Distributed Simulation of hybrid systems or integrated circuits (Analogue/Digial)

Bruno Böttcher
Main supervising Professor
Prof Dr. -Ing. Dr. h.c. D. Mlynski
Koreferent
Dr. Z. Kuznicki

01.1995 - 31.12.1996
Version 5A19,
from Tue Oct 24 16:38:07 MET 1995

This is a preliminary Version of my Thesis thus still under continuous development.

Internal use only!!

Abstract

The aim was to find a way to simulate big mixed mode circuits in competitive time. Mixed mode here means that we have for simulation a system which consist of digital parts coupled to analog parts.

Contents

Chapter 1

Introduction

In the last Years analog chip design made big progress, thus becoming competitive with digital design. Lots of functionalities are easier to implement in analog way (applications for telecommunication, high definition TV, etc.). On the other hand, the increasing clock frequencies imposed to digital circuits make analog verification imperative and today both can be implemented in C-MOS [Taha93].

The evolution of circuit design goes therefore towards analog/digital mixed design.

[Magi92] makes a comparision of the main high description level languages (HDL's) that claim to cope with analog/digital mixed design.

A general description of HDL's is achieved at the example of VHDL, VeriLog and M. As can be extracted from this paper: VHDL is a very unspecified general language good at high abstraction level, minor at low level (e.g. switch level), in confront to that a specification like UDL/I can onlx be seriously used for chip layout design (At the moment there are no tools for using UDL/I).

Big efforts are made from the IEEE committees to build an extension to the existing VHDL standard to make it viable with analog concepts.

Since VHDL is a standard and the extension will also be standardized, further because this standard is not proprietary to any commercial organization, this languages was choosen to be the top level of our mixed mode, parallel simulator.

The following chapters give first an introduction to parallel computing strategies, continue on the problematics of analog simulation for finally arriving at the specifities of mixed mode simulation and continuing towards the retained concepts for a VHDL-A compliant simulator.

Chapter 2

Parallel execution

In the following section are discussed the main strategies for distribution. [Righ89] has made an overview. The main points in his research point out the problem of finding the next element to execute and the load balancing problem. These factors influence mainly the gain of the parallel execution.

2.1 The load balancing problem

Synchronous event driven simulation means that a central event sceduler defines a list of tasks to be executed. It is the easiest way to parallelize, but the synchronization stage done by the master represents a bottleneck. This strategy works only well with lots of big tasks to distribute on few processors (see [Soul88]) because of increasing queue contention [Dand95] (means simultaneous requests to the queue). In confront to other strategies the load balance can easily be achieved.

Distributed event queues cope very well with huge amounts of processors but offer only inferior load balancing.

A try was given by statically partitionned problems, here even better load balancing was reached but unless to have an optimal partitioner a good syncronous algorithm performs better (see [Soul88]).

2.1.1 Hierarchical queue system

[Dand95] presents a compromise offering the best out of the two extremes. A variable number of intermediary queues are arranged between the central queue and the distributed ones (e.g. arranged as a binary tree). Empty queues request input from the next higher queue, thus distributing the contention problem.

The two extreme cases can be achieved using either n-leave one dimensional tree for the distributed case or the one node one dimensional tree for the central queue system.

With this system a better work load with little speed loss was achieved versus the distributed case.

2.1.2 Circuit Activity

[Soul88] evaluates to 10-15% the gain to be obtained by asynchronous strategies versus synchronous. This due to the fact that in the examples they examined they had only little global activity (less than five events at 50% of the time). In a previous publication (see [Soul87]) they tested digital circuits at different abstraction levels.

In their study they found 0,1-0.5% activity over the whole circuit size at element level. They deducted a maximal speedup of 10-30 for those circuits for parallel execution.

For each abstraction level added they foresee a speedup factor of ten.

To achieve reasonable speedups [Soul87] proposes to use either

- Hardware accelerators, unflexible but 100-10^6 times faster as a software solution
- multiple Abstraction level simulator: only the critical parts are simulated at needed precision, the rest is simulated at maximal abstraction level. (see also [Bile85])

2.2 Asynchronous Event driven simulation: Pessimistic Strategies

In [Misr86]: Logical processes (LP) are linked to physical processes (PP).

Causality must be respected, the only break to this rule: we can extract the most probable future. out of the past behaviour

All LP are sequential code. Connected PP are represented as LP connected through communication channels. Each passed message has a timestamp.

Problem with this strategy: **deadlocks**, respecting causality means that the message with the lowest time stamp is processed first. Circular data flow arrangements produce deadlock situations, since the first element needs the output of the last one to produce the message that will trigger the last element.

It seems better to recover from a deadlock than to prevent it, see [Chan81].

Deadlock resolution

- introduction of **Null Messages**: the last message an LP sends is a null-message. this tells the other LP's that this process will send no more messages at this time point.
- **variation**: a null-message is discarded if followed by a valid message. This is done in the input buffer, which must therefore be analyzed for this scope. The sending of null-messages is delayed by a factor τ in the hope of messages.

- An untested method is the **null-message on demand**: an idle LP asks its links for time increment. If there is a dead lock the request will come back to this same LP and therefore be detected.

The critical importance of lookahead (see p. 7) was shown by [Fuji88]. The null-message system behaves well only in the case of message-avalanche, a factor coupled with the lookahead. [Fuji88] proposes to better use Time Warp strategies when solving problems with poor lookahead.

The Problem with deadlock recovery is not an easy problem since the smallest timestep and the blocking unit must be found.

[Wagn88] examined the problems of link time in sight of a shared memory sytem. they introduced as replacement for null messages flags for every process. They achieved good speedups with this mehtod.

Deadlock detection with token

A token circulates in the Network and records all nodes that have not send any message since the last visit. If all visited nodes are in the record there is deadlock.

The offending node can be extracted out of the history of the token.

possible **variation**: random visit of the token or generation of tokens for subnetworks.

The problem with all this deadlock detection and recovery is that it adds lot of network overhead on a distributed system. On a shared memory system with memory locks and global variables deadlock can be easily found and healed; see [Wagn88]

2.3 Asynchronous Event driven simulation: Optimistic Strategies

[Jeff84] makes a performance analysis of timestamp driven synchronization mechanisms. Methods for measuring performance or message overhead caused by preemption (arrival of messages with lower timestamps as current time) are presented. The concept of **Roll Back** is presented: when preemption occurs the offending processes must be brought back to the state it had to the time of the preemptive message.

In [Jeff85] the previous concepts are enlarged. The concept of time is brought to an analogy of memory in computer systems. The strategies for virtual memory are extended to what now is called **virtual time**.

The virtual time is represented as a one dimensional vector used from all processes as time identification, thus allowing synchronization.

The classic partitioning methods:

- synchronization with locks

- partition of the execution in "Atoms". For those atoms "abortion-retry" is allowed, rollback is allowed but only to the beginning of the actuall transaction time.

Time Warp

is a lookahead method using rollback as synchronizatio Each atom executes independently from the others wether there could be synchronization conflict or not.

If a conflict occurs, the individual processes are brought back to the state at the conflicting time.

The system is thought for a **message passing** machine:

The messages are more mail like as pipe like (means no channels). All messages are stamped with:

- sendername
- virtual send time
- receiver name
- virtual receive time

Semantic rules

- the virtual send time of a message must be less than the virtual receive time
- the virtual time of each event in a process must be less than the virtual time of the next event in that process

Lamports Clock condition: If event A causes event B, execution of A&B must be sceduled in real time so that A is completed before B.

A central problem: the next message in the queue is not necessarily the next message in real time.

Local control mecanism

Events are executed and messages are ordered by their timestamps.

Each process has its own virtual clock and runs as far and as fast it can. Each process has a FIFO like queue.

The local clock changes at the execution of an event to the timestamp of the message inducting that event.

Here is introduced the concept **lookahead**. lookahead means that future messages are processed.

The processes are sceduled in the rank of the timestamp of the messages.

The reasons why this strategy should be better than the others:

- distributed rollback can be elegantly (means easily) implemented.
- distributed rollback inserts dynamically, without blocking the whole system

Global control mecanism

The local control described in the previous sections does not provide the complete required functionality. Some points are defined globally:

- mesure of global progress
- detection of global termination
- error handling
- memory managment
- global virtual time

Global virtual time (GVT)

The global virtual time (GVT) is defined as min(localtime, send unprocessed messages time)

GVT progresses forward never backwards, it represents the minimum time of the system.

Estimation is performed making a broadcast of the request (O(delay)). Lots of different methods for evaluation of GVT exist, as usual they can be split in rapid, bandwith expensive and slow memory expensive methods.

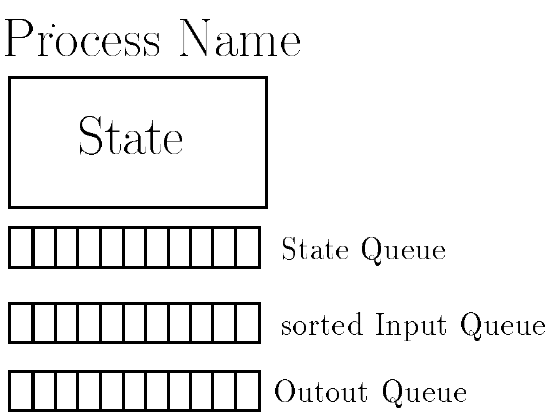
All data in the queues that has a timestamp anterior to the GVT is declared **fossil** and thus can be eliminated.

The buffersizes are fixed, if the buffers are full new messages are refused if their timestamp is greater than the one of the last element in the queue. If not the last element is send back to the sender forcing a rollback of this process (consuming time) and the arriving message is inserted at the right place in the queue.

Personnally I would prefer to see an appreciation of how often this case arrises and if this recalculation is really necessary

RollBack

At occurence of Rollback all the effects of the further processing must be annihilated. Therefore each process must have the following structure:



In the state Queue is recorded at least one state anterior to the global virtual time.

Old Messages with timestamps > GVT in the Input Queue are kept and all outgoing Messages are recorded in the ouput queue, they are also called **Anti Messages**. Those messages are send in case of Roll Back.

All messages are signed, a "-" sign defines an **anti-message**, when message and anti message met in an input queue they annihilate each other.

In the case of sending an antimessage the corresponding message is removed from the output queue of the sender.

If a process encounters an antimessage of an alredy processed message then it rollback too.

Normal Termination

Normaly this is not a trivial problem, but in this case we can say that when GVT reaches +infinite then there are no more messages to be processed and the simulation finishes.

Error Handling

Error in processing can occur, these errors can be caused by erroneous input data that can be recovered by a rollback. Thus when an error occurs a error message is send out and only when this error message becomes fossil a general error is sent out.

Input/Output

Out sended requests can be canceled until they are stamped as fossil. Therefore for executing an input/output request it must be waited until the request is fossil.

Conclusion

Unfortunately this strategy seems to fail on tightly coupled systems, where the high number of feed back loops produce rollback at high frequency. It is like the analogy to virtual memory, all works fine since the swapped memory parts are asked. [Jeff85] pretends that there is no possible dominoeffect with the rollback mecanism and that in real circuits "time locality" predomains (meaning that messages tend to arrive in the virtual future or near past, producing short rollbacks). But for non ideal circuits the overhead due to rollback and anti messages can be quite big. This can rapidly reduce the gain produced by this method. The clock and the rank ordering of latch input have great impact on the results as reported in [Soul87].

Tools using Time Warp

JADE Simulation environment, C++ based, see [Baez94]

[Wiel89] compared three different algorithms for combat simulation, his time warp implementation showed nearly linear speedup. He too evidenciated the importance of load balancing. Extensive graphics of his results are in the annex of his publication. In [Wiel89b] they implemented two aproaches: event driven and time driven on a message passing system. The event driven performed better than the time driven. This showed that the expected communication versus computation ratio determines which system performs better. High ratios seems to be better connected to an event driven simulator.

[Pres89] presented some benchmarks. They showed speedup for time warp but used extremely memory hungry methods.

2.3.1 Mixed Strategies

Bounded Lag

[Luba88] describes the coordination for parallel execution of mutually dependent events.

Two main possibilites are offered, the coordination can be synchronous or asynchronous. The synchronous strategy, also called lock step strategy consists of processing concurrent events at the same time point. This has the advantage that there will be no conflicts with causality, on the other hand this system is not very flexible.

Therefeore [Luba88] prefers the asynchronous strategy making no restriction about the time.

The here described Algorithm is based on the original Chandy Misra algorithm [Chan81],[Misr86].

The name for this variance is **Bounded Lag** and is based on the definition of a "time Window". This means that the virtual time can vary inside of a specified time window, providing some of the advantages that showed [Jeff84], [Jeff85].

- a directed graph is associated to the system, which regroups the influence the nodes towards each other and the propagation delay between them.
- Synchronization signals are send out
- Theory: only the events in the " Incoming reachability cone" are taken into acount for immediate processing (Ereignis Horizont).

A variant of the bounded lag Algorithm is the **Wolf algorithm**: when a node has to roll back, it sends a stop signal to all the nodes in its **influence sphere**. Influence sphere means here all the nodes the straggler node has send messages with a time stamp greater than the straggle time. Only if the states of the node change, a normal rollback message is issued, else the normal flow of operation continues (lazy cancellatation).

The idea behind this is that a broadcast has the same cost as a single message (Anti message in this case). Stopping all connected processes makes sure that no further "infection" is made, since under normal time warp the antimessage has the same delays as a message and that this delay causes eventually further erroneous calculation.

An other strategy for lazy cancellation is presented by [Pala92]. Since the compare actions needed for lazy cancellation are CPU intesive, they propose to insert a boolean variable with each process indicating if the process changed state in confront of the previous timestep. When a rollback occurs and the previous state changed aggressive rollback takes place.

No analitical prove exists for the Chandy Misra algorithm about its scalability.

In [Luba89] is showed that the bounded lag strategy is scalable to big examples.

This was shown on the example of a queuing network simulated on a shared memory system. Increasing the problem size and the number of processing units at the same ratio produced a time increase of no more than twice the previous time.

Optimization for Gate Level Simulation

[Hopp88] proposes to make optimizations in function of the problem to simulate.

There are several ways to achieve parllel acceleration, three of them are:

- Optimization of the input stimuli in sight of parallel execution
- task partitioning (pipelined architecture)
- circuit partitioning

[Hopp88] sees maximum parallelism in the asynchronous simulation of subcircuits with minimal global synchronization.

Therefore he proposes a modified Time Warp algorithm tailored for gate-level simulation:

The processes linked to the components have no memory, their previous state can be computed with the input queue. This produces sensible memory saving.

The rollback are different for the messages and antimessages, in opposition to the classical rollback:

Messages do rollback in the classical scheme.

Antimessages do first attempt to find their corresponding message and annihilate before doing a classical rollback. This behaviour is called **lazy rollback**

Limited Optimistic Approach

[Manj93] proposes a method for logic simulation on a work station pool.

The method is specialized for gate-level simulation (latches). The partitioning is achieved at that level. The optimal partition is found making a preliminary simulation run. In the average case the results are better for the optimistic approach exept for circuits with long feed backs.

The best approach was found to be an global link time algorithm and local time Warp.

Optimization of the next event search

[Ishi] proposes to analyse the circuit and to compile a databank with the correlation factors of the signals.

A task can subsequently only be taken in to the event list if all the connected input signals are resolved.

SPEEDES System

It is a variation of the Bounded lag algorithm from [Stei91]: All process run forward optimistically (locally regrouped process run like in Time Warp) since their local time reaches the predefined time window border. Meanwhile all issued outgoing messages are collected. After reaching the dead lock the new time window is defined and the valid messages delivered.

This way no general Rollback is ever needed and human interaction can be inserted.

They implemented a software package that include the most well known strategies. Switching between them at runtime is built in. the work continued to extend this package to parallel approach.

a more pessimistic approach based on Charm++ can be found in [Kona94], the main problem they have is the synchronization step. It blocks all the processes at regular intervalls, reducing significantly the speed gain.

Chapter 3

Analog Simulation

3.1 Different ways of describing circuits

3.1.1 Nodal Approach (NA)

$$\mathbf{YV} = \mathbf{J} \tag{3.1}$$

$\mathbf{Y}$ = admittance matrix;
$\mathbf{V}$ = node voltages (except GND), state variables;
$\mathbf{J}$ = currents; Built from KCL

Problem: No possibility to describe current dependent elements and voltage sources.

3.1.2 Modified Nodal Approach (MNA)

$$\begin{bmatrix} \mathbf{Y_R B} \\ \mathbf{CD} \end{bmatrix} \begin{bmatrix} \mathbf{V} \\ \mathbf{I} \end{bmatrix} = \begin{bmatrix} \mathbf{J} \\ \mathbf{F} \end{bmatrix} \tag{3.2}$$

$\mathbf{Y_R}$ = reduced matrix as above, reduced by current controlling elements and voltage sources.
$\mathbf{B}$ = partial derivatives of the KC equations, with resp. to additional current variables
$\mathbf{C}$,$\mathbf{D}$ = branch constitutive relations, differentiated w.r.t. unknown vector
$\mathbf{J}$ = currents, s. NA
$\mathbf{F}$ = exitations from previous timesteps corresponding to capacitors and inductors, represented in terms of implicit integration scheme. ex.:

$$i_c(t) = C\frac{dv_c}{dt} \Longrightarrow i_c(t_n) = \frac{C}{h}[v_c(t_n) - v_{cp}] \tag{3.3}$$

$v_{cp} = v_c$, previous time step; becomes part of $\mathbf{F}$
Nonlinearities must be linearized around some operating point before introducing them into the equation systems.
Pivoting schemes exist do exploit sparsity to the highest degree possible.[Ho75]

3.1.3 Tableau Approach (TA)

TA makes a very sparse tableau of the circuit. All node voltages, branch voltages, branch currents, designable parameters, fluxes and charges become state variables.[Hach71] TA does not seem to be very much used nowadays.

3.1.4 General Nonlinear Differential Equation System (NDE)

Idea: as in MNA $\mathbf{v}$ is the vector of node voltages and to be determined currents.

$$\mathbf{F}(\mathbf{v}(t), \dot{\mathbf{v}}(t), \mathbf{p}(t)) = \mathbf{0} \tag{3.4}$$

$\mathbf{p}$ is the vector of inputs.
A more common way to formulate equation systems in the circuit simulation context is:

$$\mathbf{i}(\mathbf{v}, t) + \dot{\mathbf{q}}(\mathbf{v}, t) = \mathbf{0} \tag{3.5}$$

Here the vector $\mathbf{v}$ represents the state variables, inputs included.
([Whit85/2], [Pete92] for explanation of $\mathbf{i}, \dot{\mathbf{q}}$)

3.1.5 SPICE description

In SPICE a different approach is chosen:

$$\mathbf{i}(\mathbf{v}, t) + \mathbf{C}(\mathbf{v}, t)\dot{\mathbf{v}}(t) = \mathbf{0} \tag{3.6}$$

Because of this approach in SPICE every ciruit node must be connected to ground with a capacitor. The capacitor may be of no significance to the circuit but it has to be there. This description has some draw backs as accuracy is concerned. For further detail check: [Pete92]

3.2 Replacing Differentiation with Difference Expressions

Usuallly the time derivatives in a NDE system have to be replaced by some more or less accurate difference equivalent in order to compute values for the state variables at specific timepoints. Thus nonlinear algebraic equation (NAE) systems are produced. The following methods are demonstrated using a simple one dimensional example:

$$f(v(t)) + \dot{v}(t) = 0 \tag{3.7}$$

In the more general case of vector functions the same ideas hold in conjunction with Gauss-Jacobi or Gauss-Seidel iteration schemes (described further below). Implicit methods (explained below) are often referred to as Backward Differentiation Formulae (BDF).

3.2.1 Explicit Euler (EE) Method

EE is the most intuitive way to replace the time derivative of v.

$$\dot{v}(t) = \frac{\Delta v}{\Delta t}$$

$$\Delta v = v_{n+1} - v_n$$

$$\Delta t = h_{n+1}$$

$$v_{n+1} = v_n - h_{n+1} f(v_n) \tag{3.8}$$

Explicit means that the new to be computed v_{n+1} depends only on states and inputs at the previous time point. The problem with EE is that the time step has to remain small enough to guarantee numeric stability of the method. The complex poles of the considered system have to be in a circle around $-1/h_{n+1}$ of radius $1/h_{n+1}$. This is a severe limitation for stiff systems.

3.2.2 Implicit Euler (IE) Method

In IE the problem of stability is solved by making the new to be computed v_{n+1} dependent on itself, i.e. it shows up on both sides of the equation:

$$v_{n+1} = v_n - h_{n+1} f(v_{n+1}) \tag{3.9}$$

IE is stable for all poles outside the circle around $-1/h_{n+1}$ of radius $1/h_{n+1}$.

3.2.3 Trapezoidal Rule (TR)

In TR the following mixture of past and present states on the right-hand-side is used:

$$v_{n+1} = v_n - \frac{1}{2} h_{n+1} [f(v_n) + f(v_{n+1})] \tag{3.10}$$

TR is stable for the complete left half plane.

3.2.4 Gear

In Gear - method steps reaching further into the history are used to calculate the new value of v. Additionally weighting factors are introduced:

$$v_{n+1} = \sum_{r=0}^{k} \alpha_r v_{n-r} - h_{n+1} \beta f(v_{n+1}) \tag{3.11}$$

For $k = 1$ the method becomes the two-step second order formula. It does not make much sense to use more back values of v as no A-stable formula (stable for poles in the left half plane) is ever of order greater than two.([Nich94])

3.2.5 General Formula

Most of the above formulae can be represented by the following general formula:

$$\sum_{i=0}^{k} \alpha_i v_{n+1-i} = -h_{n+1} \sum_{i=0}^{l} \beta_i f(v_{n+1-i}) \tag{3.12}$$

Thus, for $k = l = 1$, $\alpha_0 = 1$ $\alpha_1 = -1$, $\beta_0 = 0$ and $\beta_1 = 1$ one get's EE.
To make working integration methods the following conditions must hold:

$$\sum_{i=0}^{k} \alpha_i = -1$$

$$\sum_{i=1}^{k} \alpha_i + \sum_{i=0}^{l} \beta_i = 0$$

assuming that $\alpha_0 = 1$. [Whit85/2]
Using this general formula all the known integration formulae can be used to transfer NDE into NAE.
I have not yet found any program that would use any algorithm using more than 2 previous time steps for the computation of the new state. Possibilities would be: Simpson rule, Romberg inegration, TR for more than two time points . The stability regions of these methods have not been investigated so far.

3.2.6 Semi-Implicit Methods

Implicit methods may lead to NAE after applying BDF like IE:

$$\mathbf{v}_{n+1} = \mathbf{v}_n - h_{n+1}\mathbf{f}(\mathbf{v}_{n+1}) \tag{3.13}$$

with $\mathbf{f}(\mathbf{v}_{n+1})$ being a nonliner vector function. When linearizing these equations they are usually linearized around the previous timepoint and are then multiplied with a Δv composed of the to be computed and the most recent time step.

$$\mathbf{v}_{n+1} = \mathbf{v}_n - h_{n+1} \left[\mathbf{f}(\mathbf{v})_n + \left.\frac{\delta \mathbf{f}}{\delta \mathbf{v}}\right|_{\mathbf{v}_n} (\mathbf{v}_{n+1} - \mathbf{v}_n) \right] \tag{3.14}$$

Because of this mixture of previous and actual time point on the right hand side these methods are known as semi-implicit methods.
Their region of stability is good in all practical allplications but not the whole left half plane. I have not found any further hints on stability. The method seems to be reasonably robust.

3.3 Linearization Algorithms: Newton-Raphson (NR)

NR can be used for linearization purposes. The resultant LAE system is then further treated by other means. NR can also be used as a method for the solution of NAE. This approach is usually only chosen in a single variable problem as in multiple variable problems the cost of the matric inversion of the jacobian becomes too dominant.

$$\mathbf{v}_{n+1} = \mathbf{v}_n - \mathbf{J}^{-1}(\mathbf{v}_n)\mathbf{f}(\mathbf{v}_n) \tag{3.15}$$

with

$$\mathbf{J}^{-1}(\mathbf{v}_n) = \left.\frac{\delta \mathbf{f}}{\delta \mathbf{v}}\right|_{\mathbf{v}_n} \tag{3.16}$$

3.4 Solution of the Systems described in 3.1

Generally the input to an analogue simulator is a netlist of components. There type and name of the component are defined and the connexions to other components given.
A general form of transient analysis would be performed as described by picture 3.1. After feeding the simulator with the netlist the equations are extracted (e.g. Nodal analysis). At this stage is determined the initial state of the circuit (DC analysis, all Variables set to 0, etc.).
All the differential equations are discretized, and the inner loop entered.
There is solved, through iteration,the system of linearized, discretized differential equations. Time is increased in the outer loop.

3.4.1 Newton-Raphson (NR)

To solve a NAE system using NR check the paragraph above on linearization.

3.4.2 Secant Method

The secant method gets around computing derivatives. To find the solution to (cf [Bjoe72]):

$$f(x) = 0 \tag{3.17}$$

two starting values x_0 and x_1 are needed. The new value is computed as follows:

$$x_{n+1} = x_n + h_n \tag{3.18}$$

$$h_n = -f_n \frac{x_n - x_{n-1}}{f_n - f_{n-1}} \tag{3.19}$$

The iteration is ended when f_n and f_{n-1} don't differ too much. The advantage over NR is that calculation per iteration is easier and that no derivatives have to be

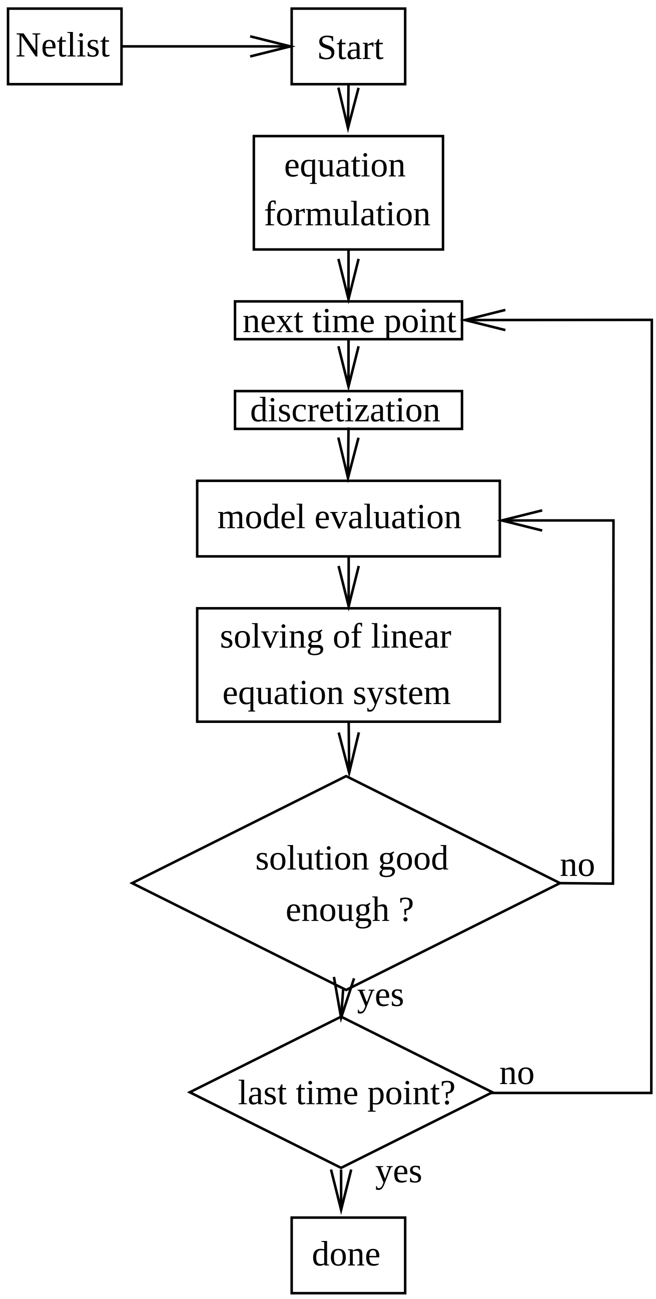


Figure 3.1: General flowchart of a typical transient analysis

computed. The draw back is that a suitable choice of the starting values is more crucial than in NR to guarantee convergence.

3.4.3 Direct Method (DM)

Solution of Linear Algebraic Equation Systems (LAE)

$$\mathbf{Ax} = \mathbf{b} \tag{3.20}$$

Either Gaussian-Elimination (GE) or LU-Factorization.
Programs to parallelize DM have been existing for a long time.
ex: Parallelism inherent in GE when all elements of one row/column have to be multiplied by the same factor ...

SPICE

SPICE executes the following steps:
1) TR to convert the system of NDE into a system of NAE
2) (damped) NR to convert to NAE into LAE
3) solve LAE using GE or LU-decomposition

If SPICE does not converge Gear may be chosen instead of TR. The region of stability for TR ideally is the whole left half plane. Sometimes the boundary becomes fuzzy if too many roundoff and truncation errors occur resulting in poles close to but left of the imaginary axis acting like poles in the right half plane. For further reference check: [Nich94]
Parallelisation: [Bisc86] shared memory multiprocessor system DIGITAL 8800/8300; most time consuming part in SPICE simulation run: calculation of solution for LAE system (about 80% for big systems). Parallelisation on 3 levels:
1) construction of the system matrix can be performed with multiple processors writing to common memory (load phase)
2) LU decomposition (poor efficiency for sparse systems)
3) time step calculation; each slave process is assigned a set of storage elements and computes the time step required by its set of energy storage elements independently.

3.4.4 Relaxation Methods

Linear Relaxation

Relaxation is applied to the system of LAE:

$$\mathbf{Ax} = \mathbf{b} \tag{3.21}$$

instead of solving with DM.

Nonlinear Relaxation

Relaxation is applied to the system of NAE, after the system of NDE has been converted to NAE system using (mostly) BDF methods.

Iterated Timing Analysis (ITA)

ITA is a nonlinear relaxation method.

repeat
  for $t = t_{start}$ to t_{end}
    convert NDE to NAE using BDF
    for $i = 1$ to n
      solve only one NR iteration for state variable i,
      with all the other state variables unchanged for the considered time point
      using GS iteration (explained below)
    next i
  next t
until all LTE small enough

with n =number of node-voltages, LTE = Local Truncation Error

Program: SPLICE
Person: Saleh '83
Literature:[Hung91]
Parallelisation described in [Hung91]. The approach relies on a circuit graph description. Whenever a fanin element of some component is changed the specific subcircuit is scheduled and calculated anew. (very optimistic). This is called selective trace technique. Not very convincing results.

One Step Relaxation (OSR)

OSR is a nonlinear relaxation method.

repeat
  for $t = t_{start}$ to t_{end}
    for $i = 1$ to n
    convert NDE to NAE using BDF
      repeat
        solve for state variable i, with all the other state variables
        unchanged for the considered time point
      using GS iteration (explained below)
      until exactly solved
    next i
  next t
until all LTE small enough

Program: ELDO of Anacad, VHDeLDO, Mix-VHDL of Mentor Graphics
There are but a few hints what solution algorithms are used in ELDO. In [Henn85/2] Hennion writes that OSR get's around computing derivatives by using the secant method instead of NR, also The difference of Waveform Relaxation (WR) and One Step relaxation (OSR) is shortly described. The difference resides in the order of time and point loops. Claiming for their OSR algorithm much less memory requirements as for WR.
[Mali94] states that "improved" NR is normally used for tightly coupled nodes and OSR is applied on loosely coupled nodes. This does not make much sense as NR could be used within OSR as solution algorithm.
The idea is probably the following: if OSR is used, the algorithm inside will be the secant method.
If OSR does not converge OSR is abolished in favor of some other algorithm.
Also, [Moye95] experienced convergence problems using OSR. They were resolved using the BE option of ELDO. It is not clear if this changes the method of changing from NDE to NAE within OSR only, or if OSR is given up completely going back to DM or some other simulation algorithm. In [Besn91] Hennion claims: 'In ELDO ... OSR and NR ... are mixed .. . ... strongly coupled elements are solved togehter by NR in so-called NR blocks, while strongly forward and weakly backward coupled elements are successively solved by relaxation iteration'
No hint about how strong/weak coupling is determined
Person: Hennion, '84, Rodriquez, '94
Literature: [Henn85/1], [Henn85/2],[Rodr94]

Iterated timing analysis and one step relaxation (as used in the ELDO simulator) are very similar. [Besn91] explains the difference between the two: the central solving algorithm for the equation system.
ITA uses ONE Newton step and assumes that after enough relaxation steps it will converge.
OSR calculates in the loop each point exactly, thus using more CPU power but having less convergence problems.
For the handling of the matrices they use both the Gauss Seidel method.
ITA has the disadvantage that a grounded capacitor must be tied to every node and that the time step must stay small enough. Floating capacitors represent a great problem for this algorithm.

Waveform Relaxation (WR)

WR is applied to the system of NDE as opposed to ITA and OSR being applied to the system of NAE.

repeat
    for $i = 1$ to n
        for $t = t_{start}$ to t_{end}

repeat
solve for state variable waveform i, with all the other state variable waveforms
unchanged for the considered time interval
until exactly solved
next t
next i
until all LTE small enough

WR offers inherent parallelism. Problems occur when tightly coupled systems are to be simulated. In [Wang91] a partitioning method for tightly coupled nodes is presented. Tightly coupled subcircuits are interconnected with a resistor network instead of their original link. The two thus decoupled circuits have an overlappimg part in common. To determine the values for the resistors the spectral radius of the submatrix connecting the subcircuits is taken into consideration.
Convergence is slow for circuits that produce oszillating waveforms within the circuit and at the outputs. (Hence these are the prefered examples to show the superiority of OSR).
Window partitioning for the time interval under simulation is introduced to overcome these limitations.
Because of these features WR is extremely suitable for simulation of MOS and other directional circuits that offer a high degree if independance between the circuit nodes. Bipolar circuits are usually difficult to simulate as coupling changes during the run. A method to group nodes together is described in [Whit85]. The node coupling two circuit elements is replaced by a simple resistor network with the resistors depending on the adjacent elements. Depending on the resistor values nodes are grouped together. Fairly heuristic approach.
Program: RELAX
Person: Leelarasmee, '82, White '84
Literature:[Whit84], [Whit85], [?], [Leel85]

Waveform Relaxation Newton-Raphson (WRN)

repeat
for $i = 1$ to n
for $t = t_{start}$ to t_{end}
solve only one NR iteration for state variable waveforms i,
with all the other state variables waveforms
unchanged for the considered time interval
next t
next i
until all LTE small enough

Program: SPLAX
Person: Saleh, '90

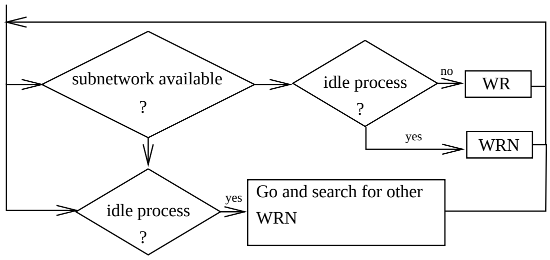


Figure 3.2: Decision tree for maximum speed gain

Literature:[Sale90a], [Oden90]

While WR computes exactly, iterating in the internal loop until convergence, WRN performs in the internal loop only one Newton Raphson step. This is less accurate but after a full relaxation it performs equal accuracy.
Parallelisation of WRN has been attempted. The method in [Oden90] relies on the fact that the most time consuming part of simulation is LU decomposition and other matrix handling operations. Different processors can parallely load and treat matrices (parallel process). When a new variable get's available the processors can start calculating straight away without having to start the matrix load phase again (sequential process). There is no circuit partitioning used in this approach.
This method causes heavy overhead and has a gain over WR only for circuits with low parallelization in respect to WR.
To gain the maximum of the two methods the decision tree as in 3.2 is applied:
Since the matrix handling operations are parallelized and the results converge back on the master process, this causes heavy network traffic on a virtual parallel machine. Therefore this sort of strategy could only be usefully used on a shared memory machine.

3.4.5 Inner loop solution proposals

[Cott90] uses for modelling frequency domain representation.
Further they calculate for each point also the gradient.
This way different time steps can be linked together, since intermediary values can be computed. This would permit to calculate asynchronously.
The disadvantage is that no future values are available, and therefore only forward Euler can be used. This algorithm has unfortunately stability problems.
The stability problem seems to be solved using for the Taylor approximation also the quadratic element. The linear calculation is preserved thanks the introduction of subsidiary variables.
One problem persists: passive arrays cannot be described with transferfunctions.
!!!! WHY NOT ?????

3.5 Iteration Methods

Whenever iterations are needed one state variable or waveform is computed with the other state variables or waveforms held unchanged. There are two commonly used ways of assigning values to the variables not considered: Gauss-Jacobi and Gauss-Seidel. When the solution of LAE systems is attempted some more algorithms are encountered. They are: Successive Over-Relaxation and Conjugate Gradient Methods and others. [Libs94]

3.5.1 Gauss-Jacobi (GJ)

In GJ the variables not computed are assigned the value they had in the previous iteration regardless of their having already been computed for the new time point or not.
This is simple and offers easy parallelism. Convergence is slow compared to GS.

3.5.2 Gauss-Seidel (GS)

In GS the variables that are not being computed actually are devided into two groups. Those that have already been computed for the actual time point and those that have not. The already computed ones get their new values assigned while those not yet dealt with remain unchanged, i.e. keep their value of the previous time step.
This offers less parallelism but converges more quickly in general.
Since Gauss Seidel is linear this slows down convergence. [Shim85] offer a modification to this algorithm introducing an error appreciation for faster convergence.
The advantage of the Gauss Seidel Algorithm lies in the fact that it can very easily be parallelized (see [Whit85]).

3.5.3 Successive Over-Relaxation (SOR)

SOR introduces in GJ or GS a weighting factor ω for the variables not calculated. This factor must be between 0 and 2 (and usually is around one) and can be increased or decreased in accordance with the observed speed of convergence. The described approach was obsered in literature and programs about the solution of LAE systems but never in combination with the simulation of NAE or NDE.

3.5.4 Conjugate Gradient Methods

Already for the solution of LAE systems Conjugate Gradient Methods are mathematically difficult to handle compared to the methods described above. Further more, Conjugate Gradient Methods require considerably more storage space than other methods. Hence, they do not seem to be a good tactic to solve large systems.

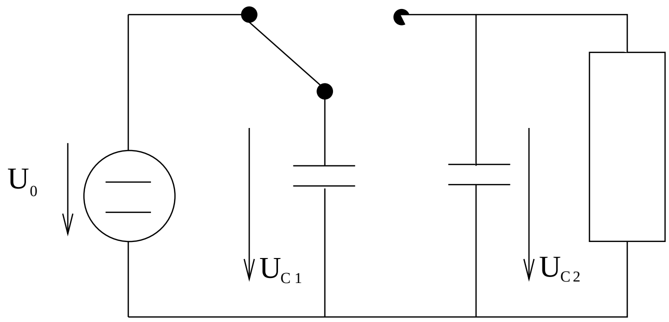


Figure 3.3: Example of initial condition problem

3.6 Initial condition problem

In the picture below is described a simple circuit:
Until the connector is open U_{C1} is equal U_0 and U_{C2} is zero. Once closed the connector we have at the same node two unequal voltages. This is called the problem of inconsistent initial condition.
Simulators working with forward Euler discretization have great problems with this. Backward Euler using as discretization the following formula:

$$x'_{n+1} = 1/n(x_{n+1} - x_n) \tag{3.22}$$

has, following to [Vlac95] less difficulty to solve the system. Furthermore they propose to solve the system of equations with the Laplace transformed. A new problem arises with the Dirac impulses of different differentiation depth.
!!!! TO REREAD !!!!

3.7 Latency and multirate exploitation

[Sale87] introduces the notion of maximum speedup due to latency and/or multirate.
Multirate means that each node can be driven at individual stepsize.
The theory was confirmed with experiments on the basis of SPLICE3. The simulator run first a direct method to gain the waveforms and make static circuit partition out of them.
Here was firstly introduced ITA with Newton extension (this means <u>one</u> Newton step inside the relaxation).
For real simulation they propose a semidynamic partitioning: from previous runs is extracted data for the next step size; if after two relaxations the function does not converge (the error remains too big), rollback is made and the stepsize is reduced.
In [Sale89] they explored further and tryed some optimistic/pessimistic approaches for the stepsize fail.
The following strategy emerged: the struggling node determines the time point at which its results become erroneous, then it sends out a rollback with that time.

With this method the rollbacks done by the nodes hopefully go reducing since absorbtion from the normal flow.
Why not complete with a varying size time window: normally the first points in the time Window converge faster than the rest, why not pushing the left limit when convergence for these points is reached?

3.8 Implementations

3.8.1 CAYENNE

Developped from [Bisc86] as a parallel extension for SPICE.
The program uses shared memory. The program is partitioned in master/slave programs. The code is nearly identical, only the call differs.
The parallel executed parts are matrix loading, LU decomposition and time step computation. SPICE uses 90% of its time resolving these phases.
No time lock system is used: each processor holds a complete matrix and writes its results in it. Afterwards all the resulting matrices are merged.
To solve the same operations parallely [Trot90] define a hypotetical parallel machine and present methods for parallelizing matrix operations.

3.8.2 iMACSIM

[Sing91] propose an very interesting system.
It is a multilevel, mixed domain analog simulator. The emulator is event driven and the whole program seems to be written in C++. The core simulator is relaxation based.
The simulator accepts electrical, functional and behavioural input. The partitioning of the input is done following the user definitions.
The input can be a HDL or s/z domain description f.e.

3.9 Problems inherent to the parallel analog mode simulation

The problem of partitioning analogue circuits is not resolved at the moment. Nearly no information could be extracted out of actual publications.
The graph partitioning strategies are possibly applied. No information about the used cost function is made.

3.9.1 Partitioning Strategies

[Arei93] presents for circuit partitioning a modified simulated Annealing algorithm.
In fact it seems to be a mix of min-cut and simulated Annealing:

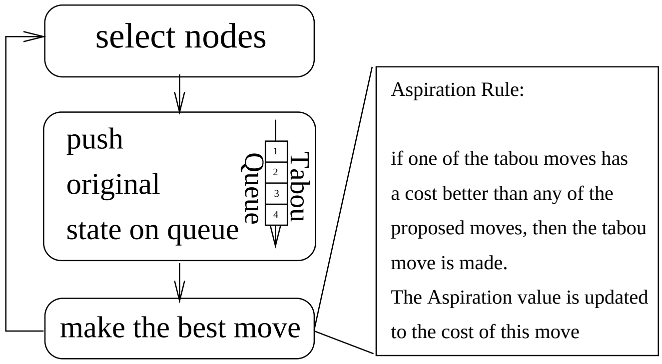


Figure 3.4: Tabou Partitioning

This technique was merged into simulated annealing and seemed to show good results.

3.9.2 Partitioning through connectivity

[Pete93] describes two completely undefined criterias, one for partitioning an the other one for merging subcircuits. The whole thing seems to work dynamically. In their thesis they are a bit more explicite, unfortunately the mathematical explanation is full of errors and the assumptions made for finding the two criterions are such that we think that there are not for general use:
For calculating the transductance of a subcircuit the impedances Z_{jj} and Z_{jm} are aproximated respectively by $1/y_{jj}$ and $1/y_{jm}$! I think that this assumption is valid only for MOS circuits.

3.9.3 Partitioning through prediction

An original thought was presented form [Vasq85]: The netlist is analyzed, setting flags predicting the future state. With this information a directed graph is extracted from the netlist and partitionned.
Unfortunately this seems to work only for digital circuits.

3.9.4 Logical partitioning

[Cox87] takes an analog circuit and partitions it with an unspecified algorithm. Each subcircuit is packed into a "interface box" with specified in and outgoing signals.
The simulation itself is done with the Newton Raphson method. Using completely separate subcircuits permits to use different timesteps for each part. [Cox87] goes further making a sort of latency exploitation: if the inputs and outputs of a subcircuit were stable over a certain time interval, and the input remain stable then no recalculation of the subcircuit is made.
Some approximations and adaptations seem to be needed, that remain unclear.

3.9.5 Device modelling

Depending of the needs on accuracy different models of transistors are used. SPICE simulates this using a standard Transistor and a variable number of resistors and capacitances.
this can cause a huge number of node creation, inflating sensibly the jacobian matrix and thus exploding the complexity of the simulation.
[Song93] propose to transform the different accuracy models in equivalent four port equations, optimize this matrix linearize and followingly use it. Each transistor can now be described by a $4x4$ matrix.

Chapter 4

Mixed Mode simulation (Analog/Digital)

4.1 Strategies for connecting an analog simulator to a digital

[Anal94] makes the statement that all is analog, the only difference is in the required resolution.
In [Moli92] for example a model simplification is proposed. The behaviour of C-MOS logic is modelled as a simplified U/I Graph. The Graph is choosen in function of the required acuracy.
But not all can be enough simplified, hence two simulators. The techniques for connecting two simulators are as following:

- **lock step** the simulators run with the same step width. Very inefficient especially for the analog simulator who has to reevaluate his matrices at every step, even if it is not necessary.
- **backplane** here synchronization between the two simulators is forced at regular intervals.
- **ping pong** synchronization is achieved only on demand of one of the simulators. The problem of rollback arises.
- **Calaveras** like ping pong, (proprietary of Analogy)the difference resides in the fact that calaveras records the history of the analogue node states skipping recalcutation of the past time point at rollback. Thus privileging the speedfactor above the memory load factor.

[Saka80] try to create with a partitionned analog an event handler, but they get problems with the thight coupling of process in analog circuits.

4.1.1 ADEPT

ADEPT from [Odry86] seems to be the first mixed mode simulator based on relaxation.
In this publication is presented the migration from direct methods towards relaxation. This system is multilevel, the user determines how the different circuit parts are modelled (digitally, switch level, analogue). The obtained results are good compared to SPICE. The strategy for initializazion is new.
Unlike other simulators they do not force all nodes to 0V and iterate towards some acceptable start value, but they make a fast switchlevel analysis. This way they produce more reliable startvalues.
The used HDL is M. But their system seems to match MOS-development.

4.2 Interfacing Analog and Digital Components

As extracted from [Berg92]:

Definition 4.1 •

- **full domain conversion** *means explicit computation of the equivalent value of a digital signal (analog voltage and current) or an analogue voltage (digital signal).*
- **partial domain conversion** *means the modelisation of digital (analogue) load seen by analogue (digital) part.*

4.2.1 Preliminary remarks

- The user must have full control over the A-to-D and D-to-A conversion mecanism. (DOD #5.1, #6.1)
- The direct connexion without conversion of a Port to a Pin is not allowed. (DOD #6.4)
- The user should not be forced to place explicitely extra components at the interface between A and D domains. This means that the apparent global connectivity of the model should not exhibit any component that will not appear in the real circuit.

4.2.2 possible Analog/Digital Interactions

- The digital signal drives an analog one: the D-signal is declared in mode **out**. A *full D-to-A* conversion is needed, in other words voltage **and** current must be determined for the analog port. I return a modificator for the fanout and the signal delay must be given back to the digital signal.

- The analog signal drives a digital Port (thus declared as in mode **in**). A *full A-to-D* conversion is needed. The analog voltage must be converted to a digital signal and the fanout for the analog signal must be modelled (as a 'stray??' capacitance for example).

- Bidirectional interaction: the digital port is declared *inout* this is still an open issue in the discussion about the inerfacing.

Full A-to-D conversion

This conversion should use a user supplied thresholding function posting an event on the associated digital signal only when significant changes occur in the analogue part (DOD #5.2). Note that this function is highly dependent on the logic system used in the digital part (2,3,9,46 states...).

Full D-to-A conversion

Thsi conversion should use a user supplied slope function to avoid instantaneous change in analogue voltage and current. The analog part may see, as a model for the digital fanin, one or several equivalent Thevenin or Norton cicuits connected to the analogue pin.
At the moment this conversion is in discussion, the actual solution tends to be an equivalent to the resolution function of VHDL.

4.3 Problems inherent to the mixed mode simulation

VHDL is a very unspecific language, very few inbuild macros are present. In general the user defines his own macro package or uses already build macro packages. Concerning the problem of interfacing the analogue parts with the digital ones is at the moment an open issue, since the standard is evolving very quickly and lacks of inbuild definition. The type of interface will mainly be leaven to be defined by the user. Some proposal have been made in the white papers of the IEEE-VHDL-A-design committe and in [Berg92].
The VHDL-ELDO[1] simulator ([Taha93]) proposes to the user three simple analog/digital interfaces:

- controlled source, the slope is controlled by the user, no conflict resolution is made. This method is fast but not very accurate,

- IRC circuit as intermediary stage. The user set the trheee values.

[1]The VHDL ELDO simulator is at the moment a mixture of VHDL and SPICE, but will be a native 1076.1 compliant simulator

- for the analog to digital conversion a simple level comparator. It assumes the digital circuit to be of high impedance and an intermediary RC-Filter emulates the digital circuit.

In any case is leaved to the user the freedom to define his own models.

Chapter 5

The VHDL-A Simulator project

5.1 The input provided by VHDL-A

[1]

Depending of the complexity of the desired design several levels of abstraction would be used. For digital design, abstraction is well defined. A level of abstraction defines the following three aspects: the modelling concept, the timing model and the observable values. For example, for the register transfer level the modelling concept corresponds to guarded commands (reactive behaviour), the timing model corresponds to discrete real time (clock ticks) and the observable value are bitstreams. It is usual to consider the abstraction level from system (high degree of abstraction) to gate (low degree) as pertaining to the digital world and the levels under gate (= electrical) are seen as analogue.
For analogue abstraction level the modelling concept is differential equation, the timing model is continuous time and the observable value are continuous values (voltage and current, e.g.). It does not however cover all possible modelling methodologies as far as analogue design is concerned.
The abstraction levels are not so well established for analogue as for digital world. Analogue higher abstraction levels differ from digital one in the timing model (beeing continuous).
Possible analogue abstraction levels are:

- **Functional Level**.

  This model is based on the signal flow model, where transfer functions map the system input to the system output in an unidirectional way. Observable values are continous values that usually do not obey conservation laws, such as Kirchoffs laws for electrical systems. (see picture 5.1)

- **Behavioural Level**.

  Equations describe the function of the component. Observable values are continuous and obey Kirchhoffs conservation laws. (see picture 5.2)

[1] extracted from [Shi95]

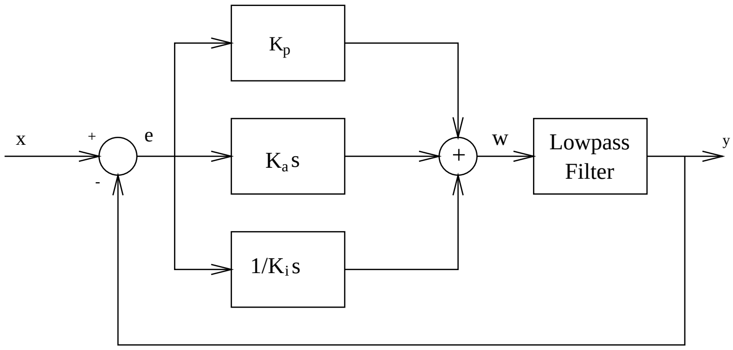


Figure 5.1: Functional abstract description of an analogue circuit

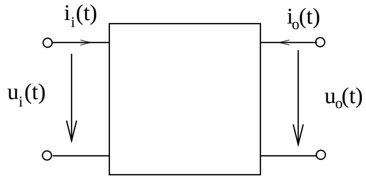


Figure 5.2: Example of behavioural description

- **Macromodel Level**.

  The component is seen as an hierarchal assembly of more primitive components (e.g. controlled sources). The macromodel may not be considered as an abstraction level, but rather as normal description strategy. (see picture 5.3)

- **Circuit Level**.

  This modeling concept is based on the hierarchical assembly of primitive circuits such as resistors, capacitors, transistors and sources, whose behaviours are pre-defined. The structure of a circuit description usually matches that of the undelying hardware. (see picture 5.4)

5.2 Possible research themes

The low level partitioning of analogue circuits seems quite compromised unless lightly coupled systems are used (like MOS circuits).

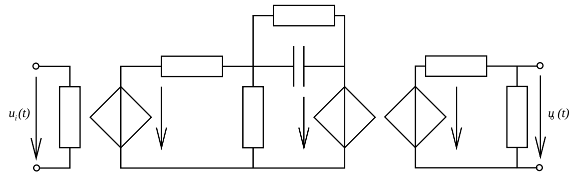


Figure 5.3: Example of macromodel description

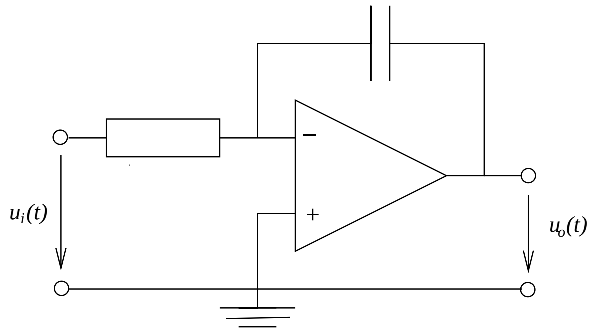


Figure 5.4: Example of circuit description

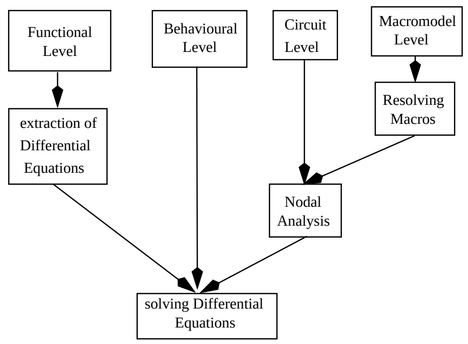


Figure 5.5: The solving Path for the different abstraction levels

5.2.1 Interconnection of different abstraction levels

An issue could be the interconnection of the different simulators or parts of the simulator using the different input models (as shown in 5.5). A study about the computational complexity of the calculation of the different problems should be made.

5.2.2 Abstract Level coupling criteria

Some criteria of static/dynamic coupling between the different models should be enounced.
The point where all models met at simulation time is at linear differentiation level.

5.2.3 Modification of the Waveform Relaxation Algorithm

A little subpart could perhaps be the investigation of the modified WR algorithm (enough for a Diplomarbeit?). (see section 3.7)

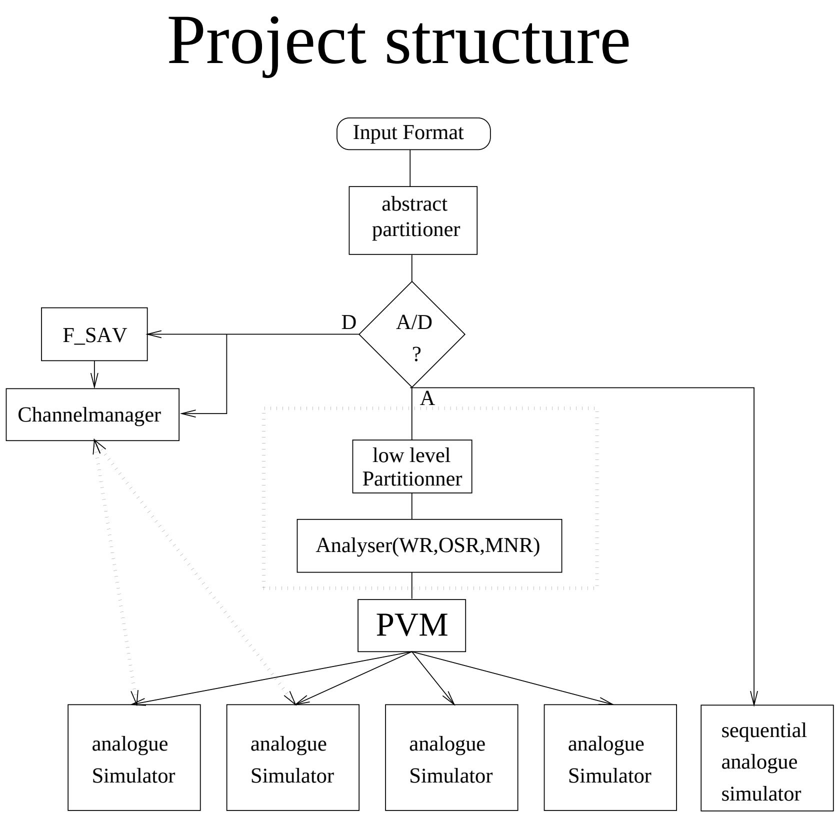


Figure 5.6: Possible structure of the project

5.2.4 Modification of available simulators

At the moment we have the sources for ACS[2]. ACS is a sort of SPICE compliant simulator with some extensions. To be of use for use it should be modified to support:

- ODE[3]'s
- multiple core Algorithms: in addition to Newton, WR[4], OSR[5].
- compilation of the simulating circuit in sight of distributed computing.

All those tasks could be given away to students for eventual "Diplomarbeit's"

5.3 The simulator

The general approach to the aimed solution could be as shown in picture 5.6. As said before no solution could be seen for the low level partitioning.

[2]see Annex B
[3]Ordinary Differentiation Equation
[4]Waveform Relaxation
[5]One Step Relaxation

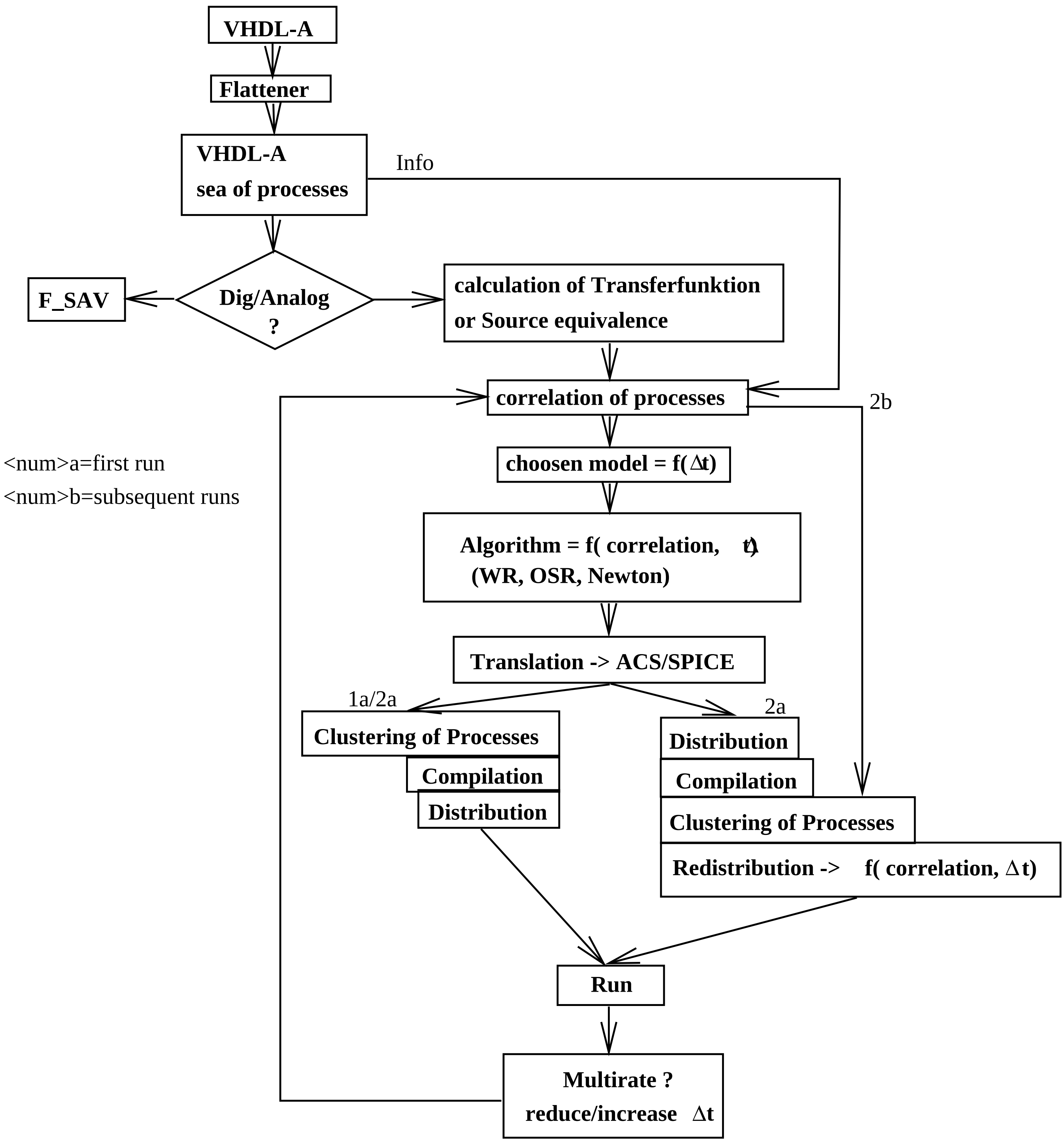


Figure 5.7: More detailed proposition for the core Simulator

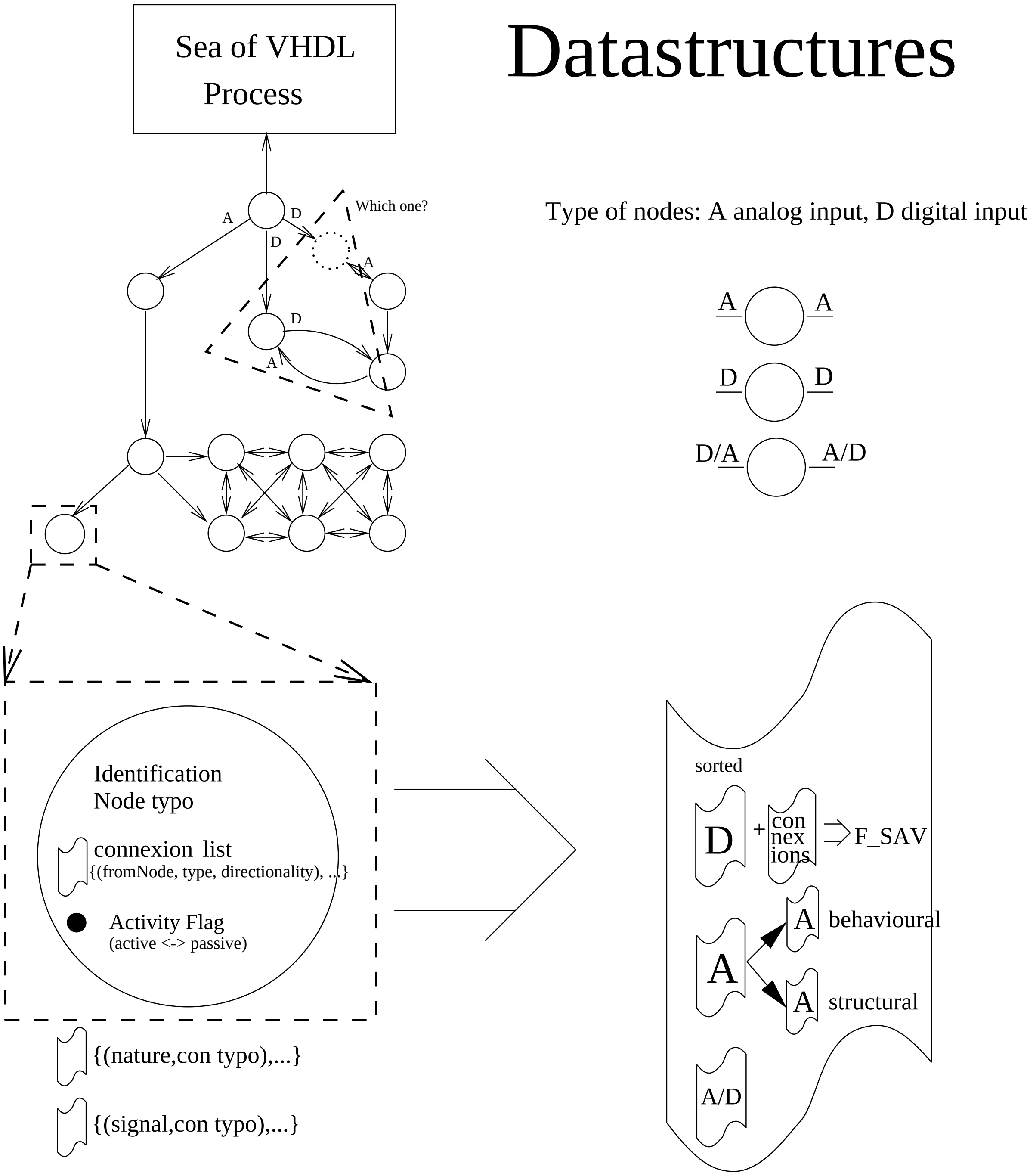


Figure 5.8: Possible decomposition of the Problem

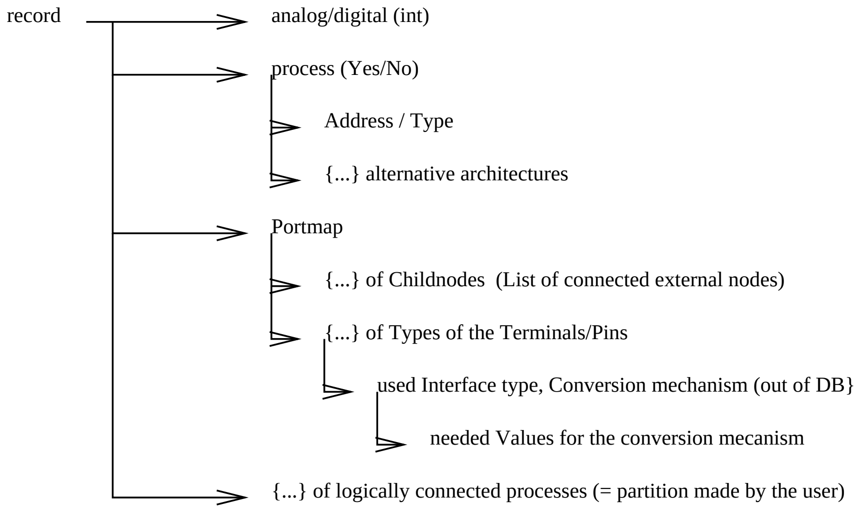


Figure 5.9: Possible datastructure for description of VHDL processes

5.8 describes the input to the simulator out of the analysis of the VHDL code. In a first step the hierarchical VHDL description is exploded in a sea of VHDL processes. The information about the hierarchical structure given by the user must be stored somewhere.
Each process must be preceeded by a header specifying its type (Analogue/Analogue, Analogue/Digital, Digital/Digital etc.). Since the discussion of mixed netlists is not resolved by the standardization committee, at the moment no mixed nodes are assumed. This means that the syntactical analyzer of the VHDL source code must extract precisely typed processes, eventually splitting a process to get "pure" processes.
Further the connection list should be included in that header. This list must transfer the information about the connected signals (A, D in/out).
Finally it was be thought of an activity flag the would determine the used timestep and precisionmodel of the specified device.
At that level arises a problem concerning the interfacing of the mixed processes. I have put the case in 5.8 just under the "Sea of VHDL Process"-Box in a dotted triangle:
How should this conversion be made? One solution consists of inserting a interface node which make the conversion between the digital and analogue world. An other solution would be to declare multiple channels for one signal and to leave their management to the process.
5.9 is the try for defining a datastructure that would hold all the needed information for simulation of the VHDL code.

Chapter 6

Why should I receive a Dr. Title for this work?

Appendix A

Interesting HTML-Links

IEEE DASC VHDL Analog Extensions Working Group
http://vhdl.org/vi/analog/
WorldWideWeb EPFL Home Page
http://www.epfl.ch/
32nd Design Automation Conference
http://www.dac.com/dac.html
IEEE DASC Standard Simulation Control Language Study Group (SCL)
http://vhdl.org/vi/scl/Welcome.html
Waves Home page
http://vhdl.org/vi/waves
IEEE Home Page
http://www.ieee.org/
CERN Welcome
http://www.cern.ch/

Appendix B

ACS (Al's Circuit Simulator)

ACS is a general purpose circuit simulator. It performs nonlinear dc and transient analyses, fourier analysis, and ac analysis linearized at an operating point. It is fully interactive and command driven. It can also be run in batch mode or as a server. The output is produced as it simulates. Spice compatible models for the MOSFET (level 1 and 2) and diode are included in this release.
This version (0.17) is a transitional release in C++. There are a few bug fixes.
Since it is fully interactive, it is possible to make changes and re-simulate quickly. The interactive design makes it well suited to the typical iterative design process used it optimizing a circuit design. It is also well suited to undergraduate teaching where Spice in batch mode can be quite intimidating. This version, while still officially in beta test, should be stable enough for basic undergraduate teaching and courses in MOS design, but not for bipolar design.
In batch mode it is mostly Spice compatible, so it is often possible to use the same file for both ACS and Spice.
The analog simulation is based on traditional nodal analysis with iteration by Newton's method and LU decomposition. An event queue and incremental matrix update speed up the solution for large circuits.
It also has digital devices for mixed signal simulation. The digital devices may be implemented as either analog subcircuits or as true digital models. The simulator will automatically determine which to use. Networks of digital devices are simulated as digital, with no conversions to analog between gates. This results in digital circuits being simulated faster than on a typical analog simulator, even with behavioral models. The digital mode is experimental and needs work. There will be substantial improvements in future releases.
ACS also has a simple behavioral modeling language that allows simple behavioral descriptions of most components including capacitors and inductors. Unfortunately, it is not well documented.
ACS uses an object oriented approach to modeling. Complex models like MOSFETS are made of simpler ones like resistors, capacitors, diodes, and any other models that may already exist. The model designer does not need to worry about details like convergence checking, bypass checking, integration, or how the new device plugs into the solution matrix because these are already taken care of by the basic models. This results in a dramatic improvement in the time it takes a researcher or model designer to install a new model, compared to Spice.
The source and documentation can be obtained by anonymous ftp from ee.rochester.edu or cs.rit.edu in /pub/acs. It can also be obtained by dial-up (USA) 716-272-1645 in /pub/acs. It may be distributed under the terms of the GNU general public license. The dial-up also has some test circuits, pre-compiled executables for Next, Sun4, MSDOS and possibly others, and documentation in dvi and postscript.
If you are tired of Spice and want a second opinion, you want to play with the circuit and want a simulator that is interactive, or you want to study the source code and want something easier to follow than Spice, try ACS.
ACS is an ongoing research project. It is being released in a preliminary phase in hopes that it will be useful and that others will use it as a thrust or base for their

research. I also hope for some comments that may help me direct my research.
Albert Davis, 136 Doncaster Rd., Rochester, NY 14623. email: al@atd.rochester.ny.us, atd@cs.rit.edu, or davis@ee.rochester.edu fax: 716-272-1645

Bibliography

[Airi94] Airiau, R., Bergé, J.-M., Olive, V.: **Circuit synthesis with VHDL**, *Kluwer, Kluwer international series in engineering and computer scienc, 261*, Boston, MA, 1994

[Anal94] Analogy **Mixed-Signal Design Simulation guide** In *Helping guide of Analogy about mixed simulation*, 1994

[Altm90] Altmaë, M.: **MINT - a VHDL Simulation System**, *Proc. EDAC-'90*, 1990, pp. 102-106

[Arei93] S. Areibi and A. Vannelli **Circuit partitioning using a tabu search approach** In *IEEE Int. Symposium, on Circuits and Systems*, Chicago, p. 1643-1646, 1986

[Baez94] D. Baezner, G. Lomow and B. Unger. **A parallel simulation environment based on time warp.** In *international Journal in computer simulation 4*, p. 183-207, 1994

[Berg92] J-M. Bergé, D. Rouquier, s. Garcia-Sabiro and A. Vachoux. **A-to-D and D-to-A Domain Conversions** In *working document*, Oct 1992

[Berg95] J-M. Bergé, E. Christen and A. Vachoux. **VHDL-A: Analog Extensions to VHDL** In *inofficial (slides)*, Feb 1995

[Besn91] J. Besnard, J. Benkoski, B. Hennion **Eldo-XL: a software accelerator for the analysis of digital MOS circuits by an analog simulator** In *Proc. EDAC 91*, p. 136-139, 1991

[Bile85] W. Biles, C. Daniels and T. O'Donnell. **Statistical Considerarions in Simulation on a Network of Microcomputers** In *Proceedings of the 1985 Winter simulation Conference*, page 388-392, 1985

[Bisc86] Bischoff G., Greenberg S. CAYENNE: A parallel implementation of the circuit simulator SPICE In *Proc. Conf. Computer Aided Design*, 1986, p. 182-185

[Casi94] Casinovi, G., Yang, J.-M.: **Multi-Level Simulation of Large Analog Systems Containing Behavioral Models**, *IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems*, Vol. 13, No. 11, pp.1391-1399

[Chad92] Chadha, R., Visweswariah, C., Chen, C.-F.: **M3 - A Multilevel Mixed-Mode Mixed A/D Simulator**, *IEEE Trans. Computer-Aided Design*, 1992, Vol. 11., No. 5, pp. 575-585

[Chan81] Chandy and Misra Asynchronous distributed simulation via a sequence of parallel computations. In *Communications of the ACM*, Vol. 24, No. 11, pages 198-206, April 1981

[Chan88] Chang, M.C., Hajj, I.N.: **iPride: A Parallel Integrated Circuit Simulator Using Direct Method**, *Proc. Int. Conf. Computer-Aided Design (ICCAD-88)*, pp. 304-307

[Cott90] R. A. Cottrel
Event-Driven behavioural simulation of analogue transfer functions In *Proc. EuroDAC 1990*, Glasgow, p. 240-243

[Cox87] Cox P., Burch R. Hocevar D., Yang P. SUPPLE: Simulator utilizing parallel processing and latency exploitation. In *Proc. Int. Conf on CAD (ICCAD -87)*, 1987, pages 368-371

[Dand95] P. Sivarama, Dandamudi, P.S.P. Cheng **A hierarchical task queu organization for shared memory multiprocessor systems** In *IEEE Trans. on P. & D. Systems*, Vol.6, No. 1, p. 1-16, 1995

[Deut84] J. Deutsch, A. Newton
A multiprocessor implementation of relaxation-based electrical circuit simulation In *21st Design Automation Conference*, 1984

[Eijn90] Eijndhoven, van J.T.J., Stiphout, van M.T., Buurman, H.W.: **Mutirate Integration in a Direct Simulation Method**, *Proc. Euro-DAC '90*, 1990, pp. 306-309

[Enge93] Engeln-Müllges, G., Reutter, F.: **Numerik-Algorithmen mit ANSI-C Programmen**, *BI-Wiss.-Verl.*, Mannheim, Leipzig ,Wien, Zürich, 1993

[Fidg91] C. Fidge Logical Time in distributed computing Systems In *Computer*, IEEE ref. 0018-9162/91/0800-0028, August 1991

[Fuji88] R. M. Fujimoto Lookahead in parallel discrete event simulation In , 1988

[Fuji90] R. M. Fujimoto Parallel discrete event simulation In *communications of the ACM* , Vol. 33, No. 10, p. 30-53, Oct 1988

[Heid86] P. Heidelberger. Statistical Analysis of Parallel Simulations In *Proceedings of the 1986 Winter simulation Conference*, page 290-295, 1986

[Henn85] B. Hennion, P. Senn
ELDO: A new third generation circuit simulator using the one-step relaxation method In *Proc. of ISCAS-85*, p. 1065-1068, 1985

[Ho75] Ho, C.W., Ruehli, A.E., Brennan, P.A.: **The Modified Nodal Approach to Network Analysis**, *IEEE Trans. Circuits and Systems*, Vol. CAS-22, No. 6, pp. 504-509

[Hopp88] F. Hoppe. Accelerated logic simulation using parallel processing. In *Comp-Euro 88* , CH2548-6/88/0000/0156, IEEE.

[Ishi] N. Ishiura, H. Yasuura and S. Yajima Time First evaluation Algorithm for high-speed logic-Simulation In *Y-58*,

[Jeff84] An approach to Performance Analysis of Timestamp-driven Synchronization Mechanisms In *1984 ACM 0-89791-143-1 84/008/0243*

[Jeff85] Virtual Time In *ACM Transactions on Programming Languages and Systems*, Vol. 7, No. 3, July 1985, Pages 404-425

[Kona94] P. Konas, P.-C. Yew. Parallel simulations of multiprocessors. In *Simulation pratice and theory*, Vol. 1, Number 6, July 1994 pages 249-266, ISSN 0928-4869

[Libs94] Libs, J.-M.: **Architecture Multiprocesseur Dédiée à la Résolution des Equations aux Dérivées Partielles en 3D**, Dissertation, Ecole Centrale de Lyon, 1994

[Luba88] B. D. Lubachevsky. Bounded lag distributed discrete event simulation. In *Distributed Simulation 1988*, ISBN 0-911801-29-4, 1988

[Luba89] B. D. Lubachevsky. Scalability of the bounded lag distributed discrete event simulation. In *Distributed Simulation 1989*, ISBN 0-911801-49-9, 1989

[Madi88] V. Madisetti, J. Walrand and D. Messerschmitt. WOLF: A rollback algorithm for optimistic distributed simulation systems In *Proceedings of the 1988 Winter simulation Conference*, page 296-305, 1988 Y-65

[Magi92] In ,

[Mali94] Malinak, L.: **A/D Simulators: An Expanding Array of Choices**, *Electronic Design, Dec. 5, 1994*, pp. 95-102

[Manj93] N. Manjikian and W. Loucks High performance parallel logic simulation on a network of Workstations In *Proceedings of the 7th Workshop on Parallel and Distributed Simulation (PADS93)*, ISBN 1-56555-055-2,Vol. 23, Number 1, page 76-84, July 1993

[Misr86] Misra Distributed discrete-event simulation. In *Computing surveys*, Vol. 18, No. 1, March 1986

[Moli92] B.-A. Molin and S. Mattisson A parallel Switch-Level Simulator for Mixed Analog-Digital Circuit Simulation In , IEEE ref. 0-8186-2645-3/92, page 202-206 1992 Y-114

[Moye95] Moyer, P.: **Rapport du Projet de Fin d'Etudes, Modelisation d'un Injecteur a l'aide du language VHDL-A**, Ecole Nationale Superieure de Physique de Strasbourg, 1995

[Oden90] P. Odent, L. Claesen, H. De Man
A combined waveform relaxation - waveform relaxation newton algorithm for efficient parallel circuit siulation In *Proc. EuroDAC '90*, Glasgow, p. 244-248, 1990

[Odry86] P. Odryna, S. Nassif
The ADEPT timing simulation algorithm In *VLSI Systems Design*, p. 24-34, March 1986

[Patr90] Patrick, D., Lyden, C.: **An Event-Driven Transient Simulation Algorithm for MOS and Bipolar Circuits**, *Proc. Euro-DAC '90*,1990, pp. 230-234

[Pala92] A.C. Palaniswamy, S. Aji and P A. Wilsey. An Efficient Implementation of Lazy Reevaluation In *Proceedings of the 25th Annual Simulation Symposium (IEEE)*, Orlando, Florida, April 1992, pages 140-146

[Pete92] Peterson, L.: **Dynamic Circuit Partitioning for Waveform Relaxation-Based Circuit Simulation** Dissertation, Department of Applied Electronics, Lund Institute of Technology, Lund, Sweden

[Pete93] L. Peterson, S. Mattisson
Dynamic partitioning for concurrent waveform relaxation based circuit simulation In *IEEE Int. Symp. on Circuits & Systems*, Chicago, p. 1639-1642, 1993

[Pres89] M. Presleyand M. Ebling, F. Wieland, D. Jefferson Benchmarking the time warp operating system with a computer network simulation In *Distributed Simulation*, ISBN 0-911801-49-9 1989

[Righ89] R. Righter and J. C. Walrand. Distributed Simulation of discrete Event Systems In *Proceedings of the IEEE*, Vol. 77, No. 1, Jan 1989

[Rodr94] D. Rodriguez
Description et simuation mixte analogique-numérique: Analyse de VHDL-Analogique, réalisation d'un simulateur mixte In *Thèse, ARTEMIS/IMAG et CIT/MOS (CNET Grenoble)*, 1994

[Ruan91] Ruan, G: **A Behavioral Model of A/D Converters Using a Mixed-Mode Simulator**, *IEEE J. Solid-State Circuits*, 1991, Vol. 26, No.3, pp.283-290

[Saka80] Sakallah, K., Director, S.W.: **An Activity-Directed Circuit Simulation Algorithm**, *Proc. IEEE Conf. Circuits and Computers*, 1980, pp. 1032-1035

[Saka85] K. Sakallah, S. Director
SAMSON2: An event driven VLSI circuit simulator In *IEEE Trans. on Computer Aided Design*, Vol. 4, No.4, p. 668-84, October 1985

[Sale87] R. Saleh, A. Newton
An event driven relaxation-based multirate integration scheme for circuit simulation In *Proc. Int. Symp. on Circuits & Systems*, Piladelphia, p. 600-603, 1987

[Sale89] R. Saleh, A. Newton
The exploitation of latency and multirate behaviour using nonlinear relaxation for circuit simulation In *IEEE Trans. on Computer Aided Design*, Vol. 8, No. 12, p. 1286-98, december 1989

[Sale90a] Saleh, R,A., White, J.K.: **Accelerating Relaxation Algorithms for Circuit Simulation Using Waveform-Newton and Step-Size Refinement**, *IEEE Trans. Computer-Aided Design*, Vol. 9, No.9, 1990, pp. 951-958

[Sale90b] Saleh, R.A., Newton, A.R.: **Mixed mode simulation**, *Kluwer, (Kluwer international series in engineering and computer science; 98)*, Boston, 1990

[Saka80] K. Sakallah and S. Director
An activity - directed circuit simulation algorithm In *IEEE Proc. Conf on circuits and computers*, p. 1032-1035, 1980

[Shi95] R. Shi, A. Vachoux
VHDL-A Design Objectives and Rationale In *Current Issues in Electronic Modeling*,Issue 2, March 1995, Kluwer Academic Publishers, 1995

[Shim85] T. Shima, C. Mizuta
Modified Gauss-Seidel iterative scheme for high speed circuit simulation In *Proc. ISCAS-85*, p. 1061-1064, 1985

[Sing91] J. Singh, R. Saleh **iMACSIM: A program for multilevel analog circuit simulation** In *Proc. IEEE Int. Conf. on computer aided design*, Santa Clara, CA, p. 16-19, November 1991

[Song93] H. Song, D. Divekar, L. Mills and P. Wang
A method for improving the efficiency of simulating large electronic circuits In *IEEE Int. Symposium on circuits and systems*, Chicago, p. 1631-1634, 1986

[Soul87] L. Soule', T. Blank Statistics for Parallelism and Abstraction Level in Digital Simulation In *24th ACM/IEEE Design Automation Conference*, Paper 32.6, page 588, no. ACM 0738-100X/87/0600-0588, 1987

[Soul88] L. Soule', T. Blank Parallel Logic Simulation on General Purpose Machines In *25th ACM/IEEE Design Autopmation Conference*, Paper 13.2, page 166, no. CH2540-3/88/0000/0166, 1988

[Stei91] J. S. Steinmann. Interactive SPEEDES In *proceedings of the 24th Annual Simulation Symposium*, Eastern Multiconference 1991, pages 149-158

[Taha93] H. El Tahawy, D. Rodriguez, S. Garcia-Sabiro and J.-J. Mayol VHD$_e$LDO: A new mixed mode simulation In *EURO-DAC '93*, Hamburg, Sep 1993

[Trot90] J. Trotter and P. Agrawal **Circuit simulation algorithms on a distributed memory multiprocessor system** In *Proc. Int. Conf. on Computer Aided Design (ICCAD-90)*, p. 438-441, Nov 1986

[Vasq85] A. Vasquez, S. Director, K. Sakallah **PRIMO: a VLSI circuit partitioner for simulation applications** In *Proc. ISCS-85*, p.1075-1078, 1985

[Vlac95] J. Vlach, J. Wojciechowski and A. Opal **Analysis of Nonlinear Networks with Inconsistent Initial Conditions** In *IEEE Trans. Circuits and systems*, Vol. 42, No 4, p. 195-200, April 1986

[Wagn88] D. Wagner, E. Lazowska, B. Bershad **Tachniques for efficient shared-memory parallel simulation** In *Technical report 88-04-05*, Department of computer science, University of Washington, 1988

[Wang91] Wang, R., Wing, O.: **Waveform Relaxation on Tightly Coupled Systems**, , pp. 2280-2283

[Whit84] White, J., Sangiovanni-Vincentelli,A.L.: **Relax 2.1: A Waveform Relaxation Based Circuit Simulation Program**, *Proc. Custom Integrated Ciruit Conf. 1984*, 1984

[Whit85] J. White, A.L. Sangiovanni-Vincentelli **Partitioning algorithms and parallel implementations of waveform relaxation algorithms for circuit simulation** In *Proc. of Int. symp. on Circuits and Systems*, Kyoto 1985, p. 221-224

[Whit85b] White, J.: **The Multirate Integration Properties of Waveorm Relaxation with application to circuit simulation and parallel computation**, Dissertation, Memorandum No. UCB/ERL M85/90, Electronics Research Lab., College of Engineering, University of Calif. at Berkeley

[Whit91] White, J., Odeh, F.: **Connecting Waveform-Relaxation Convergence Properties to the A-Stability of Multirate Integration Methods**, *COMPEL - The International Journal for Computation and Mathematics in Electrical and Electronic Engineering*, Vol. 10, No. 4, pp. 497-508

[Wiel89] F. Wieland, L. Hawley, A. Feinberg, M. Di Loreto, L. Blume, R. Reiher, B. Beckman, P. Hontalas, S. Bellenot and D. Jefferson Distributed combat simulation and time warp: The model and its performance In *Distributed Simulation*, ISBN 0-911801-49-9 1989

[Wiel89b] F. Wieland and D. Jefferson Case studies inserial and parallel simulation In *International Conference on Parallel Processing*, page III-255, 1989

[] In ,

[Nich94] Nichols, K.G., Kazmierski, T.J., Zwolinski, M., Brown, A.D.: **Overview of SPICE-like circuit simulation algorithms**, *IEE Proc.-Circuits Devices Syst.*, Vol. 141, No. 4, 1994, pp. 242-250

[Bray72] Brayton, R.K., Gustavson, F.G., Hatchel, G.D.: **A New Efficient Algorithm for Solving Differential-Algebraic Systems Using Implicit Backward Differentiation Formulas**, *Proc. IEEE*, Vol. 60, 1972, pp. 98-108

[Ho75] Ho, C.W., Ruehli, A.E., Brennan, P.A.: **The Modified Nodal Approach to Network Analysis**, *IEEE Trans. Circuits and Systems*, Vol. CAS-22, No. 6, pp. 504-509

[Hach71] Hachtel, G.D., Brayton, R.K., Gustavson, F.G.: **The Sparse Tableau Approach to Network Analysis and Design**, *IEEE Trans. Circuit Theory*, Vol. CT-18, No. 1, 1971, pp. 101-113

[Sand67] Sandberg, I.W., Shichman, H.: **Numerical Integration of Systems of Stiff Nonlinear Differential Equations**, *Bell Syst. Tech. J.*, 1968, pp. 511-517

[Casi94] Casinovi, G., Yang, J.-M.: **Multi-Level Simulation of Large Analog Systems Containing Behavioral Models**, *IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems*, Vol. 13, No. 11, pp.1391-1399

[Whit84] White, J., Sangiovanni-Vincentelli,A.L.: **Relax 2.1: A Waveform Relaxation Based Circuit Simulation Program**, *Proc. Custom Integrated Ciruit Conf. 1984*, 1984

[ElTa93] El Tahawy, H., Rodriguez, D., Garcia-Sabiro, S., Mayol, J.-J.: **VHDeLDO: A New Mixed Mode Simulation**, *Proc. Euro-DAC/ Euro-VHDL 93*, 1993, pp. 546-551

[Witt90] Wittum, G.: **Mehrgitterverfahren**, *Spektrum der Wissenschaft*, Apr. 1990, pp. 78-90

[Grim95] Grimm, C., Oehler, P., Waldschmidt, K.: **Modellierung gemischt analog/digitaler Systeme zur Entwurfsunterstützung**, , pp. 1-2

[Chan88] Chang, M.C., Hajj, I.N.: **iPride: A Parallel Integrated Circuit Simulator Using Direct Method**, *Proc. Int. Conf. Computer-Aided Design (ICCAD-88)*, pp. 304-307

[Cox87] Cox, P., Burch, R., Hocevar, D., Yang, P.: **SUPPLE: Simulatorr Utilizing Parallel Processing and Latency Exploitation**, *Proc. Int. Conf. Compute-Aided Design (ICCAD-87)*, 1987, pp. 368-371

[Bisc86] Bischoff, G., Greenbeg, S.: **CAYENNE: A Parallel Implementation of the Circuit Simulator Spice**, *Proc. Int. Conf. Computer-Aided Design*, 1986, pp. 182-185

[Naka91] Nakano, M., Youichirou, M., Shirakawa, I.: **A Waveform Relaxation Method Applicable to Bipolar Digital Circuits**, , pp. 2276-2279

1991 IEEE Int. Symp. on Circuits and Systems, Singapore 06/11-14/91

[Hung91] Hung, G.G., Gallivan, K., Saleh, R.: **Parallel Circuit Simulation Based on Nonlinear Relaxation Methods**, , pp. 2284-2287

[Song93] Song, H., Divekar, D., Mills, L., Wang, P.: **A Method for Improving the Efficiency of Simulating Large Electronic Circuits**, *1993 IEEE Int. Symp. Circuits and Systems*, 1993, pp. 1631-1634

[Saka80] Sakallah, K., Director, S.W.: **An Activity-Directed Circuit Simulation Algorithm**, *Proc. IEEE Conf. Circuits and Computers*, 1980, pp. 1032-1035

[Trot90] Trotter, J.A., Agrawal, P.: **Circuit Simulation Algorithms on a Distributed Memory Multiprocessor System**, *Proc. Int.Conf. Computer-Aided Design (ICCAD-90)*, 1990, pp. 438-441

[Henn85/1] Hennion, B., Senn, P.: **ELDO: A New Third Generation Circut Simulator Using the One-Step Relaxation Method**, *Proc. Int. Conf. Circuits and Systems (ICCAD-85)*, pp. 1065-1068

[Henn85/2] Hennion, B., Senn, P.: **A New Algorithm for Third Generation Circuit Simulators: The One-Step Relaxation Method**, *22nd Design Automation Conference (ACM IEEE)*, 1985, pp. 137-143

[Vasq85] Vasquez, A.V., Director, S.W., Sakallah K.A.: **PIMO: A VLSI Circuit Partitioner for Simulation Applications**, *Proc. Int. Symp. on Circuits and Systems (ISCAS-85*, 1985, pp. 1075-1078

[Shima85] Shima, T., Mizuta, C.: **Modified Gauss-Seidel Iterative Scheme for High Speed Circuit Simulation**, *Proc. Int. Symp. on Circuits and Systems (ISCAS-85*, 1985, pp. 1061-1064

[Deut84] Deutsch, J.T., Newton, A.R.: **A Multiprocessor Implementation of Relaxation-Based Electrical Circuit Simulation**, *Proc. 21st Design Automation Conf.*, 1984, pp. 350-357

[Whit85] White, J., Sangiovanni-Vincentelli, A.L.: **Partitioning Algorithms and Parallel Implementations of Waveform Relaxation Algoritms for Circuit Simulation**, *Proc. Int. Symp. on Circuits and Systems (ISCAS-85*, 1985, pp. 221-224

[Wang91] Wang, R., Wing, O.: **Waveform Relaxation on Tightly Coupled Systems**, , pp. 2280-2283

[Vlac95] Vlach, J., Wojciechowski, J.M., Opal, A.: **Analysis of Nonlinear Networks with Inconsistent Initial Conditions**, *IEEE Trans. Circuits and Systems*, Vol. 42, No. 4, 1995, pp. 195-200

[Sale90a] Saleh, R,A., White, J.K.: **Accelerating Relaxation Algorithms for Circuit Simulation Using Waveform-Newton and Step-Size Refinement**, *IEEE Trans. Computer-Aided Design*, Vol. 9, No.9, 1990, pp. 951-958

[Gall88] Gallivan, K.A., Koss, P., Lo, S., Saleh, R.A.: **A Comparison of Parallel Relaxation-Based Circuit Simulation Techniques**, *Electro/88 Conference Record*, 1988, pp. 1-6

[Cott90] Cottrell, R.A.: **Event-Driven Behavioural Simulation of Analogue Transfer Functions**, *Proc. Euro-DAC '90*, 1990, pp. 240-243

[Sale89] Saleh, R.A., Newton, R.: **The Exploitation of Latency and Multirate Behaviour Using Nonlinear Relaxation for Circuit Simulation**, *IEEE Trans. on CAD*,Vol. 8, No. 12, 1989, pp. 1286-1298

[John91] Johnson, T.A., Zukowski, D.J.: **Waveform-Relaxatio-Based Circuit Simulation on the Victor V256 Parallel Processor**, *IBM J. Res. Develop.*, Vol. 35, No. 5/6, 1991, pp. 707-720

[Pete93] Peterson, L., Mattison, S.: **Dynamic Partitioning for Concurrent Waveform Relaxation-Based Circuit Simulation**, *1993 IEEE Int. Symp. on Circuits and Computers*, 1993, pp. 1639-1642

[Whit91] White, J., Odeh, F.: **Connecting Waveform-Relaxation Convergence Properties to the A-Stability of Multirate Integration Methods**, *COMPEL - The International Journal for Computation and Mathematics in Electrical and Electronic Engineering*, Vol. 10, No. 4, pp. 497-508

[Oden90] Odent P., Claesen, L., De Man, H.: **A Combined Waveform Relaxation - Waveform Relaxation Newton Algorithm for Efficient Parallel Circuit Simulation**, *Proc. Euro-DAC '90*, 1990, pp. 244-248

[Patr90] Patrick, D., Lyden, C.: **An Event-Driven Transient Simulation Algorithm for MOS and Bipolar Circuits**, *Proc. Euro-DAC '90*,1990, pp. 230-234

[Eijn90] Eijndhoven, van J.T.J., Stiphout, van M.T., Buurman, H.W.: **Mutirate Integration in a Direct Simulation Method**, *Proc. Euro-DAC '90*, 1990, pp. 306-309

[Besn91] Besnard, J., Benkoski, J., Hennion, B.: **ELDO-XL: A Software Accelerator for the Analysis of Digital MOS Circuits by an Analog Simulator**, *Proc. EDAC '91*, 1991, pp. 136-141

[Ko85] Ko, F.H., Sangiovanni-Vincentelli, A.: **BLOSSOM: An Algorithm and Architecture for the Solution of Large-Scale Linear System**, *IEEE Int. Conf. on CAD: VLSI in Computers*, 1983, pp. 400-403

[Duml85] Dumulgol, D., Cockx, J., De Man, H., Odnet, P.: **Segmented Waveform Relaxation Algotithms for Large Scale Circuit Simulation**, *Proc. Int. Symp. on Circuits and Systems (ISCAS-85*, 1985, pp. 1069-1072

[Carl85] Carlin, C.H.: **DC Operating Point Computation for Waveform Relaxation Analysis**, *Proc. Int. Symp. on Circuits and Systems (ISCAS-85*, 1985, pp. 1073-1074

[Nord85] Nordsieck, A.W., Lin, P.M., Hsieh, H.Y.: **Exploitation of Latency and Near-Latency for Transient Analysis of Large Scale Circuits**, *Proc. Int. Symp. on Circuits and Systems (ISCAS-85*, 1985, pp. 1079-1082

[Casi85] **Aggregation Methods for the Solution of Large Scale Systems of Linear Algebraic Equations**, *Proc. Int. Symp. on Circuits and Systems (ISCAS-85*, 1985, pp. 1055-1058

[Leel85] Leelarasmee, E.: **Exact Simulation of Large Scale Circuits by Using Approximating Models**, *Proc. Int. Symp. on Circuits and Systems (ISCAS-85*, 1985, pp. 1059-1060

[Odeh83] Odeh, F., Carlin, C.H.: **Robustness Aspects of an Adaptive Waveform Relaxation Scheme**, *IEEE Int. Conf. on CAD: VLSI in Computers*, 1983, pp. 396-399

[Wyat83] Wyatt, J.L., Zukowski, C., Glasser, L.A., Bassett, P., Penfield, P.: **The Waveform Bounding Approach to Timing Analysis of Digital MOS IC's**, *IEEE Int. Conf. on CAD: VLSI in Computers*, 1983, pp. 392-395,

[Guar83] Guarini, M., Palusinski, O.A.: **Integration of Partitioned Dynamic Systems unsig Waveform-Relaxation and Modified Functional Linearization**, *Computer Simulation, Summer Conference Papers, 1983 JUL Vancouver*, 1983, pp. 72-77

[Kim89] Kim, Y.H., Hwang, S.H., Newton R.: **Electrical-Logic Simulation and It's Application**, *IEEE Trans. on Computer-Aided Design*, Vol. 8, No. 1, pp. 8-22

[Odry86] Odryna, P., Nassif, S.: **The ADEPT Timing Simulation Algorithm**, *VLSI Systems Design*, Mar. 1986, pp. 24-34

[Altm90] Altmaë, M.: **MINT - a VHDL Simulation System**, *Proc. EDAC-'90*, 1990, pp. 102-106

[Saka85] Sakallah, K.A., Director, S.W.: **SAMSON2: An Event Driven VLSI Circuit Simulator**, *IEEE Trans. on CAD*, Vol. CAD 4. No. 4, 1985, pp. 668-684

[Chan93] Chang, M.-C., Chern, J.-H., Yang, P.: **Efficient and Robust Path Tracing Algorithm for DC Convergence Problems**, *1993 IEEE Int. Symp. on Circuits and Systems*, 1993, pp. 1635-1638

[Sale87] Saleh, R.A., Newton, A.R.: **An Event-Driven, Relaxation-Based Multirate Integration Scheme for Circuit Simulation**, *Proc. Int. Symp. Circuits and Systems*, May, 1985, pp. 600-603

[Cell95] Cellier, F.E., Otter, M., Elmqvist, H.: **Bond Graph Modeling of Variable Structure Systems**, *Proceedings ICBGM'95, Second Int. Conf. on Bond Graph Modeling and Simulation, Las Vegas, Nevada, January 15-18*, pp.49-55.

[Elmq95] Elmqvist, H., Otter, M., Cellier, F.E.: **Inline Integration: A New Mixed Symbolic/Numeric Approach for Solving Differential-Algebraic Equation Systems**, *Keynote Address, Proc. ESM'95, European Simulation Multiconference, Prague, Czech Republic, June 5.-8.*, pp. 1-12

[Lin95] Lin, S., Kuh, E.: **SWEC Speeds VLSI Simulation**, *IEEE Circuits and Devices Magazine*, Vol. 11, No. 1, pp. 10-15

[Pelz95] Pelz, G., Bielefeld, J., Zappe, F.-J., Zimmer, G.: **Simulating Micro-Electromechanical Systems**, *IEEE Circuits and Devices Magazine*, Vol. 11, No. 2, pp. 10-13

[Ruan91] Ruan, G: **A Behavioral Model of A/D Converters Using a Mixed-Mode Simulator**, *IEEE J. Solid-State Circuits*, 1991, Vol. 26, No.3, pp.283-290

[Yama90] Yamamura, K., Horiuchi, K.: **A Globally and Quadratically Convergent Algorithm for Solving Nonlinear Resistive Networks**, *IEEE Trans. Computer-Aided Design*, 1990, Vol. 9., No. 5, pp. 487-499

[Chad92] Chadha, R., Visweswariah, C., Chen, C.-F.: **M3 - A Multilevel Mixed-Mode Mixed A/D Simulator**, *IEEE Trans. Computer-Aided Design*, 1992, Vol. 11., No. 5, pp. 575-585

[Airi94] Airiau, R., Bergé, J.-M., Olive, V.: **Circuit synthesis with VHDL**, *Kluwer, Kluwer international series in engineering and computer scienc, 261*, Boston, MA, 1994

[Enge93] Engeln-Müllges, G., Reutter, F.: **Numerik-Algorithmen mit ANSI-C Programmen**, *BI-Wiss.-Verl.*, Mannheim, Leipzig ,Wien, Zürich, 1993

[Sche89] Schendel, U.: **Sparse matrices: numerical aspects with applications for scientists and engineers**, *Horwood: Mathematics and its Applications*, Chichester, 1989

[Bane94] Banerjee, P.: **Parallel Algorithms for VLSI Computer-Aided Design Applications**, *Prentice Hall*, Englewood-Cliffs, NJ, 1994

[Hämm91] Hämmerlin, G., Hoffmann, K.H.: **Numerische Mathematik**, 2. Aufl., *Springer*, Berlin, Heidelberg, New York, 1991

[Tuin88] Tuinenga, P.W.: **SPICE, a Guide to Circuit Simulationt and Analysis with PSPICE**, *Prentice Hall*, Englewood-Cliffs, NJ, 1988

[Arms93] Armstrong, J.R., Gray, F.G.: **Structured Logic Design with VHDL**, *Prentice Hall*, Englewood-Cliffs, NJ, 1993

[Deuf93] Deuflhard, P., Hohmann, A.: **Numerische Mathematik: eine algorithmisch orientierte Einführung**, *de Gruyter*, Berlin, 1993

[Zurm84] Zurmühl, R., Falk, S.: **Matrizen und ihre Anwendungen für Angewandte Mathematiker, Physiker und Ingenieure**, 5. Aufl. *Springer*, Berlin, Heidelberg, 1984

[Pres92] Press, W., Flannery, B.P., Teukolsky, S.A., Vetterling, W.T.: **Numerical recipes in Pascal, the art of scientific Computing**,

[Whit85/1] White, J., Odeh, F., Sangiovanni-Vincentelli, A., Ruehli, A.: **Waveform Relaxation: Theory and Practice**, Memorandum No. UCB/ERL M85/65, Electronics Research Lab., College of Engineering, University of Calif. at Berkeley

[Whit85/2] White, J.: **The Multirate Integration Properties of Waveorm Relaxation with application to circuit simulation and parallel computation**, Dissertation, Memorandum No. UCB/ERL M85/90, Electronics Research Lab., College of Engineering, University of Calif. at Berkeley

[Rodr94] Rodriguez, D.: **Déscription et Simulation Mixte Analogique-Numérique: Analyse de VHDL Analogique, realisation d'un simulateur mixte**, Dissertation, Université Joseph Fourier, Grenoble

[Merm92] Mermet, J.: **VHDL for Simulation, Synthesis and Formal Proofs of Hardware**, *Kluwer, Kluwer international series in engineering and computer science; 183*, Dordrecht, 1992

[Harr91] Harr, R.E.: **Applications of VHDL to circuit design**, *Kluwer*, Boston, MA, 1991

[Stoe94] Stoer, J., Bulirsch, R.: **Numerische Mathematik 1, Eine Einführung-unter Berücksichtigung von Vorlesungen von F. L. Bauer**, 7. Aufl., *Springer*, Berlin, Heidelberg, New York, 1994

[Stoe90] Stoer, J., Bulirsch, R.: **Numerische Mathematik 1, Eine Einführung-unter Berücksichtigung von Vorlesungen von F. L. Bauer**, 3. Aufl., *Springer*, Berlin, Heidelberg, New York, 1990

[Breh92] Brehm, J.: **Parallele lineare Algebra**, *Dt. Univ.-Verl., 1992*, Dissertation Univ. Erlangen, Nürnberg

[Bjoe72] Bjoerck, A., Dahlquist, G.: **Numerische Methoden**, *Oldenbourg*, München, 1972

[Jord76] Jordan-Engeln, G., Reutter, F.: **Formelsammlung zur numerischen Mathematik, Anhang: Fortran-IV Programme**, *BI- Hochschultaschenbücher/ Bibliographisches Institut: 106*, Mannheim, 1976

[Kiel94] Kielkowski, R.M.: **Inside SPICE: overcoming the obstacles of circuit simulation**, *McGraw-Hill*, New York, 1994

[Vlad94] Vladimirescu, A.: **The SPICE Book**, *Wiley*, New York, 1994

[Orte88] Ortega, J.M.: **Introduction to parallel and vector solution of linear systems**, *Plenum Pr.*, New York, 1988

[Scho92] Schoen, J.M.: **Performance and fault modeling with VHDL**, *Prentice Hall*, Englewood Cliffs, NJ, 1992

[Sche79] Scheid, F.: **Numerische Analysis: Theorie u. Anwendung**, *McGraw-Hill*, Düsseldorf, 1979

[Scha92] Schaback,R., Werner, H.: **Numerische Mathematik**, 4. Aufl., *Springer*, Berlin, Heidelberg, 1992

[Geor93] George, A., Gilbert, J.R., Liu, J.W.H.: **Graph theory and sparse matrix computation**, *Springer*, New York, Berlin, Heidelberg, 1993

[Beck77] Becker, J.: **Numerische Mathematik für Ingenieure**, *Teubner*, Stuttgart, 1977

[Pete92] Peterson, L.: **Dynamic Circuit Partitioning for Waveform Relaxation-Based Circuit Simulation** Dissertation, Department of Applied Electronics, Lund Institute of Technology, Lund, Sweden

[Mali94] Malinak, L.: **A/D Simulators: An Expanding Array of Choices**, *Electronic Design, Dec. 5, 1994*, pp. 95-102

[Moye95] Moyer, P.: **Rapport du Projet de Fin d'Etudes, Modelisation d'un Injecteur a l'aide du language VHDL-A**, Ecole Nationale Superieure de Physique de Strasbourg, 1995

[Libs94] Libs, J.-M.: **Architecture Multiprocesseur Dédiée à la Résolution des Equations aux Dérivées Partielles en 3D**, Dissertation, Ecole Centrale de Lyon, 1994

[Sale90b] Saleh, R.A., Newton, A.R.: **Mixed mode simulation**, *Kluwer, (Kluwer international series in engineering and computer science; 98)*, Boston, 1990