p}$, $n \le p$ is a matrix of rank n such that $Ey(t)$ is the state of the system at time t.

Note that for any lumped nonlinear dynamical circuit, (2.1) is a general formulation which subsumes all other well known formulations such as Nodal Analysis [11], Modified Nodal Analysis [12], Hybrid Analysis [13] and Tableau Analysis [14]. To simplify the notations, we shall drop the time argument whenever there is no ambiguity. Hence we rewrite (2.1) as

$$F(\dot{y}, y, u) = 0 \tag{2.2a}$$

$$E(y(0) - y_0) = 0 \tag{2.2b}$$

The general structure of a WRM algorithm for analyzing (2.2) in a given time interval $[0, T]$ consists of two major processes, namely the *assignment-partition* process and the *relaxation* process.

In the assignment-partition process, each unknown variable is assigned to an equation of (2.2a) in which it is involved. However, no two variables can be assigned to the same equation. Then (2.2a) is partitioned into m disjoint*

*There are cases in which the algorithm has better convergence properties if the subsystems are nondisjoint. For such cases, we can consider the nondisjoint subsystems as being obtained from partitioning an augmented system of equations with an augmented set of unknown variables.

subsystems of equations, each of which may have only differential equations or only algebraic equations or both. Then, without loss of generality, we can rewrite (2.2) after being processed by the assignment-partition process as follows:

$$\begin{bmatrix} F_1(\dot{y}_1, y_1, d_1, u) \\ \vdots \\ F_m(\dot{y}_m, y_m, d_m, u) \end{bmatrix} = 0 \qquad (2.3a)$$

$$E(y(0) - y_0) = 0 \qquad (2.3b)$$

where, for each $i = 1,2,\ldots,m$, $y_i \in \mathbb{R}^{p_i}$ is the subvector of unknown variables assigned to the i-th partitioned subsystem, $F_i : \mathbb{R}^{p_i} \times \mathbb{R}^{p_i} \times \mathbb{R}^{2p-2p_i} \times \mathbb{R}^r \to \mathbb{R}^{p_i}$ is a continuous function, and

$$\begin{aligned} d_i = \ \mathrm{col}\,(&y_1, \ldots, y_{i-1}, y_{i+1}, \ldots, y_m, \\ &\dot{y}_1, \ldots, \dot{y}_{i-1}, \dot{y}_{i+1}, \ldots, \dot{y}_m)^* \end{aligned} \qquad (2.3c)$$

With respect to the i-th subsystem, y_i and y_j, $j \neq i$ are called vectors of *endogeneous* and *exogeneous* variables respectively. It is clear that if the vectors d_i, $i = 1,2,\ldots,m$, are treated as inputs, then (2.3a) can be solved by solving m independent subsystems. Therefore they are called the *decoupling* vectors of the subsystems. This gives rise to the notion of the decomposed system as given in the following definition.

Definition 2.2. The *decomposed system* associated with an assignment-partition process applied to (2.2) consists of m independent subsystems, called *decomposed subsystems*, each of which is described by

$$F_i(\dot{\tilde{y}}_i, \tilde{y}_i, \tilde{u}_i, u) = 0 \qquad (2.4a)$$

* $\mathrm{col}\,(a, b) \triangleq \begin{bmatrix} a \\ b \end{bmatrix}$

$$E_i(\tilde{y}_i(0) - y_i(0)) = 0 \tag{2.4b}$$

where $\tilde{y}_i \in \mathbb{R}^{p_i}$ is the vector of unknown variables, $y_i(0) \in \mathbb{R}^{p_i}$ is the subvector of the given initial values, $u \in \mathbb{R}^r$ is the vector of the given inputs, $\tilde{u}_i \in \mathbb{R}^{2p-2p_i}$ is the vector of the decoupling inputs, $E_i \in \mathbb{R}^{n_i \times p_i}$, $n_i \le p_i$ is a matrix of rank n_i such that $E_i\tilde{y}_i$ is a state vector of the i-th decomposed subsystem, and F_i is the continuous function defined in (2.3a). ■

The relaxation process is an iterative process. For simplicity, we shall consider two most commonly used types of relaxation namely the *Gauss-seidel* (GS) and the *Gauss-Jacobi* (GJ) relaxation. The relaxation process starts with an initial guess of the waveform solutions of the original dynamical equations (2.2) in order to initialize the approximated waveforms of the decoupling vectors. During each iteration, each decomposed subsystem is solved for its endogeneous variables in the given time interval [0,T] by using the approximated waveform of its decoupling vector. For the GS relaxation, the waveform solutions obtained by solving one decomposed subsystem are immediately used to update the approximated waveforms of the decoupling vectors of the other subsystems. For the GJ relaxation, all waveforms of the decoupling vectors are updated at the beginning of the next iteration. The iterative process is carried out repeatedly until satisfactory convergence is achieved.

The algorithm shown below sets up a general structure of all WRM algorithms which we shall be working on throughout this paper. Therefore it is called the WRM Algorithm Model. Let the superscript index k denote the iteration count. Then the WRM Algorithm Model can be formally described as follows:

WRM Algorithm Model 2.1

Step 0: (Assignment-partition process)

Assign the unknown variables to equations in (2.2) and partition (2.2) into m subsystems of equations as given by (2.3).

Step 1: (Initialization of the relaxation process)

Set k = 1 and guess an initial waveform $(y^0(t)\,;\, t \in [0,T])$ such that $y^0(0) = y(0)$.

Step 2: (Analyzing the decomposed system at the k-th iteration)

For each $i = i,2,\ldots,m$, set

$$d_i^k = \text{col}\,(y_1^k, \ldots, y_{i-1}^k, y_{i+1}^{k-1}, \ldots, y_m^{k-1}, \dot{y}_1^k, \ldots, \dot{y}_{i-1}^k, \dot{y}_{i+1}^{k-1}, \ldots, \dot{y}_m^{k-1})$$

for the GS relaxation, or

$$d_i^k = \text{col}\,(y_1^{k-1}, \ldots, y_{i-1}^{k-1}, y_{i+1}^{k-1}, \ldots, y_m^{k-1}, \dot{y}_1^{k-1}, \ldots, \dot{y}_{i-1}^{k-1}, \dot{y}_{i+1}^{k-1}, \ldots, \dot{y}_m^{k-1})$$

for the GJ relaxation, and solve for $(y_i^k(t)\,;\, t \in [0,T])$ from

$$F_i(\dot{y}_i^k, y_i^k, d_i^k, u) = 0 \tag{2.5a}$$

$$E_i(y_i^k(0) - y_i(0)) = 0 \tag{2.5b}$$

Step 3: (Iteration)

Set $k = k+1$ and go to step 2. ■

Remarks.

1) A simple guess for $(y^0(t)\,;\, t \in [0,T])$ is $y^0(t) = y(0)$ for all $t \in [0,T]$.

2) In the actual implementation, the relaxation process stops the iteration when the difference between $(y^k(t)\,;\, t \in [0,T])$ and $(y^{k-1}(t)\,;\, t \in [0,T])$, i.e. $\max_{t \in [0,T]} \|y_k(t) - y_{k-1}(t)\|$, is sufficiently small.

3) In analogy to the classical relaxation methods for solving linear or non-linear algebraic equations (see [15] for examples), it is possible to modify a WRM algorithm by using a relaxation parameter $\omega \in (0,2)$. With ω, the iteration equation (2.5) is modified to yield

$$F_i(\dot{\tilde{y}}_i^k, \tilde{y}_i^k, d_i^k, u) = 0 \tag{2.6a}$$

$$E_i(\tilde{y}_i^k(0) - y_i(0)) = 0 \tag{2.6b}$$

$$y_i^k = y_i^{k-1} + \omega(\tilde{y}_i^k - y_i^{k-1}) \tag{2.6c}$$

4) Note the following two important characteristics of the WRM Algorithm Model 2.1.

   a) The analysis of the original system is decomposed into the independent analysis of m subsystems.

   b) The relaxation process is carried out on the entire waveforms, i.e. during each iteration each subsystem is individually analyzed for the entire given time interval $[0, T]$. ■

III. Examples and their physical interpretation.

In this section, we shall use a few specific examples to demonstrate the applications of the WRM Algorithm Model 2.1 in the analysis of lumped dynamical circuits and to give the circuit interpretation of the decompostion. Different formulations of the dynamical equations will be used to illustrate the resulting decompositions.

The first example is a ring oscillator shown in Fig. 3.1a. The dynamical behavior of the circuit is described by

$$f_1(\dot{v}_1, v_1, \dot{v}_2, \dot{v}_3, v_3, \dot{u}, u) = 0 \qquad (3.1a)$$

$$f_2(\dot{v}_1, v_1, \dot{v}_2, v_2, \dot{v}_3) = 0 \qquad (3.1b)$$

$$f_3(\dot{v}_1, \dot{v}_2, v_2, \dot{v}_3, v_3) = 0 \qquad (3.1c)$$

where v_1, v_2, v_3 are the node voltages, u is the triggering input voltage and the functions f_1, f_2, f_3 represent the sums of all currents entering (or leaving) nodes 1,2,3 respectively. Let v_1, v_2 and v_3 be assigned to (3.1a), (3.1b) and (3.1c) respectively and let the system be partitioned into 3 subsystems consisting of {(3.1a)}, {(3.1b)} and {(3.1c)}. Applying the WRM Algorithm Model 2.1, the k-th iteration of the corresponding GJ-WRM algorithm is given by

$$\begin{aligned}
f_1(\dot{v}_1^k, v_1^k, \dot{v}_2^{k-1}, \dot{v}_3^{k-1}, v_3^{k-1}, \dot{u}, u) &= 0; \quad v_1^k(0) = v_1(0)\\
f_2(\dot{v}_1^{k-1}, v_1^{k-1}, \dot{v}_2^k, v_2^k, \dot{v}_3^{k-1}) &= 0; \quad v_2^k(0) = v_2(0)\\
f_3(\dot{v}_1^{k-1}, \dot{v}_2^{k-1}, v_2^{k-1}, \dot{v}_3^k, v_3^k) &= 0; \quad v_3^k(0) = v_3(0)
\end{aligned}$$

where $v_1(0), v_2(0), v_3(0)$ are the given initial values of v_1, v_2, v_3 respectively. The circuit interpretation of the decomposed system at the k-th iteration is shown in Fig. 3.1b. ■

The second example is a dynamic shift register shown in Fig. 3.2a. The dynamical behavior of this circuit is described by

$$f_1(\dot{v}_1, v_1, i, \dot{u}_1, u_1, \dot{u}_2) = 0 \tag{3.2a}$$

$$f_2(v_1, \dot{v}_2, v_2, \dot{v}_3, u_2, \dot{u}_2) = 0 \tag{3.2b}$$

$$i - f_{ds}(v_1, v_2, u_2) = 0 \tag{3.2c}$$

$$f_3(\dot{v}_2, v_2, \dot{v}_3, v_3) = 0 \tag{3.2d}$$

where f_{ds} is the function describing the drain-to-source current i of the pass transistor. The rest of the notations are the same as in the first example. Let v_1, v_2, i and v_3 be assigned to (3.2a), (3.2b), (3.2c) and (3.2d) respectively and let the system be partitioned into 3 subsystems consisting of {(3.2a)}, {(3.2b),(3.2c)} and {(3.2d)}. Applying the WRM Algorithm Model 2.1, the k-th iteration of the corresponding GS-WRM algorithm is given by

$$\begin{aligned} f_1(\dot{v}_1^k, v_1^k, i^{k-1}, \dot{u}_1, u_1, \dot{u}_2) &= 0; \quad v_1^k(0) = v_1(0) \\ f_2(v_1^k, \dot{v}_2^k, v_2^k, \dot{v}_3^{k-1}, u_2, \dot{u}_2) &= 0; \quad v_2^k(0) = v_2(0) \\ i^k - f_{ds}(v_1^k, v_2^k, u_2) &= 0 \\ f_3(\dot{v}_2^k, v_2^k, \dot{v}_3^k, v_3^k) &= 0; \quad v_3^k(0) = v_3(0) \end{aligned}$$

The circuit interpretation of the decomposed system at the k-th iteration is shown in Fig. 3.2b. ■

The third example is shown in Fig. 3.3a and its dynamical behavior is described by

$$c_1\dot{v}_1 - i_1 = 0 \tag{3.3a}$$

$$c_2\dot{v}_2 - i_2 = 0 \tag{3.3b}$$

$$c_3\dot{v}_3 - i_3 = 0 \tag{3.3c}$$

$$v_3 - v_1 + v_2 = 0 \tag{3.3d}$$

$$\frac{v_1}{R_1} + i_1 + i_3 - u = 0 \tag{3.3e}$$

$$\frac{v_2}{R_2} + i_2 - i_3 = 0 \tag{3.3f}$$

Let $v_1, v_2, i_3, v_3, i_1, i_2$ be assigned to (3.3a) through (3.3f) respectively and let the system be partitioned into 3 subsystems consisting of $\{(3.3a),(3.3e)\}$, $\{(3.3b),(3.3f)\}$ and $\{(3.3c),(3.3d)\}$. Note that we cannot assign v_1, v_2, v_3 to (3.3a), (3.3b), (3.3c) respectively since one of them has to be assigned to (3.3d). Applying the WRM Algorithm Model 2.1, the k-th iteration of the corresponding GJ-WRM algorithm is given by

$$c_1\dot{v}_1^k - i_1^k = 0; \quad v_1^k(0) = v_1(0) \qquad (3.4a)$$

$$\frac{v_1^k}{R_1} + i_1^k + i_3^{k-1} - u = 0 \qquad (3.4b)$$

for the first subsystem,

$$c_2\dot{v}_2^k - i_2^k = 0; \quad v_2^k(0) = v_2(0) \qquad (3.4c)$$

$$\frac{v_2^k}{R_2} + i_2^k - i_3^{k-1} = 0 \qquad (3.4d)$$

for the second subsystem, and

$$c_3\dot{v}_3^k - i_3^k = 0 \qquad (3.4e)$$

$$v_3^k - v_1^{k-1} + v_2^{k-1} = 0 \qquad (3.4f)$$

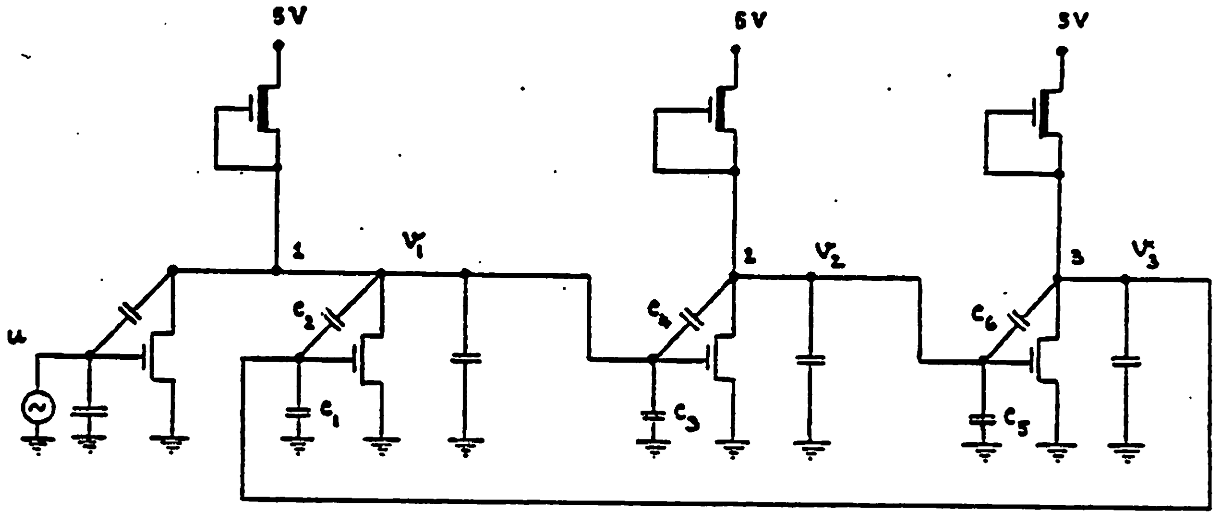
for the third subsystem.

Note that v_3^k is not initialized at time $t = 0$ since it is not a state variable of the decomposed system. However, one can check that $v_3^k(0) = v_1(0) - v_2(0)$ for all k. The circuit interpretation of the decomposed system at the k-th iteration is shown in Fig. 3.3b. ■

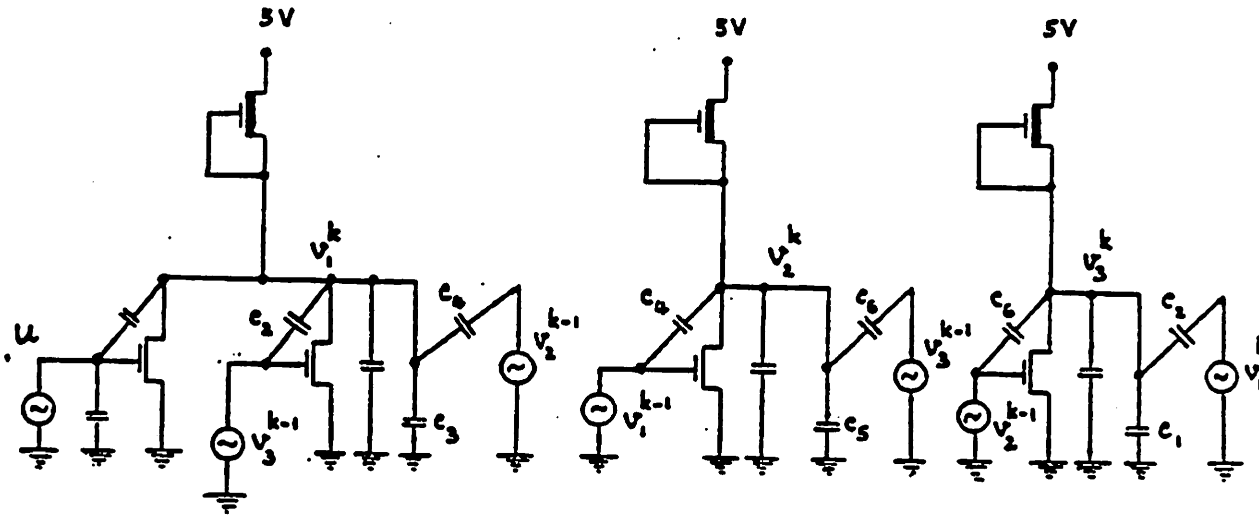
Fig. 3.1

a) A MOS ring oscillator.

b) The circuit interpretation of its decomposed circuit at the k-th iteration of the GJ-WRM algorithm.



(a)

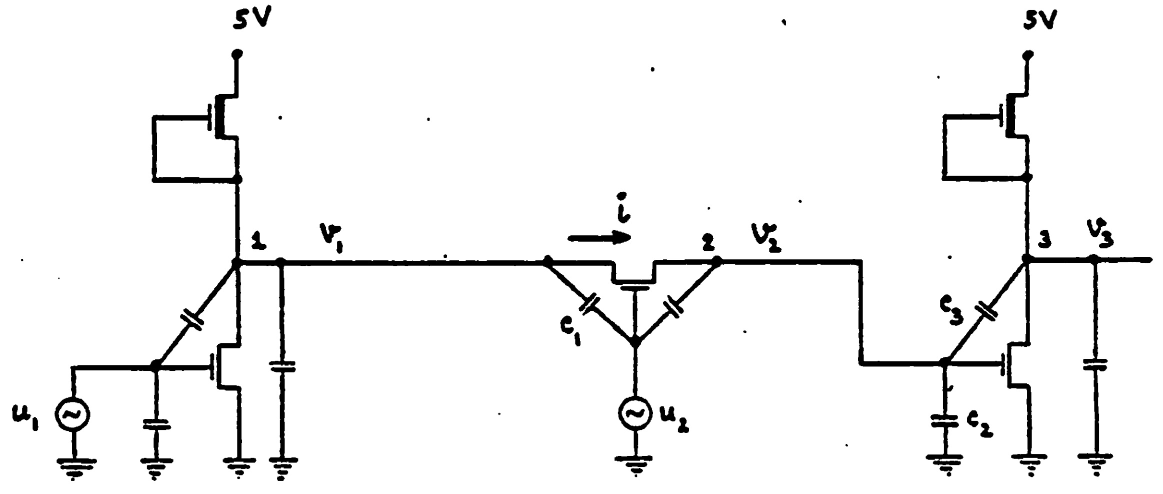


(b)

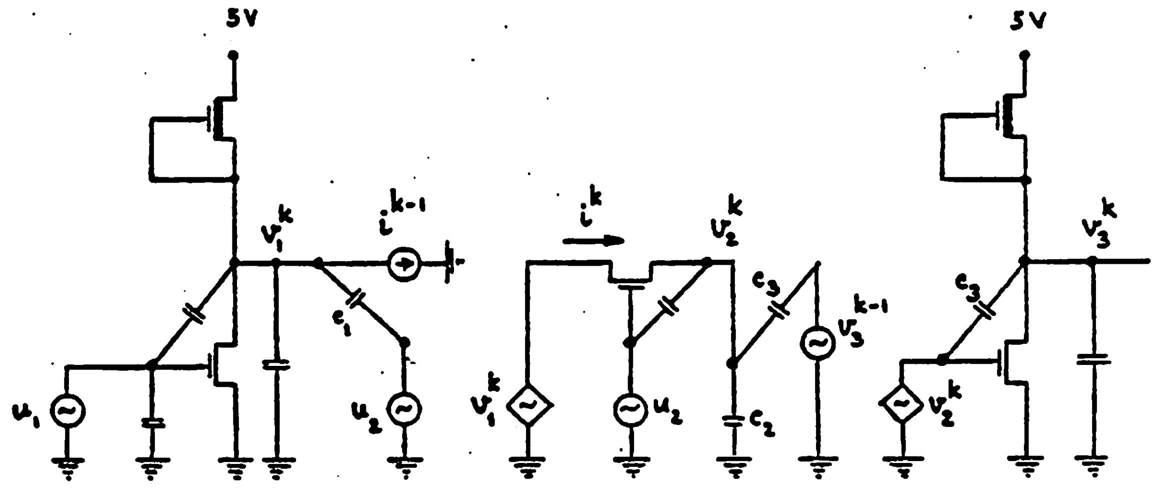
Fig. 3.2

a) A MOS dynamic shift register.

b) The circuit interpretation of its decomposed circuit at the k-th iteration of the GS-WRM algorithm.



(a)

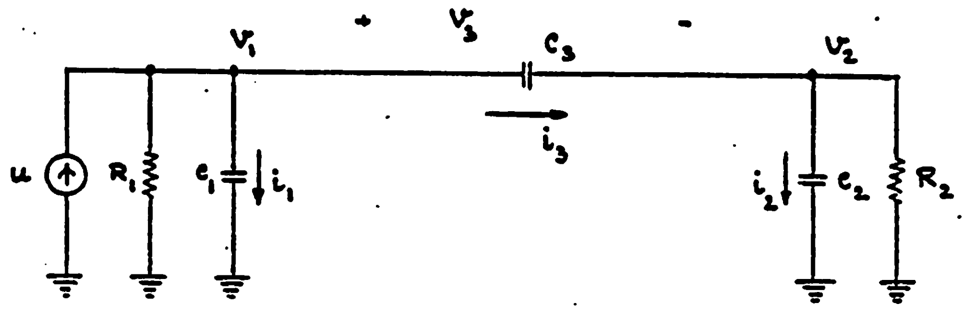


(b)

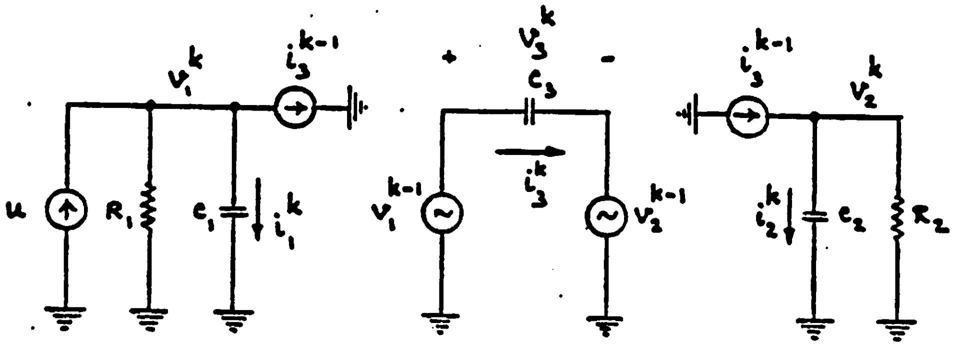
Fig. 3.3

a) A simple RC circuit.

b) The circuit interpretation of its decomposed circuit at the k-th iteration of the GJ-WRM algorithm.



(a)



(b)

IV. Convergence of WRM.

In this section we shall derive sufficient conditions to guarantee that the WRM Algorithm Model 2.1 converges, i.e. it generates a converging sequence of iterated solutions whose limit satisfies the dynamical equations (2.2a) and the given initial conditions (2.2b). As in most literature on iterative methods, much results on the convergence of any iterative method are stated when the iteration equation can be written in an explicit form, i.e. the iterated variables can be written as functions of their previous values and other non-iterated variables. Hence, we shall later introduce an explicit form of the iteration equation (2.5) of the WRM Algorithm Model 2.1 and refer to it as the canonical representation of the algorithm. Sufficient conditions to ensure the existence of the canonical WRM algorithm will also be given. These conditions are basically related to how the assignment-partition process treats the state variables of the original dynamical system and give rise to the following definition of the compatibility of an assignment-partition process.

Definition 4.1 An assignment-partition process is said to be *compatible with a given dynamical system* (2.2) if the choice of the state vector of its associated decomposed system (2.4) is also a valid choice of the state vector of the original system (2.2), i.e. there exist matrices $E_1, E_2, \ldots, E_m$ and E as defined in (2.4) and (2.1) such that

$$\text{col}\,[E_1 y_1, E_2 y_2, \ldots, E_m y_m] = Ey$$

Any WRM algorithm which uses a compatible assignment-partition process is called a *compatible WRM algorithm.* ∎

Note that in the WRM Algorithm Model 2.1, only the state variables of the decomposed system are initialized at time $t = 0$ by the given initial conditions of the original system. The initial values of other variables of the decomposed system (except the inputs u) can vary from one iteration to the other. However, if the algorithm is compatible, (2.2b) will be satisfied at every iteration. Hence, when the sequence of the iterated solutions converges, its limit will also satisfy (2.2b). Due to the definition of the state variables, this also implies that, for a compatible WRM algorithm, when the sequence of iterated solutions converges, the sequence of initial values of the non-state variables also converges to the values given by the initial conditions of the original system. On the other hand, for a non-compatible WRM algorithm, it is possible that the sequence of iterated solutions converges to the limit which satisfies the original dynamical equations with another set of initial conditions. An example of this phenomenon is given in the appendix. The same example also indicates that non-compatible WRM algorithms tend to have poor convergence properties. Therefore, throughout this paper, we consider only compatible WRM algorithms.

In most engineering designs, the assignment-partition process can be guided by the physical interpretation of the states of the dynamical system. For example, the state variables of lumped electrical circuits are usually the voltages across the capacitors and the currents through the inductors. Based on this physical interpretation, a designer can select a compatible assignment-partition process as demonstrated in the examples of section III. However, for the most general case, there is a need for a systematic method for finding a compatible assignment-partition process, especially when the number of equations is large. Such a method is discussed in [18].

We now give the definition of the canonical forrr of a WRM algorithm, followed by conditions which guarantee that the WRM Algorithm Model 2.1 can be tranformed into a canonical form. It should be noted that we do not have to perform the tranformation explcitly in order to implement the WRM Algorithm Model 2.1. Based on its canonical form, we can state sufficient convergence conditions of the WRM Algorithm Model 2.1 in a simplified form as given in Theorem 4.5.

Definition 4.2. A *canonical* WRM algorithm is characterized by the following iteration equations.

$$\dot{x}^k = f(x^k, x^{k-1}, \dot{x}^{k-1}, z^{k-1}, u) \tag{4.1a}$$

$$z^k = g(x^k, x^{k-1}, \dot{x}^{k-1}, z^{k-1}, u) \tag{4.1b}$$

where $x \in R^n$, $z \in R^l$, $u \in R^r$ and f, g are continuous functions. ■

Lemma 4.3 Assum~~ming~~e that for each decomposed subsystem described by (2.4), there exist continuously differentiable functions $\tilde{f}_i$ and $\tilde{g}_i$ and a nonsingular matrix $\begin{bmatrix} E_i \\ D_i \end{bmatrix} \in R^{p_i \times p_i}$ such that (2.4) can be rewritten as

$$\dot{x}_i = \tilde{f}_i(x_i, \tilde{u}_i, u) \tag{4.2a}$$

$$z_i = \tilde{g}_i(x_i, \tilde{u}_i, u) \tag{4.2b}$$

$$x_i(0) = E_i(y_i(0)) \tag{4.2c}$$

$$\begin{bmatrix} x_i \\ z_i \end{bmatrix} = \begin{bmatrix} E_i \\ D_i \end{bmatrix} y_i \tag{4.2d}$$

where $x_i \in R^{n_i}$ and $z_i \in R^{p_i - n_i}$, i.e. each decomposed subsystem has a state-equation representation. If the WRM Algorithm Model 2.1 is compatible, then it can be tranformed into a canonical WRM algorithm. ■

Instead of proving the above Lemma, we give the following example to show how such a tranformation can be done for a simplified case. The formal proof of this Lemma is given in [16].

Example 4.4 Consider the compatible WRM Algorithm Model 2.1 which satisfies the assumption of Lemma 4.3. Hence we can rewrite the i-th subsystem of (2.3) as

$$\dot{x}_i = \hat{f}_i(x_i, \hat{d}_i, u) \tag{4.3a}$$

$$z_i = \hat{g}_i(x_i, \hat{d}_i, u) \tag{4.3b}$$

$$x_i(0) = E_i(y_i(0)) \tag{4.3c}$$

$$\begin{aligned}\hat{d}_i = \operatorname{col}(&x_1, \ldots, x_{i-1}, x_{i+1}, \ldots, x_m,\\ &\dot{x}_1, \ldots, \dot{x}_{i-1}, \dot{x}_{i+1}, \ldots, \dot{x}_m,\\ &z_1, \ldots, z_{i-1}, z_{i+1}, \ldots, z_m,\\ &\dot{z}_1, \ldots, \dot{z}_{i-1}, \dot{z}_{i+1}, \ldots, \dot{z}_m)\end{aligned} \tag{4.3d}$$

where $x_i \in \mathbb{R}^{n_i}$, $z_i \in \mathbb{R}^{p_i - n_i}$, $\hat{d}_i \in \mathbb{R}^{2p-2p_i}$ and $\hat{f}_i, \hat{g}_i$ are continuously differentiable functions. For a simplified case, we shall assume that $\hat{f}_i$ and $\hat{g}_i$ are independent of $\{\dot{z}_i, i=1,2,\ldots,m\}$. Note that this assumption verifies the compatibility of the algorithm. Hence we can rewrite the iteration equation (2.5) in the following form:

$$\dot{x}_i^k = f_i(x_i^k, \tilde{d}_i^k, u) \tag{4.4a}$$

$$z_i^k = g_i(x_i^k, \tilde{d}_i^k, u) \tag{4.4b}$$

$$x_i^k(0) = E_i(y_i(0)) \tag{4.4c}$$

where

$$\begin{aligned}\tilde{d}_i^k = \operatorname{col}(&x_1^{k-1}, \ldots, x_{i-1}^{k-1}, x_{i-1}^{k+1}, \ldots, x_m^{k-1},\\ &\dot{x}_1^{k-1}, \ldots, \dot{x}_{i-1}^{k-1}, \dot{x}_{i+1}^{k-1}, \ldots, \dot{x}_m^{k-1},\\ &z_1^{k-1}, \ldots, z_{i-1}^{k-1}, z_{i+1}^{k-1}, \ldots, z_m^{k-1})\end{aligned} \tag{4.5a}$$

for the GJ relaxation, or

$$\begin{aligned}\tilde{d}_i^k = \operatorname{col}(&x_1^k, \ldots, x_{i-1}^k, x_{i-1}^{k+1}, \ldots, x_m^{k-1}, \\ &\dot{x}_1^k, \ldots, \dot{x}_{i-1}^k, \dot{x}_{i+1}^{k-1}, \ldots, \dot{x}_m^{k-1}, \\ &z_1^k, \ldots, z_{i-1}^k, z_{i+1}^{k-1}, \ldots, z_m^{k-1}) \qquad (4.5b)\end{aligned}$$

for the GS relaxation and f_i, g_i are continuously differentiable functions.

Hence the GJ-WRM algorithm can be tranformed into its canonical form by substituting (4.5a) into (4.4) and concatenating the subsystems $i=1,2,\ldots,m$ together. For the GS-WRM algorithm, we substitute (4.5b) into (4.4) and make forward substitutions of the first $i-1$ subsystems into the i-th subsystem in order to eliminate $\dot{x}_1^k, \ldots, \dot{x}_{i-1}^k, z_1^k, \ldots, z_{i-1}^k$ from the right hand side. The canonical form is then obtained by concatenating the resulting substituted subsystems. ■

In the above example, we can see that the differentiability of $\tilde{f}_i$ and $\tilde{g}_i$ is not needed in the transformation of the WRM Algorithm Model 2.1 into its canonical form. Also the number of state variables in the canonical form is equal to that of the decomposed system, i.e. $n = \sum_{i=1}^{m} n_i$ and $l = \sum_{i=1}^{m}(p_i - n_i)$. However, when the assumption that $\hat{f}_i$ and $\hat{g}_i$ are independent of $\{\dot{z}_i, i=1,2,\ldots,m\}$ does not hold, we have to differentiate some of the equations in (4.4) in order to tranform the algorithm into its canonical form. For example, in the third example of section III, we have to differentiate (3.4f) in order to obtain the canonical form. As a result of the differentiation, some extra state variables are created in the canonical form, e.g. v_3 in the third example of section III. The initial values of these extra state variables may be fixed or vary from one iteration to the ot.er. In this paper, we shall consider the convergence conditions of the compatible WRM Algorithm Model 2.1 in which the initial values of the extra state variables (if exist) of its canonical form are fixed. These conditions are given in the following theorem which is proved in the appendix. The convergence of the canonical WRM algorithm in which the initial values of some or all state variables are not fixed is discussed in [16].

Theorem 4.5 (Convergence Theorem of WRM Algorithms).

Consider the compatible WRM Algorithm Model 2.1 which can be transformed into the following canonical form:

$$\dot{x}^k = f(x^k, x^{k-1}, \dot{x}^{k-1}, z^{k-1}, u) \tag{4.6a}$$

$$z^k = g(x^k, x^{k-1}, \dot{x}^{k-1}, z^{k-1}, u) \tag{4.6b}$$

$$x^k(0) = x(0) \tag{4.6c}$$

where $x \in R^n$, $z \in R^l$ and $u \in R^r$. Assum~~ing~~e that

a) $u(\cdot) : [0,T] \to R^r$ is a given piecewise continuous* function.

b) there exist norms in $R^n \times R^l$ and R^n, $\lambda_1 \geq 0$, $\lambda_2 \geq 0$ and $\gamma \in [0,1)$ such that for any $a, b, s, \tilde{a}, \tilde{b}, \tilde{s} \in R^n$, $v, \tilde{v} \in R^l$ and $u \in R^r$

$$\left\| \begin{matrix} f(a,b,s,v,u) - f(\tilde{a},\tilde{b},\tilde{s},\tilde{v},u) \\ g(a,b,s,v,u) - g(\tilde{a},\tilde{b},\tilde{s},\tilde{v},u) \end{matrix} \right\| \leq \lambda_1\|a-\tilde{a}\| + \lambda_2\|b-\tilde{b}\| + \gamma \left\| \begin{matrix} s-\tilde{s} \\ v-\tilde{v} \end{matrix} \right\|$$

i.e. (f, g) is globally Lipschitz continuous with respect to x and globally contractive with respect to $(\dot{x}, z)$.

c) both f and g are continuous with respect to u.

Then, for any initial guess $(x^0(t), z^0(t) ; t \in [0,T])$ such that $\dot{x}^0(\cdot)$ and $z^0(\cdot)$ are piecewise continuous, the sequence $\{(\dot{x}^k(t), x^k(t), z^k(t) ; t \in [0,T])\}_{k=1}^{\infty}$ generated by the canonical WRM algo-

* A function $u(\cdot) : [0,T] \to R^r$ is piecewise continuous if it is continuous everywhere except at a finite number of points and at any discontinuity point, the function has finite left- and right-hand limits.

rithm (4.6) converges uniformly to $(\dot{\hat{x}}(t), \hat{x}(t), \hat{z}(t) ; t \in [0,T])$ which satisfies

$$\dot{\hat{x}} = f(\hat{x}, \hat{x}, \dot{\hat{x}}, \hat{z}, u) \tag{4.7a}$$

$$\hat{z} = g(\hat{x}, \hat{x}, \dot{\hat{x}}, \hat{z}, u) \tag{4.7b}$$

$$\hat{x}(0) = x(0) \tag{4.7c}$$

■

Remark: We do not need condition (b) of Theorem 4.5 to hold for the entire spaces $R^n \times R^l$ and R^n, i.e. global Lischitz and global contractive properties of (f, g) are not necessary. The convergence is still guaranteed as long as the condition (b) holds for subsets of $R^n \times R^l$ and R^n containing the sequences $\{(\dot{x}^k(t), z^k(t) ; t \in [0,T])\}_{k=0}^{\infty}$ and $\{(x^k(t) ; t \in [0,T])\}_{k=0}^{\infty}$ respectively. ■

It is possible to justify intuitively the derivation of the convergence conditions given in Theorem 4.5 if one is familiar with the contraction mapping theorem (see [15] page 120) and the Picard-Lindelof theorem on the existence and uniqueness of the solutions of ordinary differrential equations (see [17] page 18). From the contraction mapping theorem, the conditions (b) and (c) guarantee that (4.7) can be written equivalently as

$$\dot{\hat{x}} = \hat{f}(\hat{x}, u) \tag{4.8a}$$

$$\hat{z} = \hat{g}(\hat{x}, u) \tag{4.8b}$$

$$\hat{x}(0) = x(0) \tag{4.8c}$$

where $\hat{f}$ and $\hat{g}$ are Lipschitz continuous with respect to $\hat{x}$ and continuous with respect to u. Hence, by the Picard-Lindelof theorem, (4.8) has a unique solution $(\hat{x}, \hat{z})$ for any given initial condition and any given piecewise continuous input. Theorem 4.5 simply shows that the canonical WRM algorithm is in fact a constructive proof of the existence and uniqueness of the solution. ∎

V. Application of the WR method for the time domain solutions of MOS integrated circuits.

In this section we shall apply the WRM Algorithm Model 2.1 to analyse an important class of dynamical systems: MOS digital integrated circuits. In fact, this was the original motivation behind the development of WRM. A typical large scale digital circuit is usually an interconnection of several basic subcircuits called "gates". Hence the analysis of this class of circuits lends itself to the decomposition technique in the most natural way. We shall propose two WRM algorithms for analyzing MOS digital integrated circuits and show that, under very mild assumptions which are usually acceptable in practice, the proposed algorithms converge. Although both GS and GJ relaxations can be used in these algorithms, the GS relaxation is preferred since it requires only one copy of the iterated solutions, as opposed to two copies required by the GJ relaxation, and its speed of convergence is faster, especially for MOS circuits where unidirectional models are used to model MOS devices (for example see [3,4,5]), provided that the equations are properly ordered (see [10] for a discussion of this aspect). For the sake of simplicity, both algorithms use the simplest guessing scheme and an assignment-partition process in which each partitioned subsystem ia a single equation. The generalization of both algorithms to allow more than one equation per subsystem is straightforward and will not be discussed.

Our first standing assumption is that each MOS device and its interconnections can be modelled by lumped (linear or nonlinear) voltage controlled capacitors, conductors and current sources. The next assumption is that every (internal or external) node in the circuit has a (linear or nonlinear) capacitor to either ground or dc supply voltage rails. These assumptions are commonly acceptable in practice.

For the first algorithm, due to the above assumptions, we can use Nodal Analysis to formulate the dynamical equations of any MOS integrated circuit to obtain

$$C(v, u)\dot{v} + q(v, u) = 0 \tag{5.1a}$$

$$v(0) = v_0 \tag{5.1b}$$

where $v \in \mathbf{R}^n$ is the vector of all unknown node voltages, v_0 is the given initial values of v, $u \in \mathbf{R}^r$ is the vector of all inputs and their time derivatives, $q : \mathbf{R}^n \times \mathbf{R}^r \to \mathbf{R}^n$ is a continuous function each component of which represents the net sum of currents charging the capacitor at each node due to the conductors and the controlled current sources, $C : \mathbf{R}^n \times \mathbf{R}^r \to \mathbf{R}^{n \times n}$ is a symmetric diagonally dominant matrix-value function in which $-C_{ij}(v, u)\,;\, i \neq j$ is the total capacitance between nodes i and j, and $C_{ii}(v, u)$ is the sum of the capacitances of all capacitors connected to node i.

Due to the grounded capacitors, an obvious choice of the state vector of the circuit is the vector of all unknown node voltages. It is thus natural to assign v_i to the i-th equation of (5.1a). Consequently, the assignment-partition process is compatible with (5.1). Applying the WRM Algorithm Model 2.1 to (5.1), we obtain Algorithm 5.1 as described below.

Algorithm 5.1.

Step 1: Set $k = 1$ and $v^0(t) = v(0)$ for all $t \in [0, T]$.

Step 2: For $i = 1, 2, \ldots, n$, solve for $\{v_i^k(t)\,;\, t \in [0, T]\}$ from

$$\sum_{j=1}^{i} C_{ij}(v_1^k, \ldots, v_i^k, v_{i+1}^{k-1}, \ldots, v_n^{k-1}, u)\dot{v}_j^k \; +$$

$$\sum_{j=i+1}^{n} C_{ij}(v_1^k, \ldots, v_i^k, v_{i+1}^{k-1}, \ldots, v_n^{k-1}, u)\dot{v}_j^k \ +$$

$$q_i(v_1^k, \ldots, v_i^k, v_{i+1}^{k-1}, \ldots, v_n^{k-1}, u) \quad = \quad 0 \ \ ; v_i^k(0) = u_i(0).$$

Step 3: Set $k = k+1$ and go to step 2. ■

The second algorithm is intended for MOS circuits containing pass transistors (or transmission gates), as for example the circuit in the second example of section III. Modified Nodal Analysis [ ] is used to formulated the dynamical equations of the circuit to obtain

$$C(v,u)\dot{v} + \tilde{q}(z,v,u) = 0 \tag{5.2a}$$

$$z - g(v,u) = 0 \tag{5.2b}$$

$$v(0) = v_0 \tag{5.2c}$$

where C, v, u, v_0 are as defined in (5.1), $z \in \mathbb{R}^l$ is the vector of drain currents of the pass transistors, $g : \mathbb{R}^n \times \mathbb{R}^r \to \mathbb{R}^l$ is a continuous function each component of which describes the drain current of each pass transistor in terms of its terminal node voltages, and $\tilde{q} : \mathbb{R}^l \times \mathbb{R}^n \times \mathbb{R}^r \to \mathbb{R}^n$ is a continuous function each component of which represents the net current charging the capacitor at each node due to the pass transistors ,other conductive elements and controlled current sources.

We choose to assign v_i to the i-th equation of (5.2a) and z_i to the i-th equation of (5.2b). Hence the assignment-partition process is compatible with (5.2). Applying the WRM Algorithm Model 2.1 to (5.2), we obtain Algorithm 5.2 as described below.

Algorithm 5.2.

Step 1: Set $k = 1$, $z^0(t) = 0$ and $v^0(t) = v(0)$ for all $t \in [0,T]$.

Step 2: a) For $i = 1,2,\ldots,n$, solve for $(v_i^k(t) : t \in [0,T])$ from

$$\sum_{j=1}^{i} C_{ij}(v_1^k, \ldots, v_i^k, v_{i+1}^{k-1}, \ldots, v_n^{k-1}, u)\dot{v}_j^k \quad +$$

$$\sum_{j=i+1}^{n} C_{ij}(v_1^k, \ldots, v_i^k, v_{i+1}^{k-1}, \ldots, v_n^{k-1}, u)\dot{v}_j^k \quad +$$

$$\tilde{q}_i(z^{k-1}, v_1^k, \ldots, v_i^k, v_{i+1}^{k-1}, \ldots, v_n^{k-1}, u) \quad = \quad 0 ; \quad v_i^k(0) = v_i(0).$$

b) Compute $z^k(t) ; t \in [0,T]$ from

$$z^k \quad = \quad g(v^k, u)$$

Step 3: Set $k = k+1$ and go to step 2. ∎

Theorem 5.1. Assum~~ing~~e that

a) The charge-voltage characteristic of each capacitor, or the volt-ampere characteristic of each conductor, or the drain current characteristic of each MOS device is Lipschitz continuous with respect to its controlling variables,

b) $C_{\min} > 0$ and $C_{\max} < \infty$ where

$C_{\min} \in \mathbf{R}$ is the minimum value of all grounded capacitances at any permissible values of voltages, and

$C_{\max} \in \mathbf{R}$ is the maximum value of all capacitances between any two nodes at any permissible values of voltages,

c) The current through any controlled conductor, e.g. the drain current of an MOS device, is uniformly bounded throughout the relaxation process.

Then, for any MOS circuit with any given set of initial conditions, and any given piecewise continuous $u(\cdot)$, Algorithm 5.1 or 5.2 generates a converging sequence of iterated solutions whose limit satisfies the dynamical equations of the circuit and the given set of initial conditions. ∎

The proof of the above theorem is in the appendix. Note that the first assumption implies that for any capacitor, conductor or MOS device, its incremental (or small signal) characteristic, i.e. capacitance, conductance or transconductance, any any permissible dc operating point must be uniformly bounded. Note also that the third assumption implies that during the relaxation process, the current through any conductor or MOS device must not grow arbitrarily large. These three assumptions are usually very mild in practice and hence either Algorithm 5.1 or 5.2 is guaranteed to converge for any MOS integrated circuit.

In both algorithms the initial guesses are chosen, for convenience, to bè constant waveforms. From Theorem 4.5, we know that other choices of initial guèsses will not destroy the guaranteed convergence of both algorithms if they are piecewise continuous waveforms. Hence, for MOS digital integrated circuits, a logic simulation could be used to generate the initial guesses for these two algorithms. It is also possible to show that, under the same assumptions of Theorem 5.3, the corresponding GJ relaxation versions of Algorithm 5.1 and 5.2 are guaranteed to converge. Moreover, a relaxation parameter ω can be introduced into these GJ- or GS-WRM algorithms (as described in section II) without destroying their guaranteed convergence, provided that $\omega \in (0,2)$.

As an example, the ring oscillator described in section III is used to verify the guaranteed convergence of Algorithm 5.1 The circuit is schematically redrawn in Fig. 5.1a. The circuit interpretation of the relaxation process is already shown in Fig. 3.1b. The resulting waveforms at different iterations of the algorithm are shown in Fig. 5.1b through 5.1e and, for comparison the resulting waveform obtained from a standard circuit simulator SPICE is shown IN Fig. 5.5.

Note that since the oscillator is highly non-unidirectional due to the feedback from to the input of the NOR gate, the convergence of the interated solutions is achieved with the number of iterations being proportional to the number of oscillating cycles of interest. More examples as well as the techniques to increase the computational efficiency of WRM algorithms are given in the next section.

Fig. 5.1

a) Schematic diagram of a ring oscillator with its triggering input waveform.

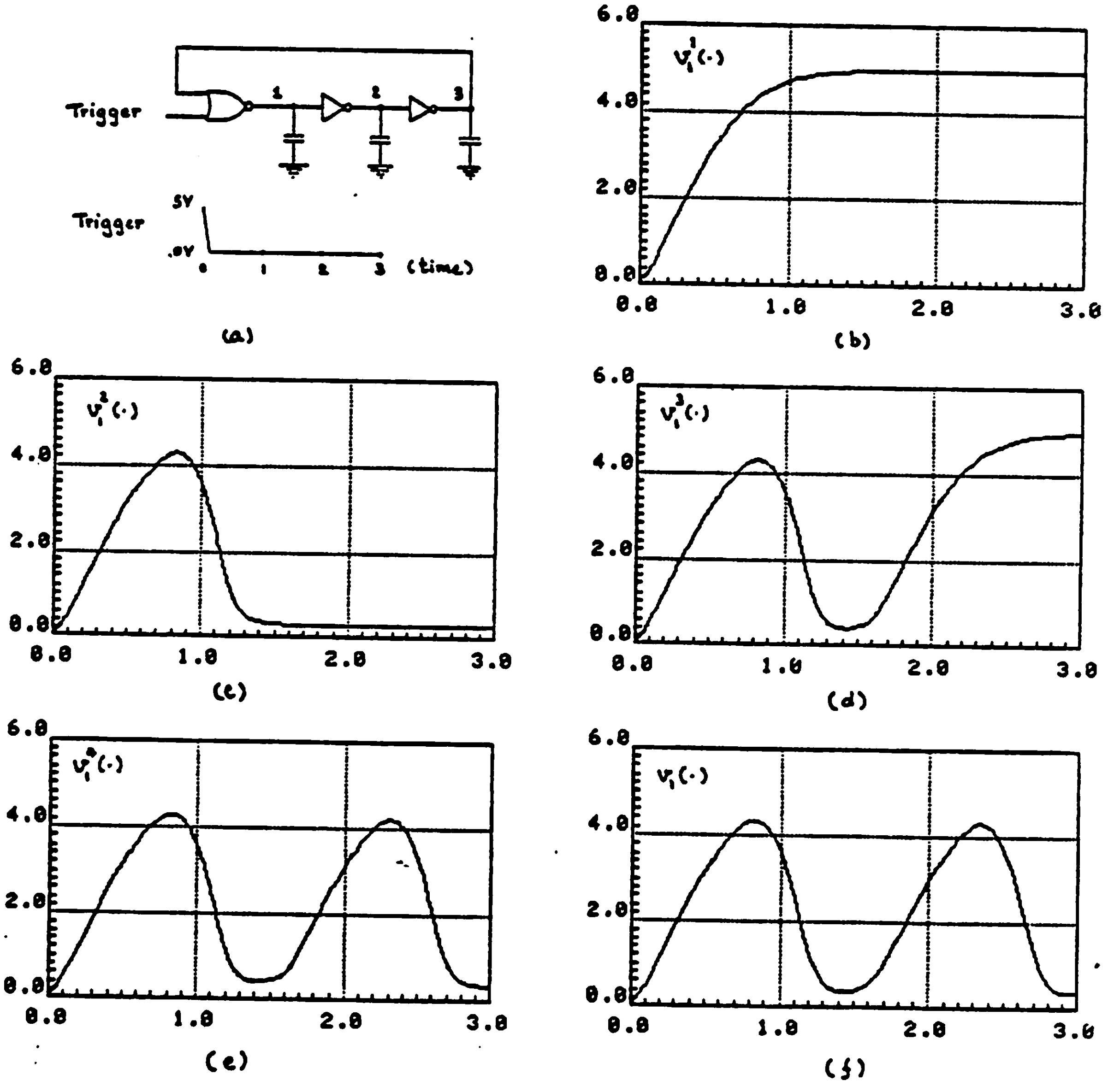
b) The waveform of v_1 after the first iteration of Algorithm 5.1.

c) The waveform of v_1 after the second iteration of Algorithm 5.1.

d) The waveform of v_1 after the third iteration of Algorithm 5.1.

e) The waveform of v_1 after the fourth iteration of Algorithm 5.1.

f) The waveform of v_1 obtained by SPICE circuit simulator.



VI. Experimental program RELAX: A computational study of WRM.

To study the computational efficiency of WRM, an experimental FORTRAN program called RELAX has been written for simulating MOS digital integrated circuits. The algorithm implemented in RELAX is based on Algorithm 5.1 with the following modifications.

1) RELAX allows each partitioned subsystem to have more than one equation so that each subsystem corresponds to a physical digital subcircuit, e.g. NOR, NAND, FLIP-FLOP etc.

2) The first iteration of RELAX is essentially the first iteration of Algorithm 5.2, but after that RELAX switches back to use Algorithm 5.1 for the rest of the relaxation process. That is, in the first iteration of RELAX, the drain currents of pass transisters do not contribute any loading effect on the subcircuits to which they are connected. This is done because, in the first iteration when all initial guesses are constant waveforms, a pass transistor can be driven continuously into its conductive region and may adversely effect the speed of convergence if its current is treated as a load of the other subcircuit.

Note that these modifications are just examples of specializing WRM for a particular class of dynamical systems so that some techniques that take advantage of its characteristics can be exploited to improve the speed of convergence. In addition, RELAX incorporates two techniques to speed up the analysis of the decomposed circuit. The first technique is based on the latency of each decomposed subcircuit and is similar to the technique described in [7]. The second technique, which takes effect after the second iteration, is based on the partial convergence of iterated waveforms. That is, when the difference between a waveform at the current iteration and the same waveform at the previous iteration is within the specified convergence error over a subinterval of the analysis time, the waveform will not be recomputed for that subinterval in the next iteration.

Several MOS digital circuits have been analysed by RELAX and the results have been compared with those obtained by two other simulation programs, SPICE [1] and SPLICE [4]. In these tests, SPLICE has been run as a timing simulator. All three programs use the Schichman-Hodges equation [9] to model the drain current of an MOS device and linear capacitors to model the charge storing mechanism of the device and its parasitic capacitances. Since the version of SPLICE used in this test does not allow the timing analysis of circuits containing floating capacitors*, each test circuit has been analysed with no floating capacitors for the comparison with SPLICE and with floating capacitors for the comparison with SPICE. Since all three programs use numerical integration methods with adjustable timesteps, the same maximum timestep, which is an input parameter of each program, is specified for the numerical integration routine of each program. For RELAX, the specified convergence error is 0.05 volt, i.e. the relaxation process stops when the maximum difference of all voltage waveforms between the current iteration and the previous iteration is less than 0.05 volt. However, as described earlier, the same convergence error is also used by RELAX to bypass unnecessary analysis when a partial convergence is detected.

The schematic diagram of MOS circuits being tested and their input waveforms are shown in Fig. 6.1a through 6.5a. Comparisons of output waveforms obtained by RELAX and SPICE are shown in Fig. 6.1b, 6.1c through 6.5b, 6.5c. For RELAX outputs, each rectangular mark denotes the computed

* A floating capacitor is a capcitor between two dependent nodes in a circuit. For example, if the input of an MOS inverter is not connected to an ideal voltage source, then the drain-gate capacitor of its pull-down device is a floating capacitor.

value after every two internal timesteps to illustrate the effect of the implemented latency technique. A comparison of the analysis time in CPU seconds spent by each program is given in Tables 6.1 and 6.2. All three programs run on a VAX 11/780 using UNIX* operating system. The tabulated figures do not include the time spent in the read-in, set-up and read-out phases of each program. For RELAX, the tabulated figure is the sum of the analysis time spent at each iteration and hence the total number of iterations is also included. As seen from the tables, the analysis time of RELAX is at least one order of magnitude less than SPICE and at the same order as SPLICE. These are accounted for by the following factors.

1) Both RELAX and SPLICE use decomposition techniques, although of different natures. It is known that the complexity of the analysis of a circuit without decomposition is proportional to n^m where n is the size, i.e. the number of unknown variables, of the circuit and m is usually between 1.2 and 3 depending on the sparseness of the circuit, i.e. m is small (large) if the number of interactions between the unknown variables is small (large). With decomposition, the complexity is usually proportional to n. Hence, the decomposition is a major factor in reducing the analysis time of a circuit. The larger the circuit, the more reduction is achieved as shown in Table 6.1.

2) Both RELAX and SPLICE incorporate latency techniques, although of different types. These techniques take advantage of the latency of a circuit by bypassing unnecessary computations. RELAX also has an additional technique that takes advantage of the partial waveform convergence which enables it to reduce the analysis time of the decomposed circuit at later iterations.

3)
Due to the decomposition used in RELAX and SPLICE, each decomposed subcircuit can be analysed with its own integration method and optimal timestep sequence. Without decomposition as in SPICE, the waveforms of all unknown variables are computed from the same timestep sequence.

4) Both RELAX and SPLICE are specialized programs for analysing MOS integrated circuits, as opposed to SPICE which is capable of analysing circuits using different types of devices, e.g. MOS, bipolar transistors etc. Therefore, many of the overheads that have to be included in SPICE in order to be able to deal with different models and different devices are eliminated in RELAX and SPLICE.

As for a comparison between RELAX and SPLICE, it is important to note that while the analysis method in the timing simulation of SPLICE is only an approximated method that sometimes suffers from instability and inaccuracy problems (see [4] and [8]), the analysis method WRM, used in RELAX is reliable and accurate. Note also that, without floating capacitors, the circuits of Fig. 6.4a and 6.5a possess a unidirectional property which is referred to as *one-way* property in [10]. Circuits having a unidirectional property can be analyzed exactly by performing only one iteration of GS relaxation. Since the present version of RELAX does not recognize this property, it has to perform two iterations before the convergence can be detected. ■

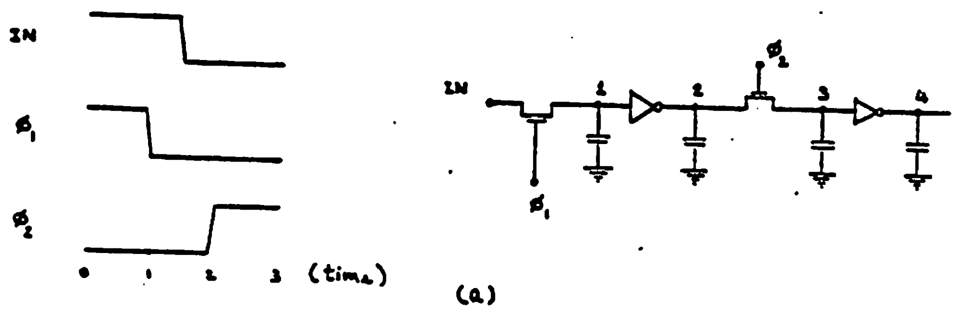
* UNIX is a trade mark of Bell Laboratories.

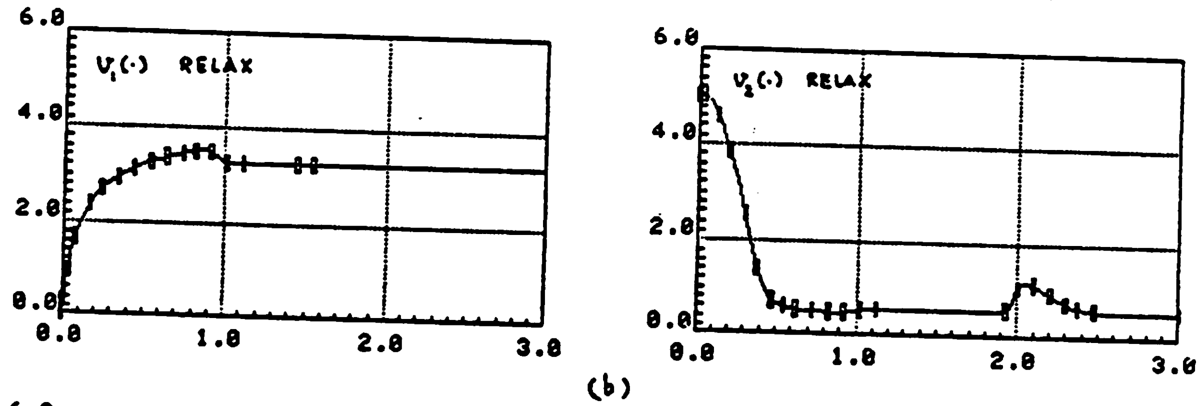
Fig. 6.1

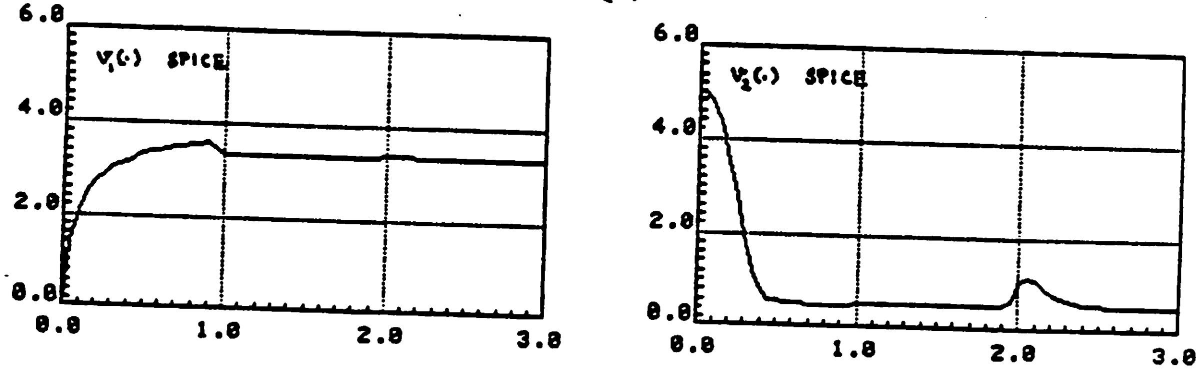
a) Schematic diagram of a dynamic shift register.

b) Output waveforms obtained by RELAX.

c) Output waveforms obtained by SPICE.







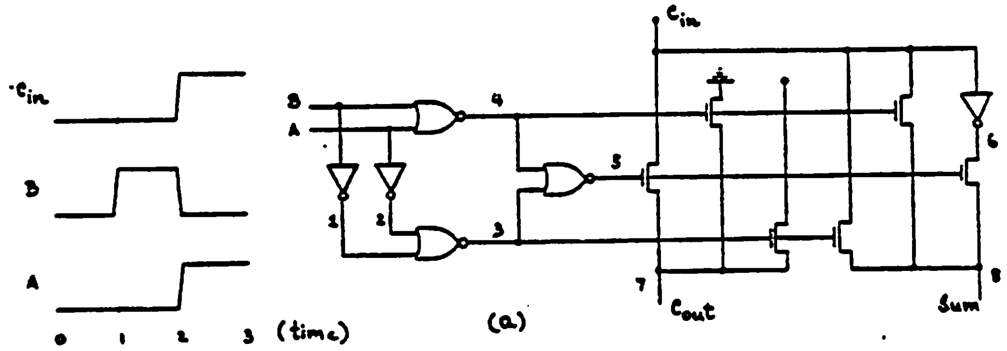
(c)

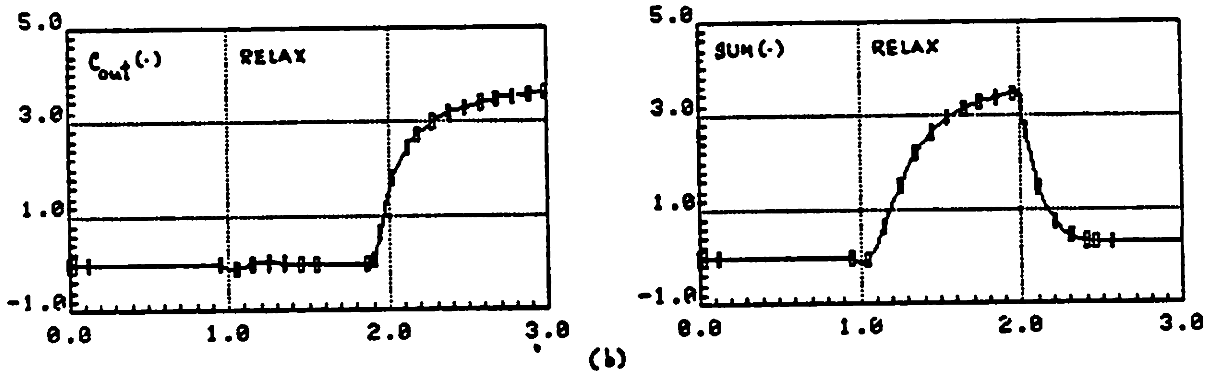
Fig. 6.2

a) Schematic diagram of a one-bit full adder with a pass transistor carry-chain.

b) Output waveforms obtained by RELAX.

c) Output waveforms obtained by SPICE.





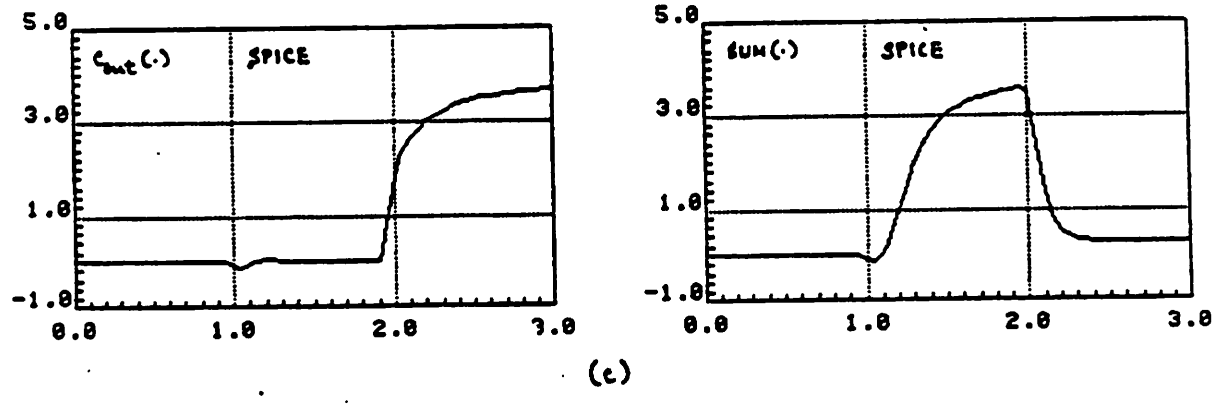


Fig. 6.3

a) A two-bits full adder obtained by cascading two one-bit full adder of Fig. 6.2a.

b) Output waveforms obtained by RELAX.

c) Output waveforms obtained by SPICE.

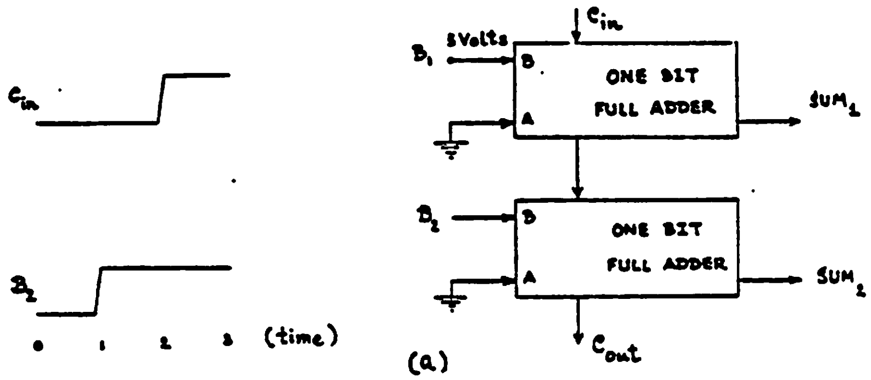
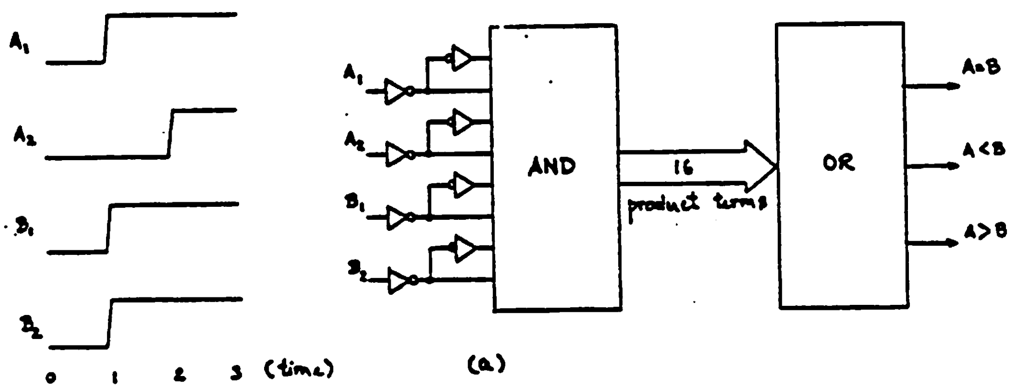


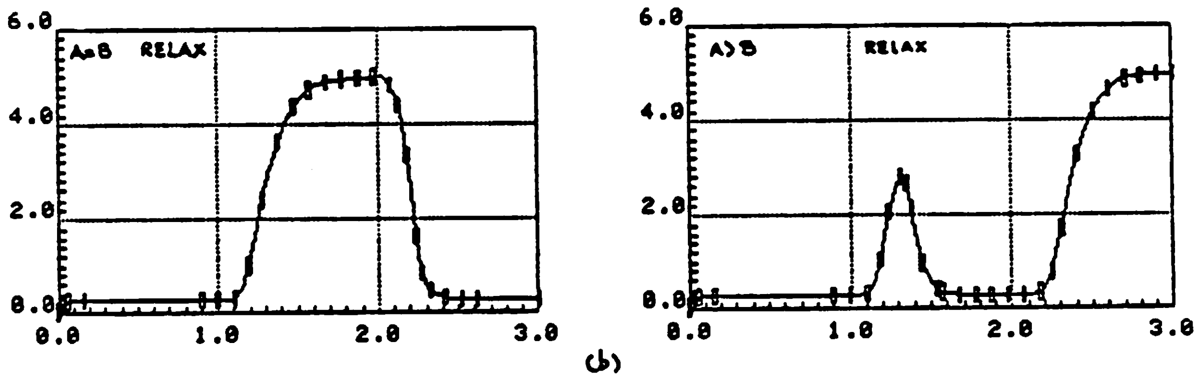
Fig. 6.4

a) Schematic diagram of a two-bits magnitude comparator implemented by a NOR-NOR PLA with no minimization of the product terms.

b) Output waveforms obtained by RELAX.

c) Output waveforms obtained by SPICE.





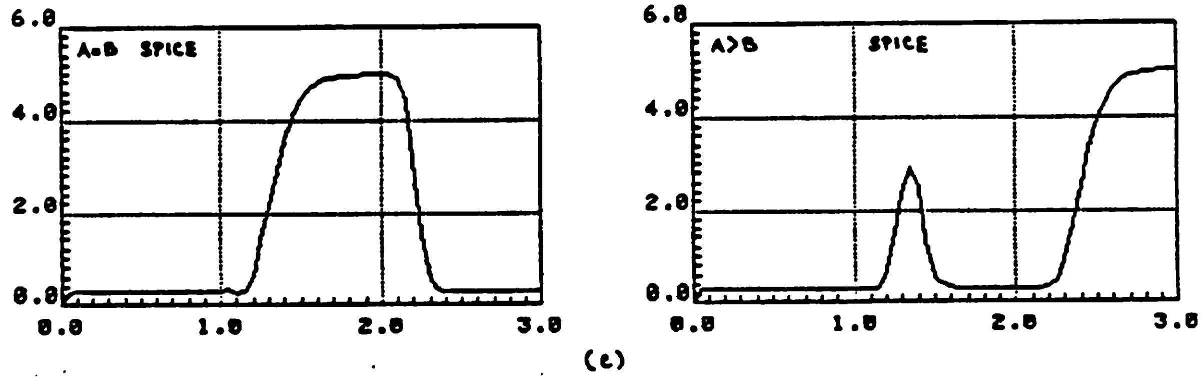


Fig. 6.5

a) Schematic diagram of a two-bits full adder implemented by a NOR-NOR PLA with no minimization of the product terms.

b) Output waveforms obtained by RELAX.

c) Output waveforms obtained by SPICE.

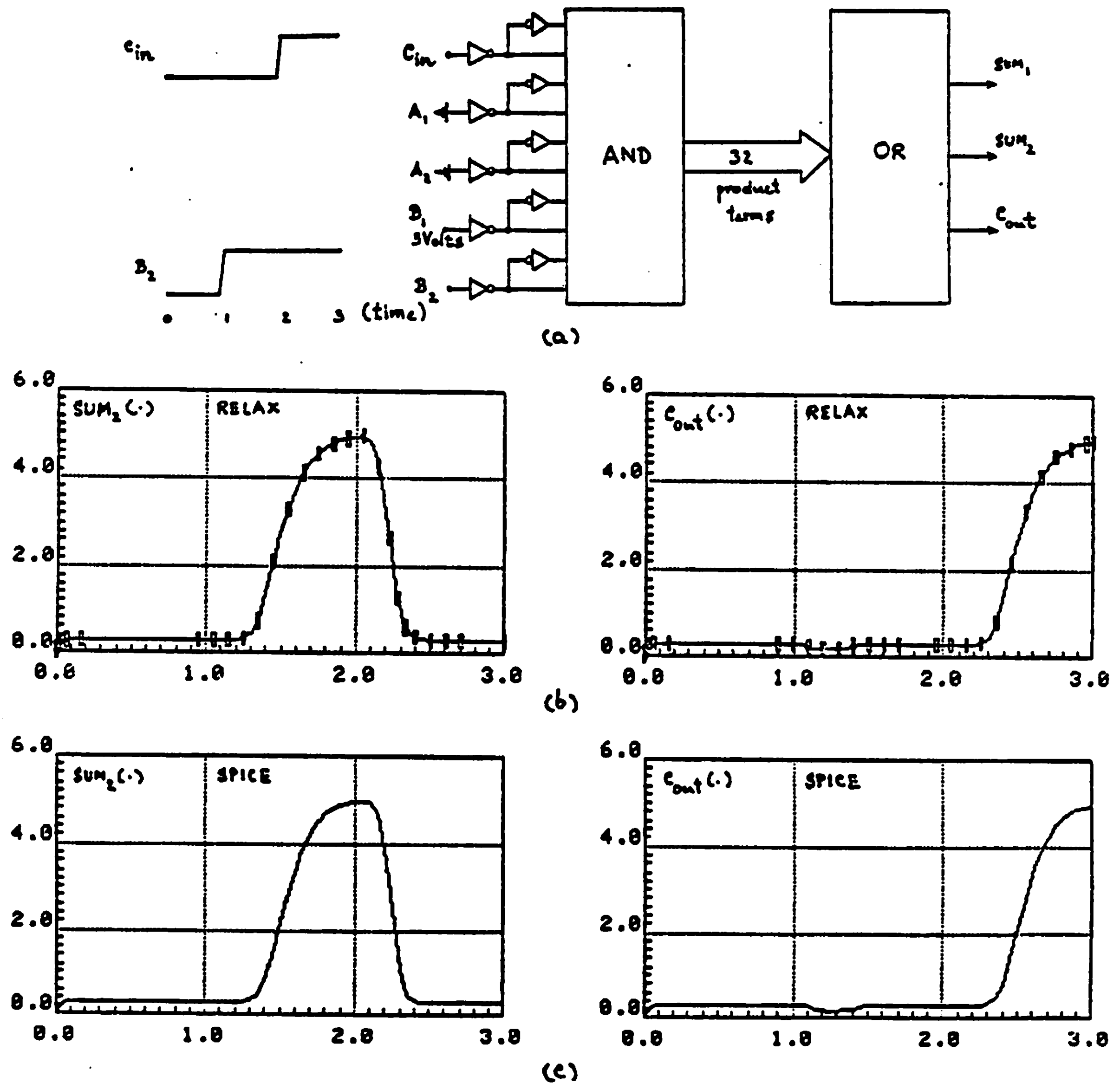


Table 6.1

A comparison of the analysis time between SPICE and RELAX

Circuit[1] of	Fig. 6.1	Fig. 6.2	Fig. 6.3	Fig. 6.4	Fig. 6.5
# of unknown nodes	4	8	16	27	45
# of MOS devices	6	21	42	131	263
CPU-SPICE (sec)	21.30	121.57	211.53	818.00	1334.80
CPU-RELAX (sec)	1.08	4.38	5.85	18.42	22.30
# of RELAX iterations	5	5	7	5	4
CPU-SPICE/CPU-RELAX	19.70	27.73	36.16	44.42	59.86

[1] With floating capacitors. The ratio of a floating capacitance to a grounded capacitance is approximately 1 to 12.

Table 6.2

A comparison of the analysis time between SPLICE and RELAX

Circuit[2] of	Fig. 6.1	Fig. 6.2	Fig. 6.3	Fig. 6.4	Fig. 6.5
# of unknown nodes	4	8	16	27	45
# of MOS devices	6	21	42	131	263
CPU-SPLICE (sec)	1.18	5.00	8.62	20.95	37.47
CPU-RELAX (sec)	0.53	4.68	7.82	27.00	44.73
# of RELAX iterations	2	3	7	2	2

[2] With no floating capacitors. Although the circuits and their input waveforms used in Table 6.1 and 6.2 are the same, the initial conditions are different.

VII. Conclusion and discussion.

WRM is a new method for the time domain analysis of large scale dynamical systems based on the relaxation of the nonlinear algebraic-differential equations describing the system to be analysed. Since the method decouples these equations, independent integration of the each decomposed subsystem can be performed with different integration formulae and timesteps. In addition, latency of subsystems can be easily detected and exploited to reduce the analysis time. This method is particularly suitable for VLSI circuits. In comparison with the relaxation methods used by timing simulators, e.g. MOTIS and the timing simulation part of SPLICE, WRM is guaranteed to converge for a wide class of circuits while sharing the main advantages of these techniques.

On the negative side, for WRM we need to store the waveforms at the current iteration for use in the next iteration. For large time intervals, the amount of needed storage can be very large. However, in the same way as is normally implemented in any computer using a time sharing operating system, the waveforms can be stored in the secondary storage and brought back into the primary storage only when they are needed. To avoid delays due to the slow access time of the secondary storage, a buffering scheme can be implemented to transfer the waveforms from primary to secondary storage and back without stopping the execution of the method.

The computational efficiency of WRM can be further improved by the following techniques:

1) The use of an adaptive error control scheme. In this scheme, the errors incurred in computing the solutions of the decomposed system during the initial iterations are allowed to be large and are progressively

decreased as the sequence of iterated solutions converges to the final solution. Various approximation techniques can be used to compute the solution of the decomposed system during the initial iterations of the method. These are discussed together with the proof of convergence of WRM with this adaptive error control scheme in [16].

2) The implementation of WRM on pipeline or parallel processors. Both types of relaxation, i.e. GJ and GS relaxation, can be implemented. However, parallel processors are better suited to GJ-WRM since the analysis of all decomposed subsystems for any given iteration can be done concurrently. ■

VIII. Acknowledgement.

The authors wish to thank V. Visvanathan, R.K. Brayton, G.H. Hachtel, A.R. Newton, N.B.G. Rabbat, R.J. Kaye and M.J. Chen for many helpful discussions. Also the authors wish to thank W. Nye and B.S. Landman for suggestions in developing RELAX. ■

References.

[1] L. W. Nagel, "SPICE 2: A computer program to simulate semiconductor circuits," Electronics Research Laboratory Rep. No. ERL-M520, University of California, Berkeley, May 1975.

[2] W. T. Weeks, A. J. Jimenez, G. W. Mahoney, D. Mehta, H. Qassemzadeh and T. R. Scott, "Algorithms for ASTAP - A network analysis program," *IEEE Transactions on Circuit Theory*, Vol. CT-20, pp. 628-634, November 1973.

[3] B. R. Chawla, H. K. Gummel and P. Kozak, "MOTIS - an MOS timing simulator," *IEEE Transactions on Circuits and Systems*, Vol. CAS-22, pp. 901-910, December 1975.

[4] A. R. Newton, "The simulation of large scale intregrated circuits," *IEEE Transactions on Circuits and Systems*, Vol. CAS-26, pp. 741-749, September 1979.

[5] A. R. Newton, "The simulation of large scale integrated circuits," University of California, Berkeley, Electronics Research Laboratory, Memo. No. ERL-M78/52, July 1978.

[6] G. Arnout and H. De Man, "The use of threshold functions and boolean-controlled network elements for macromodelling of LSI circuits," *IEEE Journal of Solid-State Circuits*, Vol. SC-13, pp. 326-332, June 1978.

[7] N. B. G. Rabbat, A. L. Sangiovanni-Vincentelli and H. Y. Hsieh, "A multilevel Newton algorithm with macromodelling and latency for the analysis of large-scale nonlinear circuits in the time domain," *IEEE Transactions on Circuits and Systems*, Vol. CAS-26, pp. 733-741, September 1979.

[8] G. DeMicheli and A. L. Sangiovanni-Vincentelli, "Numerical properties of algorithms for the timing analysis of MOS VLSI circuits," *Proceedings ECCTD'81* The Hague, August 1981.

[9] A. Vladimirescu and S. Liu, "The simulation of MOS integrated circuits using SPICE2," University of California, Berkeley, Electronic Research Laboratory, Memo. No. UCB/ERL M80/7, October 1980.

[10] A. E. Ruehli, A. L. Sangiovanni-Vincentelli and N. B. G. Rabbat, "Time analysis of large scale circuits containing one-way macromodels," *IEEE Proceedings Int. Symposium Circuits and Systems*, 1980.

[11] C. A. Desoer and E. S. Kuh, "Basic Circuit Theory," McGraw-Hill, 1969.

[12] C. W. Ho, A. E. Ruehli and P. A. Brennan, "The modified nodal approach to network analysis," *IEEE Transactions on Circuits and Systems*, Vol. CAS-22, pp. 504-509, June 1975.

[13] L. O. Chua and P. M. Lin, "Computer-aided analysis of electronic circuits: algorithms and computational techniques," Prentice Hall, 1975, Chap. 10. pp. 410-431.

[14] G. D. Hachtel, R. K. Brayton and F. G. Gustavson, "The sparse tableau approach to network analysis and design," *IEEE Transactions on Circuit Theory*, Vol. CT-18, pp. 101-113, January 1971.

[15] J. M. Ortega and W. C. Rheinboldt, "Iterative solution of nonlinear equations in several variables," Academic Press, 1970.

[16] E. Lelarasmee, "The waveform relaxation method for the time domain analysis of large scale nonlinear dynamical systems," Ph.D. Dissertation, University of California, Berkeley, 1981.

[17] J. K. Hale, "Ordinary differential equations," McGraw-Hill,. 1969. ■

Appendix

Example A1. The purpose of this example is to illustrate some behaviors of a WRM algorithm that use a non-compatible assignment-partition process, as defined in Definition 4.1. The dynamical equations to be analyzed are:

$$\dot{y}_1 + y_2 - u = 0; \quad y_1(0) = 0 \tag{A1.1a}$$

$$y_1 - y_2 = 0 \tag{A1.1b}$$

Let y_1 and y_2 be assigned to (A1.1b) and (A1.1a) respectively and let (A1.1) be partitioned into 2 subsystems consisting of $\{$(A1.1a)$\}$ and $\{$(A1.1b)$\}$. Applying the WRM Algorithm Model 2.1, the k-th iteration of the resulting GS-WRM algorithm is given by

$$y_2^k = -\dot{y}_1^{k-1} + u \tag{A1.2a}$$

$$y_1^k = y_2^k \tag{A1.2b}$$

Notice that, while the original system (A1.1) has one state variable, the decomposed system (A1.2) at the k-th iteration has no state variable since it consists of only algebraic equations. Hence the above assignment-partition process is not compatible. It is easy to show that the solutions of (A1.2) are

$$y_1^k(t) = y_2^k(t) = \sum_{j=0}^{k-1}(-1)^j \frac{d^j}{dt^j}u(t) + (-1)^k \frac{d^k}{dt^k}y_1^0(t)$$

Thus both $y_1^k(t)$ and $y_2^k(t)$ diverge when the initial guess is $y_1^0(t) = e^{-\alpha t}$, $\alpha>1$. Now suppose that the initial guess is of the form $y_1^0(t) = \sum_{j=0}^{l} a_j t^j$ where l is a finite positive integer and a_j, $j=1,2,\ldots,l$ are constants. Then we can deduce the following results:

a) When the input $u(t) = \alpha ; t \geq 0$, the solutions of (A1.1) are $y_1(t) = y_2(t) = \alpha(1 - e^{-t})$ but both $y_1^k(t)$ and $y_2^k(t)$ converge to $\hat{y}_1(t) = \hat{y}_2(t) = \alpha$ which are solutions of (A1.1) with a different initial condition, i.e. $\hat{y}_1(0) = \alpha$.

b) When $u(t) = e^{-\alpha t}$, $\alpha > 1$, both $y_1^k(t)$ and $y_2^k(t)$ diverge while the solutions of (A1.1) are $y_1(t) = y_2(t) = \frac{1}{\alpha - 1}(e^{-\alpha t} - e^{-t})$.

c) If $u(\cdot)$ is piecewise continuous with at least one discontinuity point, then both $y_1^k(\cdot)$ and $y_2^k(\cdot)$ will be unbounded at those discontinuity points but the solutions of (A1.1) will be continuous and bounded within the given time interval.

This example shows that for a non-compatible WRM algorithm, the iteration may converge to solutions of the original dynamical system with a different set of initial conditions. Furthermore, the convergence to the correct solutions arises only under special conditons on the input waveforms, initial guesses and initial conditions. Hence non-compatible WRM algorithms tend to have poor behaviors and should not be used. ∎

The following facts are useful for the proofs of Theorem 4.5 and Theorem 5.3.

Lemma A2. Let $\|\cdot\|_a$, $\|\cdot\|_b$ be norms in $\mathbb{R}^n$, $\mathbb{R}^{n+l}$ respectively. Then there exists a constant μ such that

$$\|x\|_a \leq \mu \left\| \begin{matrix} x \\ z \end{matrix} \right\|_b \qquad \text{for all } x \in \mathbb{R}^n \text{ and } z \in \mathbb{R}^l$$

Proof. We use the fact that all norms in a finite dimensional space are equivalent (see [15] page 39). That is, if $\|\cdot\|_a$ and $\|\cdot\|_{\tilde{a}}$ are norms in $\mathbb{R}^n$, there exist constants μ_1 and μ_2 such that for any $x \in \mathbb{R}^n$

$$\|x\|_a \le \mu_1 \|x\|_{\tilde{a}} \quad \text{and} \quad \|x\|_{\tilde{a}} \le \mu_2 \|x\|_a \tag{A2.1}$$

Define

$$\|x\|_{\tilde{a}} \triangleq \left\| \begin{matrix} x \\ 0 \end{matrix} \right\|_b \quad \text{and} \quad \left\| \begin{matrix} x \\ z \end{matrix} \right\|_{\tilde{b}} \triangleq \left\| \begin{matrix} x \\ 0 \end{matrix} \right\|_b + \left\| \begin{matrix} 0 \\ z \end{matrix} \right\|_b \tag{A2.2}$$

Then from (A2.1), there exist μ_1, $\tilde{\mu}_2$ such that

$$\|x\|_a \le \mu_1 \|x\|_{\tilde{a}} \quad \text{and} \quad \left\| \begin{matrix} x \\ z \end{matrix} \right\|_{\tilde{b}} \le \tilde{\mu}_2 \left\| \begin{matrix} x \\ z \end{matrix} \right\|_b \tag{A2.3}$$

(A2.2) and (A2.3) imply that

$$\|x\|_a \le \mu_1 \|x\|_{\tilde{a}} \le \mu_1 \left\| \begin{matrix} x \\ z \end{matrix} \right\|_{\tilde{b}} \le \mu_1 \tilde{\mu}_2 \left\| \begin{matrix} x \\ z \end{matrix} \right\|_b$$

Therefore, the proof is completed. ■

Lemma A3. Define $\|\cdot\|_\infty$ in $\mathbb{R}^n$ and $\mathbb{R}^{n \times n}$ as follows:

$\|x\|_\infty \triangleq \max_{1 \le i \le n} |x_i|$ where x_i is the i-th component of $x \in \mathbb{R}^n$, and

$\|A\|_\infty \triangleq \max_{1 \le i \le n} \sum_{j=1}^{n} A_{ij}$ where A_{ij} is the (i,j)-th element of $A \in \mathbb{R}^{n \times n}$.

Let L and U be strictly* upper and lower triangular matrices in $\mathbb{R}^{n \times n}$ such that $L \ge 0$, $U \ge 0$** and

$$\|L+U\|_\infty = \max_{1 \le i \le n} \left\{ \sum_{j=1}^{n} (L_{ij} + U_{ij}) \right\} \le 1 \tag{A3.1}$$

where L_{ij}, U_{ij} are the (i,j)-th elements of L, U respectively. Then

$$\|(I-L)^{-1}U\|_\infty \le \|L+U\|_\infty \le 1$$

Proof. Let $\hat{x} = \mathrm{col}(1,1,\dots,1) \in \mathbb{R}^n$, then from (A3.1) we have

* A strictly triangular matrix is a triangular matrix whose diagonal elements are zero.

** A vector x or a matrix A whose elements are all nonnegative is denoted by $x \ge 0$ or $A \ge 0$ respectively.

$$(I-L-U)\hat{x} \ \geq \ 0 \tag{A3.2}$$

Since L is strictly lower triangular, we have

$$(I-L)^{-1} \ = \ I + L + L^2 + \cdots + L^{n-1}$$

Therefore

$$\begin{aligned}(L+U)-(I-L)^{-1}U &= [(I-L)^{-1}-I](I-L-U) \\ &= (L+L^2+\cdots+L^{n-1})(I-L-U) \\ (L+U)\hat{x}-(I-L)^{-1}U\hat{x} &= (L+L^2+\cdots+L^{n-1})[(I-L-U)\hat{x}] \qquad \text{(A3.3)}\end{aligned}$$

(A3.2), (A3.3) and $L \geq 0$ imply that

$$(L+U)\hat{x}-(I-L)^{-1}U\hat{x} \ \geq \ 0 \tag{A3.4}$$

Since $(L+U) \ \geq \ 0$ and $(I-L)^{-1}U = (I+L+\cdots+L^{n-1})U \ \geq \ 0$, we have

$$\|L+U\|_\infty \ = \ \|(L+U)\hat{x}\|_\infty \text{ and } \|(I-L)^{-1}U\|_\infty = \|(I-L)^{-1}U\hat{x}\|_\infty$$

Therefore (A3.4) implies that

$$\|L+U\|_\infty \ \geq \ \|(I-L)^{-1}U\|_\infty \qquad \blacksquare$$

Lemma A4. Let $f_1, f_2 : \mathbf{D} \subset \mathbf{R}^n \to \mathbf{R}$ be two bounded[*] Lipschitz continuous functions, then the product function f_1f_2 is also bounded Lipschitz continuous in $\mathbf{D}$. If f_2 is also bounded away from zero, i.e. $\inf_{x \in \mathbf{D}} |f_2(x)| = \bar{f}_2 > 0$, then $\dfrac{f_1}{f_2}$ is also bounded Lipschitz continuous in $\mathbf{D}$.

Proof. Let $\hat{f}_1 = \sup_{x \in \mathbf{D}} |f_1(x)|$, $\hat{f}_2 = \sup_{x \in \mathbf{D}} |f_2(x)|$ and λ_1, λ_2 be the Lipschitz constants of f_1, f_2 respectively. Then for any $x, y \in \mathbf{D}$

[*] If $\mathbf{D}$ is a bounded subset of $\mathbf{R}^n$, then any function which is Lipschitz continuous in $\mathbf{D}$ is also bounded in $\mathbf{D}$.

$$\begin{aligned}
|(f_1f_2)(x)-(f_1f_2)(y)| &= |f_1(x)f_2(x)-f_1(y)f_2(y)| \\
&= |[f_1(x)-f_1(y)]f_2(x)+[f_2(x)-f_2(y)]f_1(y)| \\
&\le |f_1(x)-f_1(y)|\,|f_2(x)|+|f_2(x)-f_2(y)|\,|f_1(y)| \\
&\le \lambda_1\hat{f}_2\|x-y\|+\lambda_2\hat{f}_1\|x-y\| \\
&= (\lambda_1\hat{f}_2+\lambda_2\hat{f}_1)\|x-y\|
\end{aligned}$$

Therefore f_1f_2 is Lipschitz continuous in D and is bounded by $\hat{f}_1\hat{f}_2$.

Now, if f_2 is bounded away from zero, then

$$\begin{aligned}
\left|\frac{1}{f_2(x)}-\frac{1}{f_2(y)}\right| &= \left|\frac{f_2(y)-f_2(x)}{f_2(y)f_2(x)}\right| \\
&\le \frac{\lambda_2}{\bar{f}_2^2}\|x-y\|
\end{aligned}$$

That is $\frac{1}{f_2}$ is Lipschitz continuous in D and is bounded by $\frac{1}{\bar{f}_2}$. Therefore by using the previous result, the product $f_1\frac{1}{f_2}=\frac{f_1}{f_2}$ is also a bounded Lipschitz continuous function in D. ∎

Proof of Theorem 4.1. Define

$\mathbf{D} \overset{\Delta}{=} \{t\in[0,T] \mid t$ is a discontinuity point of either $u(\cdot)$, $\dot{x}^0(\cdot)$ or $z^0(\cdot)\}$

$\mathbf{X}\times\mathbf{Z} \overset{\Delta}{=} \{(x(\cdot),z(\cdot)):[0,T]\to\mathbf{R}^n\times\mathbf{R}^l \mid \dot{x}(\cdot)$ and $z(\cdot)$ are piecewise continuous with possible discontinuity points in $\mathbf{D}\}$

$\mathbf{F} : \mathbf{X}\times\mathbf{Z}\to\mathbf{X}\times\mathbf{Z}$ such that $\begin{bmatrix}\tilde{x}(\cdot)\\ \tilde{z}(\cdot)\end{bmatrix}=\mathbf{F}\begin{bmatrix}x(\cdot)\\ z(\cdot)\end{bmatrix}$ satisfies

$$\dot{\tilde{x}} = f(\tilde{x},x,\dot{x},z,u);\quad \tilde{x}(0)=x(0) \tag{A4.1a}$$

$$\tilde{z} = g(\tilde{x},x,\dot{x},z,u) \tag{A4.1b}$$

Since f is Lipschitz continuous with respect to $\tilde{x}$, by the Picard-Lindelof Theorem on the existence and uniqueness of the solutions of ordinary differential equations (see [17] page 18), the above equations have a unique solution $(\tilde{x}(\cdot), \tilde{z}(\cdot))$ which belongs to X×Z. Therefore **F** is well defined. From these definitions the canonical WRM algorithm described by (4.6) can be written as

$$\begin{bmatrix} x^k(\cdot) \\ z^k(\cdot) \end{bmatrix} = \mathbf{F}\begin{bmatrix} x^{k-1}(\cdot) \\ z^{k-1}(\cdot) \end{bmatrix} \tag{A4.2}$$

Let $\hat{\gamma} = \max(\gamma, 0.1)$ and α be a positive constant whose value will be chosen later. Define norms in $\mathbf{R}^n \times \mathbf{R}^l \times \mathbf{R}^n$ and X×Z as follows:

$$\left\| \begin{matrix} \dot{x}(t) \\ z(t) \\ x(t) \end{matrix} \right\| \triangleq \left\| \begin{matrix} \dot{x}(t) \\ z(t) \end{matrix} \right\| + \frac{\lambda_2}{\hat{\gamma}} \| x(t) \| \tag{A4.3}$$

$$\left\| \begin{matrix} x(\cdot) \\ z(\cdot) \end{matrix} \right\| \triangleq \max_{t \in [0,T]} e^{-\alpha t} \left\| \begin{matrix} \dot{x}(t) \\ z(t) \\ x(t) \end{matrix} \right\| \tag{A4.4}$$

where γ, λ_2 and the norms in $\mathbf{R}^n \times \mathbf{R}^l$ and $\mathbf{R}^n$ are as given in Theorem 4.5. Since

$$\left\| \begin{matrix} x(\cdot) \\ z(\cdot) \end{matrix} \right\| \geq e^{-\alpha T} \max_{t \in [0,T]} \left\| \begin{matrix} \dot{x}(t) \\ z(t) \\ x(t) \end{matrix} \right\|$$

Therefore, it can be shown that the space $(X \times Z, \|\cdot\|)$ is closed (hence it is a Banach space). Next we shall show that **F** is contractive in $(X \times Z, \|\cdot\|)$.

Let $\begin{bmatrix} \tilde{x}^1(\cdot) \\ \tilde{z}^1(\cdot) \end{bmatrix} = \mathbf{F}\begin{bmatrix} x^1(\cdot) \\ z^1(\cdot) \end{bmatrix}$ and $\begin{bmatrix} \tilde{x}^2(\cdot) \\ \tilde{z}^2(\cdot) \end{bmatrix} = \mathbf{F}\begin{bmatrix} x^2(\cdot) \\ z^2(\cdot) \end{bmatrix}$, then from (A4.1) and the condition (b) of Theorem 4.5, we have that

$$\left\| \begin{matrix} \dot{\tilde{x}}^1(t) - \dot{\tilde{x}}^2(t) \\ \tilde{z}^1(t) - \tilde{z}^2(t) \end{matrix} \right\| = \left\| \begin{matrix} f(\tilde{x}^1, x^1, \dot{x}^1, z^1, u) - f(\tilde{x}^2, x^2, \dot{x}^2, z^2, u) \\ g(\tilde{x}^1, x^1, \dot{x}^1, z^1, u) - g(\tilde{x}^2, x^2, \dot{x}^2, z^2, u) \end{matrix} \right\|$$

$$\leq \lambda_1 \|\tilde{x}^1(t) - \tilde{x}^2(t)\| + \lambda_2 \|x^1(t) - x^2(t)\| + \gamma \left\| \begin{matrix} \dot{x}^1(t) - \dot{x}^2(t) \\ z^1(t) - z^2(t) \end{matrix} \right\|$$

After some algebraic manipulations using (A4.3), $\hat{\gamma}$ and the above inequality, we have that

$$\left| \begin{matrix} \dot{\tilde{x}}^1(t) - \dot{\tilde{x}}^2(t) \\ \tilde{z}^1(t) - \tilde{z}^2(t) \\ \tilde{x}^1(t) - \tilde{x}^2(t) \end{matrix} \right| \leq \hat{\gamma} \left| \begin{matrix} \dot{x}^1(t) - \dot{x}^2(t) \\ z^1(t) - z^2(t) \\ x^1(t) - x^2(t) \end{matrix} \right| + (\lambda_1 + \frac{\lambda_2}{\hat{\gamma}}) \|\tilde{x}^1(t) - \tilde{x}^2(t)\| \quad \text{(A4.5)}$$

From Lemma A2, there exists a constant μ such that

$$\|\dot{\tilde{x}}^1(t) - \dot{\tilde{x}}^2(t)\| \leq \mu \left| \begin{matrix} \dot{\tilde{x}}^1(t) - \dot{\tilde{x}}^2(t) \\ \tilde{z}^1(t) - \tilde{z}^2(t) \\ \tilde{x}^1(t) - \tilde{x}^2(t) \end{matrix} \right| \quad \text{(A4.6)}$$

From (A4.5) and (A4.6), letting $\mu_1 = \mu\hat{\gamma}$ and $\mu_2 = \mu(\lambda_1 + \frac{\lambda_2}{\hat{\gamma}})$, we have

$$\|\dot{\tilde{x}}^1(t) - \dot{\tilde{x}}^2(t)\| \leq \mu_1 \left| \begin{matrix} \dot{x}^1(t) - \dot{x}^2(t) \\ z^1(t) - z^2(t) \\ x^1(t) - x^2(t) \end{matrix} \right| + \mu_2 \|\tilde{x}^1(t) - \tilde{x}^2(t)\| \quad \text{(A4.7)}$$

Applying the fundamental results in differential inequalities to (A4.7) (see [17] corollary 6.2, pages 30-32), using the fact that $\tilde{x}^1(0) - \tilde{x}^2(0) = 0$, we have

$$\|\tilde{x}^1(t) - \tilde{x}^2(t)\| \leq \mu_1 e^{\mu_2 t} \int_0^t e^{-\mu_2 \tau} \left| \begin{matrix} \dot{x}^1(\tau) - \dot{x}^2(\tau) \\ z^1(\tau) - z^2(\tau) \\ x^1(\tau) - x^2(\tau) \end{matrix} \right| d\tau$$

$$= \mu_1 e^{\mu_2 t} \int_0^t e^{(a-\mu_2)\tau} e^{-a\tau} \left| \begin{matrix} \dot{\tilde{x}}^1(t) - \dot{\tilde{x}}^2(t) \\ \tilde{z}^1(t) - \tilde{z}^2(t) \\ \tilde{x}^1(t) - \tilde{x}^2(t) \end{matrix} \right| d\tau \quad \text{(A4.8)}$$

From (A4.4) we have

$$e^{-\alpha\tau}\left|\begin{matrix}\dot{x}^1(\tau)-\dot{x}^2(\tau)\\ z^1(\tau)-z^2(\tau)\\ x^1(\tau)-x^2(\tau)\end{matrix}\right| \le \left\|\begin{matrix}x^1(\cdot)-x^2(\cdot)\\ z^1(\cdot)-z^2(\cdot)\end{matrix}\right\| \qquad \text{for all } \tau\in[0,T]$$

Substituting this inequality in (A4.8), we have

$$\begin{aligned}\|\tilde{x}^1(t)-\tilde{x}^2(t)\| &\le \mu_1 e^{\mu_2 t}\left\|\begin{matrix}x^1(\cdot)-x^2(\cdot)\\ z^1(\cdot)-z^2(\cdot)\end{matrix}\right\|\int_0^t e^{(\alpha-\mu_2)\tau}d\tau\\ &= \frac{\mu_1 e^{\mu_2 t}}{\alpha-\mu_2}\left\|\begin{matrix}x^1(\cdot)-x^2(\cdot)\\ z^1(\cdot)-z^2(\cdot)\end{matrix}\right\|(e^{(\alpha-\mu_2)t}-1)\\ &\le \frac{\mu_1 e^{\alpha t}}{\alpha-\mu_2}\left\|\begin{matrix}x^1(\cdot)-x^2(\cdot)\\ z^1(\cdot)-z^2(\cdot)\end{matrix}\right\| \qquad \text{assuming } \alpha-\mu_2>0\end{aligned}$$

Substituting this inequality in (A4.5) and multiplying by $e^{-\alpha t}$, we have

$$e^{-\alpha t}\left|\begin{matrix}\dot{\tilde{x}}^1(t)-\dot{\tilde{x}}^2(t)\\ \tilde{z}^1(t)-\tilde{z}^2(t)\\ \tilde{x}^1(t)-\tilde{x}^2(t)\end{matrix}\right| \le \hat{\gamma}e^{-\alpha t}\left|\begin{matrix}\dot{x}^1(t)-\dot{x}^2(t)\\ z^1(t)-z^2(t)\\ x^1(t)-x^2(t)\end{matrix}\right| + \frac{\mu_1(\lambda_1+\frac{\lambda_2}{\hat{\gamma}})}{\alpha-\mu_2}\left\|\begin{matrix}x^1(\cdot)-x^2(\cdot)\\ z^1(\cdot)-z^2(\cdot)\end{matrix}\right\|$$

Hence, using (A4.4), the above inequality implies that

$$\left\|\begin{matrix}\tilde{x}^1(\cdot)-\tilde{x}^2(\cdot)\\ \tilde{z}^1(\cdot)-\tilde{z}^2(\cdot)\end{matrix}\right\| \le \hat{\gamma}\left\|\begin{matrix}x^1(\cdot)-x^2(\cdot)\\ z^1(\cdot)-z^2(\cdot)\end{matrix}\right\| + \frac{\mu_1(\lambda_1+\frac{\lambda_2}{\hat{\gamma}})}{\alpha-\mu_2}\left\|\begin{matrix}x^1(\cdot)-x^2(\cdot)\\ z^1(\cdot)-z^2(\cdot)\end{matrix}\right\|$$

That is

$$\left\|F\begin{bmatrix}x^1(\cdot)\\ z^1(\cdot)\end{bmatrix}\right\| \le \left[\hat{\gamma}+\frac{\mu_1(\lambda_1+\frac{\lambda_2}{\hat{\gamma}})}{\alpha-\mu_2}\right]\left\|\begin{matrix}x^1(\cdot)-x^2(\cdot)\\ z^1(\cdot)-z^2(\cdot)\end{matrix}\right\|$$

Since $0.1 \le \hat{\gamma} < 1$, it is possible to choose α large enough so that

$$\hat{\gamma}+\frac{\mu_1(\lambda_1+\frac{\lambda_2}{\hat{\gamma}})}{\alpha-\mu_2} < 1$$

Therefore F is contractive. Hence, by the contraction mapping theorem (see

[15] page 120), it has a unique fixed point in X×Z satisfying

$$\begin{bmatrix} \hat{x}(\cdot) \\ \hat{z}(\cdot) \end{bmatrix} = \mathbf{F}\begin{bmatrix} \hat{x}(\cdot) \\ \hat{z}(\cdot) \end{bmatrix}$$

i.e.

$$\dot{\hat{x}} = f(\hat{x}, \hat{z}, \dot{\hat{x}}, \hat{z}, u)\,; \quad \hat{x}(0) = x(0)$$

$$\hat{z} = g(\hat{x}, \hat{z}, \dot{\hat{x}}, \hat{z}, u)$$

and for any given initial guess $(x^0(\cdot), z^0(\cdot)) \in$ X×Z the sequence $\{(x^k(\cdot), z^k(\cdot))\}_{k=1}^{\infty}$ generated by (A4.2) converges uniformly to $(\hat{x}(\cdot), \hat{z}(\cdot)) \in$ X×Z. Since we can choose $(x^0(\cdot), z^0(\cdot)) \in$ X×Z such that $x^0(t) = x(0)$ and $z^0(t) = 0$ for all $t \in [0,T]$, therefore we conclude that the discontinuity points of $(\hat{x}(\cdot), \hat{z}(\cdot))$ belong to the set of discontinuity points of $u(\cdot)$ only, i.e. they do not depend on $\dot{x}^0(\cdot)$ and $z^0(\cdot)$. Hence the proof is complete. ∎

Proof of Theorem 5.3. Define

$$L,\, U : \mathbb{R}^n\times\mathbb{R}^n\times\mathbb{R}^r \to \mathbb{R}^{n\times n} \qquad f : \mathbb{R}^n\times\mathbb{R}^n\times\mathbb{R}^n\times\mathbb{R}^r \to \mathbb{R}^n$$

$$H : \mathbb{R}^n\times\mathbb{R}^n\times\mathbb{R}^r \to \mathbb{R}^{n\times n} \qquad \tilde{F} : \mathbb{R}^j\times\mathbb{R}^n\times\mathbb{R}^n\times\mathbb{R}^r \to \mathbb{R}^n$$

$$F : \mathbb{R}^n\times\mathbb{R}^n\times\mathbb{R}^r \to \mathbb{R}^n \qquad \tilde{f} : \mathbb{R}^n\times\mathbb{R}^n\times\mathbb{R}^n\times\mathbb{R}^j\times\mathbb{R}^r \to \mathbb{R}^n$$

as follows:

$$L_{ij}(a,b,u) \triangleq \begin{cases} 0 & ; \quad \text{if } i \le j \\ -\dfrac{C_{ij}(a_1, \ldots, a_i, b_{i+1}, \ldots, b_n, u)}{C_{ii}(a_1, \ldots, a_i, b_{i+1}, \ldots, b_n, u)} & ; \quad \text{if } i > j \end{cases}$$

$$U_{ij}(a,b,u) \triangleq \begin{cases} -\dfrac{C_{ij}(a_1, \ldots, a_i, b_{i+1}, \ldots, b_n, u)}{C_{ii}(a_1, \ldots, a_i, b_{i+1}, \ldots, b_n, u)} & ; \quad \text{if } i < j \\ 0 & ; \quad \text{if } i \ge j \end{cases}$$

$$H(a,b,u) \triangleq [I - L(a,b,u)]^{-1}U(a,b,u)$$

$$F_i(a,b,u) \triangleq -\frac{q_i(a_1,\ldots,a_i,b_{i+1},\ldots,b_n,u)}{C_{ii}(a_1,\ldots,a_i,b_{i+1},\ldots,b_n,u)}$$

$$f(a,b,s,u) \triangleq [I - L(a,b,u)]^{-1}F(a,b,u) + H(a,b,u)s$$

$$\tilde{F}_i(z,a,b,u) \triangleq -\frac{\tilde{q}_i(z,a_1,\ldots,a_i,b_{i+1},\ldots,b_n,u)}{C_{ii}(a_1,\ldots,a_i,b_{i+1},\ldots,b_n,u)}$$

$$\tilde{f}(a,b,s,z,u) \triangleq [I - L(a,b,u)]^{-1}\tilde{F}(z,a,b,u) + H(a,b,u)s$$

where C, q, $\tilde{q}$ are previously defined in Algorithm 5.1 and Algorithm 5.2. Then Algorithm 5.1 can be tranformed into the following canonical form

$$\dot{v}^k = f(v^k, v^{k-1}, \dot{v}^{k-1}, u) \qquad : \; v^k(0) = v(0)$$

And Algorithm 5.2 can be tranformed into the following canonical form

$$\begin{aligned} \dot{v}^k &= \tilde{f}(v^k, v^{k-1}, \dot{v}_{k-1}, z^{k-1}, u) \, ; \;\; v^k(0) = v(0) \\ z^k &= g(v^k, u) \end{aligned}$$

From Lemma A4 and the assumptions of Theorem 5.3, we can deduce that

a) f, $\tilde{f}$, g are continuous functions and are Lipschitz continuous with respect to v^k and v^{k-1}.

b) $\tilde{f}$ is also Lipschitz continuous with respect to z, i.e. there is a constant $\lambda>0$ such that

$$\|\tilde{f}(a,b,s,z,u) - \tilde{f}(a,b,s,\tilde{z},u)\|_\infty \le \lambda\|z - \tilde{z}\|_\infty \qquad \text{(A5.1)}$$

where $\|\cdot\|_\infty$ denotes the standard max-norm as defined in Lemma A3.

Applying the result of Lemma A3, we have

$$\|H(a,b,u)\|_\infty \le \|L(a,b,u) + U(a,b,u)\|_\infty$$

$$= \max_{1 \le i \le n} \frac{-\sum_{\substack{j=1 \\ j \ne i}}^{n} C_{ij}(a_1, \ldots, a_i, b_{i+1}, \ldots, b_n, u)}{C_{ii}(a_1, \ldots, a_i, b_{i+1}, \ldots, b_n, u)} \tag{A5.2}$$

From the definition of C and the assumptions of Theorem 5.1, we have

$$C_{ii}(a_1, \ldots, a_i, b_{i+1}, \ldots, b_n, u) \le C_{\min} - \sum_{\substack{j=1 \\ j \ne i}}^{n} C_{ij}(a_1, \ldots, a_i, b_{i+1}, \ldots, b_n, u) \tag{A5.3}$$

$$-\sum_{\substack{j=1 \\ j \ne i}}^{n} C_{ij}(a_1, \ldots, a_i, b_{i+1}, \ldots, b_n, u) \le (n-1)C_{\max} \tag{A5.4}$$

Since, for all $\sigma \le 0$, the function $\frac{\sigma}{C_{\min} + \sigma}$ is monotonically increasing with respect to σ therefore (A.15), (A.16) and (A.17) imply that

$$\|H(a,b,u)\|_\infty \le \frac{(n-1)C_{\max}}{C_{\min} + (n-1)C_{\max}} = \gamma < 1 \tag{A5.5}$$

Therefore Algorithm 5.1 satisfies all the conditions of Theorem 4.1 and hence it converges for any piecewise continuous input u.

For Algorithm 5.2, we define $\|\cdot\|$ in $R^n \times R^l$ such that for any $s \in R^n$ and $z \in R^l$

$$\left\| \begin{matrix} s \\ z \end{matrix} \right\| \triangleq \|s\|_\infty + \frac{\lambda}{\gamma}\|z\|_\infty$$

where λ is as given in (A5.1) and γ is as given in (A5.5). Then

$$\left\| \begin{matrix} \tilde{f}(a, b, s, z, u) - \tilde{f}(a, b, \tilde{s}, \tilde{z}, u) \\ g(a, u) - g(a, u) \end{matrix} \right\| = \|\tilde{f}(a, b, s, z, u) - \tilde{f}(a, b, \tilde{s}, \tilde{z}, u)\|_\infty$$

$$\le \lambda\|z - \tilde{z}\|_\infty + \|H(a,b,u)\|_\infty\|s - \tilde{s}\|_\infty$$

$$\le \gamma\left[\, \|s - \tilde{s}\|_\infty + \frac{\lambda}{\gamma}\|z - \tilde{z}\|_\infty \,\right]$$

$$= \gamma \left\| \begin{matrix} s - \tilde{s} \\ z - \tilde{z} \end{matrix} \right\|$$

Therefore Algorithm 5.2 also satisfies the conditions of Theorem 4.1 and hence it converges for any piecewise continuous input u. ∎